Patrick Blackburn
Hans van Ditmarsch
María Manzano
Fernando Soler-Toscano (Eds.)

LNAI 6680

Tools for Teaching Logic

Third International Congress, TICTTL 2011
Salamanca, Spain, June 2011
Proceedings

Springer

Lecture Notes in Artificial Intelligence 6680

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

FoLLI Publications on Logic, Language and Information

Patrick Blackburn Hans van Ditmarsch
María Manzano Fernando Soler-Toscano (Eds.)

Tools for Teaching Logic

Third International Congress, TICTTL 2011
Salamanca, Spain, June 1-4, 2011
Proceedings

Springer

Volume Editors

Patrick Blackburn
INRIA Nancy Grand-Est
615, rue du Jardin Botanique, 54602 Villers lès Nancy Cedex, France
E-mail: patrick.blackburn@loria.fr

Hans van Ditmarsch
Fernando Soler-Toscano
Universidad de Sevilla
Dpto. Filosofía y Lógica
C/ Camilo José Cela s/n, 41018 Sevilla, Spain
E-mail: {hvd,fsoler}@us.es

María Manzano
Universidad de Salamanca
Dpto. de Filosofía
Campus Unamuno, Edificio FES., 37007 Salamanca, Spain
E-mail: mara@usal.es

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-21349-6 e-ISBN 978-3-642-21350-2
DOI 10.1007/978-3-642-21350-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: Applied for

CR Subject Classification (1998): F.4, I.2.3, K.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains papers presented at TICTTL2011, the Third International Congress on Tools for Teaching Logic, held during June 1–4, 2011 in Salamanca, Spain. There were 62 submissions. Each submission was reviewed by at least two program committee members.

The first Tools for Teaching Logic congress took place in 2000. It was the idea of an international group of logicians that in 1998 created ARACNE, a European Community ALFA (America Latina Formación Académica) network. The second congress took place in 2006. The webpages of TICTTL and of these past events are:

`http://aracne.usal.es/congress/congress.html`
`http://logicae.usal.es/SICTTL/`
`http://logicae.usal.es/TICTTL/`

The congress focusses on a variety of topics including: logic teaching software, teaching formal methods, logic in the humanities, dissemination of logic courseware and logic textbooks, methods for teaching logic at different levels of instruction (secondary education, university level, and postgraduate), presentation of postgraduate programs in logic, e-learning, logic games, teaching argumentation theory and informal logic, and pedagogy of logic.

Various logicians focussed their efforts on elaborating Tools for Teaching Logic, working on the interface between philosophy, linguistics, mathematics, computer science and related disciplines. We want to thank those logicians and all the people that participated in our Tools for Teaching Logic congresses. We would also like to remember the Mexican logician José Alfredo Amor Montaño who sadly passed away shortly before the conference.

The gathering of submissions for this conference, the reviewing process and the selection of papers, as well as the production of this proceedings was made much, much easier by the quite appropriately named EasyChair Conference System. Its facilities surpass anything comparable of which we know.

March 2011

Patrick Blackburn
Hans van Ditmarsch
María Manzano
Fernando Soler-Toscano

Organization

Program Committee

Jesús Alcolea Banegas	Universidad de Valencia, Spain
Atocha Aliseda	UNAM, Mexico
Colin Allen	Indiana University, USA
Andrew Arana	Kansas State University, USA
Carlos Areces	LORIA, France
Dave Barker-Plummer	Stanford University, USA
Johan van Benthem	University of Amsterdam / Stanford University, The Netherlands/USA
Patrick Blackburn	INRIA, Lorraine, France
Krysia Broda	Imperial College London, UK
Begoña Carrascal	Euskal Herriko Unibertsitatea, Spain
Hans van Ditmarsch	University of Seville, Spain
Susanna Epp	DePaul University, USA
María José Frápolli Sanz	Universidad de Granada, Spain
Dov Gabbay	King's College, London, UK
Francisco José García Peñalvo	University of Salamanca, Spain
David Gries	Cornell University, USA
Marcia Groszek	Dartmouth College, USA
Jan Jaspars	Universiteit van Amsterdam, The Netherlands
Joost Joosten	Universitat de Barcelona, Spain
Tamara Lakins	Allegheny College, USA
Fenrong Liu	Tsinghua University, Bejing, China
Josje Lodder	Open University, The Netherlands
Itala Loffredo D'Ottaviano	UNICAMP, Brazil
Maria Manzano	Universidad de Salamanca, Spain
Huberto Marraud González	Universidad Autónoma de Madrid, Spain
Concepción Martínez Vidal	Universidade de Santiago de Compostela, Spain
Ángel Nepomuceno-Fernández	Universidad de Sevilla, Spain
Manuel Ojeda-Aciego	University of Malaga, Spain
R. Ramanujam	Institute of Mathematical Sciences, Chennai, India
Diógenes Rosales	Pontificia Universidad Católica del Perú, Peru
Fernando Soler-Toscano	Universidad de Sevilla, Spain
Bernard Sufrin	University of Oxford, Uk
Luis Vega Reñón	UNED, Spain
Richard Zach	University of Calgary, Canada

Additional Reviewers

Aguilera, Gabriel
Benotti, Luciana
Bou, Felix
García-Cobián, Ramón
Grabmayer, Clemens
Trelles, Oscar
Valverde, Agustín
Velázquez-Quesada, Fernando R.
Zenil, Héctor

Table of Contents

Teaching Argumentation Theory and Practice: The Case of *12 Angry Men*

Jesús Alcolea-Banegas

Universitat de València, Dept. de Lògica i Filosofia de la Ciència, Av. Blasco Ibañez, 3046010 Valencia, Spain
jesus.alcolea@uv.es

Abstract. The aim of this contribution is to illustrate how the pragma-dialectical model of critical discussion may be explained and studied with the jurors' deliberations in the film *12 Angry Men*. The film itself may be understood as an argument by example, and to defend this idea we take into consideration the thesis that the filmmaker wants to establish, the constraints of the medium, and the three classical perspectives on argumentation.

Keywords: Argumentation theory, pragma-dialectics, critical discussion, deliberation, (visual) rhetoric, argument by example.

1 Introduction

Argumentation is a particular communicative activity in our social life, and society needs it to be free and to live free. This means that argumentation must be present as a practice, but also as an object that must be theorized upon. The subject has a respectable tradition —as venerable at least as Aristotle's work—, according to which it is easy to recognize three perspectives on argumentation: logical, dialectical, and rhetorical [13]. For the last fifty years new theories of argumentation and new models of arguing have been developed. Among them, the pragma-dialectical school of argumentation in Amsterdam (van Eemeren and his colleagues [5, 6, 7, 8]) pursues a normative and descriptive approach, and focuses on the real argumentative practice in order to analyze and evaluate it in relation with an ideal model of *critical discussion*. In this sort of discussion two parties (*protagonist* and *antagonist*) are committed to solve a *difference of opinion* arguing reasonably, and testing critically their proposals.

Following our experience in teaching argumentation theory, our aim is to illustrate how the model of critical discussion may be explained and studied with the film *12 Angry Men* (Sidney Lumet, 1957) (from now on *12AM*), and how critical argumentation works in it. Years ago Nardone [10] used it to teach critical thinking in connexion with informal fallacies. But beyond a simple illustration, the model allows us to get a greater appreciation of Lumet's goals in shooting the film.[1] On one hand, we attend to a narrated story, a jury's deliberations in a case of murder (*internal discourse*), and on

[1] Although this is the main reason why we prefer the pragma-dialectical perspective, there are others. For instance, a referee called my attention about an abstract point of view that focuses on the actions that change the involved agents' knowledge by using also *12AM* [9].

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 1–8, 2011.

the other, we may appreciate a strong case against the death penalty (*external discourse*). For different reasons the jury may fail to provide a right verdict and send an innocent to his death, a boy accused of killing his father. As in any other situation, uncovering the truth may be a difficult matter, if not impossible. But somehow to get the truth or to reach a reasonable doubt is a collective enterprise. Hence, the critical discussion perspective imposes upon us. However, although we face the film as a tool for teaching argumentation theory and practice, the film itself is a piece of argumentation, and in order to defend this idea we should take into consideration the constraints of the medium [1]. Hence, the (visual) rhetorical perspective imposes upon us. So, it will become clear how the three perspectives on argumentation are present when arguing (critically): from the *logical* one, analysing and evaluating arguments (the product); from the *dialectical* one, the (argumentative) moves to keep on arguing according to certain rules (the procedure); and from the *rhetorical* one, the way to be efficient (the process).

If all teaching should encourage critical thinking, higher education requires students to make judgments about the evidence and the arguments placed before them. But judgments and arguments have an aesthetic component [2]: a student must be struck by the validity of an argument, must be touched by its elegance, and must feel the weight of the evidence. Students should perceive that logic is concerned with an objective relation between evidence and conclusion. However, in a determinate context, if we want to convince somebody of something, we must reach his/her beliefs and argue from them as premises, as Quine and Ullian once explained [11, pp. 130-131]. And reaching those beliefs is not a matter of convincing, but of persuading, a matter that is not the business of logic. So, in *12AM*, one of the jurors must start trying to persuade his mates with an appeal to pity. As we will see, formal rationality is not natural to the jurors. As all human beings, jurors are passionate, and they will fight through passionate language to find dispassionate arguments and reach a unanimous reasonable doubt. After being introduced into the pragma-dialectical model, students may like to discover the stages of the model in the way jurors proceed, the fallacies committed by them, and the affected rules for a critical discussion. In addition, they may appreciate the difference between getting to the truth and the way to get to the truth and its difficulties. Finally, students may be asking about the theses, if any, Lumet wants to establish with his film, and here as we will see the rhetorical perspective through an *argument by example* enters.

2 The Internal Discourse in 12*AM*: *The Pragma-Dialectical Model of Critical Discussion*

According to the pragma-dialectical model, a discussion proceeds along the following four stages: (*1rst stage*) Some difference of opinion is expressed at the *confrontation stage*: "it becomes clear that there is a standpoint that is not accepted because it runs against doubt or contradiction" [6, p. 60]. The protagonist has a point of view, and the antagonist casts doubt on it or presents an alternative. In *12AM*, all jurors but Juror #8 seem to agree that the accused boy is guilty. This is the result of a preliminary ballot.

(*2nd stage*) In the *opening stage*, both parties "try to find out how much relevant common ground they share (as to the discussion format, background knowledge,

values, and so on), in order to be able to determine whether their procedural and substantive 'zone of agreement' is sufficiently broad to conduct a fruitful discussion" [6, p. 60]. The argumentative activity depends on the degree of confidence both parties have in their ability to find a reasonable solution while discussing matters, and to find a common starting point (*common premises*) upon which to evaluate their difference of opinion. The jurors share the information provided along the trial, but they feel and interpret the evidence in a different way. Juror #12 suggests that each one should try to convince the Juror #8 that he is wrong. This just means that the jury agrees to re-examine the evidence that is based mainly on: (1) the knife used to kill the father seems to be unique; (2) the testimony of the old man, who lives in the apartment just below where the father was killed, that claims that (a) he heard the boy yelling out "I'm gonna kill ya", and a second later the body falling, and (b) after walking to his apartment's door he saw the boy running down the stairs; and (3) the testimony of a woman, that saw the boy killing his father from her room across the street, just at the moment an elevated train was passing. All this evidence is consistent with the boy's prior record of crime, and strengthens the idea that the boy is guilt.

(*3rd stage*) In the *argumentation stage*, arguments supporting the standpoint(s) are advanced and critically tested. This is the most important part of the model and correspondingly the main part in the film. After undermining the evidences, and voting four times more, the jurors have changed their initial positions, and so the initial lack of unanimity yields to a reasonable doubt. Along the process each juror reveals his mood and prejudices, which are manifested in the frequent fallacies they commit. In this stage, when students are evaluating an argument they must be interested in three factors: (1) the validity or strength of the connection between its conclusion and its premises, (2) the truth or acceptability of its premises, and (3) the freedom from informally fallacious elements. In *12AM* the second factor is the most important, because the jurors' verdict will depend on it: those premises are the evidence to conclude with a reasonable doubt. However, in our experience, the students are happy detecting fallacies and relating them with the ten rules of critical discussion in each stage [5, pp. 208ff]. In this third one, for instance, by saying from insufficient evidence "You can't believe a word they say", in a clear reference to the boy's ethnic group, Juror #10 commits a *hasty generalization*, against the rule 7, according to which a standpoint may not be regarded as conclusively defended if the defence does not take place by means of an appropriate argumentation scheme that is correctly applied (*The argument scheme rule*). On the other hand, Juror #3 initially accepts the statements made by the old witness as corresponding to facts, but later on arguing against Juror #9 he shouts out "He was an old man! He was confused! How could he be positive about anything?" Here Juror #3 is caught in a clear inconsistency of which he himself is fully aware as Lumet's camera highlights concentrating on his face. The reasonable attitude requested by argumentation presupposes rationality, and Juror #3 is violating the rule 8, according to which the reasoning must be logically valid (*The validity rule*).

Students may appreciate how the context is particularly relevant to analyze and evaluate the argumentative moves. As another important example, we may also mention two instances of the *argumentum ad populum*, one fallacious and one legitimate. This sort of argument comes to refer to joining a cause because of its popularity, or to imply that something is right because everybody is doing it or believing it. Soon after

the first ballot, when Juror #8 is the only one to pronounce a verdict of innocence, Juror #10 says, "Boy, oh boy! There's always one," committing a fallacious move that appeals to *the feeling of group solidarity* as a reason to accept the boy's guilt against Juror #8's position, and so against the rule 4, according to which a party may defend his/her standpoint only by advancing argumentation related to that standpoint (*The relevance rule*). However, when the deliberations are finishing, because all jurors but #3 have *consensually* arrived to a reasonable doubt, Juror #8 makes a legitimate move telling Juror #3 that he is alone. Then Juror #3 reaffirms his arguments, and Juror #8 answers that they are unconvincing, that he has time to examine them properly. In fact, we know that Juror #3 has a serious difficulty in convincing or persuading, because he is a very aggressive man in contrast with Juror #8. Some time for talking was the only thing that Juror #8 asked for at the beginning of the meeting when he was the sole dissenter, and right at the moment when Juror #3 asked, "What's there to talk about? Eleven of us think he's guilty. No one had to think about it twice except you." Was this deeply problematic man making a legitimate appeal? It wasn't, because none of the jurors had discussed the different standpoints.

In this context students do not find problems with the concept of truth. It can be defined rather simply as a relationship that may or may not obtain between a statement or an opinion and what the statement or the opinion is about. So, a statement is true if (and only if) it reports an actual state of affairs, that is, some present, past, or future event. If the boy killed his father, then the statement that the boy killed his father is true. Otherwise it is false. Opinions and statements are subject to error and dispute and, therefore, also to verification, which means simply seeing whether they agree with reality. Are true the witnesses' statements in *12AM*? Do they agree with reality? Answering this question probably would require some research. If the research provides conclusive evidence in favour of the premise, then we can accept the premise as true. But what constitutes *conclusive* evidence? How much evidence do we need to accept a statement as true? There is no simple answer here. Now suppose, however, that the research left reasonable doubt. Jurors in *12AM* must pronounce a verdict of not guilty.

Students are taught that problems involved in verifying statements depend partly on what type of statement is involved: verbal, factual, evaluative or interpretative. In our story the clue is mainly centred on factual statements. Because these are contingent, verifying them is problematic. In general, there are some key points to consider in evaluating observations. For instance, in relation to the film: (1) The *physical conditions* refer to the conditions under which the observations were made. As the woman who claims to have seen the boy killing his father saw him when a train was passing the street, her report would be in serious doubt. (2) The *sensory acuity* refers to the sensory abilities of the observer. As the woman needs glasses and probably she was not wearing them, because she was supposedly in bed, her report would also be in serious doubt. And (3) the *objectivity* refers to the ability to view ourselves and the world without distortion. We should be aware of people's frames of reference, their interests, and their assumptions. This doesn't mean that we should automatically dismiss the views of those who have already established views or vested interest, although they may be 'colouring' their observations and thus 'colouring' the evidence they present as justification for a statement. On watching *12AM*, students may notice two different and opposed things: (a) that in most cases personal facts about the

speakers are not relevant *to the truth-value* of their claims or *to the validity* of their arguments; and (b) that attention to the background and intentions of them *do* seem to have some relevance *on the acceptance* of their claims and arguments. Jurors have serious reasons to think that witnesses may be colouring their observations. And so the reasonable doubt makes its way towards them.

(*4th stage*) In the *concluding stage*, the critical discussion comes to an end, "in agreement that the protagonist's standpoint is acceptable and the antagonist's doubt must be retracted, or that the standpoint of the protagonist must be retracted" [6, p. 61]. The available evidence is not conclusive, and the hypothesis that the boy killed his father is a controversial one. So, the jurors claim to have a reasonable doubt in agreement with Juror #8.

The four stages just examined must be completed with some rhetorical and dialectical aspects. In fact, Van Eemeren and Houtlosser introduced the notion of *strategic manoeuvring* to take into consideration the arguers' personal aim to win the discussion (*rhetorical perspective*), which, in actual argumentative practices, is always on a pair with their wishes to conduct it under reasonable standards (*dialectical perspective*). So this notion "is directed at diminishing the potential tension between pursuing at the same time a 'dialectical' as well as a 'rhetorical' aim" [7, p. 135]. The benefits for the reconstruction of the argumentative discourse in the pragma-dialectical analysis are at least three. Let's see them in relation to the film: (1) The *rhetorical* dimension allows us a better grasp of the argumentative reality. The rhetorical appeal to pity by Juror #8, calling attention to the social circumstances of the boy, not only motivates the beginning of the argumentative process, but allows the argumentative skills of each juror to be uncovered. It will come to reveal the ability that rhetoric has to transform the audience and, eventually, reality. Students like to discover how Juror #8 attempts to free his mates from their prejudices or fears, and how all this is put to the service of the reasonable doubt that he has from the very beginning. (2) The *dialectical* dimension allows us a better understanding of the rationale behind the instantiations of the argumentative moves. Students can appreciate here the importance of the jurors' cognitive contexts and backgrounds in order to explain the flaws committed by some people in arguing, as Jurors #2, 3, 6, 7, 10, 12. But they can also respond favourably to those jurors who faithfully follow the rational use of principles, as Jurors #4, 8, 9, or 11. (3) The combination of the two dimensions allows us to understand the reasons and the sense of the fallacious moves in the argumentative practice. Along the process the jurors show their argumentative skills, and the motivations they have to proceed as they do. The impatience or simple indifference for the boy's fate (Jurors #7 and 12) and the aggressiveness or personal conflict (Jurors #3 and 10) are reflected in different kinds of fallacies to be identified and classified by the students, according to the rules of critical discussion.

Here dialectic, as "a method of regimented opposition (…) amounts to the pragmatic application of logic, a collaborative method of putting logic into use so as to move from conjecture and opinion to more secure belief" [8, p. 214]. In the resulting approach, rhetoric is seen as "the theoretical study of the various kinds of practical persuasion techniques" [7, p. 138]. Viewed dialectically, the parties are trying to decide which standpoint should be maintained. But viewed rhetorically, each one will try to claim victory, and will design accordingly the strategic maneuvering. This design will depend on (a) the *topical potential* of the relevant alternatives in each

stage, selecting the material the parties can handle well; (b) the selection of the responsive adaptation to the *audience demands*, developing the point of view most agreeable to the audience, and (c) the exploitation of the appropriate *presentational devices*, presenting the positions in the most effective way. The whole idea may be illustrated through the knife's episode in the film. Juror #8 takes advantages of the moves made by Jurors #3 and 4, who are defending the uniqueness of the knife. Juror #4 invites Juror #8 to "take a look at this knife", while sticking it on the table. Juror #8 responds to the challenge by presenting in the most effective way the best available material to him with the aim to refute both jurors: drawing from his pocket, and sticking it next to the other one, a knife that is identical and that he had bought "at a pawn shop two blocks from the boy's house". Although Juror #3 still claims victory —"Eleven of us still think he's guilty", he says—, some jurors have been persuaded and are beginning to change their mind into believing that somebody else could have done the stabbing with exactly the same kind of knife.

3 The External Discourse in *12AM*: *The Argument by Example*

Now the film is the text used to construct a complete and intelligible interpretation of it. The interpreter should decompose the special linguistic and cinematic conventions in which the film is cast in order to understand them and explicate what is implicated. At the end, the filmic text should be designed to reward such interpretive activity. The extraordinary strength of *12AM* is also based on the way in which this fictional form incorporates rational principles within the narration. We follow the jurors moving through the room, while some of them, in turn, are in the process of constructing hypotheses of the crime or of the witnesses' claims. From discrete clues, they construct a narrative of the facts, which finally verifies or corrects the provided evidence. In this way the film shows how search and (critical) argumentation can change people's beliefs and the conditions that allow their beliefs to change through search and (critical) argumentation. *If* we accept that a belief system is a potential description of what someone might believe (or accept cognitively) in some situation, *and* that the purpose of the search and (critical) argumentation is to improve a belief system that has proven to be cognitively unsatisfactory (in some sense), *then* the search and (critical) argumentation are attempts to change deliberately one's or another person's beliefs, by collecting or delivering new information to a person. But is this not what the film is providing to its (potential) viewers as well?

It seems that for Lumet the boy's fate is less significant than the ways in which it affects the jurors' beliefs, minds and sensibilities to decide his destiny. Underlying the cognitive context of those beliefs, minds and sensibilities, the film becomes a *visual argument* against death penalty. But this argument encloses an *argument by example* to be used as an invalidating case. In the argument by example, inferences move from specific to general: what is true of this case is true in general. It is indifferent to use past facts or fictitious examples, both of them can help support the case. Students may notice that we form many of our opinions through proofs provided from arguments by example. They may be also happy asking these questions: Is the example enough to prove the point? Is it representative? Is it relevant? Is it unambiguous? Could it be that the connection of general and specific doesn't hold in this case? And so on.

While watching the film, the reaction of the students ranges from boredom to the desire of viewing it again. Some of them say that it is unlikely to have a case like *12AM* in real life. Even Juror #8 may be thought to be so incredible that he would be wiped immediately from the scene by the other jurors, who are willing to end their action. However, students must be explained that the interpretation of the film as an argument by example is reinforced by the idea that, as a fictitious narration, it has a didactic and moral purpose. The judge appeals to the jurors' good conscience, and exhorts them to "try and separate the facts from the fancy". This means that they should deliberate or reason normatively from the evidence keeping in mind that they are moral subjects. In other words, each juror should be a specimen of the *vir bonus peritus dicendi*. But the film will illustrate that this is more a dream than a reality, and so, if witnesses are wrong or jurors are wrong in their deliberation, the boy's life will be at stake.

On the other hand, we know that, as a case, we can use the example to remember a general proposition. But it can also be used to explain and to persuade. In the *Rhetoric to Alexander* (1429a), the example is one of the main ways to argue. Aristotle [3] says that the example is a rhetorical induction, and clarifies that, in order to prove, everybody provides proofs by persuasion citing examples or enthymemes (*Rhetoric* 1356b5). Who is addressing to an audience cannot provide a set of particular instances to substantiate a generalization, depending on the time available, the media constraints, and to avoid boring the audience. Usually, s/he will be content to refer to one or two solid examples to support her/his generalizations. For these reasons, *12AM* is a case in point. If somebody is arguing against death penalty, s/he may be happy using the film as a practice. If somebody is trying to prove how to be critical, s/he may present Juror #8 as a good illustration. If somebody is trying to prove how prejudices lead us to commit fallacies, s/he may present Jurors #3 or 10 as terrible illustrations. All this is consistent with the author of the *Rhetorica ad Herennium*, according to which the example helps to get the ornament and so "to amplify and enrich the argumentation" [4, II.29.46]. An argument by example does not prove much really, and always may be open to refutation or may be put into question, because from a logical point of view it is just as one swallow that does not make a summer. However, from a rhetorical point of view, it has persuasive value, because that is what usually happens or what is thought to occur.

To sum up, the visual rhetoric in the film highlights how Lumet is trying to promote a cooperative model under the coverage of certain principles. His cinematic style blends with the theme and the moral meaning of the film, transforming it in a work of art to the service of the thesis he argues for. Lumet is not simply intended to show us that a group of jurors has a reasonable doubt about whether a boy is guilt or not. This is the story that we are told. But, rather, he seeks to emphasize that democracy is a collective enterprise based on critical, collaborative and public action, through the use of the word in deliberation and debate. The search for the truth, albeit a difficult undertaking, should not be overlooked. But what matters is the critical path and how it is undertaken. Lumet visually enhances this idea when shows Jurors #6 and 7 trying to open together and with a great effort the window of the jury's small room where they just walked into. Later on Jurors #1 and 8 will be closing that window when the storm dumped about the building. This is a persuasive metaphor of the collective enterprise that means to think critically. But as the oppressive smallness of

the room reflects this task should be carried out under the constraint of certain principles. Even an initial shot of the Corinthian columns of the courthouse's façade is suggesting to viewers the rational way according to which the forthcoming action will / should proceed, if the prejudice must be defeated, and so death penalty. When we invite the students to watch the film from these theoretical and practical perspectives, we may remember Seneca's recommendation to Lucilius, "… the living voice and the intimacy of a common life will help you more than the written word. You must go to the scene of action, first, because men put more faith in their eyes than in their ears, and, second, because the way is long if one follows precepts, but short and helpful, if one follows patterns (*exempla*)" [12, VI.5].

Acknowledgments. The author is grateful to the Spanish Ministry of Science and Innovation for supporting this work (Projects FFI2008-00085/FISO & FFI2008-01169/FISO). He is also grateful to V. Claramonte, M.C. Fuster, M. Parreño, J. Valor, and two anonymous referees for their thoughtful comments and suggestions.

References

1. Alcolea, J.: Visual Arguments in Film. Argumentation 23, 259–275 (2009)
2. Alcolea, J.: Relacions entre Bellesa i Arguments Visuals. In: Monserrat, J., Roviró, I. (eds.) La Bellesa. Col·loquis de Vic XIV, pp. 188–197. SCF, Barcelona (2010)
3. Aristotle: The Complete Works of Aristotle, edited by J. Barnes. Princeton University Press, Princeton, 4th printing, 2 vols. (1991)
4. Cicero, M.T.: Rhetorica ad Herennium. Harvard University Press, Cambridge (1981)
5. van Eemeren, F.H., Grootendorst, R.: Argumentation, Communication and Fallacies: A Pragma-Dialectical Perspective. L. Erlbaum, Hillsdale (1992)
6. van Eemeren, F.H., Grootendorst, R.: A Systematic Theory of Argumentation: The Pragma-Dialectical Approach. Cambridge University Press, Cambridge (2004)
7. van Eemeren, F.H., Houtlosser, P.: Strategic Maneuvering. Maintaining a Delicate Balance. In: van Eemeren, F.H., Houtlosser, P. (eds.) Dialectic and Rhetoric, pp. 131–159. Kluwer, Dordrecht (2002)
8. van Eemeren, F.H., et al.: Argumentation. In: van Dijk, T.A. (ed.) Discourse as Structure and Process, pp. 208–229. SAGE, London (1997)
9. Grossi, D., Velázquez-Quesada, F.R.: Twelve Angry Men: A Study on the Fine-Grain of Announcements. In: He, X., Horty, J., Pacuit, E. (eds.) LORI 2009. LNCS, vol. 5834, pp. 147–160. Springer, Heidelberg (2009)
10. Nardone, H.F.: Using the Film 12 Angry Men to Teach Critical Thinking. In: van Eemeren, F.H., et al. (eds.) Proceedings of the Third ISSA Conference on Argumentation, vol. III, pp. 313–326. Sic Sat, Amsterdam (1995)
11. Quine, W.V.O., Ullian, J.S.: The Web of Belief. Random House, New York (1978)
12. Seneca, L.A.: Ad Lucilium Epistulae Morales. Harvard University Press, Cambridge (1989)
13. Wenzel, J.W.: Perspectives on Argument. In: Benoit, W.L., et al. (eds.) Readings in Argumentation, pp. 121–143. Foris, Berlin (1992)

E-learning and Semantic Technologies: Tools and Concepts

Enrique Alonso

Dept. Lingüística, Lenguas Modernas, Lógica y Filosofía de la Ciencia
Facultad de Filosofía y Letras
UAM
enrique.alonso@uam.es

Abstract. The use of automated tools for teaching logic shows a set of stages on which it is worth reflecting. Each of them make a different interpretation of the task which is strongly determined by the resources available. At present the generalization of *Markup Languages* commonly used to compose the documents populating the Web can pose a new model for tools employed to teach logic. After a brief historical overview of the previous stages, I will analyze the architecture of this new model and will offer two case studies on which there has already been some work done.

Keywords: Tools for teaching logic, Semantic Web, Web ontologies, WYSIWYM editors.

1 Automated Deduction Applications

Tools originally used for teaching logic could be considered as automatic calculators mainly oriented to show the students strategies to solve problems. These were small applications designed to offer to the beginners assisted examples of techniques they were supposed to acquire as a fundamental part of their formal skills in logic. Their finest moment arrives in the 90's. During this period the main programs that played some roles in the teaching of logic were published and distributed. Nevertheless their roots can be found in developments obtained during the second half of the 80's. Contrary to what one might expect, the first initiatives did not occur in the field of logical calculi, but in a mixed area heavily inspired in model theory. *Tarski's World* began to be distributed with *The Language of First-Order Logic* by J. J. Barwise and Etchemendy in 1990 but it is possible that early releases were already available some time before.

Today CSLI still maintains an active site, *The Openproof Project*, devoted to the study and design of software useful for teaching logic. According to the organizers words it is an initiative born in the early 80's focused almost entirely on applications aimed at what they describe as:

> application of software to Problems in logic.

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 9–23, 2011.

Tarski's World was able to evaluate a set of formulas interpreted on an elementary geometric universe[1] and solve the question of the satisfiability with respect to the given model. Later, in 1994, *Hyperproof* appears, also under Barwise and Etchemendy's address, whose main novelty is the introduction of a system of rules capable of guiding the student to obtain consequences from an initial situation. Probably these are the most commonly used tools in real experiences of teaching logic over the years.

Although this kind of automated calculator represent the oldest use of informatic tools in teaching logic, the truth is that nowdays we can still find several research programmes focused on the design of calculators for diverse logical calculi. Former editions of Tools for Teaching Logic Congress show a useful guide through the main results in this field.[2] We can find two main developments, one oriented to the implementation of environments based on *natural deduction* and another centered in *analytic tableau*. In the first group, we should mention the initiative led by W. Sieg since late 80's. This work is still open under the name of the *AProS Project* -*A*utomated *Pro*of *S*earch Project. In its last release it included new topics going far beyond the strict domain of clasical logic. This is the case of the so called *Strategic Thinking* to which Sieg's group has devoted its attention. It is also worth mentioning the project *Pandora* led by K. Broda. It is surely one of the best software tools designed to show our students typical strategies for solving exercises in natural deduction calculus. In the field of analytic tableau we should include *Tableau III* developed in Oxford in 2001 and *LoTrec*, created under an IRIT initiative at Lilac. While the first one focuses on sentential logic and first order logic, the second is noteworthy because it is focused on modal and descriptive logics which is certainly a novelty.

I do not intend to go into details because the impression we get from a brief tour of these tools points to the existence of a jungle of initiatives. The success of these initiatives depends most of the time on the potential of the institution in which they were developed, and the capacity of enthusiasm and dissemination of their leading researchers. Those places where dialogue among computer scientist and logicians was fluid have promoted, as it is easy to imagine, earlier and more durable initiatives. In the others, and here I include almost every university department or research center with an active line of work in logic, we can also find at one time or another, initiatives aimed at creating some kind of automatic calculator with a potential use in teaching logic. Many of these have ceased, as is evidenced by the large number of abandoned sites, and others have never reached fully operational releases. But, perhaps this is not what really interests me. The interpretation of logic as a tool for the conceptual analysis of discourse and therefore as a discipline to evaluate argumentation is deeply rooted in our community. I have no objection against this way of seeing things, but it is not the way I understand the role of logic. It is true that almost any introductory course in logic includes among its objectives the description of several formal systems, calculus,

[1] It consists of a grid where users can locate a number of bodies such as spheres, pyramids, tetrahedrons, etc. The formulas to be evaluated make reference to the size and relative position of these figures on the grid.

[2] Cite the previous monograph.

and the exposition of the corresponding skills to solve problems with them. But I think the aim of logic as a discipline goes far beyond these very limited targets. Nevertheless, the software available from the late 80's and 90's does not allow the imagination of very different implementations of those just discussed. Therefore we should not be too critical of these projects. The lest pleasant result of this view was to reinforce the conception of logic as a tool for analysis and assessment of argumentation, something which conflicts with the objections coming from informal logic and argumentation theory. The relative lack of applications focusing on translation from ordinary language to different formal languages has been identified, with some irony, as evidence against the practical value of these methods in real case studies. Another objection points to the fact that sometimes what the students had just learned was actually the management of a program rather than the formal tools this application was intended to teach. There were other cases in which on the contrary, the implementation required an extensive prior knowledge of logic to achieve an efficient use of the environment, so that the desired effect was not achieved in either case.

I do not think we can judge this model harshly because it responds to a sincere attempt to provide our students with a user-friendly access to technicalities that have always been considered complex and have caused some rejection among our students. It should also be noted that in most cases it was quite difficult to design such programs so that the work done was really good. Perhaps the most interesting thing in this period was the experience acquaired by many researchers in combining logic and software engineering to design tools that currently go far beyond a limited educational use.

2 Transition: From Presentations to E-learning Platforms

2.1 Presentations

Since the end of the 90's the use of presentations has become common, initially in Congresses and scientific meetings, and later in the classroom. It is true that this tool is not specific to logic, but it would be unfair not to consider its role in support of teaching and it is necessary for me to clarify the point I want to make. The presentations preserve a structure developed a long time ago from the use of slides or overhead projectors. In fact it could be said that the normal use of these resources is nothing but a translation into the digital age of what many wanted to do with an overhead projector but couldn't. From my point of view this tool is of fundamental importance because it opens the use of new technologies to aspects of teaching of logic that goes beyond the typical homework. For a time it seemed that the only possible use of software for teaching logic was the applications designed to solve problems. It was as if the logical nature of programming was the only point of contact with logical enquiry. The proof of a theorem, the explanation of the reasons or motivations of some fundamental result and its historical context, were matters that laid outside the scope of teaching tools specifically designed for logic. This is the kind of conception that I will criticize later.

The use of presentations in the classroom was not generally widespread until the early twenty-first century and the reason lies in the cost of the equipment needed. Professionals had previously acquired a lot of experience from its use at conferences and scientific meetings so that the advantages were obvious for every one. Even though these advantages actually came from its non-digital ancestor: the slides used in overhead projectors. One of the most important features of presentations is the ability to reuse materials created for a session at any time and in any other disposition: the *cut and paste* resources became to be used by everybody. The ability to reuse materials justified investments in time and effort that would otherwise be undesirable. Our way to practice and devise teaching would not be understood without widespread use of these resources.

One of the obvious disadvantages of some of these tools is their mishandling of special symbols, and in particular, of logical symbols. This fact oriented its employment to a very general use avoiding any kind of technical terminology. Only the explanation of very general ideas and facts could obtain some kind of gains from the use of presentations. The emergence of LaTeX classes specifically aimed at the production of presentations in late 90's[3] solved the problem but always with the specific costs of production which has the composition of documents in LaTeX.

Despite these problems, presentations are interesting because of their undeniable contribution to a much broader understanding of the use of new technologies in teaching. What is needed is not only software oriented to solving problems, but perhaps to the construction of explanations, test batteries, or to the description of more complex matters such as proofs of some fundamental theorems. Why not?

It is true that the inability to access the code of the software used by the most popular tool, *Microsoft PowerPoint*, did not allow the user to imagine the design of specialized tools that could contribute to an automatic handling of new aspects of teaching. But this was not the case for free software environments such as LaTeX in which the software itself was available to the community of developers. In fact, I think that the philosophy supporting LaTeX can contribute decisively to define new models of tools oriented to teach logic. It is true that these applications, presentations, beamers, etc, are considered more often as simple audio-visual resources. However to the extent that it is possible to conceive them as the result of some programming it is also possible to imagine more sophisticated interventions for which the programming plays a role. This will be discussed later.

2.2 E-learning Platforms

For some time, presentations were used as desktop apps running from the client side therefore rejecting any dynamic component[4] from its behaviour. Today, we

[3] The first distribution of the *Beamer* class, perhaps the most widespread at present, dates form 2003.

[4] *Dynamic* behaviour does not reduce to the presence of mobile components such as pop-ups, emerging windows, advertisements, banners and so on, but a real interaction between client and server sides.

live in what is probably the golden age of the so-called *e-learning platforms*. I think everyone has suffcient experience with these resources and so I will spare unnecessary descriptions. The great contribution of these tools is the integration of almost every relevant component of the traditional teaching in a single environment. The main consecuence of this movement towards an unified enviroment for teaching tools is the integration of all those applicactions into a common technical infrastruture. The teacher's task now consists of gathering texts and materials used in class presentations and explanations, tests, bibliographies, links to external materials, applications to solve exercises, and so on in a site having the aspecto of a single web page. All this has to be put together inside a common technical ground offered by the encoding of the e-learning platform.

At present all these materials are developed using very different techniques, some are presentations in one of the formats characteristic for this type of tool, others are mere text files, forms in HTML, videos, hyperlinks, etc. There is no kind of unity among them except that they are introduced by teacher to gather them into the same environment. This situation is increasingly strange because in practice all of them are related to each other in several ways. A teacher may have uploaded a text that the student should read at some point, in turn, that material is relevant to the explanation which is summarized in a presentation also displayed in the site. Finally to check the undestanding of a problem it is possible that the teacher has designed a test that students have try to answer, and which will be in some way evaluated. To address questions relating to this test, it is highly desirable to have a forum and perhaps, if the teacher has enough time, it is also very useful to produce some feedback with corrections and comments. As we have seen, there is a continuity between all these materials that, however, can never be transmitted through an e-learning platform adequately. The only way to do this is to gather all the material into the same virtual unit establishing in a fully manual way the mutual references. Teacher are supossed to express through appropiate instructions the relationship of each item with the rest of the material stored. So if a paper, book or video is neccesary to understand a presentation, the teacher has to inform them of that by means of the corresponding directions. The materials needed to answer a test or a question have also to be indicated in some way. But can this be done in a better way? To the extent that each of the contents included in on-line courses are built from various and incompatible resources it seems very difficult to move towards a greater integration of all these resources. Presentations, texts, forms and so on are at this moment resources not compatible from the point of view of software engineering .

E-learning platforms can be done in several ways, but the truth is that the technical infrastructure can be extremely simple if desired. The architecture of these tools does not differ substantially from that work on a blog or a social network: all these resources are examples of what we call today called Dynamic Web, or also Web 2.0. It is based on a combination of programming and markup languages that has proven extremely effective.

The base infrastructure is provided by the HTML, the language in which the web is weaved, the interaction between users and sites becomes possible thanks to a typical bundle of forms and WYSIWYG editors based on JavaScript, while the interaction between user and server runs thanks to the combined action of PHP and MySql. The importance of this model is based on its simplicity and its definition from open source and free software initiatives, all of them oriented to Web design. With these initial conditions, it is not difficult to imagine a greater integration of the different resources that are now part of an online course. To define new models of relation between human action and automated processes in teaching is not a matter of discovering new technologies. It has to do with our own philosophy and general ways of conceiving the interaction and possibilities of techonolgies which are present among us.

3 Mark-Up Technolgies

Applications oriented to problem solving skills have promoted the habit of designing our own software solutions. Programing an application was not thought to be a waste of time, nor a task independent of logical concerns. The popularization of presentations had the advantage of extending the use of new technologies to very general matters like explanations, general results, historical background, and so on. Finally, e-learning platforms allow intergration of almost every kind of resource into a single working environment. Moreover, the technological ground of e-learning platforms is the same that makes possible the communication through the Net, a fact of fundamental importance to understanding the next step.

3.1 Semantic Technologies and Tagged Text

In May 2001, Sir Tim Berners-Lee published an extraordinarily thought provoking paper which raised the possibility of pushing the network architecture in a new direction: the Semantic Web. Since then intensive work has been carried out on the design of new formalisms in which logic has played some role[5] obtanining relevant advances in the definition and implementation of conceptual maps, *ontologies*, for many different areas. Nevetheless, the progress of semantic techniques and the semantic web programm has been less successful than expected.

At present, semantic web seems to have become a sort of vague claim for more active and efficient techniques to connect human bahaviour with automatic enviroments in the Net not propertly substantiated by an unique research programm. To obtain this result I think we should improve our understanding of content tagging through metadata and the ways in which machines and humans interact. Sematic web is no more the particular and very concrete research

[5] Some of the most popular languages in Semantic Web initiative such as OWL, come from *descriptive logics*.

program started years ago, but a general philosophy I prefer to associate with the *technologies for tagging text* and metadata handling. And now the question is how can we make use of these techniques to reconsider the use of software in teaching and in particular in the teaching of logic?

Teaching is an activity which, despite pedagogy demands, can be divided into a very specific set of events or stages:

i. The construction of explanations for a number of issues previously taken as a target.
ii. The compilation of lists of support materials or literature.
iii. The development of tests and questions designed to check students understanding.
iv. The evaluation of these tests and possible explanation for the errors.

E-learning platforms are designed to gather all these stages with some efficiency. Nevertheless, this can only be achived through the manual assembly of a large number of very different materials and resources.

To compound this the teacher will take as a starting point texts that are either theirs, or have been cited previously in the literature or even have been created expressly for the occasion. Presentations are usually drawn up on existing material cut and pasted onto slides to apply a certain format, animation, transitions, etc. Questions included in tests are often constructed from texts containing statements that actually are their answers. The procedure just described, which is the most common, suggests a considerable investment of effort and what is worse, the repetition of work that, at least in its most creative stages, has already been done.

Tagging or markup technologies, generally understood, meet all the requirements which implement metadata structures on an existing content or allow the elaboration of a text including metadata on items typed during the same process. I think it is possible to use some of these resources to develop simultaneously or at least in a coordinated way, some of the materials that make up the different teaching stages described above. In particular, starting from an existing text, I think it is possible to imagine a *modus operandi* to obtain both an interactive explanation of its content and a test or collection of test incluiding some feeback with a minimun of teacher intervention. The new model of tools for teaching would be oriented to the use of a minimal effort to obtain a large amount of items that now have to be independently developed. To achive this goal we would make use of different metadata structures, taxonomies,[6] that could be recursively or simultaneously used on the same content thereby generating different outcomes.

I do not pretend to eliminate any creative aspect of the various stages of teaching, but to reduce the extra effort and repetitive tasks presently involved in e-learning standards. From this point of view traditional teaching still results a more creative and amazing practice for teachers who do not find it very exciting

[6] We could have made use of the term *ontology* to describe this structure of metadata, but there is some controversy with respect to scope of this term, so that we avoid its use -see Gruber.

to retype contents they have written on previous occassions. The technique I propose to solve this disappointing look at e-learning technologies is not new, nor does it require either great effort in the field of programming. Luckily, it is something much more modest: just a little change in the way some existing technologies can be used and applied to the making-up proccess of teaching materials.

3.2 Towards a New Model

The main tool for producing content in the digital age is the *word processor*. Almost any other activity having to do with teaching makes, at one time or another, use of a word processor. Originally word processors were unable to display text on screen consistent with the final printed page. The speed limits of computer processors did not allow that outcome. The text present on the display was a combination of data and tags used to apply some styles to plain text.[7] This situation was understood from the outset as a flaw to be overcome in some way thereby fixing the *WYSIWYG* -What You See Is What You Get- philosophy, majority today. For some time the only alternative to this editing text philosophy was the different environments for editing in LaTeXdetermined to keep in view all metadata structure that was applied to the text for further compilation. This strategy resulted in a degree of unpopularity and the fame to be a tool exclusively oriented to scientific or professional editing. But the truth is that time has changed many of our views on this edition model to the point of giving it a label that identifies its ideological position. Word processor showing the code used to add metadata layers, semantic, typographical or whatever, received the name of *WYSIWYM* -What You See Is What You Mean- editors in a clear gesture of complicity to the initiatives coming from Semantic Web. Another factor that has contributed to a new understanding of these editors is the extension and popularization of HTML and all its cognates. Web pages and sites are supported by a structure written in HTML whose code can be seen at all times. HTML is a markup language designed to facilitate the display of any kind of data. The web exists thanks to a philosophy strongly based on the distinction and independence of data and metadata structure so that the net itself can be seen as a vast collection of documents linked by metadata structures of increasing complexity.

The recurring tendency in recent years has been to assess very positively the separation of data and metadata structures in a document, but I think we still have not drawn all the consequences of this fact. Something a user might want to do is to create their own taxonomy so that it could be efficiently incorporated in a user-friendly text editor. Perhaps they might want something else, such as applying not one but, several taxonomies simultaneously on a text that is being typed. Some of these taxonomies could be theirs while others may have been taken from other environments. The reason there hs been so little progress in

[7] WordStar or WordPerfect, two classics in text editing, started that way.

this direction may be due to the fact that using a taxonomy, let alone its design, to tag some text is seen as a highly specialized process far from simple user skills, but perhaps this is what is changing.

To apply a taxonomy on a text, usually a metadata structure, is something we should reflect on for a moment. The simplest way to do this is by using a plain text editor and choosing a sufficiently general syntax for the taxonomy terms, one that could be recognized as a standard. At present that standard is offered by XML syntax widely adopted by a lot of net languages which try to follow its recomendations. An example of a more or less standard tag in a XML syntax style could look like this[8]:

```
<sentence id=1>
  <who>Gödel</who>
  <verb>proved</verb>
  <what>the incompleteness theorem for PA</what>
  <when>in 1931</when>
</sentence>
```

It seems a bit naïve to demand that users type manually the tags to be applied into their documents. This is still more evident if we consider that the model tries to make it possible to use a large set of taxonomies at the same time. It does not seem reasonable to ask users to make the extra effort of learning the exact meaning and management of the tags of every taxonomy they use. People who used to write LaTeXdocuments are well aware of the advantages of having asisted editing environments based on IDEs -Integrated Developement Enviroments- to facilitate the process of creation and typing of content. Until now, to develop a text editor capable of integrating in their structure mechanisms to tag text was a task of some complexity, not much, but enough to require a certain expertise in the design of desktop applications. Nevertheless, a few years ago some WYSIWYG editors were adapted to network environments becoming typical in webmail apps and some e-learning platforms, *Moodle* represents a good example. The decision taken by some developers to offer free versions of these word processors has allowed us to understand an extremely simple technology and to use it to obtain experimental prototypes of *tagging editors* that have exactly the required functionalities. These word proccesors are mounted on a typical network architecture incluiding HTML and JavaScript code that allows us to add toolbars containing all the buttons and commands needed to implement an taxonomy. Each button in one of these toolbars launches the corresponding tag or the corresponding complex structure obtained from combining several tags in a higher level object. These tags would be then applied to a text previoulsy highlighted through mouse actions, or would be fullfiled by typing text into their delimiters. Nothing prevents the user, among other things, to also apply typographical format in exactly the same way. In fact, for one of these tag editors typographical format is just another taxonomy which can be implemented on

[8] This example is an advance of the taxonomies we will use in one of our case studies.

a text to obtain the desired result. All metadata structures seem to share a common logical status that would be semantic in all cases, or in none, but I recognize this is a controversial issue that I do not want to deal with now.

To arrange an environment to launch taxonomies, and tag text without changing its basic structure is a fundamental part of the developement model I intend to propose, but it is obviously only a part. It implies a level of abstraction of the basic functions of text editors not fully understood until now in all its importance. Text editors would be then interfaces between a taxonomy loaded for the occasion and the computer keyboard. Without a taxonomy or library previously added, our editor could only produce plain text, nothing else. With a taxonomy, not to say a bundle of them, a text editor can compose almost every conceivable outcome based on the web infrastruture.

This new generation of tools allows us to imagine the free creation of taxonomies useful for teaching needs, although it is obvious that nothing would prevent its use in any other area. But our goal is not, as seems obvious, the mere typesetting of a tagged text but the manufacture of certain products aimed at teaching tasks and specific needs. It seems clear that a document formatted with extra metadata structure somehow contains all the instructions needed to build some other entity. Data would be the arguments and metadata would represent the functions whose definition we have to get to produce the final product. In the model I have proposed the result of applying some metadata to the data previously given should be a dynamic website. It seems obvious that it is so if our goal is, among other things, to obtain test batteries to be automatically evaluated or some interactive explanations. In more strict terms, the desired result of a tagged text composed from a teaching taxonomy should be a collection of .html, .php and .js documents grouped into a functional recognizable unit.

The transformation of a tagged document in any other entity is carried out through a compiling process[9] which can consist of several stages. In our case it is possible to identify at least three:

i. *parsing* of the original document
ii. distribution of the data in a datatable
iii. data recovery from databases and its use to build the resulting files.

It is clear that the taxonomy used to tag the text typed by a text editor forms a functional unit with the compiler needed to produce the site that itself is the result of the process. This forces us to think about the construction of complex packages that can be treated as autonomous units of tools that play a role in different parts of the process. It may seem of a little complexity, but it is actually a model typical of free software developements and clearly growing today. In fact, it is used by some of the most successful companies in the field of CMSs. This model allows individual developers to produce *modules*, *components* and *plugins*

[9] This interpretation of the process of *compiling* could be considered too wide by many, but is quite usual today.

simply by accepting a minimum of standards in the design of the aproppiate interfaces to connect new apps with the core of these environments.

The model I have proposed consists of a core and some *components* to be rejoined in the kernel and that can be changed when needed. The kernel consists of two different units, a *text editor* and a *compiler*. The components incorporate on the one hand the taxonomy load into the text editor as a toolbar, and on the other the items and directions the compiler should use to translate the text generated by the editor into a final product. The idea is simple and could be summarized in the scheme showed in fig.1:

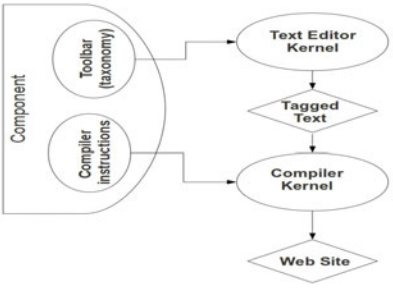


Fig. 1.

Although the work done so far can only be considered at an early stage, it is possible to analyze in more detail a couple of applications with a specific use in teaching and in particular in teaching logic.

3.3 Case Studies

As I have said throughout this brief presentation of the project, two of the most relevant stages in every teaching experience are the design of an exposition of some contents, and the construction of batteries of questions students must answer to check their comprehension of a subject. At present we already have some work done on the second of the two aspects and a series of assumptions about the first one. In the first place I will comment on the work that has already been done.

Components for designing test batteries. The target is to use an environment based on the architecture described above to automatically generate tests that students can answer receiving their qualifications and some comments about possible mistakes. The idea is to take as a starting point an already existing text and to add some format to generate questions. First, I will describe without details the taxonomy that can be used in the production of this type of content and a general example of the expected result.

An erotetic taxonomy (abstract)

- main element:

  `<sentence id=...></sentence>`

- erotetic tags:

  `<who></who>, <what></what>, <when></when>, <where></where>,`

  `<how></how>`

- sentence core:

  `<verb></verb>`

We now consider a very basic example. Let us suppose that we load into our text editor a text containing the following piece of information:

> " In 1931 Gödel probes undecidability of the First Order Peano Arithmetic frustating the formalistic program hopes sustained by Hilbert and his colleagues."

There are several ways to tag a statement like this and some of them may require some rewriting of the original text but I will avoid this aspect. A possible result is this:

```
<sentence id=1>
<when> In 1931 </when>
<who>Gödel</who>
<verb>proved</verb>
<what>the undecidability of First Orden Peano Arithmetic</what>
frustrating the hopes of formalist programm sustained by Hilbert
and his followers
</sentence>
```

We assume that the addition of tags has been achieved through the regular use of the mouse to highlight the affected data and then clicking on the tabs corresponding to the tag you want to launch. This tag is then inserted between the highlighted text introducing spaces, line breaks or other graphic resources to facilitate the proper display of the structure so generated. As can be seen in the example, not all the text has been tagged. We can make partial selections or we can also rewrite some parts to obtain a better grammatical coherence.

From this statement plus the metadata structure the system can generate at least the following questions:

i. When does Gödel prove the undecidability of ...?
ii. Who does prove the undecidability of ...?
iii. What does Gödel prove in 1931?

Depending on the grammatical structure of each language, the interface employed to generate question forms from the corresponding indicative sentences will be more or less complex. It is therefore important that this project remains open to component additions coming from independent groups interested in this type of technology. Really nothing prevents the existence of a number of taxonomies or compilers with different interpretations of the grammatical structure of erotetic logic even into the same language. What matters is the final result, and not the way to conceptualize content which may vary between authors.

The arrangment of questions in a test can adopt two possible structures: a multiple choice test, or open questions. In the first case students are suppossed to choose the right answer among a set of plausible alternatives. The system then suggests a qualification offering the correct answer and some evidence supporting it. In the second case, students can type an extended answer and the system only suggest the correct answer showing the corresponding evidence. The forms contained in fig. 2 is an example of typical multiple-choice test.

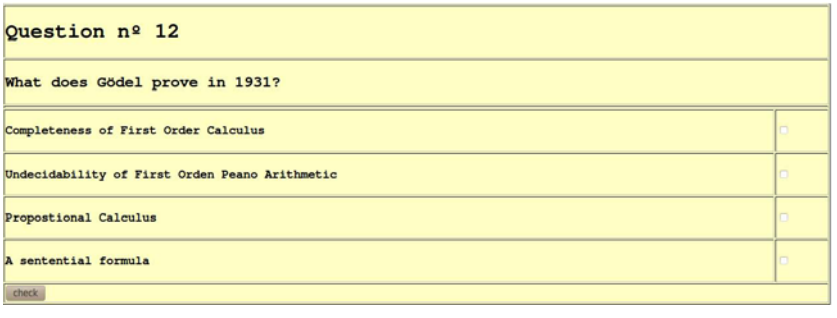


Fig. 2.

The generation of possible responses, including *inter alia* the right, is done through random selection of items within the same category the right answer belongs to. If the target of a question is of type *what*, as in the example, the other alternatives must also have been obtained from other items in the same field. This strategy ensures partial coherence among alternatives in a test, but does not offer a complete guarantee of it. The last two answers listed as alternatives in the example above, are partially inconsistent though perhaps not too much so as to be excluded. Even though this part of the process should be executed in a purely automated way, the truth is that the questions must pass a human filter to discard absurd tests of null interest for students. In order to produce batteries of questions with a suitable number of alternatives the system must possess a certain amount of data. If this critical level, calculated by the system, was not reached, it would be most feasible to propose only free-response questions in which this kind of problem does not arise.

Once the student responds by clicking the tab in the form, they should receive a reply containing at least the right answer and the evidence supporting it, as reflected in the fig. 3.

Fig. 3.

The dialogue the student establishes with the environment through their answers and their own assesment of them through the evidence supplied by the system is a really interesting field worth analyzing, but this is beyond the scope of this paper.

Components to design expositions. As I have said elsewhere in this same study, explanations or simple expositions are now provided by *presentations* built up with Microsoft *PowerPoint* or the beamer class of LaTeX. To imagine a taxonomy able to produce a bundle of dynamic web pages containing certain content is not difficult. To attach some of the dynamic behavior characteristic of presentations is trivial. To incorporate some feedback based on interactions between client and server side actions is now possible, thanks to the fact that our presentations are no different from any other dynamic web site. Its is pointless to say that this is something that is beyond the scope of the usual presentations. Now it is possible, for example, to suggest that the student answer a question, to find some data or content displayed in a short video and then follow the next step of our exposition. One advantage recently added to HTML web pages is the ability to compile text sequences encoded in LaTeXto show the corresponding result which entirely opens the possibility of using this procedure on typical contents of our discipline.

This approach only supposes some novelty with respect to the way we produce the file or final site, but I do not intend to present it as a relevant contribution in any way. It is quite evident that the latest moves in the generation of content on multiple e-learning platforms are now going in that direction. What really interests me is the application of the procedure to the presentation of contents that so far have not had an easy treatment in logic. I am thinking in particular about Theorems and other results that constitute the basic core of nonelementary courses of logic. The structure of expositions of these contents shows general patterns which are easy to obtain and translate into an appropriate taxonomy. The templates developed over the years by users of LaTeX either individually, or within specific groups,[10] show a useful example of what can be done in this

[10] The scientific publishers are the best example of this.

direction. Definitions, lemmas, theorems, corollaries form what would be some of the basic categories that should certainly be accommodated within a taxonomy oriented to presentations of logic contents. But we can go one step further and incorporate some of the basic skills needed to prove theorems in logic. I think of techniques such as *induction* and *reductio*. We all have enough experience in such techniques to know that it is usually possible to identify the basic components and display them in a reasonably organized way. For a better result, I suppose it would be nice to add the logical connectives and cuantors and also some argumentative particles usually employed in arguments, and proofs proper of purely formal contexts. This would ultimately make effective, not without some irony, the old dream of *metamathematics*, but on completely different basis.

On this occasion, I do not dare to advance a more detailed description of taxonomies that could incorporate the management of such kind of items. However the work is promising and provides not only a technology to improve teaching skills in logic, but also an elucidation of the formal devices that allow us to describe, I will not say to *formalize*, the structure of proofs of fundamental theorems in our discipline. Depending on the success of this method, it could also be considered the design of tag systems devoted to show our students typical strategies to prove theorems in logic, even in those cases in which some originality is present.

References

1. van Ditmarsch, H., Manzano, M.: Editorial 'Tools for Teaching Logic'. Logic Jnl IGPL 15(4), 289–292 (2007)
2. Broda, K., et al.: Pandora: A Reasoning Toolbox using Natural Deduction Style. Logic Jnl IGPL 15(4), 293–304
3. Sieg, W.: The AProS Project: Strategic Thinking and Computational Logic. Logic Jnl IGPL 15(4), 359–368
4. Pérez-Lancho, B., et al.: Software Tools in Logic Education: Some Examples. Logic Jnl IGPL 15(4), 347–357
5. Huertas, A.: Teaching and Learning Logic in a Virtual Learning Enviroment. Logic Jnl IGPL 15(4), 321–331
6. Berners-Lee, T., Fischetti, M.: Weaving the Web. Harper Collins Publisher, New York (1999)
7. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American (May 17, 2001)
8. Gruber, T.R.: A translation approach to portable ontology specifications. Knowledge Acquisition 5(2), 199–220
9. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies 43(4-5), 907–928
10. Openproof Project, http://ggww.stanford.edu/NGUS/Openproof/
11. W3C Semantic Web Activity, http://www.w3.org/2001/sw/

CT2.0: A Collaborative Database of Examples for Teaching Informal Logic

Peter Bradley

Department of Philosophy and Religious Studies
McDaniel College
Westminster, MD 21157, USA
http://inquiry.mcdaniel.edu/

Abstract. I propose to create a system of collaborative textbook authorship, review and delivery that would allow for multimedia interactive content and real-life contemporary examples that can take advantage of social networking and other 'web2.0' technologies. The potential for my system is great: the interactivity supported by social-networking frameworks will increase student engagement, the scalable model allows for embedded CT instruction in various disciplines, and the approach allows for a revolutionary model of textbook customization, pricing and distribution.

Keywords: Critical Thinking, Informal Logic.

1 Introduction

Most direct, sustained instruction in informal logic or Critical Thinking (hereafter 'CT') courses occurs in stand-alone classes run by Philosophy departments. While these can be very effective if taught well, they can be tedious and boring. In an effort to maintain the interest of their students, we professors frequently supplement the static prose of standard textbooks with dynamic, current, and often local examples of informal reasoning culled from editorial pages, punditry talk shows and even late night comedy, all in an effort to make direct instruction in reasoning relevant to students' interests. The irony, of course, is that the students are spending most of their time in courses that are already directly related to their interests—courses in their major. And those courses often include ample examples suitable for critical thinking instruction.

By studying student performance over a 15 year period, D. Hatcher showed that students who had small amounts of direct instruction in reasoning in the context of their regular course work performed better than students who had taken a stand-alone course in CT[1]. Contemporary CT theorists like McPeck[2] and Ennis[3] and have debated whether there is anything generalizable about CT. McPeck famously held that all CT is domain-specific, and hence cannot be taught outside of the disciplines where it is required[2]. While I disagree

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 24–29, 2011.

with McPeck and hold, as Ennis argued[5],[1], that there are generalizable CT skills such as contradiction avoidance and evidentiary relations that transcend academic domain, I also hold that CT instruction is most successful when it is integrated into discipline-specific courses.

One of the most obvious reasons why direct instruction in CT is not integrated into discipline-specific courses is the lack of supporting texts. Anthony Weston's classic *A Rulebook for Arguments*[9], which sells (currently) for $7.95 was first published in 1986. This is a great book, which can be (and is) added to many courses as a supplement. But it is limited. It does not cover scientific reasoning in any depth, and it wholly dismisses appeals to emotion and character as fallacious, rather than attempting to understand and evaluate them as attempts at persuasion.

Most importantly, however, publishers are no longer interested in small, cheap supplements. The current crop of CT and Logic books on the market blur the line between the fields. Every textbook now tries to be everything to everyone, which entails that students are paying for a great deal of content their courses will not, in fact, cover. In response, some textbook publishers are allowing for print-on-demand selections of well-known textbooks. For example, Patrick Hurley's *A Concise Introduction to Logic*[10] is now available this way. But these options still assume a stand-alone CT or Logic course.

To integrate direct instruction in reasoning into existing discipline-specific courses, professors need thin, adaptable texts. What's more, we need texts that can provide examples with commentary, hyperlinks, video, audio and interactivity. All of that is available in the medium of the internet. Internet-based texts would be even 'thinner' and more adaptable than Weston's book. They would be infinitely malleable to curricular needs. Moreover, a correctly designed system could simultaneously produce printed versions for sale as course packets.

2 Teaching from Example

CT instruction has long been based on an example-centric model. The trend started by Max Black's seminal textbook[11] is in evidence in all major textbooks used today. Most CT professors have supplemented the texts by amassing large collections of examples, many of which are withering away in file cabinets.

There are many problems with this system, but I will reflect on four:

(1) The examples become superannuated. For example, one of the best small textbooks that I have already mentioned, Anthony Weston's *A Rulebook for Arguments*, contains an example from Joseph McCarthy (pg. 85). It is a good example—a classic, in fact—but it is not cited and cannot be said to be current. One of the best recent books, Louis Vaughn's *The Power of Critical Thinking: Effective Reasoning About Ordinary and Extraordinary Claims*[12], takes the opposite approach by including examples that are meant to be contemporary. The first chapter, on the "Power of Critical Thinking," which includes a section on "Why It Matters," cites the 1994 movie *Dumb and Dumber* (pg. 5) and the

[1] For more on the issue, see McPeck's response to Ennis[6] as well as [7] and [8].

1998 movie *The Truman Show* (pg. 6). Vaughn's idea of making the content relevant to popular culture is admirable, but it falls short. First year students entering this fall were only 3 years old when *Dumb and Dumber* came out, and 7 when *The Truman Show* opened.

While making the content relevant to the students' pop culture or subculture lives is one potential path of engagement with informal reasoning, there is another obvious path staring us in the face: most students are in courses in their chosen major because they are interested in the topic. Direct instruction in reasoning can be no more relevant than when it is attached to the topic they are volunteering to study. Hence, integrated instruction will always be more engaging than stand-alone instruction.

(2) The examples are usually presented in static, printed documents. Again, Vaughn's book serves as an example of the contrived attempts of getting popular culture into textbooks. He cites Monty Python's famous 'argument clinic' sketch in a call-out-box on page 13. This example has been used in CT courses for years to present the formal notion of 'argument' as distinct from 'mere contradiction.' In Vaughn's book, it is presented as a static dialogue. If students have seen the video, they will respond better to the video than to a static transcript. A recent YouTube search found 171 versions of the sketch, most of which are reenacted by drama students, Lego men, transformers, etc. But there are at least five versions of the original.

While there is virtue in grappling with actual printed text, much of the information consumed by modern students is dynamic and interactive. The NY Times, Washington Post and BBC all allow instantaneous discussion by readers of all their content. Wikipedia and YouTube, which consist entirely of user-generated content, dominates the information life of the contemporary undergraduate. They are wholly comfortable getting their news from Jon Stewart, and sharing links, videos and commentary on social networking sites. We need to engage these skills in the direct instruction of reasoning, not avoid them.

(3) The textbook cannot incorporate student feedback. A student who comes across a questionable example cannot check to see if other students have the same question. Consider another famous example used by Robert Ennis. The Cornell Critical Thinking test X asks students to imagine that they are exploring a newly discovered planet. The question is:

Which is more believable?

(A) The health officer investigates further and says "This water is safe to drink"

(B) Several others are soldiers. One of them says "This water supply is not safe."

(C) A and B are equally reasonable.

The keyed answer is (A). As Ennis himself points out, this question is open to profound cultural influence. For example, the health department in Puerto Rico is widely known to be corrupt, but the army is trusted. When faced with this question, Puerto Ricans must make a secondary judgment such as 'what would

the authors of this test think is the most credible source?' before answering the question with the key.

If the question were presented to a student with a forum that allowed for discussion of the question itself (as all major newspapers do with their stories), those few students who recognize that their instinctive answer is not the desired one could express their reservations about the question. Not only would these students feel less marginalized by the text, but it would help us, as authors, revise the questions for future courses.

It is impossible for the author of a book to understand all the complexities of students' cultural biases. Books should be open to revision when such biases are brought to our attention. Additionally, this constant feedback will help us judge the oh-so-important line between resonant and irrelevant and replace examples before they become outdated. The web 2.0 technologies that drive the instantaneous feedback systems for newspapers, social networking tools of Facebook and the rating systems of YouTube are not proprietary. They are publicly available freeware. And employing them for instructional content will enable us to develop better content.

Finally, a great example will always become the subject of discussion. Our students discuss, share, and interact online. They consume information online. They should be able to reflect critically as a group online. We have the technology to allow it: Facebook has opened its platform for third-party applications development, and Google has launched 'OpenSocial,' which will allow third-party applications to run on multiple social networking sites.

(4) The examples used in class are not subjected to peer-review. The best examples, specifically those that appear in textbooks and journals, are subjected to peer review. But the examples that connect with the students lives are often those that appear out of the blue. They are teachable moments that occur during the course of the semester and serve to drive the point home. Those examples are not currently peer-reviewed, shared, or archived. These examples have an extraordinary power when used effectively and should be made available for future use.

We need a system to collect, share and review examples of argumentation regardless of medium. That system should embed the examples that drive instruction in the standard introductory prose found in most textbooks, while taking advantage of the social-networking and web 2.0 functionality with which our students are so familiar.

3 My Solution: CT2.0

Contemporary social networking sites and 'web 2.0' technologies present us with an opportunity to coordinate in one reusable project the efforts of large numbers of solitary CT professors. Moreover, the collaborative nature of the project allows us to foster a liberal arts community around Critical Thinking instruction, and eventually develop a repository of peer-reviewed discipline-specific examples.

The potential for this project, hereafter 'CT2.0,' is vast. First, by creating a system of peer-review, traditional standards of academic rigor can be enforced and as a result individual professors will have a reliable resource for contemporary supplementary examples. Second, by supporting all digital media, the examples can be continually refreshed in order to maintain that all-important relevancy that underlies engagement. Third, by adapting the content to take advantage of social-networking functionality, the system will engage students 'where they live' and pioneer an entirely new way of thinking about the information delivered by textbooks. Finally, by producing the examples in a variety of digital and print forms, the project can revolutionize the economics of textbook delivery.

The project takes the form of an interactive website built on a database of examples 'wrapped' in expository prose. These examples, which when wrapped become 'modules,' will be classified according to multiple browsable and searchable taxonomic schema: at least one will correspond to the standard taxonomies of reasoning, and the second according to disciplinary divisions. Further taxonomic schema may be developed. The modules along with the classification schema will be subjected to peer-review. The site allows for ratings and comments by students and instructors. The modules can then be selected and arranged by individual instructors, and delivered via the website itself, embedded into a social networking site such as Facebook, downloaded as a PDF file, or even printed via the typesetting system LaTeX.

By adapting the existing system of editorship and peer-review into a wiki-based interface, editing and text maintenance is streamlined. Updates and revisions are instantaneous, rather than annual. The content can adapt to curricular needs and individual student responses. It can support upper-level as well as lower-level courses.

This model surpasses the traditional textbook model in a number of ways. Demand from the consumer, not the producer, dictates the method of delivery. Editions become a thing of the past. A static (i.e. printed) version of the content is merely a snapshot of a textbook undergoing constant revision. The content engages the student through embedded multimedia and interactive examples. The pricing is scalable—in one extreme, it can be folded into tuition; on another, can cost pennies per module. The delivery system is optimized for the student's world, engaging them on their own terms, and allowing them to interact with the content itself, not just the 'end-of-chapter' review questions. With a well-designed interface, the content is searchable and browsable, rather than just linearly readable.

As an instructor, I select the modules I want to include in my course. This both reduces waste and focuses the class. It allows for discipline-specific examples, ready for integration into different courses. I can specify exactly what version of those modules my class accesses–so if updates are approved during my course, the content to which the students have access will not change until I have approved that change. Finally, given that I am still more comfortable reviewing texts on paper rather than on the screen, I can get the exact content—the up to the

minute version—in instantaneously generated PDF files that I can review in hard copy.

As with any academic endeavor, the success of this project will depend on the people that become involved. Tools for Teaching Logic provides a superb opportunity to develop the network of contributors and editors necessary for a truly international resource.

References

1. Hatcher, D.: Stand-Alone Versus Integrated Critical Thinking Courses. J. Gen. Ed. 55, 247–272 (2006)
2. McPeck, J.: Critical Thinking and Education. St. Martin's Press, New York (1981)
3. Ennis, R.H.: Critical Thinking: A Streamlined Conception. Teaching Philosophy 14(1), 5–25 (1991)
4. Paul, R.: Dialogical Thinking: Critical Thought Essential to the Acquisition of Rational Knowledge and Passion. In: Baron, J.B., Sternberg, R.J. (eds.) Teaching Thinking Skills: Theory and Practice, pp. 127–148. Freeman and Company, New York (1987)
5. Ennis, R.H.: Critical Thinking and Subject Specificity: Clarification and Needed Research. Educational Researcher 18(3), 4–10 (1989)
6. McPeck, J.: Critical Thinking and Subject Specificity: A Reply to Ennis. Educational Researcher 19(4), 10–12 (1990)
7. Facione, P.: Testing College-Level Critical Thinking. Liberal Education 72, 221–231 (1986)
8. Siegel, H.: Educating Reason. Routledge, New York (1988)
9. Weston, A.: A Rulebook for Arguments. Hackett, New York (2008)
10. Hurley, P.: A Concise Introduction to Logic. Thomson Wadsworth, Belmont (2008)
11. Black, M.: Critical Thinking: An Introduction to Logic and Scientific Method. Prentice Hall, New York (1946)
12. Vaughn, L.: The Power of Critical Thinking: Effective Reasoning About Ordinary and Extraordinary Claims, Oxford (2008)

Araucaria-PL: Software for Teaching Argumentation Theory

Katarzyna Budzynska

Section of Logic, Cardinal Stefan Wyszynski University in Warsaw, Poland

Abstract. The paper aims to present the software system Araucaria-PL which is the only Polish tool designed to teach argumentation theory. It is developed on the basis of Araucaria by Reed and Rowe and extended with a module capturing persuasive aspects of argumentation. The tool has been used to create a Polish online corpus of analyzed argumentation ArgDB-pl. Moreover, the paper presents the preliminary study of usefulness of Araucaria-PL for teaching about argument structures and schemes at the standard Polish courses of logic and rhetoric.

Keywords: argumentation, pedagogy, software tools.

Introduction

One of the most important methods in pedagogy of argumentation theory is the graph-theoretic technique of argument diagramming. An argument is described as a directed graph $D = (V, E)$, where V is a set of vertices representing premises or conclusion, and E is a set of arrows which are 2-element ordered pairs of V representing the relation of support or attack between statements. Diagrams allow the visualization of the structure of argumentation, i.e. its components and interrelations among them (see e.g. [21, 12, 2]), as well as the schemes of argumentation, i.e. patterns of reasoning (see e.g. [23, 8, 10, 12]). Visualization can then be used by an analyst as a starting point to critically evaluate the argument properties such as acceptability of premises or the strength of the inference.

The pioneering application of argument diagramming ("mapping") in pedagogy of logic was proposed by the English logician Whately in 1830s [25]. He proposed to start analysis with finding a main claim of a text and then its premises. The procedure should be iterated so many times so that all the premises, including the basic ones, are found. Nowadays, argument diagrams have been attracting increasing attention and have become a standard topic in textbooks on informal logic (see e.g. [24]) and in philosophy classes [6]. The diagramming method has been shown to have a significant effect on teaching philosophy [22]. An interest in diagramming methods triggered the development of numerous software tools supporting argument mapping.[1] While there is a growing number of the tools which require the command of English to use them,

[1] Examples of softwares for argument analysis: Argunet (http://www.argunet.org/debates/), Argumentative (http://sourceforge.net/projects/argumentative/), Athena [15], Carneades [5], Cohere [17], Debategraph (http://debategraph.org/), Parmenides [1], Rationale (http://rationale.austhink.com/), TruthMapping (http://truthmapping.com/).

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 30–37, 2011.

Araucaria-PL is a first Polish tool supporting analysis of argumentation. Araucaria-PL is the Polish version of Araucaria developed by Reed and Rowe [13] extended with a module capturing persuasive aspects of argumentation.

The main contribution of the paper is the presentation of the software Araucaria-PL, its module extending the original Araucaria with the persuasive context of arguments, a first corpus of analyzed Polish arguments and preliminary pilot evaluation study of the usefulness of the software for teaching logic and argumentation theory in Poland.

1 Araucaria

Araucaria [13] is a software system which supports an analyst in reconstructing and diagramming argumentative texts. Even though there are numerous software systems supporting argument analysis, the strong advantage of Araucaria is its rich argument-theoretic background. The tool does not only map a main claim, arguments pro- and con- or evaluation of argument, but it also allows to visually represent the deep structure of argumentation. It also makes use of argumentation schemes which allow the analyst to describe reasoning patterns employed in a given case. Araucaria provides five argumentation schemesets, e.g., Perelman's schemeset [10] consisting of 38 schemes.

The rich theoretical background of Araucaria makes it especially suitable for pedagogical aims. It allows a teacher to introduce different models of structure and schemes of argumentation. Its influence on enhancing skills in argumentation theory can then support skills in other disciplines (as shown for philosophy in [16]). Araucaria has been showed to be useful by its successful application in teaching philosophy, argumentation theory and law in many universities in America and in Europe.

2 Teaching Argumentation Theory in Poland

Poland has a strong tradition in formal logic, however, in the last decade the tendency to more informal approach can be observed. This tendency is particulary strong in introductory logic courses in the humanities. At the same time, a few Polish textbooks on argumentation theory were published: [19], which was the first textbook entirely dedicated to pedagogy of argumentation theory and critical thinking, and [20,7].

The first two textbooks dedicate a lot of attention to argumentation schemes and structure and to the method of argument diagramming. Still, there was no software designed to support courses in argumentation theory without the additional burden of students having to work in English. Such a situation is all the more difficult as in the humanities, where there is the most interest in the informal approach, the majority of students do not speak English to a degree which would allow them to use a non-Polish software. That was the motivation for developing Araucaria-PL.

3 Araucaria-PL

Araucaria-PL is a software tool for argument analysis based on AML (the Argument Markup Language) designed in XML. The tool has two main functionalities: it supports

argument diagramming and argument evaluation. Araucaria-PL is the Polish version of Araucaria [13] and is the first and only tool for argument analysis that has a Polish language interface and Polish schemesets.

The diagramming starts with uploading an argument saved as plain text. The text appears in the blue box on the left of the main window (see Fig. 1). The first node of a diagram is created by selecting (with the mouse) the portion of the text corresponding to a premise or a conclusion of argumentation. When the mouse is clicked in the white box on the right, the node appears at the yellow bar at the bottom. When two or more argument components are identified, their interrelation can be represented. The analyst should select one node with the mouse and drag it to the other node. The first node will be diagrammed as the premise of an argument and the second one as its conclusion. Once other nodes are identified, the structure of arguments is created by dragging the mouse from the new nodes to the node at the existing diagram. The nodes can show either a letter assigned to a unit of text (as in Fig. 1) or its full text (as in Fig. 2).

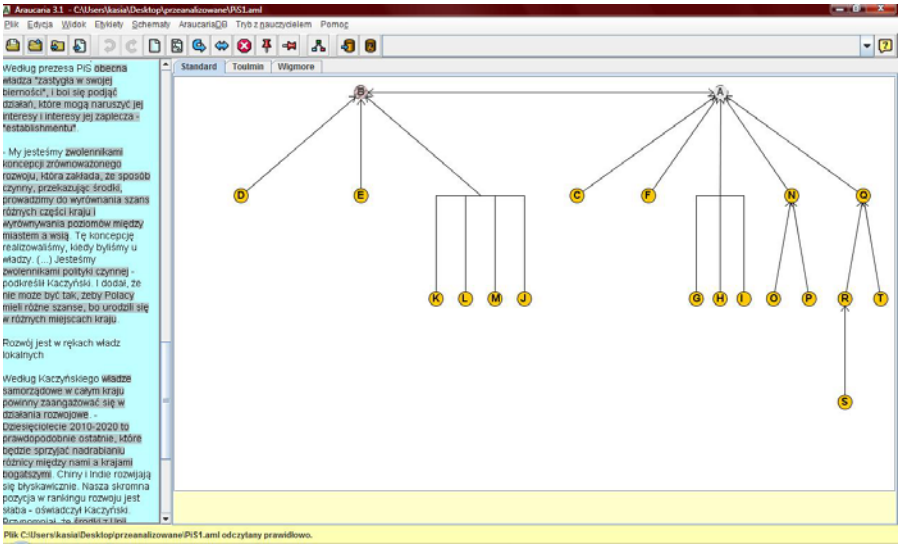


Fig. 1. Araucaria-PL main window (the example taken form ArgDB-pl corpus, the text comes from the Polish leading newspaper *Gazeta Wyborcza*, 16th Oct 2010)

Araucaria-PL supports representation of various argument structures according to different models: standard, Toulmin [21] and Wigmore [26]. In the standard model, Araucaria-PL allows visualization of structures such as linked argument (e.g. the nodes K, L, M, J, B in Fig. 1), convergent argument (e.g. O, P, B), serial argument (e.g. the chain of nodes S-R-Q-A), enthymemes (missing components of argumentation structure, e.g., A) and refutations (e.g. the node B is the refutation of A). An enthymeme is typed in by the analyst and visualized as a grey node with dashed borders. A refutation is represented as a pink node. Araucaria-PL allows also to mark the owner of the claim as e.g. Ann or Bob in Fig. 2.

One of the important features of Araucaria-PL is its support for argumentation schemes showing reasoning patterns used to draw conclusions in a given argument. For instance, the diagram in Fig. 2 visualizes the argument that was created according to the Argumentation Scheme from Consequences. The analyst can choose from different schemesets proposed by various theorists of argumentation: Walton [23], Pollock [12], Perelman and Olbrechts-Tyteca [10], Katzav and Reed [8] and Budzynska [3] (the schemeset unique for Araucaria-PL). Moreover, Araucaria-PL allows the analyst to define a new schemeset and save it in a schemeset file. When a schemeset is uploaded, the components of an argument can be selected with the mouse. Then, an appropriate scheme can be chosen from the list of the schemeset. The scheme is visualized as a shaded region (see Fig. 2). Both structure and schemes of argumentation can easily be changed with the help of 'undo' and 'redo' options. Analyzed arguments are saved in a format AML or as a JPEG image.

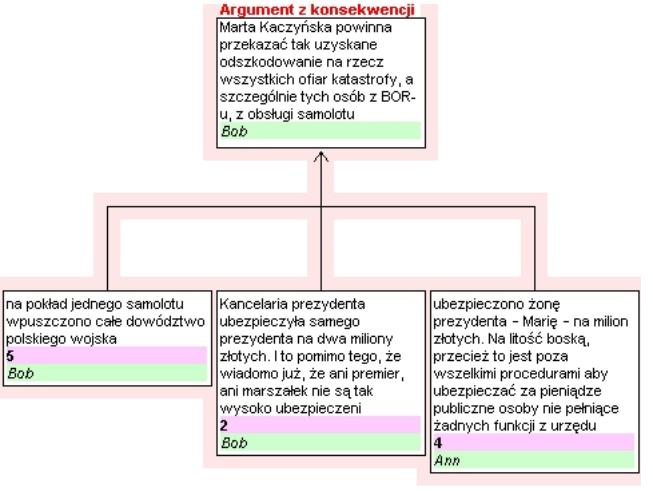


Fig. 2. Argument from Consequences (the example taken form ArgDB-pl corpus, the text comes from the blog of Polish politician Janusz Palikot, 31st May 2010)

Araucaria-PL supports argument evaluation in two different ways: (1) through evaluation labelling, or (2) through critical questions associated with a particular argumentation scheme. The evaluation labelling allows the analyst to show his confidence in the acceptability of premises or the strength of the relation of support or attack. To assign the evaluation, a node or an arrow can be selected with the mouse and the editor should be used to type in evaluation. The strength of this feature in Araucaria-PL is that the analyst can create his own scale of evaluation (e.g. 'bad–neutral–good' or '1–2–3–4–5'), however, such an assessment is arbitrary. Yet, this feature can be compensated for by the method of testing the argument with the use of critical questions. Once the analyst assigns the argumentation scheme to a given argument, the editor for that scheme provides the set of critical questions that can help the analyst to evaluate the plausibility

of the conclusion inferred on the basis of the given premises and scheme. In a pedagogical setting, the critical questions are powerful tool of enhancing students' skills of argument evaluation, since they give a "hint" as to what should be carefully considered as a potential weak point of an argument which follows the specific reasoning pattern.

Araucaria-PL is free, open-source software. The entire package can be downloaded as a zip file from the project webpage: http://argumentacja.pdg.pl

4 Application in Pedagogy

The potential utility of Araucaria-PL for teaching logic and argumentation theory in Poland was tested in a small pilot evaluation study. The character of the study is exploratory, since the tool is newly released (May 2010). In the future, further large scale studies will be conducted to measure pre- and post-performance in various student groups. The study was conducted during a course of logic for logic major graduates (a class of 6 students) and during a course of rhetoric for management major undergraduates (a class of 7 students). In addition, the feedback was gathered from logic teachers.

Student feedback was very positive, but also suggested a number of improvements (some of the comments repeat those discussed in [6]; only unique comments are presented here): (1) it should be possible to select two separate fragments of text which constitute one component of argument and associate them to one node; (2) when the text of two components overlaps, the text should be inserted twice into both nodes; (3) editing the text in a node should be possible; (4) the bigger parts of a diagram should be selectable in an easier way e.g. by using left key of the mouse (instead of clicking on subsequent nodes with "shift" pressed); (5) the same statement appearing in different places of a text should be represented with the same letter; (6) the yellow bar at the bottom could be removed – all operations could be done in the white panel; (7) different owners could be marked with different colors, which is particularly important in a big diagram with a lot of interpersonal interaction; (8) the option of determining one's own types of arguments/evidence (as in Wigmore model) and one's own list of available functionalities or argument descriptors should be allowed; (9) the legend of functions of the tool should be available.

Generally, students pointed out the following strengths of Araucaria-PL: (1) the possibility of building complex diagrams; (2) support for understanding arguments provided in a text; (3) the simplicity of operating; (4) the ease of correcting the diagram and archiving the analyzed arguments; (5) possibility of copying the results of analysis (e.g. to a Word file); (6) applicability to different domains (such as business, law etc.).

Teachers also offered very positive feedback and suggested some possible – if rather demanding and idealistic – enhancements: (1) automated reasoning (other relations among statements should be represented, e.g., the introduction of logical connectives could enable automated reasoning); (2) automated recognition (the system should recognize, e.g., inference, argumentation schemes, evaluate arguments); (3) self-learning (the teacher would prepare the set of examples of arguments with their determined structure and once the student who learns on his own makes a mistake, then the system could inform him about it, e.g., by marking the fault part of the diagram with red color).

5 Persuasive and Dialogical Context of Argumentation

In real-life practice, argumentation is most often performed in persuasive and dialogical contexts. Araucaria was designed purely to support the analyst in describing the process of argumentation, however, some natural arguments can not be adequately represented without the possibility of capturing argumentation context. From educational experience in courses of "pure" argumentation theory (which mostly ignores the dialogical and persuasive context), the majority of students fail to analyze argumentation in which this context played significant role, such as e.g.:

(1) a. Ann: I think that the attorney who proves the innocence of the defendant should be certain that he didn't commit a given crime.
b. Bob: Really? And should the doctor be certain that the person has a given disease before starting examining him? [19, p12]

(2) a. I know my son. If he lied, then he certainly did it only because he was afraid of being rejected by his peers. [7, p112]

For (1), the majority of students did not recognize the existence of argument from analogy in (1b). They had to be guided to analyze the subsequent moves of the dialogue and to identify in each of step who is the speaker and what does s/he claims. For (2), the majority of students identified the argument in (2a) as argument from authority, while the mother's statement "I know my son" is an appeal to her ethos (the credibility of a proponent) as a proponent of the argument supporting the claim that her son is not guilty. Ethos (credibility of a proponent) is a crucial mechanism of persuasion recognized by Aristotle in his theory of rhetoric [9]. The students had to be guided to consider whose credibility is at issue by distinguishing an argument's proponent from an authority to whom a proponent may refer.

"Pure" argumentation theory does not support students' skills in analysis of this type of argumentation, nor do tools which are based on such a theoretical background. Some attempts have been made to capture the dialogical context of argumentation [14], however, the theoretical results are not yet implemented in Araucaria or Araucaria-PL.[2] On the other hand, [3] proposed to represent persuasive aspects of arguments using one of the most influential contemporary theories of persuasion, the Elaboration Likelihood Model (ELM) [11]. The theory was implemented in Araucaria-PL as an additional module. The ELM assumes that there are two routes to persuasion: central and peripheral. The central route to persuasion is related to content-based arguments, while the peripheral route is related to a more insubstantial facet, such as credibility or attractiveness of the proponent. Using the ELM and Walton's schemeset [23], AraucariaPL provides a schemeset for capturing arguments' persuasive context. Among peripheral argumentation schemes, are distinguished, e.g.:

x says α, x is credible, therefore (plausibly) α,
x says α, x is attractive, therefore (plausibly) α.

[2] The results are implemented in a tool OVA [18], which slightly differs in functionality from Araucaria.

6 Corpus of Analyzed Polish Argumentation

Araucaria-PL has been used to create the first Polish corpus of analyzed natural argument, ArgDB-pl. ArgDB-pl uses the open AIF standard for argument representation [4]. The AIF aims to bring together a wide variety of argumentation technologies so that they can work together. ArgDB-pl is developed as a Polish version of ArgDB (http://argdb.computing.dundee.ac.uk/). This corpus was collected from May 2010 from a variety of domains such as newspapers, news webpages and blogs of politicians. ArgDB-pl allows users to search the database for annotated arguments using several search criteria.

The corpus is freely available online (http://argumentacja.pdg.pl/argdbpl/) for both access and update, so it can be easily used for the educational purposes. Its advantage is that it consists of real world arguments, which enrich the teaching experience with realistic examples. ArgDB-pl is useful for both the teacher (as a resource of examples) and the student (for practicing outside the regular classes). The corpus allows the student to first read a text (on the list of argumentative texts) and to try to analyze the text on his own, and then see the exemplary solution by pushing the "see the diagram" button. Since outcomes of analyses are saved not only in the AML format used by Araucaria-PL, but also in the AIF format, a student or a teacher can open an analyzed example of argument in any other software tool which uses the AIF standard language.

7 Conclusions

The paper has presented Araucaria-PL which is the only Polish software supporting pedagogy of argumentation theory. It extends the original Araucaria with a module for handling the persuasive context of argumentation. Araucaria-PL has been demonstrated to be useful in an educational setting not only directly by aiding argument analysis, but also by allowing the creation of ArgDB-pl, which is the first Polish corpus of analyzed natural argument. The corpus can be used by teachers and students as a resource of real-life examples of arguments. The small pilot study showed that students find Araucaria-PL helpful and are satisfied by the functionality of the tool. The students and teachers suggested some useful improvements for Araucaria-PL. In the future, Araucaria-PL will be further used in courses of logic and will be tested for its effectiveness in a larger scale study with more rigorous investigation of the objective impact on teaching argumentation theory in Poland.

Acknowledgments

The author gratefully acknowledges the support from Polish Ministry of Science and Higher Education under grant N N101 009338. The author would also like to thank Chris Reed for helpful comments and Joanna Skulska for aid in the collection of interviews with students.

References

1. Atkinson, K., Bench-Capon, T.J.M., McBurney, P.: Parmenides: facilitating deliberation in democracies. Artificial Intelligence and Law 14(4), 261–275 (2006)
2. Budzynska, K.: Argument analysis: Components of interpersonal argumentation. In: Frontiers in Artificial Intelligence and Applications, vol. 216, pp. 135–146. IOS Press, Amsterdam (2010)
3. Budzynska, K.: Towards the model of central and peripheral arguments. In: Proceedings of 10th International Conference on Computational Models of Natural Argument (CMNA 2010), pp. 5–9 (2010)
4. Chesnevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S.: Towards an argument interchange format. The Knowledge Engineering Review 21(4), 293–316 (2006)
5. Gordon, T.F.: Constructing arguments with a computational model of an argumentation scheme for legal rules. In: Proc. of the Conference on Artificial Intelligence and Law (2007)
6. Harrell, M.: Using argument diagramming software in the classroom. Teaching Philosophy 28(2), 163–177 (2005)
7. Hołówka, T.: Kultura logiczna w przykładach (Logical education in examples). PWN (2007)
8. Katzav, J., Reed, C.: On argumentation schemes and the natural classification of arguments. Argumentation 18(2), 239–259 (2004)
9. Kennedy, G.: Aristotle, On Rhetoric. Oxford University Press, Oxford (1991)
10. Perelman, C., Olbrechts-Tyteca, L.: The New Rhetoric. Notre Dame Press (1969)
11. Petty, R.E., Cacioppo, J.T.: Communication and persuasion: Central and peripheral routes to attitude change. Springer, New York (1986)
12. Pollock, J.: Cognitive Carpentry. MIT Press, Cambridge (1995)
13. Reed, C., Rowe, G.: Araucaria: Software for argument analysis, diagramming and representation. International Journal of AI Tools 14(3-4), 961–980 (2004)
14. Reed, C., Wells, S., Budzynska, K., Devereux, J.: Building arguments with argumentation: the role of illocutionary force in computational models of argument. In: Frontiers in Artificial Intelligence and Applications, vol. 216, pp. 415–426 (2010)
15. Rolf, B., Magnusson, C.: Developing the art of argumentation. a software approach. In: SicSat (ed.) Proc. of ISSA 2002 (2002)
16. Rowe, G., Macagno, F., Reed, C., Walton, D.: Araucaria as a tool for diagramming arguments in teaching and studying philsophy. Teaching Philosophy 29(2), 111–124 (2006)
17. Shum, S.B.: Cohere: Towards web 2.0 argumentation. In: Proceedings of the 2nd International Conference on Computational Models of Argument, COMMA (2008)
18. Snaith, M., Devereux, J., Lawrence, J., Reed, C.: Pipelining argumentation technologies. In: Frontiers in Artificial Intelligence and Applications, vol. 216, pp. 447–454 (2010)
19. Szymanek, K., Wieczorek, K., Wójcik, A.: Sztuka argumentacji (The art of argumentation). PWN (2003)
20. Tokarz, M.: Argumentacja, Perswazja, Manipulacja (Argumentation, Persuasion, Manipulation). GWP (2006)
21. Toulmin, S.: The Uses of Argument. Cambridge University Press, Cambridge (1958)
22. Twardy, C.R.: Argument maps improve critical thinking. Teaching Philosophy 27(2), 95–116 (2004)
23. Walton, D.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum, Mahwah (1996)
24. Walton, D.N.: Fundamentals of Critical Argumentation. Cambridge University Press, Cambridge (2006)
25. Whately, R.: Elements of Logic, 6th edn. B. Fellowes, London (1836)
26. Wigmore, J.: The Principles of Judicial Proof. Little, Brown and Company, Boston (1913)

Teaching Logic in Philosophy

Begoña Carrascal

Department of Logic and Philosophy of Science,
University of the Basque Country
Avda. de Tolosa 70
20018 Donostia–San Sebastian, Spain
b.carrascal@ehu.es

Abstract. Logic considered as a general model of reasoning is equated in most cases with argumentation. The symbolic component of logic is the main component of the teaching of the subject, even in a first course in Philosophy, but for students with poor technical skills, formal logic is not the best way to enhance their thinking and their initiation to the inquiry. To improve their reasoning, it would be better to begin with different problem solving tasks in natural language and only after their resolution should we try to symbolize them. Moreover, argumentation and reasoning are related but they aren't the same thing. In argumentation the linguistic and social components can't be forgotten. Training in argumentation is left to logic or never taught. Students' argumentative skills should be developed in parallel both in oral discussions and by writing.

Keywords: logic, critical thinking, argumentation, teaching.

1 Introduction

Logic was born from a concern with the correctness of argumentation in the Ancient Greece, and it has been considered one of the basic subjects in Philosophy since it was established as a discipline by Aristotle. As a consequence, logic is part of the curriculum of Philosophy in most of the proposals of the different Universities. Moreover, in the actuality, logic has become a formal discipline by its own and its methods and applications go usually far away from this first goal and intuitive definitions. Nowadays, the methods of logic, even at elementary level, are formal and abstract, because they try to grasp the essence of good reasoning by appealing to the form of the argument, formalizing it in a symbolic language. This said, it naturally follows that the logic we teach at undergraduate level, i.e. in most cases, first order classical deductive logic, also deals with the forms of the arguments which correspond to appropriate forms of inference, not with the actual arguments we use in particular cases. In the classroom, the more usual way of teaching logic is to recur to the use of formal languages and formal methods of decision to 'see' the form of an argument and to decide whether or not it is valid. Of course, we present examples to illustrate those forms, but the main goal of an actual course of logic is, in general, to define the meaning of

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 38–45, 2011.

the different connectives or logical symbols in a formal way, the definition of the notions of validity and logical consequence, and to master some formal methods of proof. In this sense, we present logic as stated in the Stanford Encyclopedia of Philosophy:

> Logic is often seen to be topic neutral. It applies no matter what one is thinking or reasoning about. And this neutrality, or complete generality of logic, together with its normativity, is often put as "logic is about how we ought to think if we are to think at all" or "logic is the science of the laws that we ought to follow in our thinking no matter what we think about". (Hofweber, T. 2010, p. 4)

As a consequence, after having presented the original relationship between logic and argumentation, in many of the undergraduate courses in logic (at least in Spain), we don't deal with the material content of actual cases of argumentations uttered in a definite context, if not to illustrate a formal problem. We adopt the above definition and the actual state of development of logic as a good starting point to teach it as a theory of reasoning, and moreover, the limitation of the time of the course don't give us a different choice, at least if we want our students to master some of the formalities of symbolic logic.

In logic courses is usual also to stress the possibility of using logic as a tool to a better understanding of other core subjects in the curriculum of Philosophy as, for example, Philosophy of Language or Philosophy of Science, but those subjects come later in the curriculum, and the questionnaires of assessment of the students about the teaching clearly show that they don't see the actual connection with other subjects in the curriculum and with possible applications of logic in their intended future jobs.

2 Logic and Undergraduate Studies

Logic courses are usually taught during the first years of Philosophy studies. For an average student of Philosophy the concepts behind the mathematical notation and the abstract methods used in logic get obscured and, in most of the cases, lost by the difficulty they have to grasp the formalities of the languages and the methods. Moreover, when coming for the first time to the University, many of the students of Philosophy don't expect to meet mathematical and symbolical notation and concepts again and, in general, they reject at a first glance everything that has to do with symbols, mainly because they are not very good at it. This difficulty to master the pattern of reasoning in an abstract way makes somehow impossible for them to equate it with actual reasoning forms in natural language. In consequence, for the majority of them, the teaching of logic is of no real use to improve their reasoning. In general, they only learn to read simple formulas, to translate simple sentences from natural language and to do some very easy deductions in a natural method of decision, but, after passing the compulsory test, they usually forget all formalism rapidly. The current educational reform in which we are now involved in Europe hasn't changed too much the

actual contents of the courses of logic, in the different universities of Spain. Only in a couple of Universities other approaches to the subject are taken.

On the contrary, logic is not usually taught in Mathematical studies or in many schools of Computer Sciences in a specific way. The students learn to use logical methods in an intuitive way or integrated with other subjects (i.e., Foundations of Mathematics, Axiomatic Set Theory, Discrete Mathematics, etc.). Students on these fields are supposed to know the rules of deductive reasoning (and maybe of other types of reasoning too) to use them in practice, but they almost never have a course in formal logic as part of their training.

On other countries that we know the study of logic is different, at least at undergraduate level. For example, the American Philosophical Association (2007) on its website has a statement on the subject of teaching logic as part of the curriculum in undergraduate studies in Philosophy. This is what they have to say about the issue:

> Logic may be studied in a number of different ways. No one of them is essential to a sound major in philosophy, but a course of some sort dealing with the principles of logic and logical reasoning is highly desirable. One version of such a course is an introduction to symbolic logic, which may be supplemented by more advanced courses. Another is an "informal logic" or "critical thinking" course, emphasizing the study of forms of sound reasoning, inference and argument. For students who choose philosophy as a good "liberal education" major and do not intend to pursue its study beyond the undergraduate level, the latter may be sufficient. Those who intend to take advanced courses dealing with contemporary treatments of philosophical issues in the central areas of the discipline, however, will find familiarity with symbolic logic very helpful; and it is indispensable for those who contemplate going on to graduate study in philosophy. (American Philosophical Association 2007).

From the point of view of the American Philosophical Association a course on logic is highly desirable but not essential at the undergraduate level. As a consequence, many colleges have only compulsory critical thinking courses at undergraduate level and symbolic logic as optional. Those courses, in general, are not taught for real experts on the field of argumentation or in logic, maybe because the current status on the field of argumentation theory, still developing and sometimes equated with rhetoric, has low status within philosophy and philosophers are generally not interested in methodological issues.

When teaching to think critically[1] the emphasis is put in actual examples of argumentation and the students have to face actual argumentative discussions and learn to defend their proposals in an active way, and this point of view is

[1] I am using "critical thinking" in a broad sense as to include all the different schools which aim is improving the actual argumentative skills of the students. Maybe, the critical thinking movement is not the better suited to be taken as a representative of the option we would choose and, so, we should, instead, talk better about arguing in general, or about reasoning in an informal way.

somehow different at that taken in logic courses. We want to stress the fact that we aren't denying the importance of logic in Philosophy and other disciplines, and we think, that a course on symbolic logic is indispensable for those who have in mind to pursue graduate studies or simply to introduce themselves in the field of Analytic Philosophy.

The main goal when teaching reasoning in its "informal" form in America is clear if we take a look at the orders that introduced this subject, several years ago, in the curriculum of the different majors at college level. To put an example, California State University Executive Order (1980) supposed a significant impetus in the development of the informal logic and critical thinking movements. This order required that post secondary education included formal instruction in critical thinking:

> Instruction in critical thinking is to be designed to achieve an understanding of the relationship of language to logic, which should lead to the ability to analyze, criticize, and advocate ideas, to reason inductively and deductively and to reach factual or judgmental conclusions based on sound inferences drawn from unambiguous statements of knowledge or belief. (Dumke 1980, Executive Order 338).

These educational interests on improving the thinking of the students and the form to express it have been the object of hundred of books and articles in English that show the different ways or possibilities to do so. They are currently used in the majority of the English spoken Universities, in Canada, the United States, the United Kingdom, and in a constantly growing number of other countries. They are based on actual examples of argumentative texts or oral discussions taken from different contexts. In the majority of the cases, the use of symbolic expressions is reduced to a minimum, if any. There are very few books written in Spanish on the subject, they have mainly a linguistic non-logical orientation, and, in general, they are not used in the classrooms in Philosophy.

Looking at the above comments, we think that it is clear that, when talking about teaching logic at the beginning of Philosophy studies, there can be different proposals with different practical goals and, in our opinion, we should differentiate them if we want our students to improve and profit from our teaching.

3 Logic and Argumentation

First of all, it is important for us to highlight the fact that we don't consider reasoning and argumentation as synonyms. It is clear that, when we argue we use reasoning, but reasoning can appear also outside an argumentative setting. For example in mathematical papers, in the proof of theorems, there is only reasoning and no argumentation involved. For us, for argumentation to take place, first there has to be a debatable question at stake, say because somebody has doubts about it, or maybe because it has been overly contested. Afterwards, we will use reasoning (but not necessarily deductive reasoning) to defend or

to clarify the doubts presented by an opponent (not necessarily present) to our point of view, in an ongoing back and forth process that should be finished when the matter is settled by some normative standard. It can be inferred then, that we take a dialectical conception of argumentation that involves more than pure reasoning.

In general, we think that the paper of deduction in the reasoning we use in everyday argumentation is limited. Although many researchers in the field of argumentation try to reconstruct the ordinary argumentations as deductive, because it simplifies the task of assessing and evaluating the arguments, and it opens also a way to bring to light the implicit possible information that an argumentation leaves without saying, this position has been criticized from almost as many other scholars. In real life reasoning, we use inference to produce new information based on previously held beliefs or known facts and combine it with new information we usually grasp through communicative practices. The uncertainty involved in the interpretation of ordinary reasoning makes difficult to fulfill the demands of deductive reasoning and, even after a careful reconstruction of the argument, it is problematic to consider most of the ordinary reasoning as deductive. In ordinary argumentation, the recourse to inductive inferences and the use of heuristics, best explanations, analogies and other resources are necessary and frequent. Some arguments can be strong, others are weaker. Moreover, there may be different and competing arguments to defend a point of view or even we can consider, in some cases, the possibility to maintain temporally two opposite claims until having more information to decide, although, in practice, we may prefer one to another. The reconstruction of the reasoning done in practical argumentation as deductive is helpful to assess it, but in general does not correspond to what happens in the actual process of arguing. Validity or soundness and fallacy can be considered as the two extremes of what we consider a much more complex classification of argumentative practices.

Teaching symbolic logic as a model of argumentation, the students notice those facts. First, they lack the dialectical component of the argumentation that as language users they are used to, and then, they are presented the logical deductive reasoning as an ideal model they should adopt. Although we remark the fact that other logical models of reasoning exist, non classical logics or non-deductive models of reasoning are mainly mentioned in undergraduate programs.

Informal logic and some critical thinking courses employ not only logic but other ingredients combined with logic to try to find a more close approach to the actual pattern of reasoning in natural language. They use arguments worded in natural language and the students have to make strategic decisions about the selection of them, their order and relevance, the choice of the words to be accurate and precise, and the amount of information they want to make explicit. These choices don't depend only on the logical form of the argumentation but also on contextual elements in which we have to situate it. Traditionally it has been said that those other elements don't help to grasp the essence of the reasoning, but we think otherwise. On the one hand, it is true that logical rules can get obscured when more elements come to play but, on the other, to isolate arguments of

their context does not help either to explain the communicative side of the argumentative practice.

Different experiments on reasoning in the fields of cognitive psychology show that people reason poorly and fail at doing simple logical tasks and other kind of probabilistic reasoning, but, they also show that participants perform better if instead of abstract information, real sentences are used (Evans 2002). This has led to think of a possible reformulation of the mechanism we use for reasoning. Some researchers think that the function of reasoning is to enhance or correct the beliefs we have about the world and, in consequence, to help us in decision making, others think of reasoning as having also a communicative and social function (Eemeren and Grootendorst 2004, Mercier and Sperber, 2011) and show that people reason better if this task is fulfilled. In fact, the same experiments from cognitive psychology show a definite increment in good reasoning when the possibility of arguing among the participants is allowed.

Taking into account those facts, we think that in an introductory course of logic in Philosophy we should work different fields in a parallel way. First, to improve the reasoning of the students, we think, as Hintikka (1989) does, that we should teach logic as a kind of problem solving and creative thinking subject. A first ingredient of Hintikka's model corresponds to a series of interrogative steps to gather all the information around the problem presented. After that, students has to organize this information and the inferential stage proceeds. Adapting this program, we propose to present the students different logical problems, in an informal way, so that they have to gather the information they need and construct their own thinking strategies to work out the solution. Differently as what Hintikka proposes, we think that semantics tableaux are not an ideal method for the beginner and we prefer, for example, the use of different logical puzzles expressed in natural language, as in Manzano and Huertas (2004). Only after being solved in natural language, should we try to formalize them. Resolution of logical puzzles can be a good way to learn how to order and give priority to the tasks necessary to reach an objective, to work out the information given and to look for relationships and consequences from this information, to examine implicit or background contents, to text partial conclusions in order to reconstruct prior beliefs and, also, to make new strategies to work forward toward the resolution of the problem.

There is another line to develop if we want to improve not only the logical reasoning of the students, but also their critical skills. For Philosophy students it is important to acquire actual skills to argue in natural settings, first in oral settings and then in writing because they are going to need those skills in their development as students, and in their jobs when graduating. In oral discussions the students are, in general, quite skillful dealing with the dialectical process and the communicative techniques of arguing in an automatic way, but they don't make a conscious use of those procedures and moreover, the argumentations they produce are not usually very good.

> In every day issues we are generally highly skillful in challenging, counterchallenging, justifying or agreeing during conversation but the

> arguments we hold are mediocre according to analytical criteria...We know "to move forward" but we don't know very well "where to go"... (Schwarz and Gassner 2003, p. 232).

In oral discussions, we should emphasize the search of counterexamples to hastily done generalizations, the lack of coherence of the arguments presented, or the lack of relevance of them with respect to the claim to defend. After having debated a topic, the students should be made to write down their argumentations in order to commit to the points defended and to integrate the whole argumentation in a text. As stated in Carrascal and Mori (2011) oral and written argumentation are different processes; in oral argumentation the statements are generally shorter; we have an immediate feedback from the opponent that helps us to find the path to retrieve the necessary information; it is almost always possible to give some kind of answer to the objections the opponent raises by weakening or negotiating our point to accommodate the challenges, to facilitate the communication and to build consensus; and our performance has to take into account both, the objections that make shift the burden of the proof back and forth between the two parts, and the conversational turns of it; In written argumentation, the opponent is not present and the abstraction to represent her makes more difficult the articulation of the arguments. The physical absence of the audience is one of the most salient characteristics of written argumentations (Bereiter and Scardamalia 1987, Kellogg 1994); For example, we need to use more stylistic resources to make our point, because we have no access to non-verbal communication. Moreover, the ordering and linearization of the text has to make sense, because there is no chance to improve it with the immediate feed-back of the opponent.

These different factors interact also with other elements of the social context, as, for instance, the status of the participants and their interest in maintaining the quality of the relationship between the interlocutors. In many everyday discussions the logical, linguistic and social components are of importance and, so, to improve adequately our argumentative skills we can't look only to the cognitive side of the activity.

After those steps, we could finally examine the written product to extract the arguments used, and assess them from a logical point of view, to see whether they are acceptable, sufficient and relevant to maintain the claim and, also, to assess whether it is a reasonable (non necessarily sound) argumentation relative to the context and the audience towards which it was directed.

Software tools designated to facilitate the analysis of argumentations and the production of good reasoning in learning environments can be of help to reinforce the process of writing and the posterior analysis of the arguments presented to defend a claim, because, in general, they provide a (reduced) list of sources for arguments to support a claim and also a way of schematizing and ordering the arguments with respect to the claim. Nevertheless, they are few software tools in language other than English and what we have in Spanish are in general linguistically oriented instead of trying to combine it with the logical and the social component of the argumentation.

4 Conclusion

In our opinion, in an introductory course in logic for students of Philosophy the symbolic component should be kept at a minimum. We should try to improve the reasoning of the students by giving them different problem solving tasks to enhance their thinking and their initiation to the inquiry. Furthermore, argumentation includes reasoning but also other components related with communication and social interaction, so, to develop those argumentative skills we need the students to practice arguing in oral discussions and by writing texts. Courses on symbolic logic should be taught latter in the curriculum.

Acknowledgments. This work was supported by the Research Project FFI2010-20118 of the Ministry of Science and Innovation of the Spanish Government.

References

American Philosophical Association: American Philosophical Association Statement on the Philosophy Major. Proceedings and Addresses of the APA 80(5), 76–89 (2007)

Bereiter, C., Scardamalia, M.: The psychology of written composition. LEA, Hillsdale (1987)

Carrascal, B., Mori, M.: Argumentation schemes in the process of arguing. In: Proceedings of the 7th International Conference on Argumentation, Sic Sat, Amsterdam (2011)

Dumke, G.: Chancellor's Executive Order 338. California State University, Chancellor's Office, Long Beach (1980)

van Eemeren, F.H., Grootendorst, R.: A systematic theory of argumentation. The pragma-dialected approach. Cambridge University Press, Cambridge (2004)

Evans, J.S.B.T.: Logic and Human Reasoning: An Assessment of the Deduction Paradigm. Psychological Bulletin 128(6), 978–996 (2002)

Fisher, A.: Critical Thinking: An Introduction. Cambridge Univ. Press, Cambridge (2001)

Groarke, L.: Informal Logic. The Stanford Encyclopedia of Philosophy, Zalta, E.N. (ed.), http://plato.stanford.edu/archives/fall2008/entries/logic-informal/ (Fall 2008 Edition)

Groarke, L., Tindale, C.: Good Reasoning Matters!, 3rd edn. Oxford University Press, Toronto (2004)

Hintikka, J.: The role of logic in argumentation. Argumentation 15(3), 347–362 (1989)

Hofweber, T.: Logic and Ontology. The Stanford Encyclopedia of Philosophy. Zalta, E.N. (ed.), http://plato.stanford.edu/archives/fall2010/entries/logic-ontology/ (Fall 2010 Edition)

Kellogg, R.T.: The Psychology of Writing. Oxford University Press, New York (1994)

Manzano, M., Huertas, M.A.: Lógica para principiantes. Alianza Univ., Madrid (2004)

Mercier, H., Sperber, D.: Why do humans reason? Arguments for an argumentative theory. Behavioral and Brain Sciences (to appear, 2011)

Schwarz, B., Gassner, G.A.: The blind and the paralytic: fostering argumentation in social and scientific issues. In: Andriessen, J., Baker, M., Suthers, D. (eds.) Arguing to Learn: Confronting Congnitions in Computer-Supported Collaborative Learning Environments, pp. 227–260. Kluwer Academic, Dordrecht (2003)

ORGANON:
Learning Management System for Basic Logic Courses

Ludmila Dostalova and Jaroslav Lang

Dept. of Philosophy, Faculty of Philosophy and Arts, University of West Bohemia in Pilsen
Sedlackova 19, 306 14 Plzen, Czech Republic
ldostal@kfi.zcu.cz

Abstract. We would like to introduce a new e-learning system ORGANON designed for teaching logic. In comparison to regular e-learning systems it is able to handle logic exercises including correcting, grading and providing elementary feedback. In comparison to didactic logic software it is able to handle the administration of teaching, namely assigning individual homework, collecting delivered homework, grading, storing achieved results and providing data-mining. Hence the web tutor fulfills two requirements: it helps students during their study to practice exercises on their own and it helps teachers to diminish the burden of teaching administration and trivial consultations.

Keywords: teaching logic; e-learning; logic didactic software; computer aided teaching; computational exercises; constructional exercises.

1 Motivation of the Project

Basic logic courses are usually designed for large number of students (several hundreds are not exceptional), who have no experience with handling formalism. Successful completion of such a basic logic course usually presupposes a substantial portion of individual work because it rests on solving logic exercises in order to obtain skills how to work with formulas. Though solving exercises is the best way to learn modern logic and make students become familiar with logic notation, relevant literature is rather missing. Exercise-books with sample solutions and a key for the correct answer are very rare if not missing at all. Therefore, teachers not only have to create large numbers of exercises each year themselves, but they also have to correct and grade these exercises and finally discuss the results with students. Because of the large-scale character of basic logic courses such a task exceeds teachers' capacities. But, problems that students usually deal with and want help with are of such a character that they can be easily solved and answered in an automatic way. So the appropriate computer-based assistance may solve the problem. Therefore the learning management system ORGANON was developed to:

1. Increase the efficiency of student's homework by
   - Providing permanent control during students' practicing exercises
   - Answering students' questions immediately as they arise

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 46–53, 2011.

2. Diminish the teachers' burden by
   - Reducing the amount of consultations
   - Administering and grading students' homework.

Originally, we looked for some existing software, which would be able to supply the basic logic course. The amount of didactic software for introductory logic courses is quite large. Most elaborated of them are probably CSLI's packages (Tarski's worlds, Hyperproof, Boole, Fitch etc.). Such software usually performs sophisticated didactics. However, it ignores the administration of teaching, like correcting, grading and namely storing achieved results.[1] Moreover, it usually provides only fixed structure of exercises.[2] On the other hand, systems designed to fulfill the administrative requirements (e.g. e-learning systems like MOODLE) are able to handle the administration of teaching, but they do not reflect special needs of logic as a discipline. They were designed for those fields of study that have an encyclopedic character, which is inadequate for logic exercises since the only way of practicing as well as examining is *via* test question while the logical exercises are mostly computational or constructional. Moreover, these systems usually do not make it possible to write logical symbols or draw diagrams. That is why we eventually decided to create a new e-learning system that will accommodate both these functions: it will be able to handle logic exercises (like specialized logical software) as well as to manage the administration of teaching (in a similar way as e-learning systems do).

2 Description of the LMS ORGANON

The e-learning system ORGANON was designed for basic logic courses. Hence, it satisfies the special needs of the discipline. Firstly, it is able to accommodate logic symbolism as well as the other sorts of visualizations like tables or diagrams. Secondly, it reflects the fact that logic exercises are not of a memorizing knowledge character (and thus cannot be practiced only by a test question); rather, they are of a computational, inferential and constructional character, which means that not only the solution but the whole process of solving must be taken into account. Therefore ORGANON accommodates not only practicing *via* test-question but also practicing *via* interactive transformations or constructions. The two main functions are: to help students in their practicing and to help teachers in grading and especially with

[1] The CSLI package includes also GradeGrinder to provide feedback when practicing as well as grading and send the results to the teachers. However it is unable to evaluate students' work with respect to the specific course and provide the statistics for the teacher. Further it is impossible to assign to students new exercises automatically according to their previous results and to settle drills.

[2] The CSLI package has a various sort of exercises but these are given in the fixed structure, which is the same for all the students. Hence the practicing is limited and not individualized. Concerning graded homework, the fixed and same structure of exercises does not prevent plagiarism. Finally, if new exercises are created, the automatic feedback (GradeGrinder) cannot be used.

individual consultations required by their students. Therefore the system consists of three segments:

1. The *Database of logic exercises*, which
   - Enables to create the structure of exercises (tests) just for the needs of the course
   - Provides enough exercises for individual practicing as well as testing
   - Generates various, though fully comparable, test variants
2. The *Grading Module* to administer students' homework, which
   - Generates individual exercises for graded homework
   - Facilitates electronic elaboration as well as delivery of the homework
   - Manages to correct and grade the homework
   - Stores the achieved results including a record of the exercise, student's solution and results of automated correcting and grading
   - Gathers statistical data to provide feedback (data-mining)
3. The *Practicing Module* to help students in practicing the exercises, which serves in three different modes:
   - Shows (step by step) the *sample solution* of the exercise and provides relevant explications
   - *Consults* with students about their own process of solving the exercise by
     - Controlling the equivalency of transformations when the step is finished (automatic)
     - Alerting when mistake (i.e. inequivalency), appears (automatic) as well as showing the mistake (if required by student)
     - Giving hints for the next step or performing that step directly including relevant explanation (if required by student)
   - *Corrects* the transformation when finished (not during the transformation) and comments on it in the same way as for graded homework

2.1 Database of Exercises

The learning management system ORGANON is built upon the database of logical exercises. The database was designed to meet two requirements. Firstly, it must be precisely sorted to fulfill the didactic needs of the course and to allow the automatic handling. Secondly, it must be considerably rich to provide individual homework or test variants for hundred students at a course.

2.1.1 The Structure of the Database

The *Database of exercises* is structured into CATEGORIES and TYPES. The typology was created according to the didactic needs of the courses. Such a precise typology also constituted the starting point for creating algorithms to solve the exercises and grade them by computer in an automatic way.

The database covers regular TOPICS of the basic logic courses – Aristotelian logic, Propositional logic, First-order logic, Validity of Arguments, Deduction and

Formalization. Within each topic, students must learn several formal methods.[3].But each method can be practiced as well as tested in various ways.[4] Hence the CATEGORY is defined within the topic by two aspects – the method to be practiced and the style of practicing. For example the truth-table method can be practiced as a constructive exercise, a Y/N-question, a multiple-choice question and so on. Each of these variants forms an individual category. Hence all the exercises within the category have identical instruction (setting of task for solution).[5]

Exercises within the same category are then structured into a hierarchy of TYPES according to their difficulty. Respectively, a type is formed by exercises having the same method of solution – the same procedures (laws of logic) must be inevitably used during the process of solution.[6] The type is always defined by such a list of laws of logic without whose knowledge the student CAN NOT solve the exercise.

The twofold sorting – into CATEGPRIES according to the topic and method of practicing and into TYPES according to the solution-method and its difficulty – allows teachers to create their own structure of exercises for their course, which would fulfill the needs of their course and respect the needs of their students. It is also possible to change the structure of exercises and homework of the course (in amount as well as in kind) arbitrarily each term. Furthermore, the same database can be used for courses of different topic structure and at different levels at the same time.

2.1.2 The Extent of the Database

The database was developed to provide enough exercises to generate individual homework as well as comparable test variants for hundreds of students in a course. Therefore the database is not filled up by concrete exercises but by PATTERNS. An exercise is automatically generated from a pattern through the principle of substitution for pattern variables. This substitution can be realized by the substitution of literals into the formulas as well as by the substitution of words into the sentence pattern. Hence the database is significantly rich. One-hundred-ninety-two different exercises with identical solution method might be generated from a single formula pattern by substitution of literals.[7] Currently, each type is formed by four patterns, which have comparable (if not identical) method of solution. That is 768 exercises of the same method of solution and difficulty in one type. A typical category consists of 15 - 20 types but can be easily extended. In the case of exercises formulated in natural

---

[3] E.g. the topic of propositional logic contains truth-table method, transformations and so on.

[4] E.g. a constructive exercise, a Y/N-question, a multiple-choice question, and so on.

[5] E.g. the truth-table method involves categories defined by instructions: Create a truth-table to given formula? (constructional exercise); Which of the formulas a-e corresponds to given truth-table? (multiple-choice question) and so on.

[6] E.g. De Morgan laws, distributivity, definitions of propositional connectives etc.

[7] Creating exercises from patterns just by the substitution of positive and negative literals might look to trivial. However, the ORGANON is designed for formalism-phobic students. Hence they are not able to see through the trick. And if yes, they do not need ORGANON any more. Anyway the principle of generating exercises allows also more sophisticated ways of substitution if required for advanced students.

language, up to 50 sentence variants might be generated from one sentence pattern (depending on the creativity of teacher), which is still a remarkably large variety.

The considerable extent of the database guarantees that students work individually and prevents plagiarism, while the structure of the database guarantees that all the students solve exercises of the same complexity.

2.2 The Grading Module of ORGANON

The *Grading Module* of the ORGANON was created to manage the whole administration of individual student's homework as well as of exam tests including generating exercises, grading delivered homework and storing the achieved results.

2.2.1 The Teacher

Teachers are allowed to define the structure of the homework for their course specifying the number of assignments to be graded, the number of exercises in each assignment, and the types of exercises from the database to be included. They have access to the whole database and may limit the structure and extent of it as they choose to accomplish the goals for the course. Moreover, they are allowed to see not only the results and grades of their students, but also the records of exercises assigned to them, the set of procedures used in their solution and (of course) the computer correction and grading of the homework in detail. That means, that in case of doubts it is always possible to control the automatic grading in person. Finally, the application gathers and provides statistical data such as average success of students, relative difficulty of exercises and so on. Such information provides relevant feedback indicating inconsistencies in the structure of the exercises and the necessity of change in the course of study.

2.2.2 The Student

Students are allowed to access only their personal accounts, which form the environment for their practicing as well as their examination. They can see exercises assigned to their homework, process of solving the exercise they have already done and saved, correction of exercises already sent and (of course) their grades, including the information about their relative successfulness in completing the course.

Exercises for the homework of a concrete student are chosen randomly by the computer from the exercises of the type assigned by the teacher. Thanks to the extent of the database, each student receives individual assignment of homework because the probability of assigning two students the same exercise is 1/768 (0.0013), and one homework is usually formed by 5 or 7 exercises. Hence, individual homework is guaranteed and plagiarism avoided. Since the exercises of each individual homework assignments are generated from the same types (exercises having the same solving procedure), achieved results are directly comparable.

Students work on their homework solutions within the environment of the ORGANON and submit it when finished. They may enter the homework several times and save the unfinished work to change or complete it later. The homework is corrected and graded by the application automatically after it has been sent. Therefore

students may immediately see their results – not only the grades, but especially the comment on mistakes they have made.

2.3 The Practicing Module of ORGANON

The *Practicing Module*, the interactive didactic section of the ORGANON, is intended to help students practice the exercises before attempting the homework to be graded. Meanwhile it can support teachers by decreasing the number of consultations with students about trivial matters.

2.3.1 The Teacher

Teachers define the structure of students' work by choosing the concrete categories (methods to be practiced and the way of practicing) and by choosing the appropriate order of practicing (the subsequence of types). Since the types are defined by their difficulty, this means that students will proceed from easier exercises towards more difficult ones. Thanks to the precise definition of the type of an exercise *via* the list of laws of logic necessary for successful completion of the exercise, teachers are able to choose exactly those exercises, which are appropriate for the needs of their students based on their previous experience. Moreover, this structure enables automatic feedback and didactic hints.

2.3.2 The Student

Each category of exercises is structured into the hierarchy of types according to their difficulty so that students proceed from the easiest to more complicated ones. Practice exercises are generated according to the students' previous solutions. If a student is successful, an exercise from the more difficult type is chosen and vice versa. If a student makes repeatedly the same mistake, an exercise is chosen from the type, which practices the law of logic in whose application the student failed. In this manner the ORGANON adapts to the rate of learning of individual students. Students may use the *Practicing module* in three different ways:

- Firstly, they can just ask for an exemplary solution. The system then shows the procedure for solving the exercise step by step and provides relevant explanation. The extent of the explanation is directed by the students – they can ask for more details as well as for shortened solution. (Fig. 1)
- Secondly, students may try to solve the exercise on their own, while the system controls their progress automatically at the end of each step and alerts them in case a transformation is not equivalent. Then, students may try to find and correct their mistake independently or they may ask for that the mistake be shown and explained. They may also ask for tips for the next step or for a demonstration and explanation of that step.
- Finally, students may solve the exercise independently (without automatic control) and when finished, they may ask that it be corrected and graded in the same way as for homework to be graded.

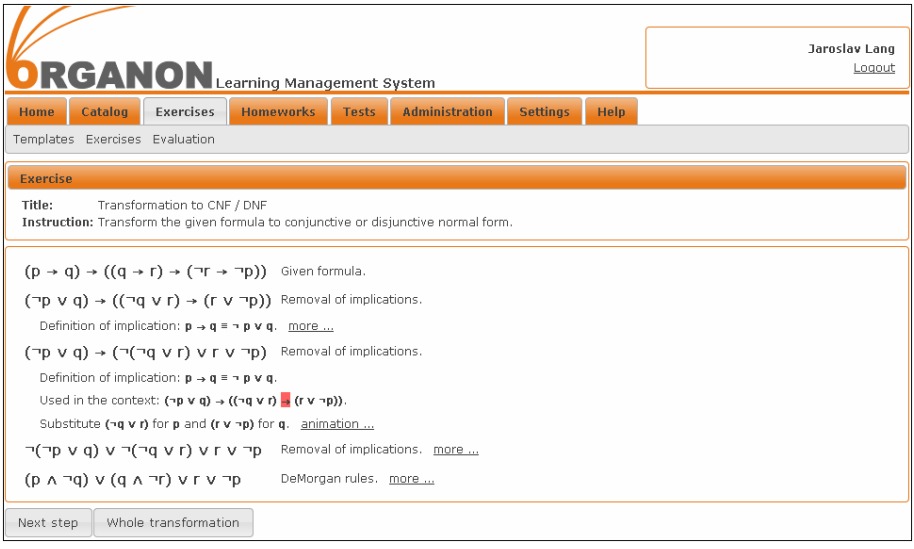


Fig. 1. The screenshot of the Practicing Module of ORGANON running in the first mode. The explanations appear step by step after clicking on the link "more..."

3 The Process of the Project

The web tutor ORGANON was first used in teaching in Fall 2006. Following experiences with the prototype a new version was introduced in Fall 2010. The database includes basic topics in modern formal logic, namely to practice syntactic and formal methods in propositional and first-order logic (recognizing well-formed formulas, truth-tables, transformations of formulas, models of formulas, counter-examples, equivalency of statements and negating statements). The database is gradually further extended to include also deduction and application of logic in natural language (formalization of natural language sentences, negation of sentences, equivalency of sentences, validity of informal arguments).

Considerable attention was given to user interface. Firstly, it was necessary to accommodate logic notation respecting variability of its conventions as well as the problem of compatibility between computers. Moreover, special interfaces were necessary to be created for various sorts of exercise like truth-tables, diagram, models and counterexample constructions.

Nevertheless, the ORGANON has no unusual software or hardware claims and only requires browser. It is user-friendly and does not require any special abilities to be operated; all that is presupposed is that the user can handle and click a mouse. Thus it can be easily introduced to other universities and organizations. Currently we prepare versions for other languages than just Czech in order to provide an international user interface.

Finally, the ORGANON conforms to *disabled students*, in particular the blind ones. Since logic is closely connected to visualization of a (logical) form (a structure) of statements and arguments, and since most of the didactic mnemonic aids are

visual-based, we had to face the problem of finding an alternative way to communicate logic and logic formalism to these students. It appears that an audio response might be a plausible solution and some positive results in this respect have been achieved.

4 Conclusion

The ORGANON system has been primarily designed for only one purpose – to make students familiar with logical formalism and diagrams and thus get rid of a useless anxiety or even formalism-phobia. The idea is that students only need to practice to obtain these skills. The precise structure of the database of exercises allows teachers to choose appropriate exercises to support their students and enables students develop necessary skills (or drills) to handle special formal tasks.

This is the first step on the way to become familiar with logic and logical thinking. Moreover, practical experience with the concepts of logic makes these concepts more familiar to students; thus logical concepts are no more just abstract inconceivable entities for them. Conceptualization and formalization helps students to effectively organize and manage their knowledge.

Of course, an e-learning system cannot fully substitute a human teacher. Yet it can be very useful by reducing the necessity to practice mechanical manipulations with formulas in lectures and seminars by providing an individual 24-hour service to each student. Thus it opens the space for the discussions on really interesting and inspiring topics during the lessons.

Acknowledgement

The project is supported by the European Social Fund and the goverment of the Czech Republic (grant ESF OPVK No. CZ1.07/2.2.00/07.0217).

The project used results of the previous ones held under the auspices of the Fund for the Development of Higher Education, Ministry of Education, Youth and Sports of the Czech Republic (grant FRVS no. 566/2006/F5/d and 65/2008/F5/d).

References

1. Barwise, J., Etchemendy, J.: Language, Proof and Logic. CSLI Publications, Stanford (1999)
2. Dostálová, L., Lang, J.: ORGANON - The Web Tutor for Basic Logic Courses. Logic Journal IGPL (2007), doi:10.1093/jigpal/jzm021
3. Dostálová, L., et al.: Odkud a jak brát stále nové příklady. ZCU, Plzen (2009)
4. LMS ORGANON Project, `http://www.organon.kfi.zcu.cz`

Variables in Mathematics Education

Susanna S. Epp

DePaul University,
Department of Mathematical Sciences, Chicago, IL 60614, USA
http://www.springer.com/lncs

Abstract. This paper suggests that consistently referring to variables as placeholders is an effective countermeasure for addressing a number of the difficulties students' encounter in learning mathematics. The suggestion is supported by examples discussing ways in which variables are used to express unknown quantities, define functions and express other universal statements, and serve as generic elements in mathematical discourse. In addition, making greater use of the term "dummy variable" and phrasing statements both with and without variables may help students avoid mistakes that result from misinterpreting the scope of a bound variable.

Keywords: variable, bound variable, mathematics education, placeholder.

1 Introduction

Variables are of critical importance in mathematics. For instance, Felix Klein wrote in 1908 that "one may well declare that real mathematics begins with operations with letters," [3] and Alfred Tarski wrote in 1941 that "the invention of variables constitutes a turning point in the history of mathematics." [5] In 1911, A. N. Whitehead expressly linked the concepts of variables and quantification to their expressions in informal English when he wrote: "The ideas of 'any' and 'some' are introduced to algebra by the use of letters... it was not till within the last few years that it has been realized how fundamental *any* and *some* are to the very nature of mathematics." [6] There is a question, however, about how to describe the use of variables in mathematics instruction and even what word to use for them.

Logicians seem generally to agree that variables are best understood as placeholders. For example, Frege wrote in 1893, "The letter 'x' serves only to hold places open for a numeral that is to complete the expression... This holding-open is to be understood as follows: all places at which '?' stands must be filled always by the same sign, never by different ones," [2] and Quine stated in 1950, "The variables remain mere pronouns, for cross-reference; just as 'x' in its recurrences can usually be rendered 'it' in verbal translations, so the distinctive variables 'x','y', 'z', etc., correspond to the distinctive pronouns 'former' and 'latter', or 'first', 'second', and 'third', etc." [4]

The thesis of this article is to suggest that the logicians' view of variables is best for the teaching of mathematics – that, right from the beginning and

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 54–61, 2011.

regardless of whether they are called "letters," "literals," "literal symbols," or "variables," they be described as placeholders, and that, to be seen as meaningful, they be presented in full sentences, especially ones with quantification. This thesis will be supported by providing a sampling of the different uses of variables and analyzing the reasons for some of the difficulties students encounter with them. Two that arise repeatedly are (1) thinking of variables as exotic mathematical objects that do not have a clear connection with our everyday universe, and (2) regarding variables as having an independent existence even though they have been introduced as bound by a quantifier.

2 Mathematical Uses of Variables

2.1 Variables Used to Express Unknown Quantities

In the early grades, students are sometimes given problems like the following:

Find a number to place in the box so that $3 + \square = 10$.

Later, however, when algebra is introduced, the empty-box notation is typically abandoned and the focus shifts to learning rules for manipulating equations in order to get a variable, typically x, on one side and a number on the other. With the resulting emphasis on mechanical procedures, the meaning of "Solve the equation for x" may be obscured, with students coming to view x as a mysterious object with no relation to the world as they know it. Pointing out that x just holds the place for the unknown quantity - perhaps even making occasional use of the empty-box notation even after variables have been introduced - can counteract students' sense that the meaning of x is beyond their understanding.

To solve an equation for x simply means to find all numbers (if any) that can be substituted in place of x so that the left-hand side of the equation will be equal to the right-hand side. In my work with high school mathematics teachers, I have found that a surprising number are unfamiliar with this way of thinking and have never thought of asking their students to test the truth of an equation for a particular value of the variable by substituting the value into the left-hand side and into the right-hand side to see if the results are equal.

By holding the place for the unknown quantity in an equation such as $\sqrt{4 - 3x} = x$, the variable x enables us to work with it in the same way that we would work with a number whose value we know, and this is what enables us to deduce what its value or values might be. In 1972, the mathematician Jean Dieudonné characterized this approach by writing that when we solve an equation, we operate with "the unknown (or unknowns) as if it were a known quantity... A modern mathematician is so used to this kind of reasoning that his boldness is now barely perceptible to him." [1]

2.2 Variables Used in Functional Relationships

Understanding the use of variables in the definition of functions is critically important for students hoping to carry their study of mathematics to an advanced

level. In casual conversation, we might say that as we drive along a route, our distance varies constantly with the time we have traveled. So if we let d represent distance and t represent time, it may seem natural to describe the relationship between t and d by saying that for each change in t there is a corresponding change in d. This language has led many to think of variables such as t and d as objects with the capacity to change. Indeed, the word variable itself suggests such a description.

Addressing this issue, however, Tarski wrote: "As opposed to the constants, the variables do not possess any meaning by themselves... The 'variable number' x could not possibly have any specified property... the properties of such a number would change from case to case... entities of such a kind we do not find in our world at all; their existence would contradict the fundamental laws of thought." [5] Quine expressed a similar caution: "Care must be taken, however, to divorce this traditional word of mathematics [variable] from its archaic connotations. The variable is not best thought of as somehow varying through time, and causing the sentence in which it occurs to vary with it." [4]

We are quick to correct students who write "let a be apples and p be pears," telling them that they should say "let a be the number of apples and p be the number of pears." Similarly, t does not actually represent time but holds a place for substituting the number of hours we have been driving, and d does not actually represent distance but holds a place for substituting the corresponding number of miles traveled during that time. *Thus it is not the t or the d that changes; it is the values (number of hours or number of miles) that may be put in their places.* However, this is a distinction that mathematics teachers rarely emphasize to their students. In fact, mathematicians frequently make statements such as, "As x gets closer and closer to 0, $1 - x$ gets closer and closer to 1." This way of describing a variable that represents a numerical quantity may contribute to students' common belief that the number 0.99999... "gets closer and closer to 1 but it never reaches 1."

Even more than in the other areas of mathematics they encounter, students must learn to translate the words we use when we describe a function into language that is meaningful to them. For example, we might refer to "the function $y = 2x + 1$." Taken by itself, however, "$y = 2x + 1$" is meaningless. It is simply a predicate, or open sentence, that only achieves meaning when particular numbers are substituted in place of the variables or when it is part of a longer sentence that includes words such as "for all" or "there exists."

Students need to learn that when we write "the function $y = 2x+1$," we mean "the relationship or mapping defined by corresponding to any given real number the real number obtained by multiplying the given number by 2 and adding 1 to the result." We think of x as holding the place for the number that we start with and y as holding the place for the number that we end up with, and we call x the "independent variable" because we are free to start with any real number whatsoever and y the "dependent variable" because its value depends on the value we start with. Imagining a process of placing successive values into the independent variable and computing the corresponding values to place into the

dependent variable can give students a feeling for the dynamism of a functional relationship. However, we need to alert students to the fact that the specific letters used to hold the places for the variables have no meaning in themselves. For example, the given function could just as well be described as "$v = 2u + 1$" or "$q = 2p + 1$," or as "$x \to 2x + 1$" or "$u \to 2u + 1$."

Another way to describe this function is to call it "the function $f(x) = 2x+1$" or, more precisely, "the function f defined by $f(x) = 2x + 1$ for all real numbers x." An advantage of the latter notation is that it leads us to think of the function as an object to which we are currently giving the name f. This notation also makes it natural for us to define "the value of the function f at x" as the number that f associates to the number that is put in place of x. Using the notation $f(x)$ to represent both the function and the value of the function at x, while convenient for certain calculus computations, can be confusing to students.

A variation of the preceding notation defines the function by writing $f(\square) = 2 \cdot \square + 1$, pointing out that for any real number one might put into the box, the value of the function is twice that number plus 1. The empty box representation is especially helpful for work with composite functions. Students asked to find, say, $f(g(x))$ often become confused when both f and g have been defined by formulas that use x as the independent variable. When the functions have been defined using empty boxes the relationships are clearer. For instance, in a calculus class students find it easier to learn to compute $f(x + h)$ if they have previously been shown the definition of f using empty boxes.

2.3 Variables Used to Express Universal Statements

Terms like "for all" and "for some" are called quantifiers because "all" and "some" indicate quantity. In a statement starting "For all x" or "For some x," the "scope of the quantifier" indicates how far into the statement the role played by the variable stays the same, and the variable x is said to be "bound" by the quantifier.

Most mathematical definitions, axioms, and theorems are examples of universal statements, i.e., statements that can be written so as to start with the words "for all." For example, the distributive property for real numbers states that for all real numbers a, b, and c, $ab + ac = a(b + c)$. The variables a, b, and c are bound by the quantifier "for all," and they are placeholders in the sense that no matter what numbers are substituted in their place, the two sides of the equation will be equal. Thus the symbols used to name them are unimportant as long as they are consistent with the original.

In mathematics classes it is common to abbreviate the distributive property (and similar statements) by saying that a certain step of a solution is justified "because $ab + ac = a(b + c)$." However, this usage can lead students to invest a, b, and c with meaning they do not actually have. For instance, some students become confused when asked to apply the distributive property to $cb + ca$ because the a, b, and c are the same symbols used in the statement of the property, and students think of them as continuing to have the same meaning as in the

statement, without realizing that the scope of the quantifier extends only to the statement's end.

A different problem arises when the omission of the quantifiers is justified by describing a, b, and c as "general numbers" because this suggests that there is a category of number that lies beyond the ordinary numbers with which students are familiar. For those with a secure sense of the way a, b, and c function as placeholders, this terminology is not misleading, but students with a shakier sense of the meaning of variable may imagine a realm of mysterious mathematical objects whose existence makes them uneasy.

By contrast, if the distributive property is simply described as a template into which any real numbers (or expressions with real number values) may be placed to make a true statement, the mystery disappears and the way is prepared for leading students to an increasingly sophisticated ability to apply the property. Again empty boxes may be helpful. For example, the property can be stated as follows: No matter what real numbers we place in boxes $\square$, $\diamondsuit$, and $\triangle$,

$$\square \cdot \diamondsuit + \square \cdot \triangle = \square \cdot (\diamondsuit + \triangle)$$

Encouraging students to test the template by substituting a variety of different quantities in place of $\square$, $\diamondsuit$, and $\triangle$ provides a gentle introduction both to the logical principle of universal instantiation[1] and to the dynamic aspect of the universal quantifier, and substituting successively more complication expressions into the boxes can develop a sense for the power of the property:

$$\begin{aligned}
\mathbf{2}\cdot s + \mathbf{2}\cdot t &= \mathbf{2}\cdot (s+t)\\
2s + 6 = \mathbf{2}\cdot s + \mathbf{2}\cdot 3 &= \mathbf{2}\cdot (s+3)\\
2^{100} + 2^{99} = \mathbf{2^{99}}\cdot 2 + \mathbf{2^{99}}\cdot 1 &= \mathbf{2^{99}}\cdot (2+1 \quad [= 2^{99}\cdot 3]\\
\mathbf{(x^2-1)}\cdot x + \mathbf{(x^2-1)}\cdot (x-3) &= \mathbf{(x^2-1)}\cdot (x+(x-3)) \quad [= (x^2-1)(2x-3)]
\end{aligned}$$

2.4 Dummy Variables and Questions of Scope

Strictly speaking, the term dummy variable simply refers to any variable bound by a quantifier, but we most often use the term when discussing summations and integrals. For example, given a sequence of real numbers a_0, a_1, a_2, . . and a function f, we make a point of referring to k, i, x, and t as dummy variables to help students understand that

$$\sum_{k=1}^{10} a_k \;=\; \sum_{i=1}^{10} a_i \qquad \text{and} \qquad \int_1^2 f(x)dx \;=\; \int_1^2 f(t)dt.$$

In fact, it may be helpful to use the term dummy variable whenever we are especially concerned about problems that can result from thinking of variable names as "exceeding their bounds," that is, as having meaning outside the scope

[1] Universal instantiation: If a property is true for all elements of a set, then it is true for each individual element of the set.

determined by their quantification. For instance, it is common to state the definitions of even and odd integers as follows:

For an integer to be even means that it equals $2k$ for some integer k.
For an integer to be odd means that it equals $2k + 1$ for some integer k.

Following such an introduction, many students try to prove that the sum of any even integer and any odd integer is odd by starting their argument as follows:

Suppose m is any even integer and n is any odd integer. Then $m = 2k$ and $n = 2k + 1 \ldots$.

For the definitions of even and odd, however, the binding of each occurrence of k extends only to the end of the definition that contains it. In order to avoid the mistake shown in the example, students must come to understand that the symbol k is just a placeholder, with no independent existence of its own. One way to emphasize this fact is to call k a dummy variable. We can reinforce this characterization by writing each definition several times, using a different symbol for the variable each time. For example we could write the definition of even as:

For an integer to be even means that it equals $2a$ for some integer a.
For an integer to be even means that it equals $2r$ for some integer r.
For an integer to be even means that it equals $2m$ for some integer m.

It is also effective to give an alternative version of the definition that does not use a variable at all:

For an integer to be even means that it equals twice some integer.

In general, asking students to translate between formal statements that contain quantifiers and variables and equivalent informal statements without them is very helpful in developing their ability to work with mathematical ideas.

A few years ago I discovered that when I asked students to write how to read, say, the following expression out loud:

$$\{x \in U \mid x \in A \text{ or } x \in B\}.$$

the most common response was to omit the words "the set of all" and write only "x in U such that x is in A or x is in B." More recently, when teaching about equivalence relations, I learned that part of students' difficulty in interpreting such a set definition was a belief that the variable x had a life outside of the set brackets. When I defined the equivalence class of an element a for an equivalence relation R on a set A as

$$[a] = \{x \in A \mid x \, R \, a\},$$

a number of students had trouble applying the definition, and the question they asked was, "What happened to the x?" However, they were successful after I showed them that the definition could be rewritten with t in place of x and that it could be rephrased without the x as "The equivalence class of a is the set of all elements in A that are related to a."

Instructors who teach students with computer programming experience can draw analogies between the ways variables are used in programs and the ways

they are used in mathematics. For example, the name for a "local" variable in a subroutine can be used with a different meaning outside the subroutine, and within the subroutine it can be replaced by any other name as long as the replacement is carried out consistently. This is strikingly similar to the way a mathematical variable acts within a definition or theorem statement.

2.5 Variables Used as Generic Elements in Discussions

A variable is sometimes described as a mathematical "John Doe" in the sense that it is a particular object that shares all the characteristics of every other object of its type but has no additional properties. For example, if we were asked to prove that the square of any odd integer is odd, we might start by saying, "Suppose n is any odd integer." As long as we deduce properties of n^2 without making any assumptions about n other than those satisfied by every odd integer, each statement we make about it will apply equally well to all odd integers. In other words, we could replace n by any odd integer whatsoever, and the entire sequence of deductions about n would lead to a true conclusion. In that sense, n is a placeholder.

To be specific, consider that, by definition, for an integer to be odd means that it equals 2 times some integer plus 1. Because this definition applies to every odd integer, a proof might proceed as follows:

> *Proof*: Suppose n is any odd integer. By definition of odd, there is some integer m so that $n = 2m + 1$. It follows that
>
> $$n^2 = (2m + 1)^2 = 4m^2 + 4m + 1 = 2(2m^2 + 2m) + 1.$$
>
> But $2m^2 + 2m$ is an integer, and so n^2 is also equal to 2 times some integer plus 1. Hence n^2 is odd.

Dieudonné's use of the word "boldness" to describe the process of solving an equation by operating on the variable as if it were a known quantity applies equally well to the use of a variable as a generic element in a proof. For instance, by boldly giving the name n to an arbitrarily chosen, but representative, odd integer, we can investigate its properties as if we knew what it was. Then, after we have used the definition of odd to deduce that n equals two times some integer plus 1, we can boldly apply the logical principle of existential instantiation[2] to give that "some integer" the name m in order to work with it also as if we knew what it was.

Occasionally we may be given a problem in a way that asks us to think of a certain variable as generic right from the start. For instance, instead of being asked to prove that the square of any odd integer is odd, we might have been given the problem: "Suppose n is any odd integer. Prove that n^2 is odd." In this case, after reading the first sentence, we should think of n as capable of being

[2] Existential instantiation: If we know or suspect that an object exists, then we may give it a name, as long as we are not using the name for another object in our current discussion.

replaced by any arbitrary odd integer, and we would omit the first sentence of the proof that is given above.

An important use of variables as generic elements in mathematics education occurs in deriving the equations of lines, circles, and other conic sections. For example, to derive the equation of the line through $(3, 1)$ with slope 2, we could start as follows: "Suppose (x, y) is any point on the line." As long as we deduce properties of x and y without making any additional assumptions about their values, everything we conclude about (x, y) will be true no matter what point on the line might be substituted in its place.

We could continue by considering two cases: the first in which $(x, y) \neq (3, 1)$ and the second in which $(x, y) = (3, 1)$. For the first case, we note that what insures the straightness of a straight line is the fact that its slope is the same no matter what two points are used to compute it. Therefore, if the slope is computed using (x, y) and $(3, 1)$, the result must equal 2:

$$\frac{y-1}{x-3} = 2, \quad \text{and so} \quad y - 1 = 2(x - 3). \ (*)$$

This concludes the discussion of the first case. In the second case, $(x, y) = (3, 1)$ and both sides of equation (*) equal zero. So in this case it is also true that $y - 1 = 2(x - 3)$. Therefore, because no assumptions about (x, y) were made except for its being a point on the line, we can conclude that every point (x, y) on the line satisfies the equation $y - 1 = 2(x - 3)$.

3 Conclusion

This paper has advocated placing greater emphasis on the role of variables as placeholders to help address students' difficulties as they make the transition to algebra and more advanced mathematical subjects. Supporting examples were given from a variety of mathematical perspectives. It is hoped that the paper will stimulate additional research to delve more deeply into the issues it raises.

References

1. Dieudonné, J.: Abstraction in mathematics and the evolution of algebra. In: Lamon, W. (ed.) Learning and the Nature of Mathematics, pp. 102–103. SRA Associates, Inc., Chicago (1972)
2. Frege, G.: The Basic Laws of Arithmetic: Exposition of the System (Translated and edited by M. Furth). University of California Press, Berkeley (1964)
3. Klein, F.: Elementary Mathematics from an Advanced Standpoint (Translated from the 3rd German edition by Hedrick, E.R. and Noble, C.A.). Dover Publications, New York (1924)
4. Quine, W.V.O.: Methods of Logic, 4th edn. Harvard University Press, Cambridge (1982)
5. Tarski, A.: Intrtoduction to Logic and to the Methodology of Deductive Sciences. Oxford University Press, New York (1941)
6. Whitehead, A.N.: An Introduction to Mathematics. Henry Holt and Co., New York (1911)

Logic Training through Algorithmic Problem Solving

João F. Ferreira [1] , Alexandra Mendes [1] , Alcino Cunha [2] , Carlos Baquero [2] , Paulo Silva [2] , L.S. Barbosa [2] , and J.N. Oliveira [2]

[1] School of Computer Science, University of Nottingham, Nottingham, England
joao@joaoff.com, afm@cs.nott.ac.uk
[2] CCTC & Dep. Informatics, Minho University, Braga, Portugal
{mac,cbm,paufil,lsb,jno}@di.uminho.pt

Abstract. Although much of mathematics is algorithmic in nature, the skills needed to formulate and solve algorithmic problems do not form an integral part of mathematics education. In particular, logic, which is central to algorithm development, is rarely taught explicitly at pre-university level, under the justification that it is implicit in mathematics and therefore does not need to be taught as an independent topic. This paper argues in the opposite direction, describing a one-week workshop done at the University of Minho, in Portugal, whose goal was to introduce to high-school students calculational principles and techniques of algorithmic problem solving supported by calculational logic. The workshop resorted to recreational problems to convey the principles and to software tools, the *Alloy Analyzer* and *Netlogo*, to animate models.

Keywords: equational logic, calculational method, problem solving, algorithm derivation.

1 Introduction and Overview

It is consensual that Logic plays an essential role in rigorous software development. But the converse is also true, even if less well-known: literacy in logic, i.e., the ability to make productive use of the logic methods and tools, can be improved through algorithmic problem solving. Actually, two decades of research on *correct-by-construction* program design have created a new discipline of algorithmic problem solving and shed light on the underlying mathematical structures, modelling, and reasoning principles. In particular, it emphasises goal-directed, calculational construction of algorithms as opposed to more traditional guess-and-verify methodologies. Starting with the pioneering work of Dijkstra and Gries [8,13], and in particular, through the development of the so-called *algebra of programming* [5,4], a *calculational style* [3,11,7] emerged, emphasising the use of systematic mathematical calculation in the design of algorithms. The realisation that such a style is equally applicable to logical arguments [8,13] and that it can greatly improve on traditional verbose proofs in natural language has led to a systematisation that can, in return, also improve exposition in the

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 62–69, 2011.

more classical branches of mathematics. In particular, lengthy and verbose proofs (full of natural language explanations for "obvious" steps) are replaced by easy-to-follow calculations presented in a standard layout which replaces classical implication-first logic by variable-free algebraic reasoning [11,12].

Such a systematisation of a calculational style of reasoning, proceeding in a essentially syntactic way, greatly improves on the way proofs are presented. In particular it may help to overcome the typical justification for omitting proofs in school mathematics: that they are difficult to follow for all but exceptional students.

However, and although much of mathematics is algorithmic in nature, the skills needed to formulate and solve algorithmic problems do not form an integral part of mathematics education. Also, the teaching of computer-related topics at pre-university level focuses on enabling students to be effective users of information technology, rather than equip them with the skills to develop new applications or to solve new problems.

In such a context, this paper reports on a concrete case-study on exploiting and combining the dynamics of algorithmic problem solving and calculational reasoning to introduce logic in high-school as a live and productive tool. A tool to boost the abilities students need to overcome the challenges they will encounter through life. This experiment shows that logic skills can be trained through simple problems that emphasise formalisation and calculation. For example, logic puzzles where the goal is to solve simultaneous equations on Booleans, can be introduced by analogy with simultaneous equations on numbers. High-school students already learn how to solve simultaneous equations on numbers; going from the reals to the simpler Boolean domain, where each variable is either true or false, seems a natural step to follow. Furthermore, illustrating how logic can be used to model and solve algorithmic problems, improves the students' abilities to solve problems in general. Related research, leading to similar conclusions, is reported in [6,1,2,10].

2 An Educational Experiment

In July 2010, we organised a one-week workshop for Portuguese high-school students (aged between 14 and 17) on algorithmic problem solving. The goal was to show, through active involvement in tackling concrete problems, how the principles and techniques developed by computing scientists can be used to model and solve complex problems. There were 13 students enrolled in the workshop, all above-average students with a high interest on mathematics.

The workshop provided the opportunity to assess how pre-university students react to the calculational method and proof format. Two tools were used to increase interactivity and to show how machines can assist in problem-solving: *Alloy Analyzer* [15], to prototype models, and *NetLogo* [16], a multi-agent programmable modelling environment.

A summary of the plan of activities is shown in table 1. The week was divided into three main parts, detailed in the sequel. The first two days were used to

introduce computer modelling, supported by Alloy. The two following days were devoted to basic concepts in algorithmic problem solving: concision in naming the elements of a problem, symmetry, calculational logic, and invariants. These concepts were introduced using a *pen-and-paper* approach to reinforce the idea that computers are not needed to explore the topics that underlie computing. The final day was devoted to modelling complex systems in *NetLogo*.

Table 1. Plan of activities (the duration of each session was 3 hours)

Day	Activities
1	Introduction to Alloy. Modelling of a simple logic problem.
2	Modelling "The chameleons of Camelot" in Alloy.
3	Importance of concision and symmetry in algorithmic problem solving. River-crossing problems.
4	Introduction to calculational logic (through a logic puzzle). Introduction to invariants. Definitive solution to the problem "The chameleons of Camelot".
5	Introduction to *NetLogo*. Modelling of the problem "The chameleons of Camelot" in *NetLogo*.

Modelling problems in Alloy. Alloy is a formal specification language based on relational logic (first-order logic enriched with relational product, composition, meet, converse and other relational operators). Alloy is getting increasingly popular in the software engineering community since it embodies a *lightweight* approach to formal methods: its minimalist syntax and straightforward semantics (centered in the unifying concept of relation) make it particularly easy to learn and well-suited for automatic verification. Bounded verification of Alloy assertions can be performed by the *Alloy Analyzer* tool: the model is translated to propositional logic and fed to an off-the-shelf SAT solver; when found, counter-examples are graphically displayed for better comprehension.

Alloy proved quite effective in this workshop. Set-theory is part of high-school curriculum, and the knowledge of the students was enough to understand the usage of relations as a specification formalism and even to develop small models after a one morning introductory course. As an example, in day 1, the students used Alloy to determine when they can be their own grandfathers (more precisely, when someone is her own step-grandfather). More important than the tackled problems, was the realisation that logic could be made "alive" with the help of computational tools: after modelling, *Alloy Analyzer* was used to explore properties of the problem in an interactive way.

Pen-and-paper approach. The pen-and-paper approach was central to the workshop dynamics. Our starting point was the following logic puzzle:

> In an abridged version of Shakespeare's *Merchant of Venice*, Portia had two caskets: gold and silver. Inside one of these caskets, Portia had put her portrait, and on each was an inscription. Portia explained to her

suitor that each inscription could be either true or false but, on the basis of the inscriptions, he was to choose the casket containing the portrait. If he succeeded, he could marry her.

The inscriptions were:

Silver: *The portrait is not in this casket.*
Gold: *Exactly one of these inscriptions is true.*

Which casket contained the portrait? What can we deduce about the inscriptions?

One way of solving the problem is to introduce the variables pg for "the portrait is in the gold casket", ps for "the portrait is in the silver casket", ig for "the inscription in the gold casket is true", and is for "the inscription in the silver casket is true". Then, we are given:

$$(ig \equiv (ig \equiv \neg is)) \wedge (is \equiv \neg ps) \wedge (pg \equiv \neg ps) \quad .$$

The calculation that determines the values of the variables is a straightforward exercise in calculational logic:

$$\begin{array}{ll} & (ig \equiv (ig \equiv \neg is)) \wedge (is \equiv \neg ps) \wedge (pg \equiv \neg ps) \\ = & \quad \{ \quad \text{associativity} \quad \} \\ & ((ig \equiv ig) \equiv \neg is) \wedge (is \equiv \neg ps) \wedge (pg \equiv \neg ps) \\ = & \quad \{ \quad \text{reflexivity and negation} \quad \} \\ & (\mathsf{false} \equiv is) \wedge (is \equiv \neg ps) \wedge (pg \equiv \neg ps) \\ = & \quad \{ \quad \text{substitution of equals for equals} \quad \} \\ & (\mathsf{false} \equiv is) \wedge (\mathsf{false} \equiv \neg ps) \wedge (pg \equiv \neg ps) \\ = & \quad \{ \quad \text{negation and substitution of equals for equals} \quad \} \\ & (\mathsf{false} \equiv is) \wedge (\mathsf{true} \equiv ps) \wedge (pg \equiv \mathsf{false}) \quad . \end{array}$$

After being introduced to the rules used in the calculation above, the students came up with the same solution very easily.

Modelling complex systems in NetLogo. The use of *Netlogo* allowed the introduction of a few examples of how very simple models can give rise to complex interactions. The covered examples included models for forest fires, community segregation, and soil erosion. Following these examples, and inspecting the code, the students could assess classical cases of complex systems and emergent behaviour.

In particular, the forest fire model shows how tree density plays an important role in the percentage of forest burned by a fire event. By playing with the model, students could perceive critical values in the density that when reached lead to a phase transition where most of the forest is burned. The language is so accessible that, even at first contact, some students could add to the model the influence of wind to the fire propagation.

Unifying the three parts. As we can see in table 1, the problem "The chameleons of Camelot" was discussed in each of the three parts. In fact, the problem was used to unify them. Using *Alloy Analyzer*, the students were able to find examples of arguments for which the problem can be solved. However, since Alloy models are verified in bounded domains, a definitive answer could not be obtained for all arguments, namely those for which there is no solution. We then modelled the problem using pen and paper and we were able to get a definitive answer. Finally, we modelled the problem in *NetLogo*. The graphical interface of the tool enriched the experience and allowed the students to interact with the problem. In the next section, we describe the problem and provide more details about how the three different approaches complement each other.

3 An Algorithmic Problem: The Chameleons of Camelot

One of the problems extensively used in the workshop was a generalisation of "The chameleons of Camelot", as stated in [14, p. 140] (a more recent and accessible reference is [17]). The problem was used to help students to recognise, model, and solve algorithmic problems. In particular, it has a good potential to introduce non-determinism, problem decomposition, invariants, and program termination.

Problem statement. On the island of Camelot there are three different types of chameleons: grey chameleons, brown chameleons, and crimson chameleons. Whenever two chameleons of different colours meet, they both change colour to the third colour. For which number of grey, brown, and crimson chameleons is it possible to arrange a succession of meetings that results in all the chameleons displaying the same colour?

For example, if the number of the three different types of chameleons is 4, 7, and 19 (irrespective of the colour), we can arrange a succession of meetings that results in a monochromatic state:

$$(4\,,7\,,19)\to(6\,,6\,,18)\to(5\,,5\,,20)\to\cdots\to(0\,,0\,,30)\quad .$$

On the other hand, if the number of chameleons is 1, 2, and 3, it is impossible to make them all display the same colour.

Modelling the problem in Alloy. The first step towards the solution was to model the problem in Alloy. After modelling the rules governing the evolution of the chameleon colony, *Alloy Analyzer* was used to explore different combinations of colours and detect which could evolve to a monochromatic state (automatically detecting the succession of meetings leading to that state). This provided useful insight into finding the logical invariant constraining the state, and motivated the students for the following *pen-and-paper* approach.

Pen-and-paper solution. Although *Alloy Analyzer* could be used to determine if given colonies of chameleons could reach a monochromatic state, it could not give a definitive answer when there was no solution. This limitation motivated a new approach. Using g, b, and c to denote, respectively, the number of grey, brown, and crimson chameleons, the first step was to model the algorithm that underlies the problem and to decompose it into two simpler parts:

$$
\begin{array}{ll}
\mathsf{do} & g \neq b \wedge g \neq c \wedge b \neq c \;\rightarrow \\
& \quad \mathsf{if} \quad 0<g \wedge 0<b \;\rightarrow\; g,b,c \;:=\; g{-}1,b{-}1,c{+}2 \\
& \quad \Box \quad 0<g \wedge 0<c \;\rightarrow\; g,b,c \;:=\; g{-}1,b{+}2,c{-}1 \\
& \quad \Box \quad 0<b \wedge 0<c \;\rightarrow\; g,b,c \;:=\; g{+}2,b{-}1,c{-}1 \\
& \quad \mathsf{fi} \\
\mathsf{od} & \\
\{ \quad g=b \;\vee\; g=c \;\vee\; b=c \quad \}\,; & \\
\textit{Two classes of chameleons are now equally numbered. Arrange} & \\
\textit{a meeting between all the chameleons of these two classes.} & \\
\{ \quad (g=0 \wedge b=0) \vee (g=0 \wedge c=0) \vee (b=0 \wedge c=0) \quad \}\quad . &
\end{array}
$$

The algorithm consists of a loop (enclosed between the keywords do and od) that terminates when two classes of chameleons are equally numbered. Once we reach such a state, the problem is easy to solve.

The loop executes while at least one of the three guards (the conditions at the left of the arrow $\rightarrow$) is satisfied. If more than one guard is satisfied, the block operator ($\Box$) ensures that only one of the three assignments is chosen non-deterministically. The first assignment, for example, corresponds to a meeting between a grey chameleon and a brown chameleon: provided that there are chameleons of both these colours, the number of grey chameleons (g) and brown chameleons (b) both decrease by 1, whilst the number of crimson chameleons (c) increases by 2.

The next step of the solution was to find an invariant of the three assignments. Based on the postcondition of the loop, we calculated the invariant

$$g \cong b \pmod 3 \;\vee\; g \cong c \pmod 3 \;\vee\; b \cong c \pmod 3 \quad .$$

This pointed to the conclusion that if the initial numbers of chameleons do not satisfy the invariant, that is, if no two initial numbers are congruent modulo 3, it is impossible to organise a succession of meetings that results in all the chameleons displaying the same colour. At this point, the students understood why *Alloy Analyzer* could not find any succession of meetings when two initial numbers were not congruent modulo 3.

Finally, we discussed how to remove the non-determinism from the algorithm shown above so that we can guarantee termination.

The *pen-and-paper* approach to this problem is fully described in [9] (including the solution, notes on how to present the problem, and exercises).

Modelling the problem in NetLogo. Modelling this problem in *NetLogo* allowed a closer simulation of how the encounters could occur in a spacial setting. The model places the chameleon population in a virtual torus and allows them to roam at constant speed. When a pair of chameleons reaches a given proximity threshold they interact and possibly change colour. As the simulation proceeds, a graph shows the population sizes for each colour along time. The interface allows either a randomised allocation of colours to a given total population size or an individual assignment of each initial colour population.

The experiment helped to determine how fast (when possible) one can observe convergence to a single colour state, in this idealised movement model. It also illustrated, empirically, that although some initial colour distributions can allow a set of encounters that leads to single colour convergence, this is statistically highly unlikely for all but the smallest sized populations.

4 Conclusions

At least from the point of view of the 13 students present, who were asked to fill an anonymous assessment survey at the end, the workshop was a success. The proposed questions sought open answers, where students could provide their opinion not being limited to a few standard cases. Although this does not allow for a statistical treatment of the results, it provides us with a more personal feedback and interesting insights about the outcome of the workshop.

Most students wrote that their expectations were exceeded. Some remarked the workshop has further stimulated their interest in mathematics, while other stated to have learned something about how software is developed and the activity of the professionals working in this field. Asked whether anything in that activity came as a surprise, most of them pointed out that the connection between programming computers and solving logical problems was by and large unexpected. They were also surprised by the accessible and interesting contents, while highlighting the overall quality of the sessions. All the students considered the pace of this workshop appropriate, although some recognised that it was faster than in high-school.

They were expecting to have more contact with programming languages and computers, but in the end, they seem to have understood that clear and structured reasoning is the key to solve problems and write good software, and that computers can be useless if the programs they put in motion are not carefully designed. They also remarked they have enjoyed approaching general problems and understood the role of abstraction and genericity in programming.

Certainly, no general conclusions can be extracted from a limited and single experiment. In general, however, the challenge placed to students was surprisingly well received and the feedback was quite positive. For example, the students *calculated* the solution of a logic puzzle very easily, which suggests that calculational logic can indeed be introduced at high-school level. Furthermore, most of them enjoyed the recreational flavour of the problems, and, at the end of the week, they were able to apply techniques like invariants by themselves. They

also liked the interactivity provided by the software tools that we have used, and, most important, they enjoyed being challenged.

Acknowledgements. On-going collaboration with Roland Backhouse is deeply acknowledged. This research was supported by the MathIS project under contract PTDC/EIA/73252/2006. The first two authors were further supported by FCT grants SFRH/BD/24269/2005 and SFRH/BD/29553/2006, respectively.

References

1. Back, R.J., Mannila, L., Peltomaki, M., Sibelius, P.: Structured derivations: A logic based approach to teaching mathematics. In: FORMED 2008: Formal Methods in Computer Science Education, Budapest (2008)
2. Back, R.J., von Wright, J.: Mathematics with a little bit of logic: Structured derivations in high-school mathematics (2006)
3. Backhouse, R.C.: Mathematics and programming. A revolution in the art of effective reasoning. Inaugural Lecture, University of Nottingham (2001)
4. Backhouse, R.C., Hoogendijk, P.F.: Elements of a relational theory of datatypes. In: Möller, B., Schuman, S., Partsch, H. (eds.) Formal Program Development. LNCS, vol. 755, pp. 7–42. Springer, Heidelberg (1993)
5. Bird, R., Moor, O.: The Algebra of Programming. Series in Computer Science. Prentice-Hall International, Englewood Cliffs (1997)
6. Boute, R.: Using Domain-Independent Problems for Introducing Formal Methods (chapter 22). In: Misra, J., Nipkow, T., Karakostas, G. (eds.) FM 2006. LNCS, vol. 4085, pp. 316–331. Springer, Heidelberg (2006), doi:10.1007/11813040_22
7. Dijkstra, E.W.: On the economy of doing mathematics. note EWD1130 (1992)
8. Dijkstra, E.W., Scholten, C.S.: Predicate Calculus and Program Semantics. Springer, New York (1990)
9. Ferreira, J.F.: Principles and Applications of Algorithmic Problem Solving. Ph.D. thesis, School of Computer Science, University of Nottingham (2010)
10. Ferreira, J.F., Mendes, A.: Student's feedback on teaching mathematics through the calculational method. In: 39th ASEE/IEEE Frontiers in Education Conference. IEEE, Los Alamitos (2009)
11. van Gasteren, A.J.M.: On the Shape of Mathematical Arguments. Springer Lect. Notes Comp. Sci, vol. (445). Springer, Heidelberg (1990)
12. Gries, D., Feijen, W.H.J., van Gasteren, A.J.M., Misra, J.: Beauty is our Business. Springer, Heidelberg (1990)
13. Gries, D., Schneider, F.: A Logical Approach to Discrete Mathematics. Springer, New York (1993)
14. Honsberger, R.: In Polya's Footsteps: Miscellaneous Problems and Essays (Dolciani Mathematical Expositions). The Mathematical Association of America (1997)
15. Jackson, D.: Software Abstractions: Logic, Language, and Analysis. The MIT Press, Cambridge (2006)
16. Tisue, S., Wilensky, U.: Netlogo: A simple environment for modeling complexity. In: International Conference on Complex Systems, pp. 16–21 (2004)
17. Winkler, P.: Puzzled: Understanding relationships among numbers. Commun. ACM 52(5), 112 (2009), doi:10.1145/1506409.1506432

Concrete Epistemic Modal Logic: Flatland

Olivier Gasquet and François Schwarzentruber

IRIT (Institut de Recherche en Informatique de Toulouse)
Université de Toulouse
Toulouse, France

Abstract. In this paper, we give a logic for perception and knowledge: Flatland. This semantics of this framework is a concrete Kripke model so that it is an easy-to-understand toy example for students. We present a piece of software called `Plaza's world` enabling to check formulas in such a concrete Kripke model and to announce formulas.

Keywords: Epistemic modal logic, Public announcements, Flatland.

1 Introduction

This work is directly inspired by some aspects of the thesis of one of the authors of this paper [9]. The initial idea was inspired by the famous `Tarski's world` [2], a software that allow undergraduate students to practice predicate logic in a concrete situation[1].

Here, we aimed at developing a concrete example of application of epistemic logic [5] to help graduate students to understand and to practice it with the help of announcements as in the logic of announcement of [7] who gave his name to our project. This idea lead to several theoretical developments (axiomatization, completeness, complexity) and it is now time to go back to the initial aim: to offer a tool for teaching epistemic logic. In this paper, we will deliberatelay omit theoretical considerations and focus on the tool itself, all details can be found in the thesis. Also, we consider the reader more or less familiar with modal logic, this paper is written for teachers and not really for students.

We consider a framework where some artificial agents have some knowledge[2] and, can see or cannot see both other agents and where they are looking at. We will be interested into questions of the type "'Do agent a sees agent b?", "Do agent a knows that agent b sees agent c?", "Do the group of agents $\{a, b\}$ share the common knowledge that each of them sees c?", ... To this aim, we will consider a concrete situation in the plane[3] where we suppose that any agent sees the entire half plane in front of her. At the moment, `Plaza's world` opens on a window where all different dispositions of three agents are visible, each of these dispositions representing a possible state of affair. In some state, let us say that of Figure 1:

[1] A universe of coloured geometrical figures in which students interpret or write formulas like $\forall x.(square(x) \rightarrow \exists y.(triangle(y) \wedge grey(y) \wedge on\_left(x, y)))$.

[2] As often in the literature, our logic for knowledge is S5.

[3] We do not present here the version where agents are situated on a line which has been the first framework investigated in [4] and [8].

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 70–76, 2011.

Fig. 1. A possible disposition of agents

> Ann sees both Bob and Chris, Chris also sees everybody but Bob sees no one. But more, "Ann knows that Chris sees Bob", and also "Ann knows that Chris knows that Bob does see Ann', ... What does Bob know? In fact nothing beside that Ann and Chris either sees him or does not see him. Bob can imagine any possible situations of Ann and Chris, but "Ann knows that Bob doesn't know she sees him" (*).

Any agent can imagine that the *actual* situation is any situation that is compatible with her partial view, with what she sees, imagination concerns what she does not see. `Plaza's world` allows two kind of actions:

1. the user may test whether such sentence may be true in some situations: by entering a logical formula representing the sentence, then `Plaza's world` will indicate those situations where the formula is true: in situation of Figure 1, the sentence (*) is true;
2. the user may *publically announce* a sentence like "Ann knows that Bob does not see Chris"(**), this would provoke the deletion from the model of situations which are not compatible with this sentence.Of course this may change the result of testing the truth of some sentences, e.g. after such an announce, it is true that "Bob knows that Ann sees him" in any situation since it is a logical consequence of the announcement which, of course has become true.

Note that at the beginning, knowledge of agents is only based on what they see, but evolve as announcements happen.

With these two possibilities, students may *play* with epistemic logic: it helps in understanding formal truth-conditions in Kripke models by interpreting formulas in a concrete and intuitive situation, and modal subtleties by experimenting the effects of various announces, in particular the classic Moore sentence [6]: "Ann sees Bob and Bob does not know Ann sees him" which becomes false after being announced.

The piece of software `Plaza's world` can been launched from the web site

`http://www.irit.fr/~Francois.Schwarzentruber/flatland/`

2 The Logic Framework of Flatland

The syntax is the following: formulas describing questions or announces are built with the usual boolean connectives ($\wedge$, $\vee$, $\neg$) together with K_a (agent a knows that...) and the only propositions of the form $(a \rhd b)$ (agent a sees agent b). Thus sentence (*) is represented by the formula: $K_{Ann}\neg K_{Bob}(Ann \rhd Bob)$.

Concerning the semantics, formulas are interpreted in Kripke models corresponding to all situations and their links $(W, R_{Ann}, R_{Bob}, R_{Chris}, m)$ where:

- W is the set of all possible situations and is depicted in Figure 2;
- m says in each situations who sees who;
- each R_a links each situation to those that agent a cannot distinguish on the basis of what she sees or knows because of an announce, they are called *accessible situations for a*.

The interpretation of boolean connectives is classical and that of $K_a\varphi$ is as usual in epistemic modal logic: $K_a\varphi$ is true in a situation iff φ is true in any accessible situation for a. For more details, the reader is invited to look at [9].

3 Running Example

This section deals with a running example with the model-checker `Plaza's world`. Let us start with the Kripke model of Flatland with 3 agents depicted in Figure 2. As you can see, the Kripke graph is not planar[4].

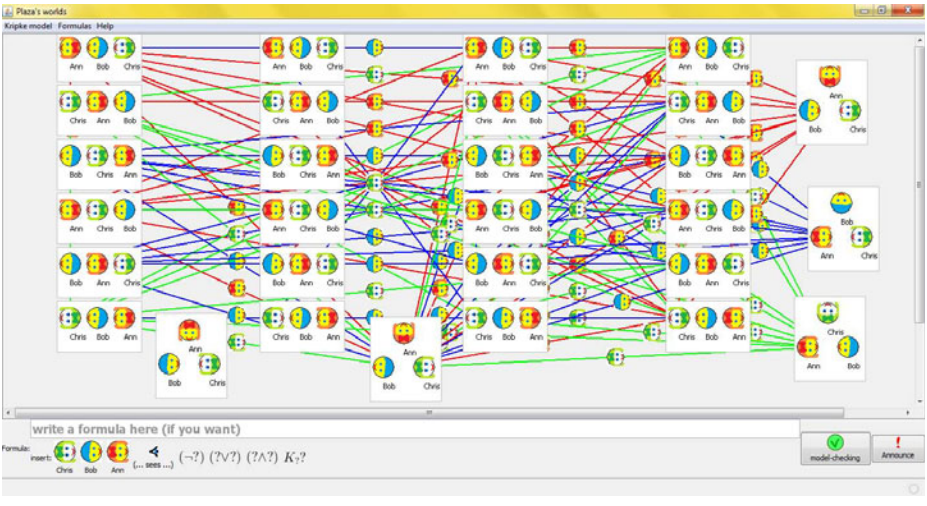
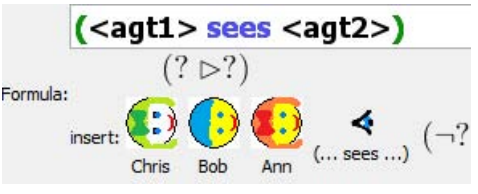


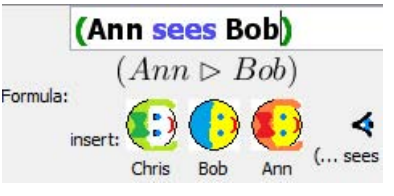
Fig. 2. 3-agents Flatland model

[4] For graph theorists: it contains the graph $K_{3,3}$.

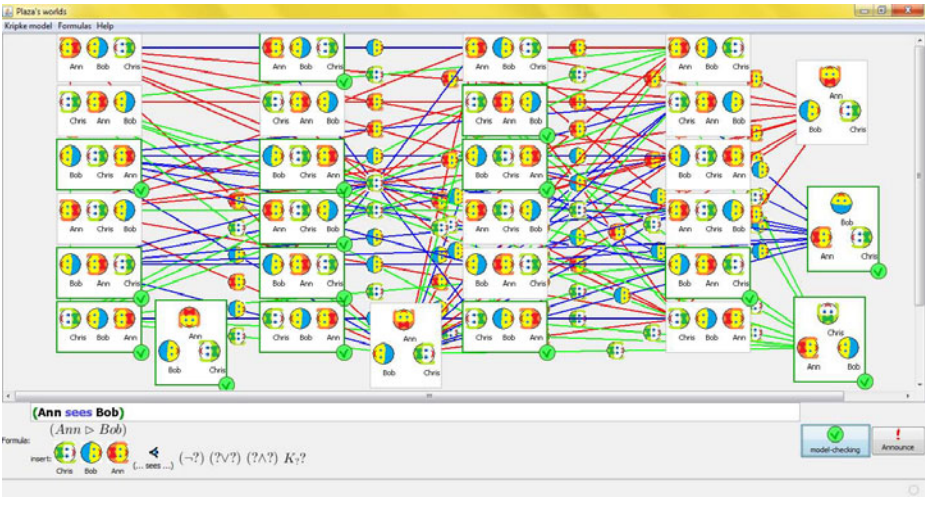
The graphical user interface enables us to see a Kripke model on the top of the screen. The bottom of the screen is devoted to write an epistemic formula. Then one button "model-checking" enables to check where this formula is true whereas the other button enables to announce the formula.

Now let us check the formula $Ann \triangleright Bob$ on this model. In order to this formula, the graphical user interface provides some buttons for each constructions. We simply click on those buttons to enter the formula:

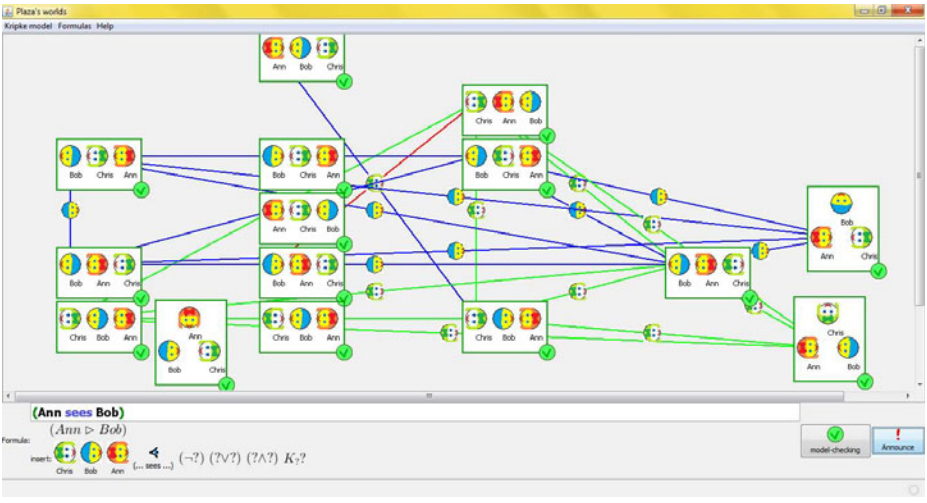




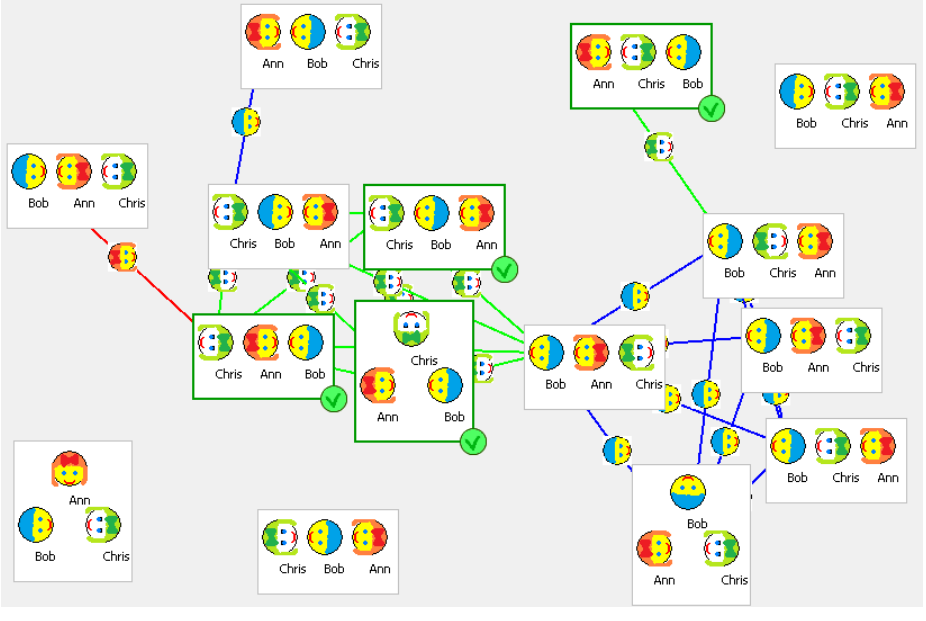
By clicking on the button "model-checking", the software highlights the worlds in which the formula $Ann \triangleright Bob$ is true.



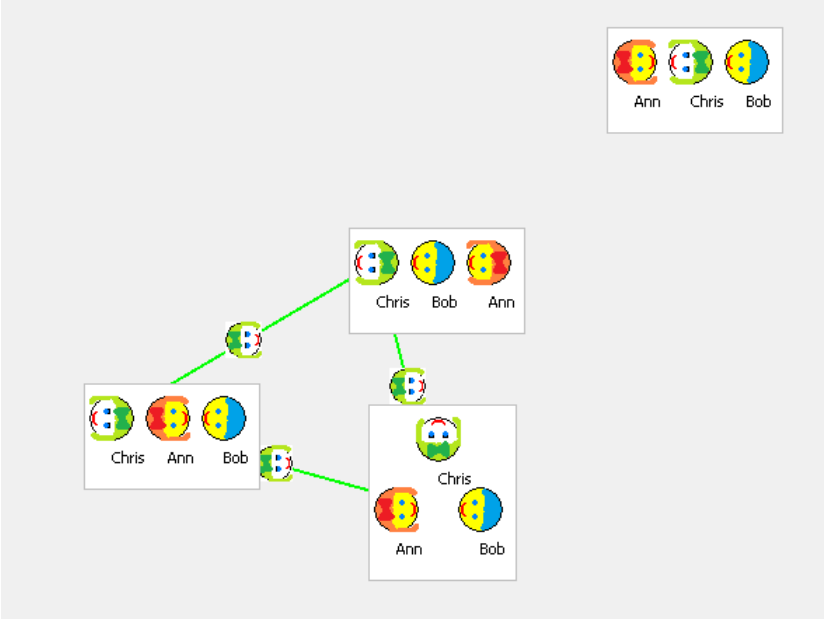
The button “announce” updates the model by deleting all worlds in which the formula $Ann \rhd Bob$ is false. We obtain:



Now we rearrange the disposition of the worlds and we check the Moore's sentence $Bob \rhd Chris \wedge \neg K_{Chris} Bob \rhd Chris$ on the model.



We can announce this Moore sentence and check that this formula is now false in all worlds of the updated model:



Of course now the student can check that $K_{Chris}(Ann \rhd Bob \wedge Bob \rhd Chris)$ in all worlds.

4 Beyond the Scene

The piece of software `Plaza's world` has been developped in JAVA for one major reason: JAVA enables the program to run on several multi-platforms. Even more: JAVA enables the software to run without any installation. Indeed the program can been launched directly from the Web via Java Web Start. This is a good requirement for students.

The other advantage of JAVA is that the language is object-oriented and many libraries already exists. For instance:

- it was possible to use the library jLatexMath (`http://forge.scilab.org/index.php/p/jlatexmath/`) in order to display epistemic modal logic formulas correctly;
- it was easy to extend the class JTextField for colouring parenthesis and key words as "sees" and "knows";
- the software is easy to extend: for instance the class KripkeGraph represents an abstract Kripke model that can be displayed. Our 3-agents Kripke model of Flatland is only a specific implementation of this class.

5 Conclusion

We found it pleased master students who could "see" epistemic logic in action in a concrete and intuitive application. It also ease the understanding of what an

epistemic relation stands for because here the epistemic relation is naturally defined. For sure `Plaza's world` is still to be improved and is under development. We already plan to do the following improvement:

- to enhance the layout of the graph especially when the Kripke graph is complicated, non-planar. Solutions are multiple: using 3D, algorithms with spring and gravity forces etc. Maybe, we are going to implement it using a graph visualization library like `Jung` (http://jung.sourceforge.net/);
- to allow the editing of the model: add nodes, remove edges by mouse clicking etc.;
- extending the language with common knowledge and allow to test or announce sentences like "All members of a group commonly knows that φ is true" (see [5] for details on common knowledge);
- add announcements in the language and allow formulas of the form "Ann knows that after she will have said that she knows that Bob does not see him, Bob will know she sees him" both for testing and announcement.

Of course there are also still open problems concerning the logical framework itself. For instance some researchers have introduced arbitrary announcement logic [1] in order to model the "knowability" of agents. Unfortunately it has been proved to be undecidable [3]. But what about arbitrary announcement logic in Flatland? Is it still undecidable? Can we implement it in `Plaza's world`?

References

1. Balbiani, P., Baltag, A., Van Ditmarsch, H., Herzig, A., Hoshi, T., De Lima, T.: What can we achieve by arbitrary announcements?: A dynamic take on Fitch's knowability. In: Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge, p. 51. ACM, New York (2007)
2. Barwise, J., Etchemendy, J.: Tarski's World: Version 4.0 for Macintosh (Center for the Study of Language and Information - Lecture Notes). Center for the Study of Language and Information/SRI (1993)
3. French, T., van Ditmarsch, H.: Undecidability for arbitrary public announcement logic
4. Gasquet, O., Schwarzentruber, F.: Knowledge in Lineland (Extended Abstract) (short paper). In: International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Toronto (Canada), 10/05/2010-14/05/2010 (2010)
5. Halpern, Y.M.J.Y.: A guide to completeness and complexity for modal logics of knowledge and belief (1996)
6. Moore, G.E.: Moore's paradox. In: Baldwin, T. (ed.) Selected Writings, pp. 207–212. Routledge, Londres (1993)
7. Plaza, J.: Logics of public announcements. In: Proceedings 4th International Symposium on Methodologies for Intelligent Systems (1989)
8. Schwarzentruber, F.: Knowledge about lights along a line (2009)
9. Schwarzentruber, F.: Seeing, knowing, doing. Case Studies in Modal Logic (2010)

SATOULOUSE: The Computational Power of Propositional Logic Shown to Beginners

Olivier Gasquet, François Schwarzentruber, and Martin Strecker

IRIT (Institut de Recherche en Informatique de Toulouse)
Université de Toulouse
Toulouse, France

Abstract. We present a tool (SATOULOUSE) that can help teachers, particularly in computer science, to convince undergraduate students that logic may be powerful. It is to be used very early in a logic course, in order to enhance students' motivation to learn propositional logic. SATOULOUSE simply consists of a friendly interface that offers several syntactic facilities and which is connected with a sufficiently powerful SAT-prover (namely SAT4J) allowing to automatically solve big instances of difficult problems (such as time-tables or Sudokus).

Keywords: Teaching logic in computer science; SAT solvers; constraint solving.

1 Introduction

The authors of this paper have been teaching logic to second-year undergraduate students in computer science for several years.

From our experience, most of our students are not sensitive to the "magic of logic", only a few are, and we found it could be useful to be helped in convincing them that logic is "at least" useful for computer scientists and that computer science does not only consist in hacking C-code or JAVA or PHP. But this has to be done at a time they do not know a lot of logic of course, the aim being to show them the "power of logic" in order to improve their motivation to study it.

Up to now, we used to motivate logic by abstract examples about its application in computer science (program verification, knowledge representation, planning, circuits design) and also by solving toy problems on a blackboard. But we began to think that it would be preferable to show and not only tell them that with only little knowledge, logic can be used to solve difficult problems whose size prevents humans from solving them by hand easily or would require rather complex programming. As we said, this has to be done very early during their course if motivation is in question. A SAT-solver can do that, but it turned out that none of the existing tools fitted our needs.

Of course, there are loads of logic tools (provers, proof assistants, truth table editors,...) on the Internet, even PROLOG could have been used, but none fits our requirements which are:

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 77–84, 2011.

– the tool must be very easy to install and to use, with no complex syntax;
– the prover can be used as a black box without knowing how it works;
– no normal forming, ordering on clauses, or PROLOG cut must be needed;
– only little knowledge in logic should be necessary.

As we could not find an existing tool fulfilling these requirements, we started to implement ours, and we came to the idea of just developing an interface that allows to very comfortably use a powerful SAT-prover (namely SAT4J, details in section 3): the whole tool being called SATOULOUSE. With this tool, students can experiment by themselves that a logical language is not only descriptive but may lead to computations that solve real-life problems. In particular, with SATOULOUSE, they solve Sudokus very easily, as well as many other combinatorial problems (Time-table, Map coloring, Electronic circuits,...). Since they are also asked in their programming course to implement a Sudoku solver in C, they can compare both approaches with many advantages for the logical one. SATOULOUSE has a real impact on students: they can compare the time needed to implement a C-program that solves Sudokus with no real warranty about its correctness, with the time needed to encode Sudokus in propositional logic. This comparison is neatly in favor of the logical approach. Of course, we also give them contextual elements about research on the SAT problem: conferences, SAT-solvers competitions, and benchmarks... Our aim is to make our students understand that being a computer scientist does not only consist in hacking C-code, but may only require them to tackle problems more abstractly so that they can re-use existing and efficient tools.

Here are the main facilities that SATOULOUSE offers:

– Input formulas need not to be in clausal form and arbitrary connectives may be used, normal forming is done dynamically during keyboarding of the user;
– Big conjunctions and disjunctions facilities are offered like in:

$$\bigwedge_{i\in\{1..9\}} \bigvee_{j\in\{1..9\}} \bigwedge_{n\in\{1..9\}} \bigwedge_{m\in\{1..9\},m\neq n} (p_{i,j,n} \rightarrow \neg p_{i,j,m})$$

– Running the solver only consists in clicking a button;
– The tool displays a model in the syntax of the input formula.

Then it is possible to show the power of propositional logic to students that have been trained a couple of hours to formalize sentences in logic and have acquired basic notions of validity and satisfiability. First, as an exercise, they formalize the rules of Sudoku on paper, as well as data corresponding to the content of already filled cells. Then they are asked to run SATOULOUSE and to solve the Sudoku (in fact, formulas are available in a menu, so they don't have to type them in). But this is not the whole story, since the same SAT-solver may be used for solving many other combinatorial problems as easily as they just did for Sudokus: they just have to formalize the constraints. Our students are asked to do so for: time-table, map coloring,... The use of SATOULOUSE is very recent and has concerned about 80 students, but it seems that for quite many of them

SATOULOUSE had a real impact on the representation they had of logic, in the sense that they were more numerous than usual to say that they believe logic to be useful for computer scientists, and it seems to us that they got better marks than usual when passing their exam. But we need to find out how to evaluate this more objectively.

SATOULOUSE is publicly available for download from the following site

`http://www.irit.fr/satoulouse/`

2 Interaction with SAToulouse

Launching SATOULOUSE displays a formula editor that allows to enter a set of formulae to be tested for satisfiability. Formulae can either be hand-written in a Lisp-like syntax, or introduced in a sort of syntax-directed editor, by progressively refining the syntax tree.

The formulae of SATOULOUSE are the traditional formulae of propositional logic, plus indexed conjunction and disjunction, which are of the forms:

- $\bigwedge_{i \in E} P_i$, where E is an enumeration of natural numbers.
  For example, $\bigwedge_{i \in \{1,2,3\}} P_i$ represents the conjunction $P_1 \wedge P_2 \wedge P_3$. It is written `(bigand j (1 2 3) (P i))` in textual syntax.
- $\bigwedge_{i \in E} P_i$, where E is a range of natural numbers (for example `(bigand i (1 .. 9) (P i))` in textual syntax).
- $\bigwedge_{i \in E | i \neq j} P_{i,j}$, where E is as above and $i \neq j$ is an inequality constraint (for example `(bigand i (1 .. 9) (diff i j) (P i j))` in textual syntax).

Corresponding indexed connectors exist for disjunction. On writing a formula in textual syntax, SATOULOUSE immediately renders it in a LaTeX-style.

2.1 Be Familiar with the Notion of Satisfiability

During our course, we give to the students easy exercises of formalization like:

"A robbery happened last night. Inspector Lestrade is on the premises. The thief has been seen: he is small and he left footprints of size 40[1]. There are three suspects: A, B or C. A is tall and takes size 43 shoes. B is small and takes size 40 shoes. C is small and takes size 43 shoes. Who is the thief?" (NB: they will have to elicit the fact that no one who takes size 43 wears 40).

SATOULOUSE is used to enter such formalization into the computer and solve this kind of problems in order to accustom students with it, but such examples can be treated by hand. Hence, we consider far bigger problems...

2.2 Logic Programming

Let's take a look at an example, the coding of a particularly simple form of Sudoku (9×9 cells) for the sake of clarity. A screenshot is shown in Figure 1.

[1] French size...

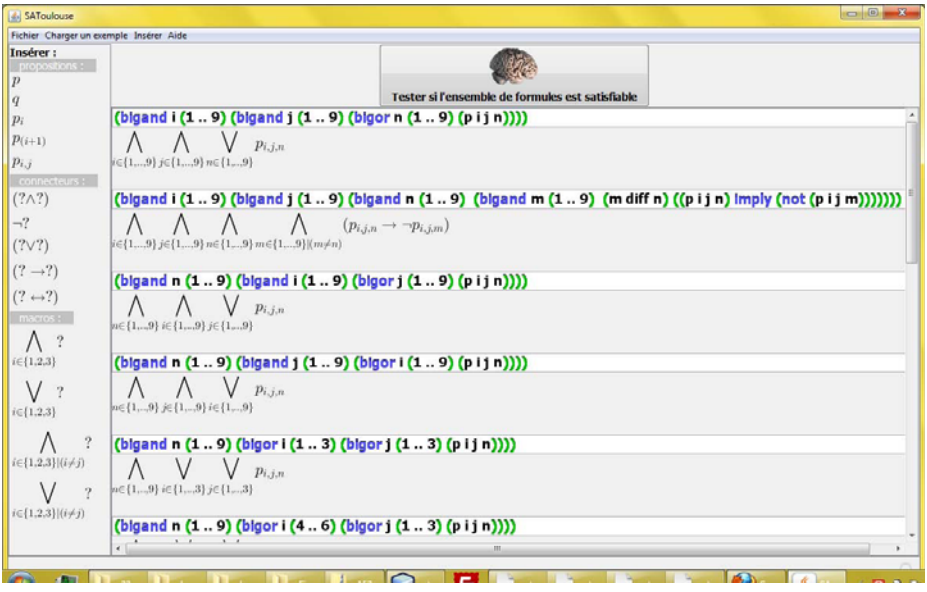


Fig. 1. Input of the Sudoku 9×9 problem in SATOULOUSE

Remember that in a Sudoku, the player is requested to fill in an 9×9 grid with numbers ranging from 1 to 9, such that some conditions are satisfied. To express the fact that at position (i, j) on the grid, there is a unique number 9, we use a number of constraints, for example (see screenshot):

- In each cell at position i, j there is at least one $n \in \{1, \ldots, 9\}$
- each cell (i, j) contains at most one n
- Each line and column contains each n of $\{1, \ldots, 9\}$ exactly once
- The game starts with already filled cells, this initial setting being described by a conjunction like $p_{1,2,3} \wedge p_{1,6,1} \wedge p_{2,3,6} \wedge \ldots$ (there is 3 in cell (1, 2), there is 1 in cell (1, 6), there is 6 in cell (2, 3), etc.)

Pressing the "brain" button transforms these formulae into the input format required by the SAT solver, and passes it the transformed input formula. SATOULOUSE then reports an outcome: Either the formula is satisfiable and then SATOULOUSE returns a validating assignment. Or SATOULOUSE returns a warning saying that the formula is unsatisfiable. For our example, SATOULOUSE returns the valuation depicted on Figure 2.

There are several other problems which can be easily implemented into SATOULOUSE (some of them are available in the menus). For instance:

Map coloring;
Planning;
Pentominoes;
Configuration of an avionics network, etc.
Cryptograms;
8 queens;
Failure in a electric circuit;

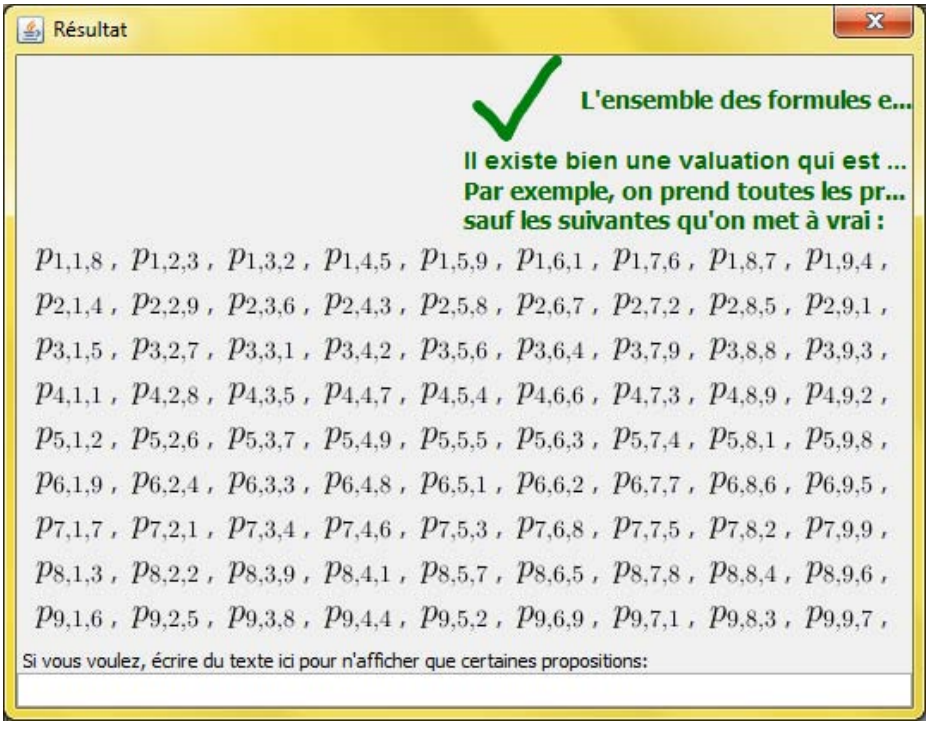


Fig. 2. Example of a valuation returned for the Sudoku problem of Figure 1

2.3 An Application to Genetics

The classical examples of the previous section are closely related to computer science. We want also our students to be open-minded and to be conscious that computer science, and especially logic, has applications in other domain, for instance in genetics.

A classical problem problem in genetics is the problem of haplotype inference by pure parsimony (HIPP) [3] [4]:

- input: n genotypes, that is to say n sequences of the form $\begin{pmatrix} hz_{i,1} & \dots & hz_{i,m} \\ g_{i,1} & & g_{i,m} \end{pmatrix}$ of m sites which can be either homozygous ($hz_{i,j}$) or heterozygous ($\neg hz_{i,j}$). In the case where the site is homozygous we specify whether the site is wild ($g_{i,j}$) or mutant ($\neg g_{i,j}$) where i ranges over $\{1, \dots, n\}$ and j ranges over $\{1 \dots, m\}$;
- output: the smaller set of haplotypes that explains the genotypes. More precisely we have to find r sequences of the form $(h_{k,1} \dots h_{k,m})$ such that for all $i \in \{1, \dots n\}$ there exists $k, l \in \{1, \dots r\}$ such that:
  - if $hz_{i,j}$, then $h_{k,j} = h_{l,j} = g_{i,j}$;
  - if $\neg hz_{i,j}$ then $h_{k,j} \neq h_{l,j}$.

2.4 Highlighting the Difficulty of SAT

Students may think that SAT is an easy problem: indeed the syntax and the semantics is so simple that it is surprising that this problem is difficult. In our course, we try to convince our students that the SAT problem is difficult in terms of complexity: indeed it is NP-complete [5]. In order to make our students aware of this difficulty, we show them that the current research in this area is prolific. We show them some big formulas of some benchmarks like on the Web site `http://www.satlib.org/` and treat them in SATOULOUSE.

3 Behind the Scenes

The piece of software SATOULOUSE is an open-source project globally written in JAVA so that it can be used on many different platforms.

3.1 The Graphical User Interface

The graphical user interface (GUI) enables the user to write formulas and to click on the button "check if the set of formulas is satisfiable". The GUI is written in JAVA in order to be able to use the suitable library SWING. In particular, as this library is object oriented, it was really simple to create our own widget to enter a formula of propositional logic. Formulas are also displayed in LaTeX style thanks to the library jLatexMath[2] which does not require any additional software (in particular it does not require a real LaTeX installation).

3.2 The SAT Solver

The SAT solver used as backend of SATOULOUSE is SAT4J [3]. The first advantage of this solver is that it is written in JAVA so it is easy to call it from the GUI. The second one is its good efficiency [1], and even if it is not the best SAT solver in the world (see SAT Race 2008), it is one of the best.

3.3 Connection between the SAT Solver and the Graphical User Interface

The translation from formulas written by the user to formulas for the SAT solver SAT4J is divided in two parts:

- expanding formulas, that is to say, rewrite macros into expanded conjunction normal forms. For instance, we expand the formula $\bigwedge_{i\in\{1..4\}} p_i$ into $(((p_1 \wedge p_2) \wedge p_3) \wedge p_4)$. Expanding formulas is done in Scheme code intepreted with the JAVA library kawa [2]. Scheme is a suitable language to perform this translation because parsing of formulas is directly done in this language: formulas are directly s-expressions.

[2] `http://forge.scilab.org/index.php/p/jlatexmath/`
[3] `http://www.sat4j.org/`

– translating the expanded conjunctive normal forms into a rigorous SAT4J input. This part is done in Java because we need to run SAT4J which is coded in JAVA. We construct a map between the propositions of the GUI (strings) and strict positive integers used by SAT4J and then build corresponding arrays of integer for SAT4J.

3.4 Loading and Saving a Problem

As we can give explicit names to propositions (and not only integers as for SAT4J), as we have macros (like $\bigwedge_{i\in\{1..4\}} switch_i$), SATOULOUSE has its own file format: we simply write s-expressions representing formulas. Nevertheless, SATOULOUSE is also able to load standard DIMACS cnf format files as depicted below.

```
c A sample .cnf file.
p cnf 3 2
1 -3 0
2 3 -1 0
1 2 -3 0
```

$$((p_1 \vee \neg p_3) \wedge (p_2 \vee p_3 \vee \neg p_1) \wedge (p_1 \vee p_2 \vee \neg p_3))$$

The standard DIMACS cnf format

4 Evaluation and Further Work

As far as we are aware, there is no other tool targeted at the same public as SATOULOUSE, since existing pedagogical tools (either implementation of truth-tables or semantic tableaux) that could do the job of searching a model cannot efficiently handle big problems, and real tools able to deal with them are definitely not designed to be used by beginners in logic.

The tool which comes closest to SATOULOUSE is Limboole[4], which also has the purpose of being used in class and is based on standard propositional logic (and not only clausal) input format. However, it does not offer the quantifiers over finite domains (indexed conjunction / disjunction) that allow for a concise statement of problems over structured domains such as Sudokus, map coloring etc. On the other end of the spectrum, there are highly developed input languages like of the Chaff[5] solver, which have a much too steep learning curve for 2nd year students.

The planning and configuration problems mentioned above can also be solved by constraint programming languages like Mozart[6], which often use dedicated solvers. The Alloy language and tool[7] allow to describe configuration problems in an "object-oriented" style and to state desired properties in an OCL-like syntax, which are then translated to SAT problems. These tools have clear merits for

[4] http://fmv.jku.at/limboole
[5] http://www.cs.nyu.edu/acsys/cvc3/
[6] http://www.mozart-oz.org
[7] http://alloy.mit.edu/

professional computer scientists, but, on the downside, have a steep learning curve and are therefore less appropriate for beginning computer science students.

We plan, in the future, to ask students to solve other problems that can be expressed in propositional logic, but to this end, we need to enrich the language of SATOULOUSE used in big conjunctions. Improvements we intend to introduce in SATOULOUSE are:

- Optimization of the translation of formulas into the internal format of SAT4J (by compiling them dynamically);
- Enriching the language of SATOULOUSE, e.g. by adding cardinality conditions for use in big conjunctions and disjunctions. For instance `(exact 3 i (1 .. 40)` ϕ_i`)` (`(atmost 3 i (1 .. 40)` ϕ_i`)`, `(atleast 3 i (1 .. 40)` ϕ_i`)`) for specifying that exactly (at most, at least) three formulas of the ϕ_i are true.

References

1. Le Berre, D., Parrain, A.: The Sat4j library, release 2.2, system description. Journal on Satisfiability, Boolean Modeling and Computation 7, 59–64 (2010)
2. Bothner, P.: Kawa: compiling dynamic languages to the java vm. In: Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC 1998 (1998)
3. Gusfield, D., Orzach, S.: Handbook on Computational Molecular Biology, vol. 9 (2005)
4. Marques-Silva, J.: Practical applications of boolean satisfiability. In: 9th International Workshop on Discrete Event Systems, WODES 2008, pp. 74–80. IEEE, Los Alamitos (2008)
5. Papadimitriou, C.H.: Computational complexity. Addison Wesley, Reading (1994)

Panda: A Proof Assistant in Natural Deduction for All. A Gentzen Style Proof Assistant for Undergraduate Students

Olivier Gasquet, François Schwarzentruber, and Martin Strecker

IRIT (Institut de Recherche en Informatique de Toulouse)
Université de Toulouse
Toulouse, France

Abstract. We present a proof assistant in Natural Deduction for undergraduate students. The system is interactive: you can combine, delete, modify proofs with a easy-to-use graphical interface. We discuss the pedagogical benefit of this tool.

Keywords: Teaching logic in computer science; Gentzen style natural deduction; interactive provers.

1 Introduction

We, authors of this paper, have been teaching logic to second-year undergraduate students in computer science for several years. We classically present syntax, model theory and proof theory (natural deduction) for both propositional logic and predicate logic. Natural deduction seems to reveal some difficulties among students: rigour and abstraction.

Curiously, while being able of writing rather *rigourously* a simple C program (at least to the point that the compiler is happy with the syntax of their program), often they still cannot rigourously write down a mathematical proof of the kind needed in graph theory, formal logic, abstract data types,... We believe that this may partly be attributed to their mathematical backgrounds: at least in France, mathematics, up to high school, is mainly taught as a science of *numbers and quantities* (relations between integers in arithmetics, between reals or complex numbers in analysis, between matrices of numbers in linear algebra, but always numbers), and not as a science of *structures*, which is a crucial point of view in logic, in computer science and other areas of discrete mathematics. This latter problem seems to us linked to a lack of abstraction capabilities.

Many courses of logic include an introduction to some proof system often based on some variant of natural deduction among the following three main approaches (according to the thorough review of natural deduction systems of

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 85–92, 2011.

[5]): Fitch's diagrams[1], Suppes' annotated proofs[2] and Gentzen's trees. Any of them of course could help students in gaining rigour in reasoning by the *use* of formal systems of deduction. So why should one choose Gentzen's trees as we did since Pelletier, in the same paper, says:

> "The Gentzen tree method did not get used much in elementary logic books with the exception of Curry [...] van Dalen [...] and Goodstein [...] and Bostock [...] In any case, this method of representing natural deduction proofs is not at all common any more."

We want here to argue in favor of Gentzen's tree. We believe that one reason of their less frequent use w.r.t. the other systems is purely technical: they are more difficult to typeset (and hence it is more difficult to implement a friendly interface for them) when compared with Fitch's and Suppes' systems. But it seems to us that Gentzen's style proofs are more readable because the tree-like structure of proofs (which is linked to their inductive construction) is more evident.

Another drawback of other systems is that on the one hand Suppes' style proofs tend to be very long and the structure of the proof tends to be lost, or at least invisible. On the other hand, Fitch's boxes for big proofs involving many subproofs tend to be inconvenient, a nice interface could be difficult to design due to the need of resizing the boxes dynamically, also, the tree-structure tends to disappear with the size of the proof. On the contrary, Gentzen's trees really do justice to the natural tree-structure of proofs, and to us, this is an underestimated feature: it is the best formalism[3] when one wants to point out the fact that *proofs are objects,* and that as such they may be combined and manipulated. It is in this sense that Gentzen's trees train for abstraction: they not only accustom students to the use of logic for proving, but also to the abstract manipulation of proofs as mathematical objects. At least for future graduate computer scientists, this is of high importance. But of course, all these theoretical considerations must be taken into account when it comes to propose an implementation. In our software Panda, students may as well build pieces of proofs, but also manipulate them by drag-and-drop actions, so they can see how easy it is to build a proof of say $A \wedge B$ once they are provided with a proof of A and a proof of B.

We initially looked for tools that could help in teaching natural deduction like *proof assistants*, or at least proof checkers, that would meet our requirements:

- Easy to install and to use;
- Allows one to check a given proof;
- Allows one to build a proof either with some help, or without help;
- Allows both backward and forward reasoning;
- Allows to easily compose proofs from subproofs;
- Uses Gentzen's proof trees style.

[1] That goes back to the "graphical method" of Jaśkowski (1934).

[2] Pelletier argues that the annotated method of the latter is different from that of Suppes, but here we will ignore this small difference.

[3] Of course, here we do not mention Sequents systems, but they are not the most natural formalism.

And we came to develop our own proof assistant PANDA. Nevertheless, we briefly compare PANDA to some existing tools in Section 4.

2 Constructing Proofs in Natural Deduction

Let us first give an example of a typical proof using PANDA and then describe the great degree of freedom to construct and compose proofs.

2.1 Starting a Proof

In its most basic form, the user first enters a formula to be proved by means of the "Add a Formula" button. A few exercises in various degrees of complexity ("Examples level 1 ... 5") are also available in the menu bar. After entering, say, the initial goal $(A \wedge B) \longrightarrow (B \vee A)$, the formula is displayed on the main canvas.

During proof construction, there is always a currently active formula, highlighted by a red frame, to which one of the rules of the calculus of Natural Deduction (or derived rules) can be applied. Applicable rules are displayed in the task bar on the left. The set of rules changes, depending on the syntactic form of the active formula. For example, the rule "And introduction" ($I\wedge$) is not displayed for the initial goal, because it would not be applicable.

2.2 Forward- and Backward Reasoning

Rules can be applied in forward- and backward-chaining mode. In the taskbar, the formula matched against the currently active formula is indicated in boldface font. Let us proceed in pure backwards style and apply the rule ($I \longrightarrow$). The subgoal $B \vee A$ is displayed directly above the original formula. Shaded in light green, there is also the hypothesis that has just been introduced and that can be used later on. By clicking on the appropriate ($I\vee$) rule, we now have reduced the current goal to B. Entering the formula (in this case A, see Figure 1) produces the available hypothesis $A \wedge B$, which automatically concludes the proof.

Proving by backward chaining is implemented in some existing proof editors (see § 4). This style of proof can be cumbersome, due to the missing subformula property of Natural Deduction. Therefore, PANDA allows arbitrary mixture of forward and backward reasoning. Partial proof trees can be constructed, moved across the canvas and easily composed with some mouse gestures. Of course, there is no guarantee that these partial proofs can be eventually extended to a complete proof. However, PANDA keeps track of hypotheses introduced in branches of the proof tree and verifies variable conditions during proof composition, which ensures that finished proofs are sound.

In order to explore some of these possibilities, let us return to the situation in the above proof directly after application of the ($I \longrightarrow$) rule. This is easily done: PANDA allows subtrees or entire regions of the canvas to be deleted, and there is a chronological undo operation that reverts the proof to a previous proof state. Activating the hypothesis $A \wedge B$ (as depicted in Figure 2 on the left) now allows to proceed in several ways:

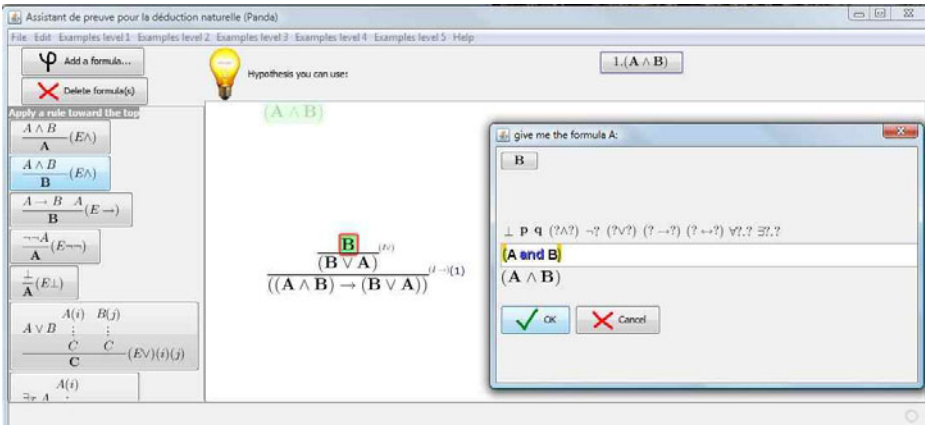


Fig. 1. Simple backward proof in PANDA

- We may select one of the elimination rules ($E\wedge$) in the task bar on the left.
- We may accept one of the formula decompositions that are already suggested by PANDA, for example by clicking on the light green A below $A \wedge B$. PANDA automatically chooses the appropriate rule and applies it.
- We may introduce the desired conclusion of the rule (for example A), using the formula editor, and then connect hypothesis and conclusion of the rule, as described further below.

2.3 Proof Tree Manipulation

In any case, the result is a partial proof tree whose contours are in light red as in the right part of Figure 2. We join the two partial prooftrees by moving the upper tree close to a leaf of the lower proof tree. By using some heuristics, PANDA will recognize that these trees are indeed compatible, by application of an ($I\vee$) rule (this is indicated by the blue line drawn between the trees). By releasing the upper tree at this position, the two trees indeed "snap" together, and the proof is finished.

Rules with several preconditions, such as ($E\vee$) or ($E \longrightarrow$), can be applied in forward chaining style with great ease: It is sufficient to align the conclusions of the subtrees on approximately the same height, and then draw a horizontal line below them, such as in Figure 3. PANDA will recognize the appropriate rule pattern and apply the corresponding rule.

2.4 First-Order Reasoning

At the moment, PANDA only treats standard first-order logic[4]. Concerning rules for handling quantifiers, and particularly $\exists$ we choose the so-called Gentzen's style vs. Quine's style. The former is based on "Elimination of Existential" ($E\exists$)

[4] Some extensions (e.g. with equality) are under development.

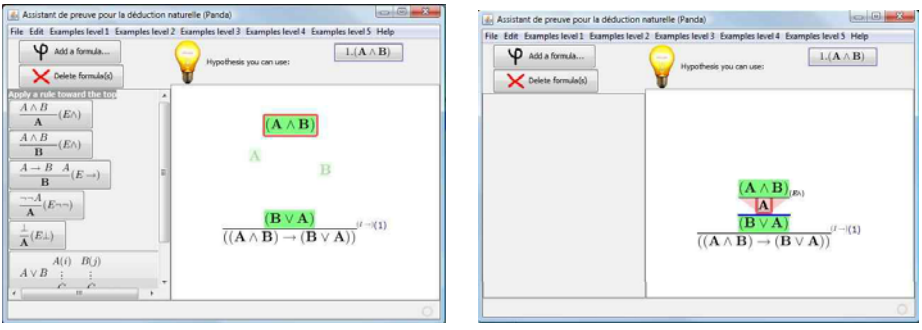


Fig. 2. Forward proofs in PANDA

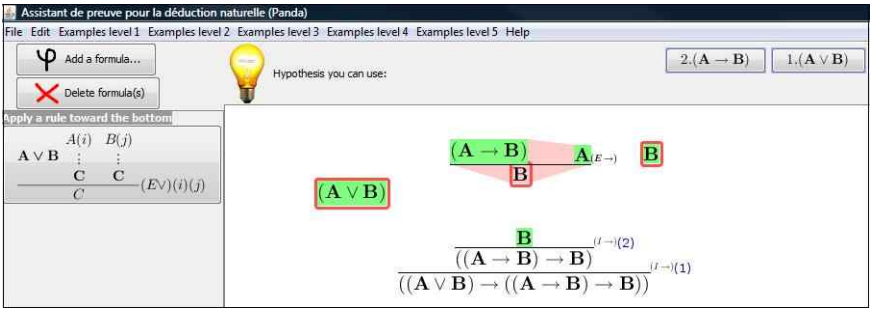


Fig. 3. Applying elimination rules

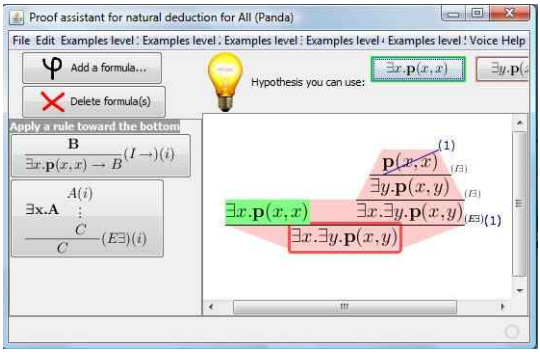


Fig. 4. Applying elimination of ∃

and the second on "Existential Instanciation" (EI). It seems to us that (EI) is always less natural than $(E\exists)$, since either it allows unsound steps in the proofs: from $\exists x\phi(x)$ infer $\phi(a)$, or requires that the a introduced at this step is not a

term, but a *parameter*[5]. We believe that $(E\exists)$ with its subproof process looks more natural: From $\exists x\phi(x)$ and a subproof $(\phi(x)\dots\Psi)$ infer Ψ (with restriction on x and on variables of Φ of course). Thus a proof of $\exists xR(x,x) \rightarrow \exists x\exists yR(x,y)$ in PANDA looks like in Figure 4.

2.5 Saving and Loading

Proofs may be saved in LaTeX format for display in other documents.They may also be saved in our own specific format which is stored as an xml file and may be reloaded for further use.

3 Behind the Scenes

The piece of software Panda is an open-source project written in JAVA for two reasons. The first one is because an object-oriented language is really adapted to create graphical user interface. The second one is for cross-platform compatibility and in order to be launched from the web without any installation.

3.1 The General Graphical User Interface

Abstraction has been adopted in order to be able to generalize easily this interface to other proof systems. The class `ProofFormulaNode` represents an abstract node in a proof tree which can have a father and children. Then the class `ProofFormulaNodeNatDet` inherits from `ProofFormulaNode` and implements the rules of Natural Deduction. Formulas are written in a Lisp/Scheme approach: for instance `(a and (b or (not c)))` stands for $(a \wedge (b \vee \neg c))$. The main advantage is that parsing s-expressions is simple since we use the library JScheme[6]. Formulas in a proof node are then translated in LaTeX code and displayed thanks to the library jLatexMath[7] (it does not require a real LaTeX installation).

3.2 Pattern-Matching to Select Rules

The class `InsertionRuleButton` represents an abstract button enabling to implement an application of a rule of natural deduction. It allows to implement a function `testIfRuleApplicable` for testing if the rule is applicable on the current selected nodes. If a rule is not applicable, the button is invisible. It offers also a function `ruleApply` in order to apply the rule. For instance, the class `InsertionRuleEliminationImplyBU` implements the functions `testIfRuleApplicable` and `ruleApply` for rule $(E \longrightarrow)$:

$$A \qquad A \longrightarrow B \qquad \longrightarrow^{\texttt{ruleApply}} \qquad \frac{A \qquad A \longrightarrow B}{B}$$

The function `testIfRuleApplicable` tests if the selected nodes match with the left-side of the rule. The function `ruleApply` adds the node B and connect it to nodes A and $A \longrightarrow B$.

[5] Also called arbitrary/quasi/pseudo/ambiguous names.
[6] `http://jscheme.sourceforge.net/`
[7] `http://forge.scilab.org/index.php/p/jlatexmath/`

3.3 Cancel and Redo

The interface `Command` provides the method `execute` to execute an action and `undo` to execute the converse of the action. For instance, the class `Command-AddNode` implements the interface `Command`: `execute` adds a node to the proof and `undo` removes this node. This way of thinking makes programming the cancel/redo mechanisms easy: we just save `Command` objects in a stack.

3.4 Load and Save Rules

There are several file formats for saving proofs: Isabelle, Coq etc. Here our needs are different: our proofs are trees with graphical positions etc. So already-known file formats are not adapted to save proofs. That is why we created our own XML file format for saving rules in PANDA.

4 Related Tools

There are many logic tools (provers, proof assistants, truth table editors,...) on the internet, but most of them are either very difficult to install (and not always work), are no more maintained, are not platform independant, are only libraries, etc. Nevertheless there are also suitable tools: PANDORA described in [3] and J∀P∃ [2], which are in Fitch style, and PROOF LAB described in [6] which uses Suppes' style annotated proofs. They are all three very mature and valuable tools, in an advanced state of development including many advanced topics (like elements of set theory for example). Note that the last version of PROOF LAB is not really a proof assistant but rather a tutor, it proposes exercises, but one cannot try to build arbitrary proofs. Their respective style of proof and that of PANDA have their respective merits and drawbacks. In general, the hypotheses available locally at a particular point in the proof can be read off more easily from the nesting structure in the box style, whereas the premises contributing to a conclusion are displayed more clearly in the tree style of PANDA (the premise-conclusion relation is obtained by numbering nodes in J∀P∃). Besides, the free composition of partial proofs, a distinctive feature of PANDA, is barely conceivable (and indeed not supported) in J∀P∃ nor in PROOF LAB and PANDORA, since it would mean introducing a box in the scope of a surrounding box.

Of course, we should say a word about Barwise and Echtemendy's well-known Hyperproof ([1]), which allows students to prove statements about situations involving color, shape and size of geometrical objects. This software, which is only a part of the OpenProof project, has clearly great educational virtues. But it runs only on MacIntosh, it is not free, and is not designed to be used alone but inside the whole courseware package. Unfortunately, the Hyperproof package is out of print. We believe that despite its high qualities, this software has too many drawbacks that limit its diffusion.

A less known tool is BONSAI [4]. The notable differences with PANDA are:

- BONSAI uses a minimalistic logic language, without "or" or "exists"
- Proof construction is bottom-up and does not allow forward reasoning.

– However, it allows different calculi to be selected (such as intuitionistic and classical logic), whereas PANDA has "hard-wired" classical logic.
– In BONSAI, there are facilities for proof search, for generating proof terms (lambda terms) and for proof normalization.

5 Further Work

Firstly, we would like to make it possible for users of PANDA to define their own rules, which can capture commonly occurring patterns of reasoning. For this, one would need an abstract language of proof rules.

Secondly, we would like to increase the automation of PANDA. Proof search is rather easy to conceive in a system that is working in backward reasoning style, and where it is necessary to close open leaves in the proof tree. However, it is more difficult to see how proof search could fill in the gaps between partially constructed proof trees that are disposed on the main canvas of PANDA.

There are extensions that might gradually be introduced, for example support for certain theories (equality, arithmetic, geometry) or a framework for reasoning about programs, in the style of a Hoare logic.

However, there are other topics that will most likely not be integrated into PANDA. For example, proof terms and normalization presuppose notions of lambda calculus that our students have not acquired at this point of their studies.

To conclude, we think that the PANDA experience was interesting for us, and helped our students improve their understanding of logic reasoning. We will especially point out that one disabled student who cannot write with a pencil, PANDA was the only reasonable means of interaction. We invite the reader to try out PANDA: an interactive and multilingual[8] version is available from `http://www.irit.fr/panda`

Acknowledgements. We would like to thank Elsy Kaddoum who has implemented a prototype of PANDA during a student internship.

References

1. Barwise, J., Etchemendy, J.: Hyperproof. CSLI Publications, Standford (1994)
2. Bornat, R.: Natural Deduction Proof and Disproof in Jape (March 2004), `http://jape.comlab.ox.ac.uk:8080/jape/`
3. Broda, K., Ma, J., Sinnadurai, G., Summers, A.: Pandora: A reasoning toolbox using natural deduction style. Logic Journal of the IGPL 15(4), 293–304 (2007)
4. Deiser, O.: Bonsai Manual (April 2004), `http://www.aleph1.info/bonsai/bonsai.html`
5. Pelletier, F.J.: A brief history of natural deduction. History and Philosophy of Logic 20(1), 1–31 (1999)
6. Sieg, W.: The AProS project: Strategic thinking & computational logic. Logic Journal of the IGPL 15(4), 359–368 (2007)

[8] At the moment only English and French are available.

The Question of the Question in Critical Thinking?

Roderic A. Girle

Philosophy Department, University of Auckland, Auckland, New Zealand
r.girle@auckland.ac.nz

Abstract. The question of the question in Critical Thinking is the same as in logic at large. The basis on which Critical Thinking is taught is usually the logic of deductive premise-conclusion arguments. The basic elements of the logic are propositional. This approach gives no basis for teaching students to think logically about either questions or commands. At the same time questions and commands are an important part of practical life. In either asking or responding to questions it is easy to commit any of several illogical moves. Although the same applies to commands, this paper will focus on questions. We raise the question of coherent systems in which questions occupy a proper place, systems which could form an appropriate theoretical background for teaching Critical Thinking and propose dialogue logic.

Keywords: Critical Thinking, questions, commands, logic, dialogue.

1 Introduction

I begin with three anecdotes. The first is about a book I saw in a tray of cheap books outside a magazine shop. The title was *Do You Think What You Think You Think?*[1] Although it's not as catching as *What is the name of this book ?*, it's still a Philosophers' trap. I glanced, bought, took home and started reading. The very first thing is that it's a book of questionnaires. It tries to establish how consistent your belief systems are. But, the first thing that struck me was how defective some of the questions are. The **very first** 'question' is, "There are no moral standards; moral judgements are merely an expression of the values of particular cultures. Agree/Disagree." Just note two things: first that no decent options are given. I mean the real question part of this is just "Agree or disagree." There is not even "Agree, disagree, or don't know." The second thing to note is that although Philosophers might say, "Oh, **we** know what it's getting at," that's not good enough. This book is addressed to the popular audience, and should not be posing such a poorly framed question. Many questions were reasonable, but too many were not good.

The second anecdote concerns a survey which was being conducted at the University of Queensland one morning. I used to drive about 30 km to University. As people drove their cars into the University they were being slowed down and the numbers of people in each vehicle were being noted on clipboards. I was curious enough to stop and ask what was going on. I was told that there was a survey to find out how many cars were arriving with no passengers so that the University could start a campaign to

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 93–100, 2011.

increase car pooling. I said, "I left home with a car full of people, one I dropped off at the Primary school just down the road, and the other two at the High School, also just down the road. Do you want to know about that?" Swiftly the answer came back, "No, not interested." "Do you want to know how far I have come?" "No." I felt like asking, "What if everyone is giving someone a lift to some intermediate point?" Defective and quite incomplete data was being collected. The wrong questions were being answered. The responses were being used as a basis for decisions about spending University funds.

The third anecdote is not an anecdote, but is just to remind ourselves that at least from the time of Aristotle logicians have been aware of the *fallacy of many questions*. "Have you stopped beating your grandmother?"

These anecdotes highlight some of the many logical problems that emerge from the use of questions in human interaction. In each of these cases we might advise people to think carefully before responding to the questions, indeed, to think carefully *and logically* before responding to these questions. But, apart from also recommending apple pie and ice-cream, on what basis are we to found such logical thought about questions and responses? How are people to proceed?

In this paper we will first look at the theoretical background and educational aims that provide the basis for teaching Critical Thinking. We consider some of actual assumptions at work, and provide an analysis and critique. Special consideration will be given to the place of questions in this twin pillar foundation or lack of foundation. Second, we consider a coherent basis for teaching Critical Thinking, especially a basis that borrows from both argumentation theory and multi-agent systems. We consider the basic elements of which a broader logic would be constructed and the way in which these might set directions for Critical Thinking. Thirdly, some standard objections and obsessions will be considered.

2 Assumptions and Processes

I would like to assume that Critical Thinking is taught against some theoretical background, and that the educational aims for teaching the subject are reasonably clear. But both of these assumptions are fraught with danger because I am sure that in many cases Critical Thinking is taught without any clear ideas in either area. After talking to lots of people who teach Critical Thinking and looking at endless text books both academic and popular, I suspect that tradition drives much of the teaching of Critical Thinking. People simply teach the course "we've always taught." A second driver is give students some ideas that might encourage them to do 'real logic'. In those Philosophy curricula where there are no follow on courses from Critical Thinking, one might suspect that academics see 'real logic' as the follow on. A third driver is to be found in communication studies relevant to professional training. For example, if one opens Business Communication texts one often finds a chapter on Critical Thinking or Clear Thinking.[1]

[1] Chapter 10, especially a section headed **Thinking**, in [2] Judith Dwyer, *The Business Communication Handbook* page 240.

Traditional Critical Thinking

The content of the traditional Critical Thinking course usually consists of three parts. The first part is deductive logic instantiated as modern diagrammatic syllogistic. This might be followed by some classical truth-table logic or some natural deduction logic. The second part is inductive logic instantiated as Mill's Methods, or some propositional probability calculus to help students play cards or roll dice. There might be other inductive logic. The third part is the fallacies. This is instantiated in just the way Hamblin described it in *Fallacies* [5] as *The Standard Teatment*.

The emphasis is totally and completely on premise-conclusion argumentation. As one academic said in a discussion about this, "It's about the evidential relationship between premises and conclusions." The underlying theory is a theory about the relationships between propositions (or statements). Special emphasis is on the evidential relationship between sets of propositions (premises) and a single proposition (conclusion).

As far as questions are concerned, they are almost entirely ignored except for the questions which commit the *fallacy of many questions* or the *fallacy of the loaded question*. And even here, the emphasis will be on the propositional content and there will be all sorts of efforts to turn the question into a premise-conclusion argument.[2] This is all in spite of there being a considerable literature on *erotetic logic*, the logic of questions.

The educational aim in teaching Critical Thinking usually centres around making students aware of the argumentative content of everyday life. And getting them to think rationally about the way in which people argue. But some texts will immediately insist that arguments aren't arguments. Argumentative interaction is usually dismissed with some sort of stipulation such as, "For our purposes arguments are not verbal disputes." One might well want to ask the question, in interactive response to such a stipulation, "Then who is to teach students about what might be involved in thinking rationally during or reflectively after verbal disputes about not only statements, but about questions and commands." Indeed, what might be involved in thinking critically or rationally about questions and commands in all sorts of contexts?

Real Logic

A second driver is the motivation of getting people into "real" logic classes, that is, into formal logic. A typical curriculum scheme is to teach Critical Thinking at first year level with formal logic being taught at second and third year levels. The formal logic which follows on from Critical Thinking is usually deductive logic and more often than not there is no inductive logic and no probability logic and nothing about fallacies. This approach can be called the "bridge" approach. Critical Thinking is a bridge to formal logic.

These courses will almost always have a content similar to the courses based on Traditional Critical Thinking teaching. But there will be lots of propaganda encouraging students to go further into formal logic. People who teach this kind of Critical Thinking course usually know nothing about Argumentation Theory, nor that there is

[2] And by the way, there will be nothing at all about commands. Even more so than questions, commands are apparently far beyond the reaches of Critical Thinking.

such an area of study and research, and hold strictly to a totally propositional approach to Critical Thinking.

Sometimes, in discussion about the reasons for teaching Critical Thinking the proponents of the bridge approach will reject any idea that Critical Thinking has to do with thinking rationally or developing reasoning skills. Both of these activities are seen to be outwith the remit of any kind of logic course, and Critical Thinking is 'baby' logic. The rejection of any skills often comes from an anti-psychologistic approach to logic. But the strange thing is that in many formal logic courses students have to develop skills in filling-in truth tables or diagrams, constructing proofs or truth-trees, or in translating from natural language to artificial languages such as first-order predicate logic.

There is a confusion here which needs to be sorted out, but that is not the main aim of this paper.

Communication and Professional Training

Clear Thinking or Critical Thinking are often mentioned when Professional Training is discussed. Critical Thinking courses are the lifeblood of many Philosophy Departments because students in Law, Business, Engineering, Information Technology and Education are other urged to take a Critical Thinking option as part of professional training. Critical Thinking can be wedded to or become part of Business Communication courses.

If one is interested in taking a broader look at what might motivate the teaching of Critical Thinking, then texts in business communication can be very suggestive. One reason for their suggestiveness is that they do discuss topics such as surveys and questionnaires (questions), and setting out clear procedures (commands). There is also no ideological bias against skills.

Summary

The traditional and bridge motivations for teaching Critical Thinking and their advocacy for a logic basis for course content are highly restrictive. At the most basic level of content they deal almost exclusively with propositions, and neglect questions and commands. For them, the question does not exist in Critical Thinking because it not part of standard proposition focussed logic. But that focus leaves us with a vastly impoverished basis for dealing with thinking rationally about the variety of things expressed in everyday communication.

3 A Broader Coherent Basis

Although there is a considerable literature on erotetic logic and a vast literature about logic based solely on propositional elements, there seems to be little in the way of an overall systematic approach to integrating logic based solely on propositional elements with logics for questions and commands. The obsession with premise-conclusion argumentation provides a Procrustean bed into which everything has to be forced, and if a limb of logic won't fit it's cut off and thrown away. This rejection is often done with an aggressive tirade against anything that does not fit the narrow preconcieved idea of logic.

It is clear that a basis for sensible integration can be found in formal dialectic or dialogue logic. This must not be confused with discourse theory nor with the question-answer approach to semantic tableau. Modern dialogue logic was set in motion by C. L. Hamblin [5] in Australia and by E. M. Barth [2] in the Netherlands. Each seems not to have known of the other person's work. My focus will be on the work of Hamblin and his student Jim Mackenzie and those who have followed their lead. Many of those who have followed in the footsteps of Barth have made *speech acts* the basic elements of dialogue systems. Hamblin and Mackenzie [8] have emphasised *utterances and their content*. The emphasis for Hamblin and Mackenzie has been on propositions and questions. Latterly there has been some work on commands within this context.

Hamblin Mackenzie Dialogue systems (HMD systems), as they are often called, give an account of the sequence of moves that dialogue participants make as they discuss and debate issues. The sequence of moves is a sequence of propositions, questions and commands. Hamblin was concerned to divide the sequences into two categories so that dialogue could be evaluated as rational or non-rational. In premise-conclusion logic the evaluative categories are *valid* and *invalid*, *sound* and *unsound*, *strong* and *weak*. Hamblin introduced the terminology of *legal* and *illegal* dialogues. Some authors see these categories as something like the categories in games where a referee has the task of enforcing rules, and some moves or plays are be illegal. In dialogue, for example, asking a loaded question is almost certainly best seen as illegal rather than fallacious, unless one extends the notion of a fallacy to include illegal moves in a dialogue.

A legal dialogue is a dialogue governed by a set of rules which, like game rules, require certain sequences, forbid certain sequences, and allow certain sequences. For example, some moves must be followed by particular moves. Early writings by both Walton [9] and Hintikka [7] referred to *dialogue games*. This is just like the game of rugby league where, if a player carrying the ball is tackled and held down firmly that player must then "play the ball" as the next move. Similarly, questions should be followed by responses which answer the question, not evade or mislead. Commands should be followed by a response which shows that the person commanded either accepts or rejects the command. In very formal command situations such as the bridge of a ship, when a command is given the person commanded repeats the command to show that they have received it, and that they either accept it or cannot follow it.

Dialogues are not all of the same kind. Walton and Krabbe [10] divide dialogues into Persuasion, Negotiation, Inquiry, Deliberation, Information Seeking, and Eristic (or Abusive). To these has been added Instructional or Command Dialogue [4]. Although some dialogues might not be of great interest to many of today's logicians, one has to be careful about this matter because some of the traditional fallacies might best be explained in terms of a slide from one kind of dialogue to another. For example, the *ad hominem* fallacy can be seen as a slide from persuasion to abuse. The *ad baculum* fallacy can be seen as a slide from persuasion to negotiation. If negotiation and persuasion are taken to be distinct, then it is a serious mistake to think that negotiation is essentially a matter of persuading people to all believe the same thing. An analysis of negotiation would almost certainly introduce an additional basic element - promises.

Questions play a leading role in all dialogue forms. And besides, there is a vast literature about the logic of questions and there are many papers about the kinds of questions. There is a quick summary to be found in Wiśniewski [11]. My preference is with Wiśniewski for a non-reductionist account of questions. Several kinds of questions are featured in HMD systems. There are questions which ask for reasons for statements or beliefs, questions which ask whether a proposition is true or false, wh-questions such as "Who is the person in the kilt?" or "When did that happen?" or "What is your offer?", method questions such as "How do you take off the wheel?", goal questions such as "What is the aim of doing C?", and epistemic questions such as "How do you *know* that P?" Some of these might be grouped together, especially under the wh-question grouping.

My proposal for seeing Critical Thinking against a dialogue logic background so that questions can properly be dealt with is not a proposal made in a vacuum. But it not a proposal for seeing Critical Thinking against a chaotic background, no matter how rich. There are few systems which bring together so many elements in a systematic way. The only other that I know of is to see the consequence relation as the glue. Both Wiśniewski, and Gabbay and Woods rely heavily on the consequence relation. But I suggest that this is too narrow. It does not cope well with the notion of *agent interaction*, a notion which is central to dialogue logic, and a notion central to understanding the practical situations with which the critical reasoner might be expected to cope.

4 Objections and Responses

There are fairly standard objections from logicians to departing from premise-conclusion argumentation. One is that truth is central, and that only logic with the bearers of truth-value at the centre can be seriously considered for logic. There are at least two problems with this approach. The first is that it is highly question-begging. It is the ultimate irony that philosophers often tell us that it's not answers (propositions) that matter, but questions. This might be dismissed as a kind of hyperbole, but why is it so seriously repeated? Perhaps what is meant is that it is not conclusions which really count, especially because there are so few which have broad acceptance, but that it's the rational discussion of topics that's central to Philosophy. In that case, one would think that a broad coherent account of discussion and dialogue including questions would be central to giving an account of the methodology of Philosophy.

A related serious issue is that even if we were getting students to look carefully at only premise-conclusion argumentation, there are those who have argued that truth need not be central. It has been argued that the focus on truth has led to a misrepresentation and misunderstanding of practical rational argumentation. In discussing everyday argumentation Gabby and Woods point out that while "Guaranteed truth preservation is a good way of avoiding error … individual agents are not in the general case dedicated to error-avoidance." ([4] pg. 42) Error-avoidance might not be a top priority in rational dialogue or deliberation where reasonable approximation and plausibility are mandated by reasoning under resource constraints. If this is so, then objections based on the claim that truth is central are less forceful than might first appear.

The second objection comes from those who see logic and Critical Thinking as applying almost exclusively to persuasion dialogue, and while they concede that the persuasion dialogue context is a neat way to approach "real" logic, they often allege that it's just cosmetic. They assert that eventually we have to get down to the real issues of the relationships between propositions. "Argument is all about persuasion, and persuasion is all about premise-conclusion argumentation." In other words, we have the question begging claim that the most important issues by far have to do with premise-conclusion argumentation, and the rest is nowhere nearly as important.

My response would be twofold. First, argument might be about which question of a range of questions is either the most important or the most relevant. Argument might be about which response to a question is an answer, and about which answer, if any, is the correct one, and about whether the correct answer will be a proposition. Second, that "the rest" is just as important in the context of what we want to accomplish in any Critical Thinking course. I would repeat that encouraging students to think rationally about at least the four basic elements of dialogues should be the aim, not just about propositions in premise-conclusion arguments. Students should be aware of the basic elements of discuss such as: propositions, questions, commands and promises. They should have a framework into which they can fit these elements and see when they form a reasonable sequence and when not in daily discussion and debate. University students meet all of the four in their classes, especially tutorials, and in discussion and debate with other students. Critical Thinking can find application in their everyday academic life if it is broad based. But if Critical Thinking is narrow in scope it will have little obvious application, and in the meantime questions, commands and promises will not be seen as matters for careful reflection.

A third objection is not so much to the issues I have raised above about a broad context for Critical Thinking, but that dialogue logic might not be the best framework. One would expect that any framework suggested here would find great use in multi-agent logic. One of the contenders in that area is dynamic epistemic logic. Would this not be a better theoretic basis? This is an interesting question because one of the features of dialogue systems is the dynamic building of *commitment stores* during dialogue. This involves elements of belief change and a dynamic epistemic logic would be the sort of system needed to oversee belief change. In this case dynamic epistemic logic would be subsumed into a dialogue logic. But commitment stores are not only stores of changing beliefs. They also incude questions not yet satisfactorily answered, commands in process of being dealt with, and promises yet to be met. So, the answer to the question would be that dynamic epistemic logic would be most useful, but as part of the theoretical basis, not all of it.

5 Conclusion

Standard logic can certainly provide a framework for understanding when sets of propositions are consistent and when not. Most of us know what that involves and what flows from it. Except for a very few of us, when we turn to the other basic elements of everyday rational communication deserving careful thought, most of us face elements for which we little theory or none. We just operate in an *ad hoc* manner, the very sort of thing Hamblin complained of half a century ago. In the end it all depends on how much attention we are willing to apply to basing the teaching of Critical Thinking on a broad systematic foundation.

References

1. Baggini, J., Strangroom, J.: Do You Think What You Think You Think? Granta Books, London (2006)
2. Barth, E.M.: A Normative Pragmatical Foundation of the Rules of Some Systems of Formal Dialectics. In: Barth, E.M., Martens, J.L. (eds.) Argumentation. John Benjamins, Amsterdam (1982)
3. Dwyer, J.: The Business Communication Handbook, 3rd edn. Prentice Hall, New York (1993)
4. Gabbay, D., Woods, J.: Agenda Relevance. North Holland, Amsterdam (2003)
5. Girle, R.A.: Commands in Dialogue Logic. In: Gabbay, D., Ohlbach, H.J. (eds.) Practical Reasoning: International Conference on Formal and Applied Practical Reasoning, Bonn, Germany, pp. 246–260 (June 1996)
6. Hamblin, C.L.: Fallacies, Methuen (1970)
7. Hintikka, J.J.K.: On games, questions and strange quantifiers. In: Pauli, J. (ed.) Philosophical Essays Dedicated to Lennart Åqvist, pp. 159–169 (1982)
8. Mackenzie, J.: Four Dialogue Systems. Studia Logica XLIX, 567–583 (1990)
9. Walton, D.N.: Logical Dialogue-games and Fallacies. University Press of America, Lanham (1984)
10. Walton, D.N., Krabbe, E.C.W.: Commitment in Dialogue. State University of NY Press (1995)
11. Wiśniewski, A.: The Posing of Questions. Kluwer, Dordrecht (1995)

Adding a Dimension to Logic Diagramming

Laurence Goldstein

School of European Culture and Languages, University of Kent
Canterbury, Kent CT2 7NF, England
L.Goldstein@kent.ac.uk

Abstract. A blind learner needs some method other than Venn diagrams to test syllogisms for validity. I present here a sketch of a three-dimensional apparatus, Sylloid, invented to fill this need and to inculcate deep learning rather than the mere ability to get answers right. What one learns in the design process is then used in designing a successor, Son of Sylloid, for sighted users that is pedagogically superior to Venn diagrams. This dog-legged approach to materials design is of wide application: first design apparatus for a user, real or imagined, weak in one or more of Howard Gardner's 'intelligences', then create a successor design for the non-deficient.

Keywords: Gardner intelligences, Venn syllogism, blind.

1 Introduction: Gardner-Inspired Design of Teaching Materials

The aim of this paper is to fundamentally re-think the design of teaching materials in the light of what is now known about cognitive deficits and about what the psychologist Howard Gardner has termed 'multiple intelligences', and to construct more effective, more attractive teaching materials as a result. What emerges is a recipe for innovatory design that promotes deep learning, and engages the different 'intelligences' of the learner. My case study here is the design of a sequence of aids for the learning of syllogistic logic.

The springboard for the present project was a request to construct specialized equipment for teaching elementary logic to blind students. The logic in question was the theory of syllogisms, and the standard method for testing syllogistic arguments for validity is to use diagrams named after their inventor, the nineteenth century logician John Venn. Obviously, the Venn-diagrammatic technique is unavailable to a blind student — someone who, in Gardnerian terms is deficient in the visual dimension of spatial intelligence — so a substitute apparatus needs to be invented that taps into a different dimension of intelligence of the blind person . And then the question arises as to whether this apparatus, or some successor of it that re-engages with the visual, ill provide a richer learning experience for non-impaired users. If it does then one can reasonably expect this design process to generalize to other areas of learning.

Let us elaborate just a little. The problem of designing teaching equipment tailored to the needs of a group of individuals whose access to certain modes of learning is restricted e.g. because of impairment or weakness in one or other 'intelligence', ought to confront designers with the challenge fundamentally and creatively to rethink

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 101–108, 2011.

questions such as 'What is the real nature of what is being taught (and why is it important, if it is, that it be taught at all)?' and 'What type of learning experience will best promote real, deep understanding of the subject matter?'. Such questions, when addressed seriously, inform the design of the new equipment and so benefit the target users. But the pedagogically important consideration is this: If the piece of equipment is a sophisticated solution to educational questions that have not previously been raised, then it will supply a superior means of learning not just for the group for whom it was designed, but for all students. And this will be true right through the age spectrum. The equipment will, however, typically need to be reconfigured so that a learner who is not restricted is not disadvantaged. For example, if braille letters are used in a device designed for blind subjects, then the braille would be replaced by standard letters or some other visible substitute in versions of the device to be used by the non-blind. And the reconfigured apparatus could also, for example, make effective use of colour. In summary, the dog-legged design process is this:

1. Identify some part of the syllabus that is taught by some traditional means, e.g. by book learning, where you feel the traditional teaching methods to be stodgy or ineffective.
2. Construct learning material X (it may be a piece of apparatus, a competitive or collaborative activity for two or more students, an interactive computer game, etc.) for a target group of students that suffers some real or imaginary cognitive deficit. The use of X will engage a range of intelligences different from that invoked by the traditional teaching method.
3. Construct a new apparatus, son-of-X that preserves all the pedagogical advantages of X but which also features elements that enhance the learning experience of students who do not suffer the cognitive deficit mentioned in 2.
4. Test the effectiveness of the new apparatus against traditional methods of teaching. Effectiveness is measured not just by the speed at which the student solves various problems, but by the depth of the knowledge imparted. (Testing for depth of learning is a by no means trivial task.)

The area of learning under review is syllogistic logic but it must be emphasized that this is for illustrative purposes only. The dog-legged design process, if effective, can be employed in any area of teaching to deliver deep learning.

2 Design for the Blind

In this section, I shall recount the evolution of the design of some equipment for teaching elementary logic. Logic is a subject that features in almost every tertiary philosophy curriculum. Syllogisms are a particularly simple type of argument originally investigated by Aristotle, and the theory of syllogisms was the core of logic right up to the last quarter of the nineteenth century. It is still regarded as an important and rather beautiful part of logic, and in recent times has been developed, most notably by Fred Sommers. A syllogism consists of two premises and a conclusion, with three noun phrases (or 'terms') each occurring twice over, and each sentence in the syllogism has to be of one of just four allowable types. In the following example, the first premise is of type I, the second premise is of type E and the conclusion is of type

O. Type A sentences are of the form ‘All Xs are Ys’, for example ‘All ducks are elephants’.

Some tax cheats are parliamentarians.
No blue-eyed people are tax cheats.
Therefore
Some parliamentarians are not blue-eyed.

This particular example elicits conflicting verdicts from people asked to say whether it is valid or invalid. This alone shows the usefulness of an objective method for determining the correct answer. The validity or invalidity of any syllogism is usually established via algebraic (Boolean) equations or graphically via Venn or Euler diagrams and make use of the notion of a set (or class), e.g. the set of parliamentarians corresponding to the common noun ‘parliamentarians’. Neither technique is readily available to the visually impaired student. It is easy enough to design software such that a visually impaired student could input a coding of the premises and the conclusion of a syllogism, hit a button and receive instantaneously a correct verdict on that argument’s validity. But, of course, the intellectual/educational value to a user of such a device would be close to zero. Hence the need, identified above, to raise the question of just what it is about the nature of syllogisms that makes them a fascinating object of study, questions about the nature of classes (sets) and the relations between them, questions about what it is to inculcate an understanding of these and of entailment and validity. Such questions are important if one cares about deep learning.

A Venn diagram consists of three intersecting circles, each (if your imagination is sufficiently vivid) to be thought of as containing all of the objects, if any, corresponding to a noun-phrase occurring in the syllogism (e.g. if we were representing the above argument Venn-diagrammatically, one circle would ‘contain’ all the tax cheats, another all the parliamentarians so that the intersection of these circles contains all the parliamentarians who are tax cheats, if there are indeed any. With three circles intersecting, there are seven distinct areas. There are two basic operations when representing premises on a Venn diagram: (i) shading areas to show that they are empty (contain no objects) and (ii) using a heavy short line (a ‘bar’) to show the presence of objects where the bar lies. (Interestingly, the invention of the bar was due not to Venn but to Charles Sanders Peirce.) Venn’s two dimensional diagrams are a much simpler way of testing syllogisms for validity than any of the earlier methods. But why not go up a dimension and produce a physical, highly tactile three dimensional model? The apparatus (called ‘Sylloid’) designed by the author for the use of blind students exchanges the seven areas of a Venn diagram for seven solid tetrahedra, and the counterpart to shading an area (Venn) is to remove a tetrahedron from the core by pulling it off. The counterpart to drawing a bar between two areas (Venn) is slapping a hinge in the valley between two tetrahedra. If, in the course of representing another premise, one of the tetrahedra on which the hinge is resting is removed, the hinge is folded back, revealing a differently textured surface and remains attached to the tetrahedron that has not been removed. This corresponds, in a regular Venn diagram, to part of a bar being eclipsed when one of the areas in which it lies is declared empty (so the objects that the bar represents as existing must lie in the area where the bar remains, uneclipsed). Blind students who have used this apparatus get the hang of it remarkably quickly.

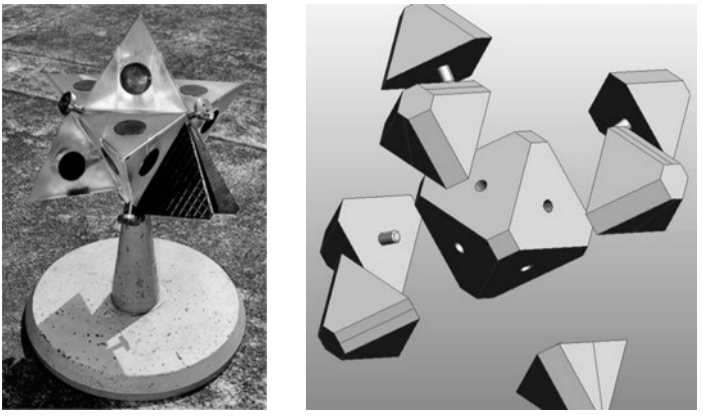


Fig. 1. SYLLOID Note that, just visible in the picture, the steel buttons are embossed with the braille equivalent of 'X', 'Y' and 'Z'. Note also the textured metal hinge. The device consists essentially of seven tetrahedra that plug into a central core – see the computer deconstruction on the right. Small round magnets are embedded in the exposed faces of the tetrahedra. Sylloid is supplied with an audiotape containing instructions for use.

3 Sight Restored

The task of designing an apparatus for the sight–impaired presented the opportunity not just to produce a device by means of which blind people can test syllogisms for validity or invalidity but to think about what a deep understanding of syllogism involves, and to ensure that it is this kind of understanding that will be delivered to the user working with the apparatus. Sylloid was a response to this opportunity, and among its pedagogical advantages are the following:

1. A solid tetrahedron is a slightly more intuitive representation of a set of objects than is a two-dimensional shape.
2. Each tetrahedron in Sylloid visibly represents a discrete set. In Venn, the intersection (lens) between sets is clearly depicted on the diagram, but the lens has two parts and it is by no means clear what set each part represents. (It is a pedagogically useful, exercise to explain to students using Venn just which sets each of the seven areas represents.)
3. Physically removing a tetrahedron from Sylloid is a much more natural and attractive way of demonstrating the absence of the relevant set of objects than is shading an area in Venn.
4. The smooth side of the hinge represents the possibility of the presence of objects, the textured side (revealed when one side of the hinge is folded back on the other) represents their actual presence. Neither Venn nor Son of Sylloid sport any counterpart to this useful feature.
5. There is a strong element of play in Sylloid — pulling out blocks, slapping on hinges etc; Venn is not quite so much fun.
6. Sylloid is a beautifully crafted object; one can thus take advantage of the Thorndike effect.

An intriguing possibility now presents itself. Because of the advantages just listed, and because of the fundamental re-thinking that went into its design, Sylloid is probably a better learning tool than Venn, and, if we modified it slightly (e.g. by replacing the brailled letters with regular letters) then it could be used to advantage by sighted students. But why not go one step further and produce a radically new design for the sighted, a Son of Sylloid, that incorporates all the virtues of Sylloid and of Venn and that makes maximum use of the visual sense? A useful first step in this process is to take a hard look at the defects of Sylloid to ensure that they are not transmitted to its heir. These defects are:

1. It is difficult to get a firm grip on the sloping sides of a tetrahedron made of perspex, especially with clammy hands. Dropping one of the pieces on the floor is obviously a nightmare for a blind user. (This hazard would have been apparent, at the outset, to a competent designer, but was not to yours truly.)
2. There is no representation of class intersection in Sylloid. This was pointed out to me by Jon Williamson at a workshop, and it is a major strike against Sylloid, since part of the deep learning of syllogistic is understanding the connection (which so excited George Boole, when he discovered it) between the four types of Aristotelian sentence and their counterpart class relations, that can be captured in algebraic equations.
3. The tetrahedra are of equal size, and this may create the false impression that the classes they represent are equinumerous.
4. This piece of equipment, the prototype of which was constructed at the Ho Tung Engineering workshop, University of Hong Kong, needs to be built to fine tolerances and is therefore expensive to produce.
5. It is also very heavy since it has not to topple over when in use. (An alternative would be to screw it down to the workbench.)

Son of Sylloid, exploits colour, and the user represents the premises of a syllogism by removing bits of the jigsaw (corresponding to shading the area in Venn) or shows existence by using a bridging piece (corresponding to the bar in Venn). When a the

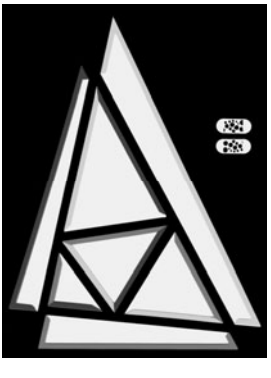


Fig. 2. SON OF SYLLOID Each of the seven coloured pieces can be removed from the black housing, though typically, when representing the premise of a syllogism, only one or two pieces are removed. The bridging bars (top right) are used when representing existential premises.

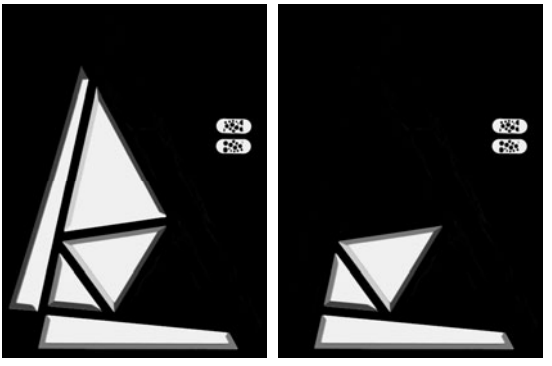


Fig. 3. All Y(ellow)s are B(lue)s FOLLOWED BY All B(lue)s are R(ed)s

bridging piece straddles two areas then, if one of those areas is declared empty in the next premise and the corresponding piece removed, the bar moves to the remaining area. This illustrates testing for validity the syllogism

All humans are mortals
All mortals are arachnophobes
Therefore all humans are arachnophobes.

Use the variables 'Y', 'B' and 'R' as stand-ins for, respectively, the nouns 'human', 'mortal' and 'arachnophobe'. The form of the first premise is 'All Ys are Bs'. Read this 'All Yellows are Blues', then simply remove from the apparatus each piece that has a yellow edge but no blue edge. For the second premise, of the form 'All Bs are Rs', remove any of the remaining pieces that have a blue edge but no red edge. By inspection of the mutilated apparatus that survives, one notes that the only piece with a yellow edge also has a red edge, hence the conclusion 'All Ys are Rs'. In the course of representing the two premises, all those pieces with yellow edges but no red edge were removed. In representing the premises, we eo ipso represented the conclusion, hence the argument is valid. Somewhat metaphorically, a deductively valid argument is often characterised as an argument the conclusion of which is contained in the premises. This notion of containment is made vivid in all three methods of testing described in this paper, but not in Aristotle's original deductions nor their mediaeval refinements nor in Boolean algebra. The invalidity of a syllogistic argument can be immediately read off a diagrammatic representation, for the conclusion is visibly not contained in the premises. Note, for example, that the conclusion 'All arachnophobes are humans' (All Rs are Ys) does NOT follow from the original two premises, for, in its final configuration (right hand figure, above), the mutilated apparatus contains red edged pieces that are not also edged in yellow.

I leave it as an exercise for the reader to imagine how testing our tax-cheating parliamentarians example above would be performed using Son of Sylloid..

4 Advantages of Son of Sylloid over Both Sylloid and Venn

1. The apparatus is visually attractive (Piet Mondriaan) and its operation makes essential use of colour.

2. The three main classes and the seven subclasses are represented by pieces of different shapes and sizes, reflecting the differences between the associated classes.
3. As in Sylloid, the emptiness of a class is signalled by physically removing the relevant piece, but this operation is easy, since hollows have been gouged out to create space for prising fingertips. Also, the operation of showing the presence of objects by the use of a bridging bar, is dead simple.
4. Coloured ridges round the sides of the 'jigsaw' pieces in Son of Sylloid give an immediate indication of the class represented by that piece. Thus a piece edged only in yellow indicates the class that contains objects, if any, that are just Y (i.e. they are not also R or also B); a piece edged in red, yellow and blue indicates a class of objects that are R, Y and B.
5. The jigsaw pieces are not contiguous; the world of parliamentarians and tax cheats and blue-eyed people also contains goats, planets, prime numbers etc..
6. When a bridging bar traverses two pieces, the subsequent removal of one of those pieces removes the support for one side of the bridge and there is just one place for it to go – onto the remaining piece. This is a more natural operation than the eclipsing of a bar by a shaded area, as in Venn.
7. When representing in Venn a type A sentence of the form 'All Xs are Ys', one first has to mentally paraphrase this as 'There is nothing that is X that is not Y' and, accordingly shade as empty the area of the X-circle lying outside the Y-circle. A lot of students find this mental manipulation difficult, and get it wrong. With Son of Sylloid, no such mental manipulation is required (see the preceding description of representing a type A sentence). Arguably, understanding why 'All Xs are Ys' is equivalent to 'There is nothing that is X that is not Y' is part of deep learning and would need to be taught as an extra to users of Son of Sylloid.
8. Tthe process of testing in Son of Sylloid is quicker, easier and more fun than in either Venn or Sylloid, BUT it is so only once you have got used to it. In his original thinking, the author badly underestimated the time it would take a student new to the subject to learn about syllogisms and how to test them for validity. It was simply unrealistic to suppose that this could be done in one hour with no prior knowledge of Venn diagrams.
9. Son of Sylloid, like Venn but unlike Sylloid, visibly represents class intersection. In Venn, the intersecting circles need to be labelled; in Son of Sylloid, intersecting quadrilaterals are identified by the colours of their edges.
10. Unlike Venn, Son of Sylloid is re-usable and is cheap and easy to produce.

5 Conclusion

The idea of constructing apparatus for reasoning is not new. A distinguished precursor is Raimundus Lullus, or Ramon Llull (ca. 1232-1316). Lull's chief invention was a so-called Ars Magna of encoded, inter-rotating wheels developed in the latter decades of the thirteenth century and articulated in a treatise called the Ars Generalis Ultima'. See [1]. The idea of switching from traditional media has also been anticipated by Barwise and Etchemendy in their logic software 'Turing's World' and 'Tarski's World', which makes use of 3-D graphic techniques for inference. They write: '[T]here is no principled distinction between inference formalisms that use text and

those that use diagrams.' [2], p.214. True, but are diagrams or solid models better as learning devices? The design methodology outlined here is applicable to the construction of all kinds of teaching material for students of all ages, and is, at root just a way of calculatingly tapping into the rich range of intelligences that learners possess. I have used, merely as an illustration, the teaching of a very narrow aspect of logic. I also run workshops in which teachers are invited to adopt a similar approach to the design of innovatory materials in their own areas of expertise, based on thinking about students weak in one or other of the Gardnerian intelligences,.

Acknowledgements. I thank Kevin Smith, for the CAD analysis of Sylloid and for building six Sons of Sylloid. For help with the photographs, I am grateful to Charles Young and, for helping to set up the instruction/testing sessions, Charlie Artingstoll.

References

Nowviskie, B.P.: Speculative Computing: Instruments for Interpretative Scholarship, a doctoral dissertation presented in 2004 to the University of Virginia, http://webcache.googleusercontent.com/search?q=cache:UsadeaJNsl0J:www2.iath.virginia.edu/bpn2f/diss/dissertation.pdf+%22use+text+and+those+that+use+diagrams%22&cd=3&hl=en&ct=clnk&gl=uk

Barwise, J., Etchemendy, J.: Heterogeneous Logic. In: Glasgow, J., Hari Narayanan, N., Chandrasekaran, B. (eds.) Diagrammatic Reasoning: Cognitive and Computational Perspectives, pp. 209–232. AAAI Press/The MIT Press, Cambridge, MA (1995)

The Many Rewards of Putting Absolutely Everything into Introductory Logic

James M. Henle

Smith College
Northampton, Massachusetts, USA
jhenle@smith.edu

Abstract. I co-teach an introductory logic course with philosopher Jay Garfield. The contains an absurd amount of material—formal and informal, theoretical and applied, trivial and profound, sacred and mundane.

The course gives ordinary students important critical skills. For some, it kindles a love of logic that shapes their academic careers.

Keywords: formal logic, teaching, critical reasoning.

I do mean everything.

In this paper I argue for an introductory logic course that does both formal logic and critical reasoning, includes serious philosophy and mathematics, hits topics in linguistics and computing, touches on law and rhetoric, and has time to visit politics, religion, poetry, meteorology and science fiction.

The benefits of a course like this are many.

- It puts logic where it belongs, at the center of the universe.
- It grows logicians.
- It fosters a vibrant logical community.
- It makes you look good to your dean.

And surprisingly, for a course so broadly conceived,

- It reaches great depths.

The contents may seem too much for one course. It is more than one instructor can reasonably do. For this reason I recommend a team. My first co-teacher was the late philosopher Tom Tymoczko. I now teach with philosopher Jay Garfield. Most of the ideas in this paper are his as much as mine.

The origin of the course is embarrassing. We designed it to please ourselves. We made it great fun to teach. We did just about whatever we wanted. Fortunately, the course has turned out to be almost as much fun for the students. It's also good for them.

There is much to explain here. I'll start by describing some of the problems the course is meant to address. Then I'll go into the composition of the course—what it must contain and what it can contain. At the end I'll discuss the consequences of offering such a course.

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 109–122, 2011.

1 What's Wrong Right Now

1.1 Irrelevance

Logic is relevant, but that's not how it's regarded, at least in the United States. Formal logic lives on a pedestal. It's respected, but distantly. Everyone admires it; a few study it; but most ignore it. In general, the public makes no connection between the strange symbols of formal logic and the great issues confronting the world.

Informal logic gets more attention, but it too has trouble. It's relevance is respected abstractly but not practically. Everyone understands that there's something special about arguments that are logical, but for many reasons logic is seldom sought and seldom achieved. Politicians, advertisers and artists prefer appeals to emotion. Critics and commentators respond in kind.

Even in academia, logic is more an ideal than a concrete goal. There are many fields where logic would expose appalling gaps and terrible inconsistencies. Naturally logic is not especially welcome in such places. I am not at liberty to be more specific.

In curricular matters, the same is true. For example, logic should be at the center of any writing course. Most courses give it only cursory attention. Almost never is it the organizing principle.

In sum, the relationship of the professoriate to logic is a bit like its relationship to quantitative reasoning. Faculties will usually vote to require that students be "quantitatively literate." But few in the faculty are themselves quantitatively literate.

1.2 Fear

Many students fear logic. The reasons vary.

Philosophy majors are required to take logic. As a result, logic is regarded by them as an unpleasant hurdle. Many majors put off taking logic until their senior year.

Logic is strongly recommended for pre-law students. The consequence of that is that all the fear that pre-law students have of the LSATs[1] transfers to logic.

"Fear" is too strong a word for most people, but many in the general population have a distinct discomfort with logic. This is true even among mathematicians. In mathematical social gatherings, I often feel like the clergyman who has walked into a party. Everyone suddenly is on their best behavior. They don't want to say the wrong thing.

1.3 Tedium

Many philosophy professors teach logic every year. For some, it is a simple chore. There are students that get it; there are students that don't. The ones that don't

[1] The Law School Admission Test, a lengthy standardized test critical in applying to law school in the United States.

are hopeless. The material doesn't change. The same exercises and the same examples trot across the blackboard term after term.

It's tedious, but tedium is not universally seen as a problem. Some faculty are more than content to teach a course that's not challenging and not surprising. They can relax teaching logic.

But even if tedium is not a problem, I have a solution.

1.4 Indifference

We (the reader and I) are logicians. We're fascinated by its power, its beauty, and the marvelous surprises it offers us time and again. But we're not very successful recruiting followers. Unlike historians, economists, and psychologists, we have few undergraduate accolytes. Certainly not as many as we deserve.

1.5 Cowardice, or Possibly Lethargy

College is the time and the place where students should challenge everything. It is the period in life which is ideal for intellectual exploration and adventure. Students should wrestle with ideas. They should try on beliefs without preconditions. But on many campuses, this is difficult.

On some issues there is a "politically correct" position. It can take courage to doubt it. Alternative ideas will be attacked. The individual who ventures the alternative idea can suffer socially.

Even in the best environments, it takes energy to tackle an issue. It takes work to achieve the intellectual distance that makes toying with unusual positions possible. It's always easier to accept what you have always accepted.

And then, of course, some widely-accepted ideas are accepted widely because they're correct. Those ideas are especially difficult to challenge. But of course, they should be.

Finally, while everyone believes that ideas should be challenged, and while everyone knows that college students have grown lazy and complacent, just about everyone is confident that they at least challenge ideas frequently and courageously. So maybe it's not a problem.

2 The Course That Will Fix All That

2.1 Two Sides

To begin with, this course needs a team. At Smith, it's taught by a mathematician and a philosopher. We add one or two graduate students from the University of Massachusetts for section help and we have a staff of undergraduate veterans to grade papers, tutor students and cheer them on.

Students take logic for different reasons. If different perspectives are offered, more students have the pleasure of seeing their issues raised. More students find meaning in logic. More students see themselves as future logicians.

There are other benefits from having two different viewpoints in the classroom. Class is more exciting with disputations. Arguments are a signal to students that disagreement is possible, that disagreement is good.

The differences don't have to be extreme; in our case they are not. We disagree, for example, over what constitutes a reasonable resolution of the Liar paradox. Most of our quarrels are about what is interesting or what is significant. Only occasionally do we differ about what is actually true. Students are surprised at first when we fuss over what seem to them trivialities. But the fuss tells them that they don't have to accept any view; they can form their own. That's liberating. And it also tells them that maybe, underneath it all, there are no trivialities.

2.2 Formal Logic and Critical Reasoning

Formal logic and critical reasoning are different sorts of logic. Their differences have kept them apart in the curriculum, to their detriment. For despite the differences there is a close kinship.

Formal logic is symbolic and abstract. Some are attracted to its intellectual challenges. But for many, Ps and Qs seem pointless. For these students, informal arguments can give meaning to formal logic. They can bring the symbols to life. The effect is especially strong when formal structure can be seen in arguments made in the world today, or in the arguments one wants to make.

Many formal logic courses spend little time with argumentation in natural language because it is feared that would take time away from deeper topics. But in fact no time need be lost. The illustration of logical ideas in language facilitates understanding. Our students, yours and mine, have an innate logical sense. That sense comes with the practice of speech. If we use that logical sense, we can move faster. Our students come to us with a foundation for formal logic. Critical reasoning strengthens that foundation; it enhances learning formal logic.

On the other side, critical reasoning lives with the ambiguity and irregularity that goes with natural language. It can be difficult to sort valid argumentation from invalid. But formal logic can reveal the hidden structure. And it's not enough to do this in simple situations. Students won't believe the formal analysis unless they get more than taste of it.

Rich logical structure can be found in the public record. As a challenging exercise, we might ask students to formalize in propositional logic:

"If any qualified retailer fails to provide the notice described in section 4041(n)(3)(A)(ii) to any seller of diesel fuel to such retailer, unless it is shown that such failure is due to reasonable cause and not to willful neglect, there shall be paid, on notice and demand of the Secretary and in the same manner as tax, by such retailer with respect to each sale of diesel fuel to such retailer by such seller

to which section 4041(n)(4) applies an amount equal to 5 percent of the tax imposed by section 4041(n)(1) on such sale by reason of paragraphs (3) and (4)(A) of section 4041(n)."[2]

Critical reasoning is often offered as a soft logic course, a course for students not ready for abstract thinking. There may be students like this, but most students, even high school students can handle formal logic and indeed are not served well by the limited ambition of critical reasoning courses. The fact that many students don't do well in critical reasoning doesn't mean that they would have failed a more rigorous course.

By analogy, consider high school algebra. Many floundering algebra students flounder because they are left with the impression that the rules they are taught are arbitrary. Lacking a formal basis for the subject, they suspect the laws of algebra were hammered out in committee after lengthy negotiations. Random rules (and to clueless algebra students, that's how they appear) are difficult to learn.

Interestingly, when Tom and I first began work on the course I am describing, it was the philosopher who yearned for symbolic logic and the mathematician who sought natural language argumentation. Both of us knew instinctively what we needed to be whole.

How far you go, either in formal logic or critical reasoning is negotiable, of course. The formal logic in our course includes predicate logic and spends four weeks on predicate deduction. Our informal logic trains students to read and diagram arguments, to rebut them, and to diagram, outline, and write their own arguments.

2.3 Client Departments

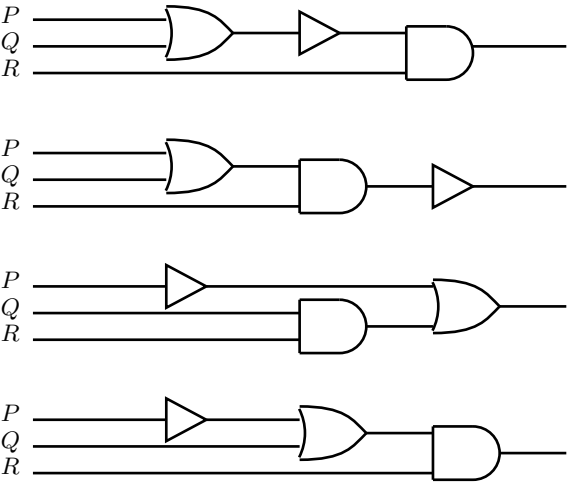
Computer scientists and linguists use logic professionally. It's important for students to understand that. It raises their struggles to a different plane. It shows them that logic is not just a game. It's serious stuff.

At the same time, each of these subjects gives students a different view of the same material, helping their understanding and confidence. We like to give them a matching problem that ties together language, computing, mathematics and logic. We give them four statements,

$$\neg(P \vee Q) \wedge R$$
$$(\neg P \vee Q) \wedge R$$
$$\neg P \vee (Q \wedge R)$$
$$\neg((P \vee Q) \wedge R)$$

to match with four logic circuits,

[2] United States Statutes at Large, Containing the laws and concurrent resolutions enacted during the second session of the ninety-ninth Congress of the United States of America.



to match with four English sentences,

"Either Harold won't come to the party or his aunt will come and it will be a great success."
"It's not true that Harold or his aunt will come to the party and it will be a great success."
"The party will be a great success and either Harold won't come or his aunt will."
"The party will be a great success but its not true that either Harold or his aunt will come."

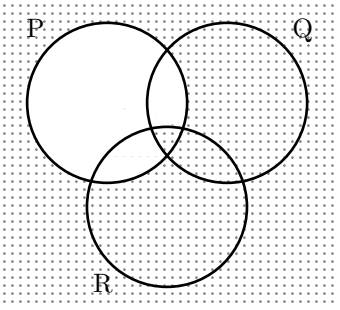
to match with four statements in Polish notation,

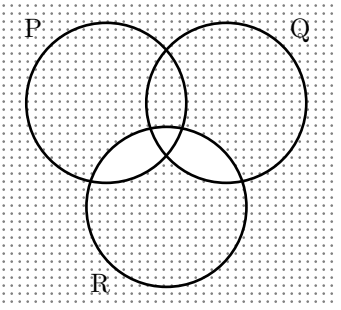
$$\neg \wedge \vee PQR$$
$$\wedge \vee \neg PQR$$
$$\vee \neg P \wedge QR$$
$$\wedge \neg \vee PQR$$

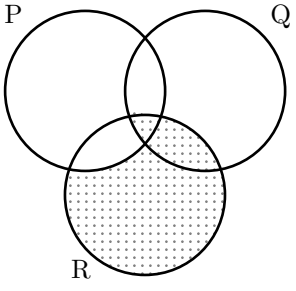
to match with four truth tables,

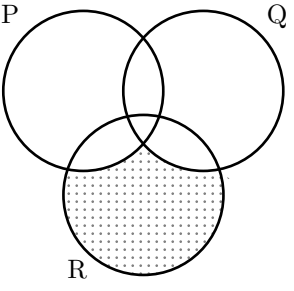
P	Q	R	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
T	T	T	F	T	F	T
T	T	F	F	F	T	F
T	F	T	F	F	F	F
T	F	F	F	F	T	F
F	T	T	F	T	F	T
F	T	F	F	T	T	F
F	F	T	T	T	T	T
F	F	F	F	T	T	F

to match, finally, with four Venn diagrams.









Students make the matches easily. The exercise convinces them that logical ideas are ubiquitous and recognizable in different contexts. We have a similar matching problem involving different ways to define an artificial language (rewrite rules, recursive definitions, finite state automata, and finite state acceptors).

It doesn't take much computer science or linguistics to have an effect. We always include both but each year we key on one of them.

2.4 Other Stuff and Lots of It

In the same spirit we include samples from the LSATs. We discuss advertisements. And we analyze presidential debates. It's surprising how many logical issues play out in debates.

"Now, Barack Obama, of course, he's pretty much only voted along his party lines. In fact, 96 percent of his votes have been solely along party line, not having that proof for the American people to know that his commitment, too, is, you know, put the partisanship, put the special interests aside, and get down to getting business done for the people of America."[3]

We discuss the scientific method. We discuss statistical reasoning. And we explore the use of logic in poetry.

[3] Sarah Palin, Vice-presidential debate, 2008.

"If this be error and upon me proved,
I never writ, nor no man ever loved."[4]

We use logic to clarify theological disputes. We use, for example, different sorts of mathematical infinity to illustrate that God is supreme, that God is everywhere, that God and Satan are polar opposites, that there is no god, that there are many gods, that God is omniscient, that God is omnipotent, and that God is so great as to be beyond human understanding.

All of this enhances student experience. It makes logic memorable. It tells them they are studying something vital.

2.5 Serious Fun

We tease our students. It's fun. It's legal.

Over the semester we toss them one paradox after another. Paradoxes amuse—but they also disturb. We write on the board,

Exactly one statement on this board is true.
$2 + 2 = 4$.
Jim Henle and Jay Garfield are the finest professors
in the history of the world.

and they lose sleep worrying about it. Some decide it is more likely that the second statement is false than that the third statement is true.

Paradoxes raise important philosophical questions, questions we pursue in class when we have the logical tools. For some students, and for us, paradoxes drive the course.

We also give the class pretty ordinary jokes and puzzles. By the end of the course it isn't easy for students to tell when joke is just a joke. That's part of the message.

One of our favorite events is to bring to life a character from our book. This is Cathy, an obnoxious junior who intimidates students with arguments for extreme positions, arguments that are difficult to rebut. Toward the end of the semester we start rumors that she has been seen on campus. Then one day she bursts in, abuses us liberally, and engages the class in an outrageous debate. It's a great scene.

2.6 Logic Outreach

Having given our students some powerful but elementary logical skills, we send them out to challenge and irritate their friends. Several times during the semester we take a statement, say, "This college should require all students to take a course in quantitative reasoning." We then ask each student to find someone outside the course to argue the point with. Our student must first ascertain their opponent's view on the issue and then take the opposite position.

[4] William Shakespeare, Sonnet 116.

We don't grade this exercise on our student's success, only on whether the argument takes place. We start with issues of local interest (as above) but later move to national, international or ethical questions. The last usually has some emotional content, which of course, students must manage to transcend.

At some time in the term we combine Logic Outreach with Serious Fun by playing a practical joke on the campus. We invent an issue. One year it was about selling paintings in the art collection to support scholarships. Of course, nothing of the sort was contemplated by the college.[5] The issues we choose are wholly fictitious.

For the practical joke, students are sent to chalk arguments on the sidewalks and scatter angry bursts on the internet. We start with one team opposing the chosen wild idea. The following day a different team rebuts the first. Two more teams follow in subsequent days with counter-proposals and counter-rebuttals.

Students have a lot of fun with this. They learn that no issue is beyond debate. They discover the pleasure of advocacy for its own sake. They experience the thrill of arguing for something ridiculous. All this is healthy.

As a class exercise, the chalking always works. Our record with the rest of the campus, however, is mixed. Sometimes we are gloriously successful in starting arguments. The proposition, "There shouldn't be male professors at a women's college" was one of our best. More recently the campus has caught on. A recent chalking campaign was dismissed with "It's just Logic 100 again." Last year we used a decoy argument.

2.7 Deep Thoughts

After all the above, there doesn't seem to be room for more. But we always save time for something special at the end. It's an important ingredient in the course. It motivates us and it motivates the students. The fact that at the close of the semester we are in a position to discuss ideas and issues at the frontier of logic is exciting. And when at last we take up advanced topics, it's a ratification of the class's progress. We're telling our students that they have achieved something significant.

The advanced topic depends intimately on the taste of the instructor. I'm a mathematician with a research interest in infinite sets. We tease the students throughout the semester with paradoxes of infinity. At the end we give them Cantor's diagonalization proof, construct infinite ordinals, and even discuss Hugh Woodin's latest work on the continuum problem.

Jay is a philosopher with a research interest in paraconsistent logic. We tease the students thoughout the semester with paradoxes of self-reference. At the end we give them a logic in which statements can be true, false, or both true and false. We find time at the end to discuss the latest work of Graham Priest and others on paraconsistent logics.

[5] We won't use this issue again—it's not a joke anymore. Brandeis University seriously proposed selling their collection.

2.8 Syllabus

This is what we cover on the formal side:

1. Sentential or propositional logic, truth tables and logic circuits
2. Validity and inference in sentential logic
3. Defining Sentential language recursively
4. Predicate logic, universes, validity, and inference
5. Defining Predicate recursively; syllogisms
6. Deduction in sentential and predicate logic with identity

On the informal side we cover:

7. Analyzing arguments, identifying conclusions, diagramming logical structure
8. Rebutting premises, rebutting inferences
9. Diagramming your own argument
10. Outlining and writing your argument
11. Argument in debate

In a typical semester we will also cover either the following from linguistics,

12. Rewrite rules
13. Finite state automata
14. Regular languages
15. Finite state acceptors

or the following from computer science,

16. Binary
17. Stacks
18. Prolog
19. The busy beaver problem
20. The halting problem

but in any case we will definitely take up these from set theory,

21. Countability
22. Uncountability
23. Defining numbers from sets
24. Ordinals and ordinal arithmetic

and these on alternative logics:

25. Polish notation
26. Many-valued logic
27. Probability and inductive logic
28. Modal logic
29. Paraconsistent logic.

2.9 Too Much Material

It appears that there is too much in our course. It covers the equivalent of two, perhaps three ordinary courses: a critical reasoning course, a course in formal logic, a survey course in the logic of computer science, linguistics, philosophy, and mathematics, with topics in cardinality, modal logic, possible worlds, and paraconsistent logic. Students can't possibly master it all. Students will certainly forget almost all they learn. There seems to be no point to it.

But there is a point. I'll explain.

3 What This Course Does

This is not so much a course *in* logic but a course *of* logic. The goal is not for students to master content; it's for them to become logical. We want students to recognize logic when they see it, and to note its absence when it's missing. The course is an experience. Every topic, every puzzle, every game, every paradox adds to the experience. In the course, logic happens. It changes the students.

Students do forget the material. Even the best, the ones who become teaching assistants, lose their grasp of some details. But the students grow, logically. They emerge ready for the next course in logic. What they don't recall they can retrieve quickly. And with their logical reflexes they can deal with new logical challenges.

The students are better prepared not just for logic courses, but for math courses, philosophy courses, economic courses—courses in all fields. Students who have passed through logic are quicker, tougher, more resilient. They're smart and they know it.

They're also better writers. They know they have to have something to say. They (some of them) will start their history papers by diagramming and outlining their arguments.

And they're better readers. They know that the English language has traps. They're alert. They're wary.

Isn't this really what we want for our students? These are the true basic skills. Today's employers want smart, knowledgeable people. But smart is far more important than knowledgeable. Smart people can be trained, they can learn what they need to know. But it's challenging to make your employees smart.

Jobs disappear. Industries collapse or move abroad. Paradigms change. Our students can only prepare for this by getting smartness. They get it by studying Logic.

3.1 It Meets Expectations

We tell our students that they will learn a ton of stuff, that they will get smart, that they will join an elite group of students. That sort of happens. Nearly all complete the course. They don't learn everything but they end up smarter and feeling special.

One reviewer for our book looked at the material (some but not all of what I am suggesting here) and said that it was far too much, that maybe Harvard students could handle half of it.

We cover all of the book and additional material we put on the web. And Smith is not Harvard. Our students are not special. Some are good, but about half of them are in their first semester at college. Others are senior philosophy majors who, fearing logic, have put it off as long as they could. A few are women of non-traditional age, returning to finish their degree. Some students struggle, but the teaching assistants keep them going.

Students respond to high expectations. Ambitious goals are critical to the course's success. We set out to move our students a great distance. It seems to work.

3.2 It Grows Logicians

At Smith and in the Five Colleges Consortium (Smith, University of Massachusetts, Amherst College, Hampshire College, and Mount Holyoke College), this course has been the engine of a strong logical community. The best of our introductory students are asked to be teaching assistants the following year. The most interested of our students take follow-up courses in incompleteness, relevance logic and set theory. The most committed students elect to minor in logic. The most obsessed major in it.

At any time we generally have one or two majors and four or five minors. The minor is on the books but students who wish to major in logic must construct a "self-designed major" to satisfy Smith's graduation requirements. The major they design can focus on philosophy, mathematics, computer science, linguistics, or even law. All logic majors are expected to take courses from several of these fields.

There are logicians at each campus of the Five Colleges. Together we have a logic program that offers a certificate to any undergraduate completing certain course requirements. Earning the certificate requires taking at least six logic courses. We are turning out logicians.

As part of Smith's logic program and in support of the larger logical community, Smith sponsors the annual Alice Ambrose Lazerowitz/Thomas Tymoczko Logic Lecture. This is timed to coincide with the end of the introductory logic course so that students are ready to understand at least half of the talk.

3.3 It Opens the Campus to Debate

Well, perhaps it does. There is no way to measure the success of our logic outreach efforts. We hope students feel more comfortable trying on positions. We have no way of knowing if they do.

3.4 It Is Immeasurably Successful

Or, to put it another way, there is no way to measure its success. Content is not the goal, the goal is smartness. I'm not convinced that smartness can measured. As a mathematician, I have a deep distrust of numerical measures of intelligence.

The only measures I see as meaningful here are the number of students in the course, the number who take a second course, and the number who continue on to major or minor in logic. Students know when they are doing something worthwhile. They may give a course high marks on evaluation sheets, but their decision to continue is more significant and sincere.

Enrollment in the course is high, three to four times higher than it was before Tom and I began teaching together. Students elect the course despite the difficulty and the workload. Many take a second course but we haven't conducted a study. There are three logic majors at the moment with two more at the planning stage. There were none before the program began.

3.5 It Pleases the Administration

The course is highly regarded by our administrators because it addresses so many basic student skills. One could almost describe the course as "reading, writing, and 'rithmetic."

We teach the students to diagram arguments they read. This is, in a sense, advanced reading. It's surprising how difficult it is for students to identify the conclusion of a typical letter to the editor of a paper. Diagramming an argument requires understanding what was in the mind of the writer. That is problematic when the writer is not skillful.

The critical reasoning portion of the course clearly teaches writing skills. We don't spend time on bibliographic concerns, spelling, or grammar (except as it affects the logic of a sentence). We don't ask students to write long papers. Despite that, our course satisfies Smith's writing requirement because it addresses the most critical of writing skills: composing a strong argument and conveying that argument to the reader.

The claim that we teach arithmetic is perhaps harder to justify. The course doesn't address quantitative skills, except for a little time spent on probability. But the course satisfies the college's recommendation for analytic reasoning.[6] The other courses with this distinction are, with the exception of a few philosophy courses, all in mathematics or computer science.

Our dean isn't disturbed by the fact that two professors are being used to teach one course. We attract a large number of students, well more than twice the average number of a course at Smith, and more than five times the size of classes satisfying the writing requirement.

About 15% of Smith students take logic, many of them in their first semester. The course does a lot to teach them how to be a successful student.

3.6 It Pleases Me

In the end, one can never be sure what students will appreciate. Since this is so, my guiding principle has been to fashion a course for my own satisfaction,

[6] Smith has no distribution requirement, but students will not be awarded Latin honors (*cum laude*, *magna cum laude*, etc) unless they take courses in a list of areas, one of which is "analytic reasoning".

hoping that coincidentally it will satisfy my students. I believe this has worked; I believe the course does please our students. But unquestionably the course pleases *me*.

I can do whatever I want. I can justify the inclusion of anything in the course. After all, in every field you find either

a. the presence of logic—in which case it makes sense to discover and analyze that presence,
   or
b. the absence of logic—in which case it makes sense to discover why this is so and how the addition of logic might change things.

The list of topics I have explored in the course is pretty long: economics, politics, law, rhetoric, theology, history, the tax code, comic opera ... The list of stunts I have tried is also pretty long. Some of them I'd rather not discuss here. The attempt to do Monty Python was better than the attempt to do Abbott and Costello. The episodes with the time machine had mixed success.

4 To Sum Up

Logic is special. What makes logic special, what makes it so thrilling and vital, is its cosmic nature. Logic's reach is global. Logic is at the core of all intellectual activity. A new logical truth reverberates up and down the centuries. When you advance logic, you advance understanding in every corner of the intellectual universe.

That being so, the introduction to logic should be special. It should include everything. It should reflect the subject's majesty—and its whimsy.

The SELL Project: A Learning Tool for E-Learning Logic

Antonia Huertas, Josep M. Humet, Laura López, and Enric Mor

Computer Science Department, Universitat Oberta de Catalunya,
Rambla Poblenou 156,
08018 Barcelona, Spain
{mhuertass,jhumet,llopezor,emor}@uoc.edu

Abstract. The SELL project described here is the design and development of a tool for assisting the learning of Logic in the context of a wholly online CS degree using a web-based learning environment. This tool should provide guidance, interactive feedback, and continuous assessment for Logic course students, covering major topics in an introductory course (natural deduction, resolution and semantics in propositional and predicate logic). The process of the design, implementation, use and development of the resulting tool, coined Logic E-learning Assistant, is presented.

Keywords: introduction to Logic**,** online higher education, e-learning, intelligent tutoring system, e-tutor, web assistant, e-assessment.

1 Introduction

The UOC (Open University of Catalonia; www.uoc.edu) is a wholly online university having an e-learning student-centred educational model. The university offers a basic Logic course as part of the Computer Science degree, which is one of the most technical programs. There are more than 700 students enrolled each semester in this course, comprising the usual topics of an introductory Logic course in two possible languages (Catalan and Spanish). The traditional Logic course is an overview of propositional and predicate logic and special attention is given to formal semantics. Logic at this level is part of mathematical logic and the subject inherits the mathematical particularities that make it rather difficult for students to grasp.

Students enrolled in a Logic course have to acquire a set of skills and a small set of contents. The continuous assessment is very important in this kind of subjects, as a way to monitoring the progress in the learning progress of the logical skills. The instructor has also an important role when acquiring these skills and the concrete guidance and interaction with the teacher is a fundamental aspect of the learning methodology [5]. In an online scenario, students have the same interaction needs but they interact with the teacher only using their computer [11]. Furthermore, this computer-mediated interaction is usually text-based, providing a narrow scope to have feedback. Therefore, this can become a concern when learning the competences of Logic.

On the other hand, web-based learning and e-learning in general can allow individual training while being easily delivered to a wide audience via the Internet at a relatively low cost. Another important advantage of e-learning is that it is based on

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 123–130, 2011.

resources and activities that are accessible via a computer device, offering a high degree of interactivity and a more dynamic type of self-assessment or immediate feedback [3]. In particular, intelligent tutoring systems could be used to improve the learning process providing customized assistance and feedback to students [6,7]. In the context of e-learning, intelligent tutoring systems can help to overcome the absence of teachers while taking advantage of self-learning. In the UOC online model and in a subject like Logic, such learning tools should be part of the solution [14].

There are many learning tools for Logic [19] but they have no standard notation, rules, logical systems, etc. Thus, the existing tools are practically not reusable for a material created independently of that tool [10], in particular our UOC courseware for distant learning. We would need to create a new tool adapted to our courseware. Therefore, an innovation project placed in charge of the UOC was proposed in which a learning tool for Logic would be developed.

This paper is organized as follows; Section 2 describes the SELL project, the design and development process of the learning tool. Section 3 describes the characteristics of the resulting tool and its use in the virtual classroom. Section 4 describes the evaluation of the learning tool. Finally, Section 5 presents the conclusions.

2 Description of the SELL Project

The goal of the SELL (Monitoring E-Learning Logic, in Catalan "Seguiment E-Learning Lògica") project described here is the design and development of a tool for assisting the learning of Logic in the context of a wholly online Computer Science degree using a web-based learning environment. This tool should provide guidance, interactive feedback, and assessment to Logic course students.

The project team have comprised staff of the UOC that specified the functional needs of the project and an analyst-programmer who analyzed, designed and implemented the project in collaboration with the staff of the UOC.

2.1 Design

The project followed the UCD (User Centred Design) [13] process that includes three main phases: gathering user requirements, designing the product iteratively and finally, evaluating the prototypes of each design iteration [4]. In addition to these phases, the UCD process applied in the design and development of learning resources and tools has to follow the specific goals of the e-learning context: a) reduce difficulty in the teaching and learning process, b) improve the learning experience and c) integrate with the existing virtual learning environment. The key element of this approach is the evaluation and iteration of the design solutions. We had two sets of requirements:

- Institutional: e-learning tools which have to be placed in the virtual classroom structure should be accessed using a standard web browser and they should not technologically interfere with existing resources.
- Users: they will be the students enrolled in the course. Teachers will also be users but they will have more functionalities and views of the learning tool. An analysis was carried out with teachers and students to identify those requirements.

In addition to that, other information has been gathered during one semester by observing the continuous assessment process and the exams at the end of term. From a test with an existing tool for learning Logic in a comparable curriculum[1] [12] we learn that students do not use the tool since it only provides voluntary practice. This is because the students perceive the time and energy using the tool as an extra effort without any clear reward. Thus, an important finding was the need to integrate the learning tool in the continuous assessment model of the pedagogical strategy used [2].

Following that analysis, the main requirements for the Logic learning tool were: providing immediate feedback; fostering learning of the strategies and skills characteristic of Logic; to be integrated in the continuous assessment of the students; ease-of- use; to be integrated into the existing virtual classroom; and being multilingual (at least Catalan and Spanish).

2.2 Implementation

After testing the initial prototypes, the developing phase followed on. The coding of the tool took several months and was done by a web developer of the team. Other UOC members of the group supervised and commented on the process of implementation which was done in an iterative way. The architecture of the tool and other technical solutions were decided in order to assure the requirements would be taken into account. The tool would be designed under an approach based on architecture in three layers: interface, domain and persistence.

It is interesting to mention that the only user requirement in order to use the tool is to have a browser compatible with Internet Explorer 5.5 or superior or with Firefox 2.0 or higher. It is also necessary that the user has enabled Javascript (option by default of the browsers). The web pages that are part of the tool were developed using PHP for the programming in the server and HTML, CSS and Javascript. It is also interesting to mention that in order to store the data in a persistent way, a MySQL database has been used, which allows the managing of sessions in an efficient way. Finally, the development of this application requires liaison with the existing UOC servers to obtain the login information of the students. This functionality has been covered through a series of web services provided by the UOC virtual campus developers.

The final tool was simply named *Logic E-learning Assistant* or even simpler "The Assistant". There have been two versions of it. The first one is the result of the initial SELL project (2008, 2009); and the second one the resulting version of the enlargement of the project after the success of the first version (2009, 2010). The functionality available for users of the tool will be different depending on their profile "student" or "tutor". The first version set out the key lines of implementation: architecture, structure, functionality of each user and early automatic tutor support; while in the second one, the continuous assessment module was added.

3 Description of the Logic E-Learning Assistant

In this section the main features of the Logic E-Learning Assistant are presented. Technical and educational features and functions are described.

[1] Josep M. Humet: LSD (accessible a http://ima.udg.edu/~humet/lsdweb/index.php).

Access: Access to the tool is through the virtual classroom of Logic. There is a direct link in the first level. Students have been identified to enter the virtual classroom and this identification is to be used in all processes that need it.

Interface and Structure: There are four different modules that are easily identifiable in the interface: Language, Exercises, Assessment, and Help.

Multilanguage: The Assistant starts in one of the two possible languages (Spanish and Catalan). In the top right corner of any screen, one or the other language can always be selected. The system is easy extensible to other languages, by just translating s a list of vocabulary. As for the exercises and other content posted by the administrators, titles and explanations should be introduced in different languages. The formulas, through the universal language of Logic, do not need translation.

Exercises: The exercises are classified according to the topic. There are two major groups: Propositional and Predicate Logic exercises. In the Propositional case, there are three types: Natural Deduction, Resolution, Truth-tables. In the Predicate case, there are two kinds: Natural Deduction and Resolution. Exercises are easily identifiable in the initial screen. Each kind links to a different interface with a list of proposed exercises with the option to create their own exercises and with the students' own statistics. The option "Solve" leads to a new screen where the final real Assistant is eventually accessible in order to introduce algorithms to solve exercises, check them, obtain feedback and grade them in an automatic way. In fact there are five different Assistants, one for each of the five kinds of exercises. A database of exercises is filled up by concrete exercises, which are chosen and introduced by the teachers for further practice. On the other hand, students can easily introduce new exercises of any type in order to obtain immediate feedback while they solve them. There is a formulae editor to answer text questions.

Interactivity: When the learning assistant is accessed from the virtual classroom, five groups of topics of exercises are available. When solving an exercise, students have to introduce the rule and the result of the rule application. AELL responds to the student's actions by either applying the rule and going to the next step or giving an appropriate error message. AELL is never solving by itself at any step. Thus, the Assistant is mainly providing feedback of correction or error at any step of the algorithm. Error messages give standard information (depending on the error) to help the student find out what was the error. All the pieces of advice are dynamically generated and context sensitive. AELL has been enhanced to facilitate automatic marking for assessed coursework. As the students are always identified, a report can be produced for each student together with statistics for their teachers involving minimal human intervention.

Assessment: This module has been created to manage the individual work of the students (both voluntary and compulsory work). There are two possible interfaces, one for each kind of user (student or teacher). Students are always identified while working with the tool and the logs produced are stored. For any category of exercise, students can see the statistics of their voluntary work (the complete, incomplete, or pending exercises) and be able to access them. Through another part of the tool, students do and deliver the exercises of the compulsory work, where each test could have exercises that are automatically graded by the system and text exercises that are the only ones that the instructor has to individually correct. Students see their results immediately with respect to the automatically grade parts of the test. From the point

of view of the instructor, the assessment module is much richer. The instructor defines and introduces the different compulsory tests in an easy way. They have access to the individual or group statistics of many different elements: the average success of students, rules that were incorrectly introduced, frequencies of use, temporal distribution of the work, and as the system stores all the logs in a server of the university, many other statistics can be obtained. Most of the grading is automatically calculated by using the solving associated procedure and the time the teacher saves not doing the automatic part of the correction can be used in the providing of individual feedback and comments.

Help module: It consists of a collection of short videos (2 to 7 minutes) in both languages, showing how to use the tool for each category. A more complete documentation of the tool is being produced. Students can always ask the teacher for some help, if required.

The E-Learning Logic Assistant (Assistant E-Learning Logic, AELL, in Catalan or Spanish) is integrated in the courseware of Logic. There are other resources in that courseware and recommended timings and paths with a detailed guide of readings, examples, and exercises to practice. Style and notation in the Assistant is similar to the one in the rest of the tutorials, which makes it intuitive.

4 Evaluation

Until now, two versions of the learning tool have been used:

- The first version corresponds to the course 2009-10 (two semesters). The covered topics were natural deduction and resolution of propositional logic.
- The second version has been used in the first semester of the course 2010-11 (one semester). The covered topics in the previous version were enlarged through truth tables to be validated in propositional logic, natural deduction and resolution of predicate logic. The Continuous Assessment Module was also added.

The tool is completely integrated with the rest of the learning material and with the continuous assessment model of the Logic course. Students and teachers easily can track the learning progress since the tool provides statistics for each individual user, for the class group and also for the different groups. Students can see their progression level and both students and teachers can find the critic points of every stage of the learning process. At present, the tool is being used in the Logic course of Computer Science degree of the UOC. Using it in a real scenario provides a set of usage data that is being use to improve both performance and usability of the tool. We already have a first feedback from students that consists on positive comments about the tool and how it helps them to learn the course.

The initial small team has grown in terms of tutors of the ten or more virtual classrooms of each semester. They have participated as key users in the evaluation and improvement of the two versions. In particular, their involvement in the design of the continuous assessment module has been most valuable and their enthusiastic involvement went beyond professional limits.

During this time, we have evaluated the tool by using anonymous feedback forms and questionnaires, by studying the performance of the final face-to-face written

Table 1. Institutional satisfaction evaluation of general resources

Evaluation of the resources in Logic	Positive
2008-09	70%
2009-10 (first course with the Web Assistant – first version)	88%

exam and by evaluating recorded detailed logs. In the general satisfaction questionnaire drawn up by the university, we have the following results in the evaluation of the resources of the Logic subject.

We also have the following results in the specific satisfaction questionnaire.

Table 2. Specific satisfaction evaluation of the Logic Assistant

2009-10	
Useful in learning DN (propositional logic)	80%
Useful in learning Resolution (propositional logic)	70%
Useful in continuous assessment	76%
Useful in reducing study-time	42%
2010-11	
Useful in learning DN (propositional and predicate logic)	71%
Useful in learning Resolution (propositional and predicate)	77%
Useful in continuous assessment	87%
Useful in reducing study-time	52%

The results of the satisfaction questionnaires show that students find AELL easy and useful in the learning process of all the topics. They have also made useful criticism.

We have compared the results in the written exams with the semester using AELL and the other semester without the tool. This did not provide a clear advantage to students using the Assistant. The reasons for it could be many. The fact that the exam is in the old-fashioned paper-and-pencil style is one of the possible reasons for so little influence. Rates are still very low, as usually occurs in other mathematics subjects. However in the percentage of participation in the continuous assessment a little more success can be appreciated.

Table 3. Evaluation of the results, previous to and with the Logic Assistant

Course	Participation in Continuous Assessment	Passed the course
2008-09 (without Logic Assistant)	55%	39%
2009-10 (Logic Assistant – first version)	59%	38%

We are waiting to know the results for the present course with the second version of the Assistant covering the major part of the curriculum and implementing the continuous assessment module. It seems that participation is on the rise but at the moment of writing this version of the article, we do not have reliable data.

On the other hand, further work will be adding new modules to cover other parts of the Logic subject, improving the feedback system and functionalities, allowing comments and notes and building a mobile version, among others.

5 Conclusions

One characteristic of online universities like that of the UOC is its intensive use of technology to enhance learning, and this has been the initial impulse for the project. There are many other teaching tools for Natural Deduction or other Logic topics [1,9,15,17] but it seems that the current trend is the e-tutor paradigm [2,8], where intelligent tutors are integrated in a more general e-learning courseware environment. What we should add to this current trend is the use of e-assessment to support continuous assessment, like the one presented here. This kind of tool allows the automatic evaluation of some individual learning activities (in the case of logic, all those related to the methods of proving) and provides students immediate and personalized feedback and statistics of their progress. From the point of view of instructors, those tools enable automated tracking of performance and progress of students and free tutors from the routine of correcting many activities. The time and effort saved can be used to provide richer individual feedback and continuous improvement of the subject.

We know that there are people in other institutions working in similar projects [8, 16, 18] and we think that the possible community of practice of all these people will be a valuable framework towards more universal projects.

Acknowledgments

This work was made possible by the support of a 2008 project of educational innovation of the UOC; the Spanish government grants HUM2006-12848-C02-01 and FFI2009-09345/FISO; and the eLearnCenter (UOC). Special thanks to teachers and students who have participated in the development of it.

References

1. Barwise, J., Etchemendy, J.: Tarski's World CSLI (2007)
2. Broda, K., Ma, J., Sinnadurai, G., et al.: Pandora: A Reasoning Toolbox using Natural Deduction Style. Log. J. IGPL 15, 293–304 (2007)
3. Buyukozkan, G., Ruan, D., Feyzioglu, O.: Evaluating e-Learning Web Site Quality in a Fuzzy Environment. Int. J. Intell. Syst. 22, 567–586 (2007)
4. Cavus, N.: The Evaluation of Learning Management Systems using an Artificial Intelligence Fuzzy Logic Algorithm. Adv. Eng. Software 41, 248–254 (2010)
5. Collins, A., Halverson, R.: The Second Educational Revolution: Rethinking Education in the Age of Technology. J. Comput. Assisted Learning 26, 18–27 (2010)
6. Corbalan, G., Kester, L., van Merrienboer, J.J.G.: Dynamic Task Selection: Effects of Feedback and Learner Control on Efficiency and Motivation. Learning and Instruction 19, 455–465 (2009)

7. Corbett, A.T., Koedinger, K.R.: Intelligent Tutoring Systems. In: Helander, M., Landauer, T.K., Prabhu, P. (eds.) Handbook of Human-Computer Interaction, pp. 849–874. Elsevier Science B.V., St. Louis (1997)
8. Dostalova, L., Lang, J.: ORGANON - the Web Tutor for Basic Logic Courses. Logic Journal of the IGPL 15, 305–311 (2007)
9. Endriss, U.: The Interactive Learning Environment WinKE for Teaching Deductive Reasoning. In: First International Congress on Tools for Teaching Logic, pp. 23–27 (2000)
10. Etchemendy, J.: Heterogeneous reasoning. In: Stapleton, G., Howse, J., Lee, J. (eds.) Diagrams 2008. LNCS (LNAI), vol. 5223, pp. 1–1. Springer, Heidelberg (2008)
11. Huertas, A.: Teaching and Learning Logic in a Virtual Learning Environment. Logic Journal of the IGPL 15, 321–331 (2007)
12. Humet, J.: LSD, una Herramienta Didáctica para el Aprendizaje de la Lógica. In: JENUI (2001)
13. Hussain, S.: Developing E-learning Materials. Applying User-centred Design. National Institute of Adult Continuing Education (England and Gales) (2005)
14. Johnson, M., Barnes, T.: Visualizing Educational Data from Logic Tutors. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010. LNCS, vol. 6095, pp. 233–235. Springer, Heidelberg (2010)
15. Lesta, L., Yacef, K.: An Intelligent Teaching Assistant System for Logic. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 421–431. Springer, Heidelberg (2002)
16. Lodder, J., Passier, H., Stuutmann, S.: Using IDEAS in Teaching Logic, Lessons Learned, p. 556 (2008)
17. Moreno, A., Budesca, N.: Mathematical Logic Tutor - Propositional Calculus. In: First International Congress on Tools for Teaching Logic, pp. 99–106 (2000)
18. Sieg, W.: The AProS Project: Strategic Thinking & Computational Logic. Logic Journal of the IGPL 15, 359–368 (2007)
19. van Ditmarsch, H.: Educational Logic Software (2005), http://www.ucalgary.ca/aslcle/logic-courseware

Ten Years of Computer-Based Tutors for Teaching Logic 2000-2010: Lessons Learned

Antonia Huertas

Computer Science Department, Universitat Oberta de Catalunya,
Rambla Poblenou 156, 08018 Barcelona, Spain
mhuertass@uoc.edu

Abstract. The First International Congress on Tools for Teaching Logic took place in 2000 and the second such congress took place in 2006. The third one takes place in 2011. In the ten years that separate the first and the third congress, the history of the tools for teaching Logic based on the paradigm of intelligent tutoring has been evolving significantly. This article provides a brief review of this history. It also presents a comparative study of 26 such tools and proposes a classification of existing tools with a specific methodology.

Keywords: Logic**,** higher education, e-learning, intelligent tutoring system, computer-based tutor, e-tutor, web-assistant.

1 Introduction

The First International Congress on Tools for Teaching Logic[1] took place in Salamanca in June 2000. The second such congress[2] took place in September 2006, also held in Salamanca. The third one will also take place in Salamanca in July 2011. In the ten years between the first and the third one, the development of information and digital technology has brought about great changes in the teaching of Logic. On the one hand, the Logic curriculum has been extended through new logics for that new technology and on the other hand, teaching itself has been transformed due to the general access to computers and the Internet.

Currently, opportunities and challenges arise from the onset of technology in educational practice [12]. Digital technologies such as computers, mobile devices, digital media creation and distribution tools, and social networking sites are transforming learning. Tensions between traditional models of schooling and the affordances of digital technologies are noted, while the initial promise of these technologies to shape a new system of education is being reviewed and many voices coming from the educational research area are claiming for the urgency of seeking a coherent model for the future of education in a technological age [8].

The tension between traditional and technology-driven learning can be summarized in the following four opposing positions:

[1] See http://aracne.usal.es/congress/congress.html
[2] See http://logicae.usal.es/SICTTL/

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 131–140, 2011.

- Age uniform learning vs. life-long learning
- Teacher as expert vs. diverse knowledge sources for acquiring knowledge
- Standardized assessment vs. performance-based assessment
- Learning by 'absorption' vs. 'learning by doing'

Computer tutoring systems are being seen as the tools to provide customization, interactivity and performance-based assessment in e-learning environments. But what about the history and the position of those tools?

We can find a first review of the subject in [2] where the 10-year history of tutor development based on the advanced computer tutoring theory is reviewed. This early evaluation showed that students could achieve at least the same level of proficiency as conventional instruction in one third of the time. Empirical studies showed that the best tutorial interaction was one in which the tutor provides immediate feedback consisting of short and directed error messages; and those systems appear to work better if they present a nonhuman appearance rather than as emulations of human tutors. Computer tutoring systems were being built during the nineties following this framework: let students do as much as possible for themselves and provide them with error messages to tell them something useful about that error; providing also some extremely short help messages (even if they sound nonhuman). Thus the system becomes an Assistant that can deal with more routine learning problems.

In the context of Logic, academic research in the nineties was conducted in software designed to help computer science students to learn formal proofs (specially, natural deduction). The appearance of Tarski's World [3] was a milestone; it was considered not only useful for learning the syntax and semantics of first order Logic but also enjoyable. Another important tool was Jape [6] that enabled students to construct, revise and test formal proofs. At the Institute of Educational Technology of the Open University, some research focused on students' experience using the program to assist proof construction [1].

That was the theoretical situation when María Manzano promoted the First International Congress on Tools for Teaching Logic in 2000 and a number of logicians from different countries met to focus on the education of Logic. They reported interesting teaching experiences; some of them were works presenting the use of intelligent tutoring systems for teaching Logic at university level [19]: An *interactive proof Assistant* [10] to support teaching Logic and deductive reasoning (based on Tableaux); a *didactic tool* [17] to help in the learning of Natural Deduction; a *Logic tutor* [21] to help students of Logic in Computer Science; an *intelligent system* [24] for learning First-order Logic.

The Second International Congress on Tools for Teaching Logic (SICTTL) was even more successful. The presentation of software tools was a very important part of it. An outline of some of the tools was published in [25]. A wholly integrated tutoring system in a Logic courseware seemed to be an improvement in the paradigm of previous tutoring systems. Three of those new kind of systems presented in SICTTL were: Pandora [7], APros [23] and Organon [9].

Between 2000 and 2006, e-learning had become a reality and a new kind of tool was the step forward of the initial computer tutors. They were web-based, fully interactive and oriented in guiding students through activities.

Since 2006, life-long, distance and online learning have reached the college educational scenario. The intensive use of IT approaches traditional education to an e-learning model where the student and its activity have become the centre of the model [12] [8] [16]. One of its main challenges is the use of technology in customizing learning and assessment [22]. There has been a considerable amount of research showing that intelligent tutoring systems provide effective instruction for math and relative subjects in online scenarios [5]. What is the situation in 2011? In the next section, I will present a brief study of existing tools for learning of Logic.

This paper is organized as follows; Section 2 describes the basic characteristics found among current tools for learning Logic. Section 3 describes a possible classification of the current tools depending on which characteristics they possess. Finally, Section 4 presents some final discussion and conclusions.

2 Basic Features of Tools for Learning Logic

In 2011, a great variety of tools for teaching Logic can be found. In [26] there is the well known list of educational Logic software compiled by Hans van Ditmarsch. Today there are 46 entries with Internet links to their web pages and some information (a brief description on the functions, the platform, developers, and a book reference). In the proceedings of the First and the Second International Congresses on Tools for Teaching Logic [19,20], there are very interesting papers on some of the tools working at each moment. Visiting these sites one by one and studying each of the tools, it is evident that there is a wide variety among them, and it is also clear that they have evolved from the first ones until now. Some were left on the way, others were adapted to the times in its subsequent versions and other new ones have appeared. To understand the present situation, a comparative study of the current tools is presented here. To perform this study, a sample of 26 has been selected among the ones in use [3].

A careful study has been made with each of these selected tools. Each tool has been studied and has been used to know how it works, and its literature has been reviewed to know how it is used in learning and how it has been evaluated. Access to the tools has been unbalanced, as some are free and others are commercial, although among the latter there are demo versions or are better documented than those of free access. From this empirical study, a qualitative study was carried out to identify the specific characteristics that allow a comparative analysis and eventually a classification. The outcomes of this study below are the basic features of those tools, clustered for presentation in the following groups:

- Functional characteristics: the basic functionalities as a learning tool and the logical content being taught are considered.
- Technical characteristics: the year of birth and the current version, developer profile, platform, the type of license and the form of access are considered. Features such as the programming language, the type of database or information system were not taken into account for this study. Only significant characteristics from the point of view of users were discussed.

[3] See the list of the studied tools and web references of them in the Annex.

- Interaction characteristics: the type of interactivity, feedback, advice, help, and the introduction of logical symbols are taken into account.
- Assessment characteristics: the way to grade, the type of assessment and the statistics that the tool provides, if any.

2.1 Functional Characteristics

The central role of the tools is intended to be used for learning purposes and therefore we have not considered in our sample those systems that automatically perform tests without any educational objective. Regarding the educational function, a wide range of characteristics have been found: guide and help in the evaluation of sentences using a simple editor; guide students in the construction of a proof (with different levels of interaction), guide learners to construct semantic tableaux or help in making proofs or normal forms; guide building truth tables and checking unification programs; checking proofs to teach basic proof-writing skills, etc.

In relation to the logical content, most tools focus on Propositional and Predicate Logics, and in particular, in Natural Deduction and other proof systems. Truth tables and normal forms are also common. There are those that focus on semantics and there are also many tools that cover all the themes in an introductory Logic course. The latter is designed to fit snugly over a course's teaching materials or a particular book. Most of the tools studied are aimed at introductory-level college students.

As for the language, most of the 26 studied tools have a monolingual interface in English, with a few exceptions in German, Catalan, Spanish and Czech. In a very few cases, multilingual systems were implemented.

2.2 Interactivity, Feedback and Advice

Since the very beginning, interactivity has been the most fundamental characteristic of an intelligent tutoring system [2]. The main issue in subjects like Logic is that feedback could be calculated automatically by e-learning tools that support problem solving [11] and that students could interact with computer-based teaching systems in solving the problems [4]. Among the tools that they studied, the following characteristics are presented:

- Different levels of interaction, from simply verifying that the problem is solved right and delivering an error or correction message, to guiding, in a step by step process, in building the solution with different levels of feedback or assistance.
- The tool emits some kind of feedback after checking whether or not a step is correct. If it is correct, an asserting message is emitted. If it is not correct, an error message is usually given and possibly, information about the kind of error is provided.
- A hint after an error message is done automatically or when the student asks for it and this is only found in some of the studied tools.

Instructional software for Logic usually incorporates this kind of feedback and hints features [4]. In the studied tools, most of them are giving error messages and some kind of feedback about the work (17 of 26). A few of them give only error messages (6 of 26). Two of them only check the work and another only gives results. The

kind of feedback after errors is immediate, sometimes only with simple information about the kind of error or detailed feedback about each proof or exercise. In some cases, there is specialized feedback depending on the mistake; or the tool gives students immediate feedback for both correct and incorrect actions.

Pandora, for instance, works in two modes. In the basic mode, at every attempt to apply a natural deduction rule, either a success or a helpful error message is found. In the mode with an enhanced support, hints and explanations are also provided [7].

Another example of high feedback is the Proof Lab of Carnegie Mellon University. It is a sophisticated application for constructing proofs that allows students to make logical forward and backward moves. The Lab also diagnoses errors and provides error-reports with links to instructional material related to the mistakes [23].

Another example would be that Organon provides both exercises for student practice as well as graded work. It shows (step by step) the example solution of the exercise and provides a relevant explanation. It also provides assistance through checking each step of a student's solution on completion, and alerting when a mistake appears as well as showing the mistake (if required by the student), giving hints for the next step or performing that step. In fact it corrects the student's solution when finished (not during the process of solving) and comments it in the same way as graded homework [9].

[18] In the IDEAS project at the Open University of Netherland, the tool is a logical exercise solver. It helps students to rewrite formulae from propositional logic into a disjunctive normal form using standard equivalences. At each step, the tool provides feedback which depends on the kind of mistake the student might have made: the error-correcting parser suggests a correction, the tool tries to find a plausible rule the student intended to use and gives a correct application of this rule.

In the case of the SELL project of the Open University of Catalonia the AELL tool is an assistant in the learning process [13,14], providing both practice and graded exercises, immediate feedback and hints in every step.

As for the introduction of the logical notation, which is usually a problem, the tools studied had two kinds of solution: either a keyboard window (web editor) or the introduction of symbols in plain text (ASCII), from the physical keyboard, equivalent to the Logic symbols.

2.3 Assessment, Statistics, Reports

Automatic grading and statistics of student activity is a characteristic to be found in almost half of the tools. This functionality requires that students access to the tool individually and be identified. In some of these cases, the tool has an evaluation module that manages the development and delivery of work, qualification and recording of continuous assessment.

In the case of Pandora, for instance, students are provided with exercises that they download from the web-based Continuous Assessment System and the first time they save a proof (usually when they have completed it), their identity is coded. Then the tool can produce a report for each student and a summary of results for their tutors with minimal human intervention [7].

The web tutor Organon also has an Assessment Module to administrate students' homework. It generates the exercises, facilitates the production as well as the delivery

of the homework, manages to correct and grade the homework, stores the achieved results including a record of the exercise and gathers statistical data to provide feedback. There is always an automatic solving procedure associated with them. The resulting grades reflect not only whether the answer is correct or not, but also the whole procedure of solving because it is a proportion of how many mistakes the student has made, and how many stages he/she has completed successfully [9].

In the case of the AELL tool [14], the assessment module has been created to manage individual work of the students (both voluntary and compulsory work). There are two possible interfaces, one for each kind of user (student or teacher). Students are always identified while working with the tool and the logs produced are stored. For any kind of exercise, students can see the statistics of their voluntary work (completed, uncompleted, or pending exercises). Students are allowed to work on compulsory exercises until the deadline has expired. Continuous assessment tests are delivered with the tool and most of the grading is automatically calculated by using the associated solving procedure. From the point of view of the instructors, the assessment module is much richer. They have access to the individual or group statistics of many different elements: the average success of students, rules that were incorrectly introduced, frequencies of use, temporal distribution of the work, and as the system stores all the logs in a server of the university, many other statistics can be obtained.

The assessment characteristic, with grade reports and statistics has appeared recently in tools with the identification of student (especially in an online environment) and which provide learning for the majority of the content of the Logic course. It is the case in 8 of the 26 studied tools.

3 A Classification of Tools for Learning Logic

After establishing the fundamental categories for the study and classification of the different tools, the specification of the values in these categories has been assigned for each of them. The chosen categories have been 1) date of creation and last version, 2) platform (web, Windows, Appel, Linux), 3) developer's profile (university staff or not), 4) logical content, 5) student level, 6) language or languages, 7) functionalities, 8) introduction of logic notation (windows keyboard or plain text), 9) kind of interactivity, 10) kind of feedback, 11) help messages and advices, 12) license, 13) statistics and assessment, 14) identification of the user, and 15) the existence of a book or courseware reference.

Once the map of the values for each tool and each category has been made[4], the result allows the identifying of five groups, which are specified below.

- Provers. They are characterized by their automatism. In these types of tools, the user is totally passive. The system calculates and shows the solution of the exercise in an automatic way. It provides examples to the student. No interactivity at all. They are usually free licensed and of open access [Z].
- Checkers. They are characterized by the rather passive behaviour of the learner. The main activity is to verify deduction or formalization. The feedback of the tool

[4] A table showing the values in these categories for the different tools can be found in http://openaccess.uoc.edu/webapps/o2/bitstream/10609/6501/1/TableofToolsPaper.pdf

is restricted to find errors when the user requests explicitly the verification of the exercise being carried out. A message error is then sent, but with no more feedback or richer interactivity. Many of them are systems for practicing with Natural Deduction (ND) and other proof systems [U, W, Y]; others also include Truth Tables and Normal Forms [T, V, X]. They are usually free licensed and of open access.

- Assistants (also named constructors in [15]) are characterized by a higher degree of interactivity with the user. They provide menus and dialogues to the user for interaction purposes. This kind of tool gives the students the feeling that they are being helped in building the solution. They provide error messages and hints in the guidance to the construction of the answer. Many of them usually offer construction of solution in Natural Deduction proofs [K, L, M, P, Q] or other kind of proofs [R]. Others assist with Semantic Tableaux [N, O]. However they are not integrated in the rest of the logic courseware notation and contents. They are usually free licensed and of open access.
- E-tutors. They are characterized by the full integration with the rest of the resources of the Logic course. The tool inherits the central element of interactivity, feedback and hints also presents in Assistants. The difference is that it is totally related with contents, exercises, exams and occasionally assessment. Students are usually identified so the system can do statistics, grade reports or more complex assessment services. It is usually a web application, especially when the e-tutor is part of an online course in the university (the users are then usually the students and instructors of the course) [C, E, G, H]. Other kinds of e-tutors are provided with books offering a complete introductory course of Logic [A, B, D, F]; in that case, they are windows or apple platform with proprietary license. In this group, we can also find logic formulae in plain text (the older systems), but the trend is for Windows editors to introduce logic notation.
- E-tutorial. It is a tool having the characteristics of Assistants but integrated in a courseware. It does not offer assessment services, which is the main difference with E-tutors. We think that E-tutors are a development of those ones. We have found a free licensed one here [I] and a free tutorial to accompany a book [J].

The number of tools in each group and their relevance varies from group to group. There are 8 E-tutors, 9 Assistants, and 6 Checkers, while only 2 are E-tutorials and 1 is a Prover.

Among the E-tutors, 4 of them are web applications and 4 are Windows and Apple platforms. The web platform tools are related to specific courses and its tutorials are accessible only to students of the course, while non-web tools are associated with the commercialization of reference books under a proprietary license and with access for the users having the book. With respect to the introduction of logical notation, we find 50% using the Windows keyboard and the other 50% using plain text

Among the Assistants, the same distribution is repeated with 50% in each type of platform, all of them are using plain text to introduce Logic notation, and they are usually specialized in some logical content (5 of the 9 in Natural Deduction). Some of them also have a book of reference.

In the case of the Ckechers, there are again 50% in each type of platform, they use plain text to introduce notation, and they have a thematic similar to the Assistants, but without offering a high level of feedback and advice.

5 Conclusions

After this study of a large sample of current tools for teaching Logic, we can try to deal with the following question: what kind of computer-based tutor for teaching Logic is the one used in the 21st century? The first clear idea is that there is no standard kind of tool for teaching Logic. The way those tools have emerged, as research projects in many cases, may partly explain this variety. But perhaps the most obvious explanation is that the change in the type of teaching that requires the use of these tools is not standard yet. It is much easier to find them in online universities or as a component for self-learning of some introductory textbooks. This is the case of the E-tutors. Checkers and Assistants are commonly products of free access that could be used to support any part of the curriculum in face-2-face courses.

Students in a Logic course have to acquire a set of skills related with the complexity of logical reasoning. The immediate feedback and continuous assessment are very important in the learning process. In particular, intelligent tutoring systems could provide customized assistance and feedback to students. Moreover, in the context of e-learning, intelligent tutoring systems can help to overcome the absence of teachers while taking advantage of self-learning. There are many case studies of tools for learning logic [9,14,15,17,18,23] where Assistants and E-tutors seem to be the preferred kind of tools to help in the learning process of a subject like Introductory Logic.

In the lifelong-learning era, assessment usually occurs as the learner progresses through a computer-learning environment in order to provide support to carry out the tasks and determine whether the learner has accomplished its goals [8]. E-Assessment, which is most commonly known Computer-Assisted Assessments or On-line Assessment, has become a very integral part of study programs that are mainly conducted in an online environment. In face-to-face education, the use of E-assessments may help academics by reducing their workload of making large volumes of papers as well as by making the assessment-related administration task efficient. E-assessments can be categorized as Diagnostic, Formative and Summative assessments based on at which stage of the learning, the assessment is carried out. In particular, using a Virtual Learning Environment only for teaching and learning and then using a manual method of assessment is not very productive. Therefore in the e-learning paradigm it is needed to integrate the task of assessment. Both E-tutors and Assistants could then have a role in a model of e-assessment and in a model of continuous assessment as well. Furthermore, most of the evaluations of these tools confirm that students use them very little if they are not associated with the work being assessed.

It seems that E-tutors suggest a future for higher education in this context. Its integration in online courses has a clear meaning, covering all kinds of exercises with maximum interactivity, while acting as a guide in the learning process and, certainly, helping when the student requires. The time factor could also be a key element in the future, both for the opportunity to work individually in the allotted time, and in passing, this has to affect the current standard of school times. On the one hand mobile devices and accessibility of digital resources are important new trend in e-learning. On the other hand, knowledge technologies like the use of ontologies and the Semantic Web could help in building new intelligent tools for enriching the learning experience in subject like Logic.

Thus, an evolution of E-tutors to become more usable, ubiquitous, accessibility and intelligent tools could be the near future for computer-based tutors for teaching Logic.

Acknowledgments

This work was made possible through the support of the Spanish government grant FFI2009-09345/FISO; and the eLearnCenter (UOC).

References

1. Aczel, J.C., Fung, P., Bornat, R.: Using Computers to Learn Logic: Undergraduates' Experiences. In: Advanced Research in Computers and Communications in Education, vol. 1(55), pp. 875–882 (1999)
2. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive Tutors: Lessons Learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
3. Barwise, J., Etchemendy, J.: The Language of First-Order Logic. Cambridge University Press, Cambridge (1992)
4. Beal, C.R., Qu, L., Lee, H.: Mathematics Motivation and Achievement as Predictors of High School Students' Guessing and Help-Seeking with Instructional Software. J. Comput. Assisted Learn. 24, 507–514 (2008)
5. Beal, C.R., Walles, R., Arroyo, I., et al.: On-Line Tutoring for Math Achievement Testing: A Controlled Evaluation. Journal of Interactive Online Learning 26, 43–55 (2007)
6. Bornat, R., Sufrin, B.: Animating the formal proof at the surface: the Jape proof calculator, Technical Report, Department of Computer Science, Queen Mary & Westfield College, University of London (1996), ftp://ftp.dcs.qmw.ac.uk/jape/papers/
7. Broda, K., Ma, J., Sinnadurai, G., et al.: Pandora: A Reasoning Toolbox using Natural Deduction Style. Logic Journal of the IGPL 15, 293–304 (2007)
8. Collins, A., Halverson, R.: The Second Educational Revolution: Rethinking Education in the Age of Technology. J. Comput. Assisted Learn. 26, 18–27 (2010)
9. Dostalova, L., Lang, J.: ORGANON - the Web Tutor for Basic Logic Courses. Logic Journal of the IGPL 15, 305–311 (2007)
10. Endriss, U.: The Interactive Learning Environment WinKE for Teaching Deductive Reasoning. In: First International Congress on Tools for Teaching Logic, pp. 23–27 (2000)
11. Heeren, B., Jeuring, J., van Leeuwen, A., et al.: Specifying Strategies for Exercises. In: Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F. (eds.) AISC 2008, Calculemus 2008, and MKM 2008. LNCS (LNAI), vol. 5144, pp. 430–445. Springer, Heidelberg (2008)
12. Huertas, A.: Teaching and Learning Logic in a Virtual Learning Environment. Logic Journal of the IGPL 15, 321–331 (2007)
13. Huertas, M.A., Mor, E.: Tool Development to Support Learning, Immediate Feedback, and Continuous Assesment in Logic. In: Proceedings of the 6th International Conference on Web Information Systems and Technologies. INSTICC Press (2010)
14. Huertas, A., Humet, J.M., López, L., Mor, E.: The SELL Project: a Learning Tool for E-learning Logic. In: Blackburn, P., et al. (eds.) TICTTL 2011. LNCS (LNAI), vol. 6680, pp. 123–130. Springer, Heidelberg (2011)
15. Humet, J.: LSD, una herramienta didáctica para el aprendizaje de la lógica. In: JENUI (2001)
16. Kennewell, S., Tanner, H., Jones, S., et al.: Analysing the use of Interactive Technology to Implement Interactive Teaching. J. Comput. Assisted Learn. 24, 61–73 (2008)

17. Llorens, F., Mira, S.: ADN (Asistente Para Deducción Natural - Natural Deduction Assistant). In: First International Congress on Tools for Teaching Logic, pp. 65–69 (2000)
18. Lodder, J., Passier, H., Stuutmann, S.: Using IDEAS in Teaching Logic, Lessons Learned. In: International Conference on Computer Science and Software Engineering, p. 556 (2008)
19. Manzano, M. (ed.): Proceedings of the First International Congress on Tools for Teaching Logic. Universidad de Salamanca (2000)
20. Manzano, M. (ed.): Proceedings of the Second International Congress on Tools for Teaching Logic. Universidad de Salamanca (2006)
21. Moreno, A., Budesca, N.: Mathematical Logic Tutor - Propositional Calculus. In: First International Congress on Tools for Teaching Logic, pp. 99–106 (2000)
22. Popescu, E.: Evaluating the Impact of Adaptation to Learning Styles in a Web-Based Educational System. In: Spaniol, M., Li, Q., Klamma, R., Lau, R.W.H. (eds.) ICWL 2009. LNCS, vol. 5686, pp. 343–352. Springer, Heidelberg (2009)
23. Sieg, W.: The AProS Project: Strategic Thinking & Computational Logic. Logic Journal of the IGPL 15, 359–368 (2007)
24. Simón, A., Martínez, A., López, M., et al.: Learning Computational Logic with an Intelligent Tutoring System: SIAL. In: First International Congress on Tools for Teaching Logic, pp. 161–168 (2000)
25. Van Ditmarsch, H., Manzano, M.: Tools for Teaching Logic. Logic Journal of the IGPL 15, 289–292 (2007)
26. Van Ditmarsch, H.: Educational Logic Software, `http://www.ucalgary.ca/aslcle/logic-courseware`

A List of Studied Tools

A. Tarsk's World, http://ggww2.stanford.edu/GUS/tarskisworld/index.jsp
B. LPL-software, http://ggww.stanford.edu/NGUS/lpl/www-csli/LPL/
C. MLT-PC, http://aracne.usal.es/congress/PDF/AntonioMoreno.pdf
D. Logic Daemon, http://logic.tamu.edu/,
E. CPT-AProS (Carnegie Proof Lab), http://www.phil.cmu.edu/projects/apros/
F. LogicCoach III, http://academic.csuohio.edu/polen/
G. Organon, http://www.kfi.zcu.cz/organon/index_aj.htm
H. AELL: Logic E-Learning Assistant, http://cimanet.uoc.edu/logica/aell
I. Logic Café, http://thelogiccafe.net/PLI/
J. Power of Logic, http://www.poweroflogic.com/
K. Pandora, http://www.doc.ic.ac.uk/pandora/
L. Jape, http://users.comlab.ox.ac.uk/bernard.sufrin/jape.html
M. Plato, http://www.utexas.edu/courses/plato/
N. Socrates, http://www.utexas.edu/courses/socrates/index.html
O. blobLogic, http://corinhowitt.com/blob/blobSplash.html
P. AND (Natural Deduction Assistant), http://www.dccia.ua.es/logica/ADN/
Q. DN-FN, http://patrice.bailhache.free.fr/dnfn/deductioneng.html
R. IDEAS, ftp://ftp.computer.org/press/outgoing/proceedings/csse08/data/3336i553.pdf
S. TPS and ETPS, http://gtps.math.cmu.edu/tps.html
T. Interactive Logic http://www.math.uwaterloo.ca/~snburris/htdocs/LOGIC/st_ilp
U. Bertie3, http://selfpace.uconn.edu/BertieTwootie/software.htm
V. Summa Logicae XXI Software, http://logicae.usal.es/
W. ProofWeb, http://proofweb.cs.ru.nl/login.php
X. Gateway to logic, http://logik.phl.univie.ac.at/~chris/gateway/formular-uk.html
Y. DC Proof, http://www.dcproof.com/
Z. Logic Animations, http://staff.science.uva.nl/~jaspars/animations/

Logic in Action*

An Open Logic Courseware Project

Jan Jaspars and Fernando R. Velázquez-Quesada

Institute for Logic, Language and Computation, University of Amsterdam,
Science Park 904, 1098 XH Amsterdam, The Netherlands
{J.O.M.Jaspars, F.R.VelazquezQuesada}@uva.nl
www.logicinaction.org

Abstract. Today, logic is taught to a much broader academic audience than just the traditional target group of mathematicians and philosophers. Logic in Action provides modern open courseware, in the form of a freely accessible animated e-book, as an extensive teaching tool for this growing student population. The project stresses the methodological merits of logic for a wide range of interdisciplinary sciences.

1 Introduction

In the Netherlands, logic is taught for a much broader group of students than what we were used to in earlier days. There are several reasons for this increase of the academic dissemination of logic. Most importantly, undergraduate programs at Dutch universities have changed considerably during the last two decades. One of the most significant changes has been the introduction of new interdisciplinary studies that offer a wider range of scientific subjects in one program, and therefore tend to attract more and more students who have just left secondary school. They prefer this stepping stone option because it offers them introductory courses on different subjects of study before fine-tuning their academic careers.

Logic, as an interdisciplinary science par excellence, fits very well within the curricula of these kind of studies. At the University of Amsterdam, for example, undergraduate programs like 'exact and social sciences' (Bèta Gamma) and 'liberal arts and sciences', welcoming together more than 300 starting students each year, have incorporated an introductory course in logic within their first year's program.

Besides this general academic trend, different departments have initiated other new studies in which logic education is involved. Examples of these are artificial intelligence at the computer science department, cognitive science at the social

* Logic in Action is a project financed by the Dutch Ministry of Science and Education (the so-called Beta Platform program) and is organized by the Center for Creation, Content and Technology (http://ccct.uva.nl/content/project/beta). Many other persons are involved in the project than only the two authors of this paper. The project has been initiated by Johan van Benthem (University of Amsterdam), Jan van Eijck (Center for Mathematics and Computer Sciene Amsterdam) and Hans van Ditmarsch (University of Seville).

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 141–153, 2011.

science faculty and computational linguistics at the department of humanities. Altogether, these relatively new programs meant a considerable increase of the local undergraduate consumption of logic, since in earlier days logic was only taught on an undergraduate level to students of mathematics, computer science and philosophy.

Although these academic tendencies are very beneficial for the spreading of the education of logic, there is a much more important 'internal' reason for the more dominant position of logic in academia. The development of logic, mainly caused by its influence and applications in computer science since the fifties, has changed its academic presentation dramatically. Whereas traditionally logic was taught as the abstract 'meta-science of valid reasoning' at mathematics and philosophy departments, modern logic education today focusses very much on the mathematical and computational methodology it offers for a wider range of studies. Every academic field that studies representation and exchange of information in one way or another, either by machines or real humans, requires logic as one of the fundamental mathematical tools.

1.1 The Logic in Action initiative

Logic in Action is an educational initiative which started at the University of Amsterdam, in cooperation with the Tsinghua University (Beijing, China) and the Stanford University (California, USA), in 2009. The main objective of the project is to provide new teaching material for modern introductory logic courses for the wider academic audience mentioned here above. Besides classical logic, the project focusses on modern logics, such as dynamic and epistemic logics, which provide formal and computational methods for representation and exchange of information rather than solitary reasoning.

Logic in Action provides open courseware in the form of an extensive animated e-book, which can be read and studied on computers, but also on other modern electronic devices such as certain type of e-readers, tablet computers and even modern smartphones. The e-book consists of texts with many animated illustrations, and also diverse applications: small ones that visualize the most important concepts, and larger ones that can be used to exercise the most important methodological skills. The complete electronic manuscript has been set up in such a way that the reader — maybe 'the user' would be a more appropriate typology here — may switch from text to animations or applications as smoothly as possible (single touch or click).

This paper is meant to give a bird's-eye view on what Logic in Action has to offer for introductory courses in logic. Section 2 focusses on the contents and the educational motivations behind it. Section 3 presents the electronic extensions of the e-manuscript together with some illustrative examples. Section 4 elaborates on the near future of the Logic in Action project, which goes in three directions: specialized advanced logic courses for graduate students, the creation of open courseware for pre-academic education on secondary schools and also an additional part that we are working on at the moment: learning logic by programming, and vice versa.

2 Manuscripts

Our courseware is based on the idea that today logic should be taught as a *methodological* science which is applicable in many different areas. The ultimate tool for teaching logic is by demonstrating the tools logic has to offer for those fields of study.

What comes first in our set-up of our courseware is *logical design*. In order to study a given phenomenon, we first capture the formal structures (models) that represent it, and then a formal language which allows us to describe these structures. The modelling enables us to abstract away from details to identify the key aspects of the phenomenon. The formal language facilitates a precise description and study of these properties in a systematic way. Once models and language have been defined we are ready to introduce *logical computation*, i.e., formal inference systems. The main goal of the open courseware is to look at logic from this broader methodological perspective. Instead of presenting it as an abstract isolated science that focusses merely on truth-conditions and truth-preserving reasoning, logic is presented as a methodology that takes the 'art' of *modelling* as its starting point.

In order to achieve this goal, the manuscript is divided into three main parts. First, as an introduction of the basic logical concepts, three classical systems are presented: propositional logic, syllogistic reasoning and predicate logic. Special emphasis is put on the semantic models in which formulas of these systems are evaluated in order to underscore the use of the languages as a tool for describing structures. Then we move to more recent logical developments that shift the main focus from the notion of truth to the concept of information. The fundamental topics that this new perspective puts into light are threefold: knowledge and the way it changes (dynamic epistemic logic in a broad sense), abstract actions (dynamic logic), and the way information and actions work together in multi-agent environments (interaction). Finally, after showing how formal structures and formal languages are useful for representation, description and therefore the formalization of different phenomena, the manuscript finishes with an extensive part on three different sorts of computational methods.

The remainder of this section presents a more detailed description and motivation for each of the three parts that constitute the core of the lecture notes.

2.1 Classical Systems

Traditionally, logic has been understood as the study of valid patterns of reasoning, and many systems have been developed with the aim of formalizing what a valid inference is. The most important classical frameworks take the truth-value of a sentence to be an objective concept, completely determined by the sentence's components and completely detached from any personal considerations. This is what has made mathematics the paradigmatic example of classical logic, with precise definitions setting the basic ground, and then the formalization of sound proofs showing what follows from initial assumptions.

The fundamental logical framework is that of propositional logic. This logic defines valid patterns of reasoning with sentences whose basic components are abstract propositional variables. One of the most important features of propositional logic is that it gives formal meaning to *logical connectives* (sentential operations) interpreted as Boolean functions. The models for formulas of this language are simple *truth-value assignments*. Our student first learns about compositional semantics on the basis of these simple models, and then uses this to distinguish valid from invalid inferences.

At this point there is not much difference with a traditional first introduction to the key constituents of a logical system. Despite its simplicity, we use propositional logic to offer our student a broader modernized perspective on logic. For example, besides the inevitable truth-tabling we also use the so-called *update* definition of validity which sheds another light on logical inference. The premises of an inference are seen as pieces of information that reduce the initial uncertainty of the reasoner. When all the premises have been processed in this way, a conclusion is then defined to be valid if it does not add any new information (no further reduction of uncertainty). Another informational perspective we use on this basic level is that of truth-value assignment of a propositional formula as a *game*, or rather a *debate*, between two opposing parties: one defending its truth (the verifier) and one claiming its falsity (the falsifier).

Still, as a purely abstract formal system, propositional logic can easily be underestimated as a simple innocent logic by a starting student. Besides many puzzles that we have added to the text to bring propositional logic more to life, we focus on more serious applications and deeper mathematical understanding of the impact of propositional logic. The most manifest example is binary arithmetic as performed by logical circuits built up from the propositional logical operations. Our student learns how these circuits work and, moreover it is taught that this can all be done due to the expressive wealth of the propositional language, i.e., its functional completeness. In addition we focus on the rude complexity of propositional logic. There is a computational price (NP-completeness) to be paid for the surprising richness of basic system.

The second framework is that of syllogistic reasoning. It 'opens the box' of propositional logic by looking closer at what an atomic proposition actually expresses: very often it is not about abstract sentences, but about objects and their properties. Syllogistics focuses on simple quantification patterns, like "All P are Q" and "Some P are Q", and it provides the first steps towards the more general system of predicate logic.

On the other hand, syllogisms can also be seen as a special form of propositional reasoning, and we demonstrate how syllogistic inferences can be dealt with by information updates very much in line with the procedure we have given for propositional logic. In the case of syllogistic reasoning, universal information like "All P are Q" can be processed as the removal of the P-individuals who are not Q. As a consequence of this informational interpretation it can be shown that, due to its moderate expressiveness, the computational complexity of syllogistic reasoning is lower than that of full propositional logic. This shows our student

that if a logic system is brought back to a natural but limited fragment there may also be some important computational benefits to be gained.

The strongest of the logical frameworks that are presented is that of *predicate logic*, extending syllogisms to deal not only with objects and their properties, but also with relations of arbitrary arity between them, all these combined in arbitrary forms of quantification. This is the most important system in logic today because it is a universal language for talking about formal *structures*, that is, collection of objects together with their properties and relations. Predicate logic has been used to increase precision in describing and studying structures from linguistics and philosophy to mathematics and computer science.

The language of predicate logic is much harder to get acquainted with for many beginning students. To overcome this difficulty, much attention in the text is given to translations of phrases of natural language to predicate logic making use of the more accessible quantificational patterns of syllogisms. Students are trained to write formulas first as trees to recognize the composition of such patterns, and then rewrite those in the standard linear logical form.

Another difficulty of predicate logic is that its semantics is relatively much more complex than that of propositional logic and that of syllogistics. The text therefore focusses first on the *informal* semantics of pictures and finding the predicate logic formulas to distinguish two of such pictures. Moreover, much attention has been given to the role of free variables as to make the working of variable assignments more comprehensible (see also Figure 1).

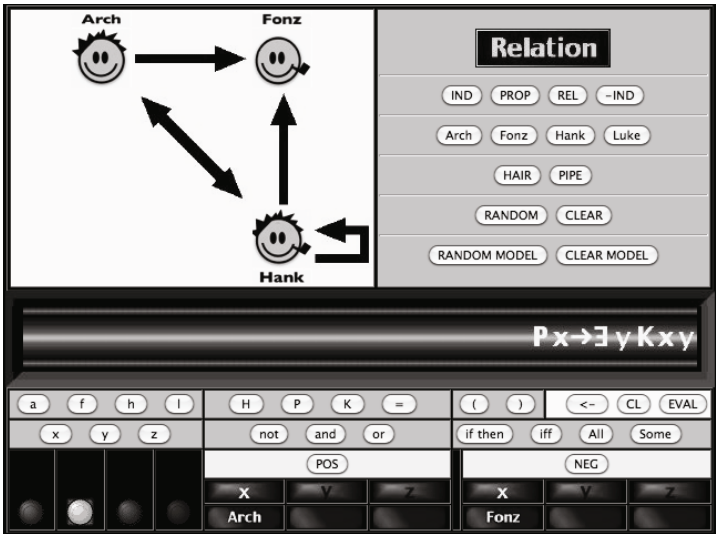


Fig. 1. The art of modeling in the context of predicate logic. This is a small application from the electronic document which illustrates model evaluation evaluation in first-order models. What is important here is that it also illustrates the working of assignments of free variables.

2.2 Knowledge, Action and Interaction

As we mentioned before, logic has been traditionally taken to be the objective study of valid reasoning, with truth-preserving inference as the most important concept and mathematical logic as the paradigmatic area. But with the emergence of artificial intelligence, the 1980s witnessed the development of many logical systems that focussed on what are nowadays called "rational agents". These proposals expanded the notion of logic: from the study of the objective notion of *truth* and the reasoning patterns that preserve it, to the study of subjective notions of *information* and the diverse actions that change it.

Three fundamental themes play a role here. The first one is the study of *information*, its diverse representations and its properties. Then, this information may provoke agents to perform *actions*, thereby changing the objective circumstances — the actual state of affairs — as well as the subjective context — the information that the agents have about the state of affairs. The third theme is *social interaction* which ties up the previous two, and enables the study of the most interesting forms of action and information flow which involves not a single but multiple agents sharing their private information by communication. For each of these main themes, the manuscript presents a logical system to formalize the most important concepts.

The first of these themes is best exemplified by epistemic logic, a logical framework that differs from classical logic in that it incorporates the reasoner(s), i.e., the agent(s), within the system. The epistemic logical language and its semantic models facilitates the representation of the private knowledge the agents have about the actual world as well as the knowledge they have about their own and other agents' knowledge.

As a simple extension of epistemic logic for modelling communication, the manuscripts outlines the so-called public announcement logic. At first, *dynamic* operators appear on stage to represent the effect of such public announcements. In addition, operators are introduced in order to express group-related notions of information, with common knowledge being the most representative. As these announcement acts are taken to be 'really public', the optimally attentive agents acquire new information, but they also get to know that the other listeners have gained the same information, and all those agents share this knowledge about their knowledge and so on (see Figure 2). The infinite nature of the common knowledge notion makes the underlying logic mathematically distinct from the classical logics of the first part of the book.[1]

In an additional chapter the manuscript focusses on the propositional dynamic logic framework, which allows us to study actions in a more abstract way. Propositional dynamic logic incorporates a more extensive combinatorics of actions and is as such a very powerful tool in many research areas. Originally the formalism has been introduced in order to develop systems to derive information about the behavior of computer programs, but nowadays it serves as a general logic of actions.

[1] For example, such extended epistemic logics lose the *compactness* property.

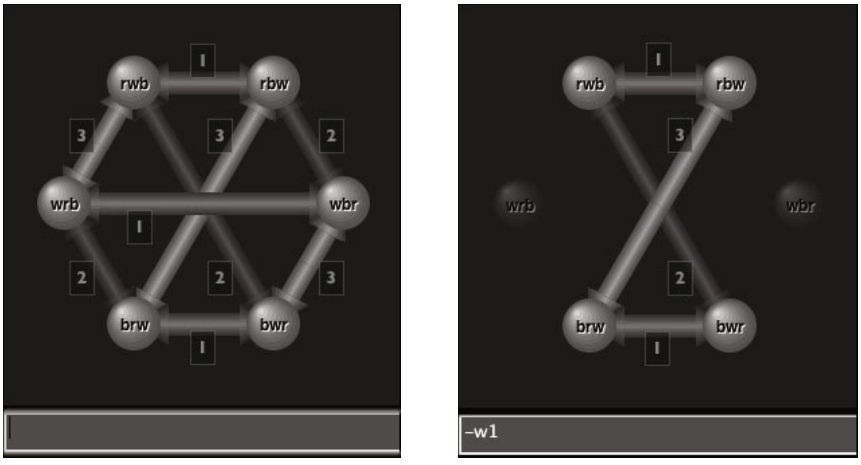


Fig. 2. A small app from the electronic document which visualizes the working of *Public Announcement Logic*. The example is about a very simple card game, three players holding a card of different colors, red, white and blue. The players can see their own card but not those of the other players. The program takes new input as public announcements, and administrates the effect as elimination of possibilities. In the second picture the first player just told (publicly) that he does not have the white card.

The third topic combines the previous two in a multi-agent environment, and deals with the broadest conception of logic nowadays: the study of agents, their private and shared attitudes towards information (knowledge, beliefs, preferences, intentions, desires, etc.), and the way these attitudes change due to the interaction between those agents. A paradigmatic application of this new understanding of logic is the analysis of competitive multi-agent situations, commonly known as *games*. Games have always played an important role in the history of logic, witness the many sorts of interactive games with two opposing players that have been used as tools for demonstrating the validity or invalidity of inferences, or for determining the truth or falsity of arguments in a dialogue. But this broader conception of logic shows how the relation works in both ways: not only games are vehicles for defining logical concepts, but logic also provides tools for analyzing games, focussing not only on their structure (the actions the players may undertake) and their payoffs (the agent's *preferences*), but also on the information agents have about each other and the way this is affected by the different actions that take place.

For the Logic in Action project this part of the book is the most challenging. Although we introduce all the aforementioned topics on an elementary level, our student becomes familiar with recent developments of logic and, at the same time, also learns about the new and central position that logic has taken up in the era of information and communication.

2.3 Methods

The final part of the lecture notes elaborately introduces three different formal systems for computing the validity of logical inferences. We start off with the tableau method since it fits best with the model-theoretic perspective of the first two parts of the book. Moreover, it is the most general method of the systems discussed in this part, because it is a testing method. Invalidity of a given inference can be determined by means of the construction of a counter-model. Validity of such an inference, on the other hand, is then shown by proving the non-existence of such counter-models. An additional important didactic feature of the tableau method is that it shows quite straightforwardly how reasoning can be implemented within computer programs.

On the other hand, a clear disadvantage of the tableau method is that it is a refutational method: instead of showing directly the truth of a given conclusion under certain circumstances a tableau demonstrates that it can not be false. An obvious consequence is that a tableau computation does not always reflect "the way people think". In the second chapter we therefore outline the method of natural deduction as a more 'human' system. The reader is taught how characteristic complete systems of axioms as introduced in the first two parts of the book can be uplifted to natural systems of reasoning by facilitating conditional reasoning by means of temporary hypotheses.

The third and final chapter of this part of the book is completely dedicated to first-order theorem proving by means of resolution and unification. This is probably the most technical chapter of the Logic in Action manuscript as it stands now, and may therefore not be suitable for every introductory course in logic. Despite the technical nature of this topic, we have chosen to incorporate this formal method since it exemplifies the practical nature of a logical formalism invented by philosophers, like Frege and Peirce, more than a hundred years ago. Today a student may as well learn predicate logic as a very useful programming language as this chapter illustrates.

3 Electronic Support

In order to provide a better understanding of the material described in the previous section, the e-book provides digital support for the users (students and teachers) in the form of animated illustrations and different sorts of applications. The former are meant as step-by-step — or rather click-by-click — visualizations of the most important concepts, and the latter are mainly meant for training logical skills. These two electronic extensions of the document are conceived as tools for the student to master technical methods, but they can also be used by teachers for demonstration purposes.

In addition to this technical support, the full electronic version of the book contains optional teaching material that relates the main logic topics with other sciences. Since the manuscript focusses on the technical methodology that formal logic offers, a student may easily lose track of the meaning of the different forms

of symbolic manipulation that he is learning. In order to put these methodological developments in a proper context, the text contains several 'outlooks' that highlight important applications of the presented material. These outlooks allow the students to see the subject from different perspectives, suitable for different backgrounds and different objectives.

In the remainder of this section we will elaborate a bit further on the electronic part of our courseware.

3.1 Animated Illustrations

Visualization is an important tool for learning formal logic. Animated illustrations provide an important enhancement of our means to picture examples of technical concepts. Firstly, animations give us the possibility of building illustration in a step-by-step fashion, therefore allowing us to provide, at each stage, a proper explanation. Moreover, the examples can be replaced by other examples, either chosen by the reader himself or generated automatically.[2] In Figure 3 we have given an example of an animated illustration from the second part of the book. The reader can freely change such examples.

3.2 Applications

Besides animations the book also contains larger applications which assign a more (inter)active role to the reader. We have built several interactive applications to make it easier for the user to exercise technical skills such as model checking, model construction and setting up derivations in formal deduction systems such as axiomatic systems, semantic tableaus, natural deduction and resolution. This sort of applications have clear advantages for exercises which involve logical derivations and computations. Editing such formal structures becomes much easier and the programs check the results. This is an advantage since, in ordinary books, the provided solutions may be misleading to the reader: his own answers may be correct as well. In Figure 4 two applications are shown as examples of applications for logical derivations.[3]

3.3 Additional Contents

Besides interactive extensions, the e-book also contains other additional material. The main purpose of these additions is to offer a broader perspective on logic for specific target groups of students. They consist of applications of logic, deeper theoretical results for the methodologies as unfolded in the main text, the philosophical and historical backgrounds of the development of logic and external links to other online documents.

[2] The final electronic version of the e-book is written in HTML (5), and the animated illustrations are coded in JavaScript such that these animations run within the electronic text.

[3] A minor technical disadvantage of these larger applications is that they have to be run in a separate window of the document browser.

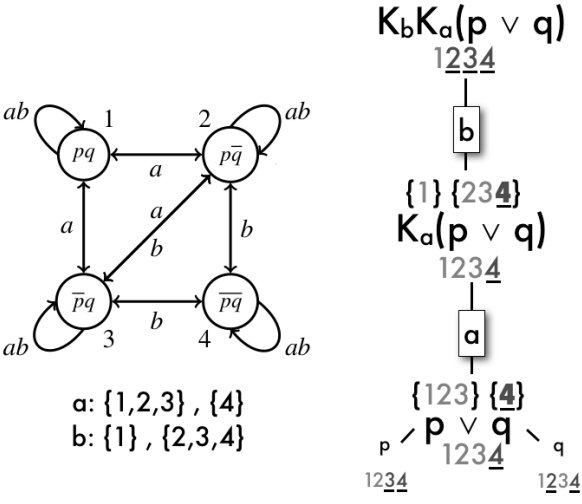


Fig. 3. This is a screen dump of how the final stage of an animation of Kripke (possible world) models are defined and how evaluation of modal formulas take place. As we found out in try-outs of our courseware students find it hard to capture all the formal details of the definition of such models. This animation generates new models on the spot for limitless training by examples. The evaluation of the formulas are built up step-by-step as to understand the compositional aspect of evaluation.

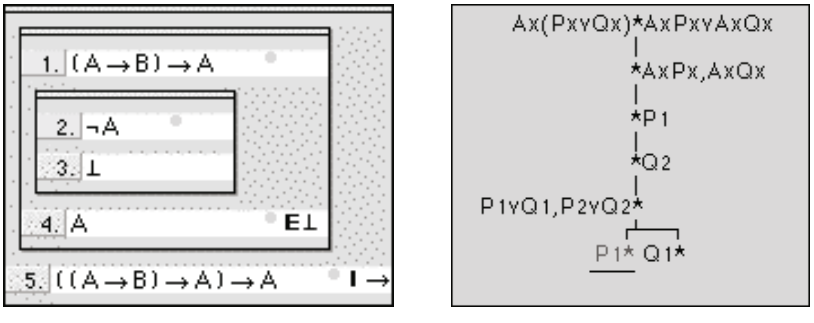


Fig. 4. Two important applications for methodological training. On the left the reader is on his way to prove Peirce's law by natural deduction (Fitch style). The editor for building the proofs facilitates bridging gaps in the proof and the program checks the soundness of every step made. On the right hand our reader is decomposing a tableau (Beth style) to test the validity of a chosen inference.

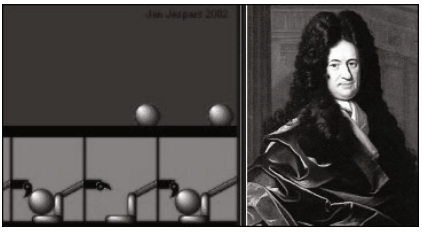


Fig. 5. A fragment of an animation of Leibniz' 'mechanica dyadica', the first binary computer (1705). This animation comes from a bibliographical outlook in our first chapter on propositional logic where the relation with binary computation is explained to our readers.

Outlooks. The most important additions in the e-book are outlooks which reflect on the applications of logic today. Such perspectives are very important to motivate the study of logic. Because a specific application may be appealing to just a selection of our potential readers, we have chosen to put these parts in the e-book version as optional extensions of the manuscript. The full electronic document is organized in such a way that the manuscript can be extended smoothly with these outlook sections.

In addition to applications of logic we have also added some outlooks on theoretical issues which are introduced on an informal level. They are meant as eye-openers for the reader, but as the course is on an elementary level these subjects are not dealt with in full technical detail. Currently we started working on a fourth part of the manuscript for graduate students for logical meta-theory (see also Section 4).

The historical and philosophical context of logic. The second group of optional extensions of our manuscripts focusses on the history and the philosophical context of logic. They provide basic background information for students of logic courses, allowing them to reach a deeper understanding and appreciation of the logical methodology. As an example see Figure 5 which shows an animated version of the first binary computing device. Although this invention has only been described on paper by its inventor Leibniz (1705), in the e-book version of our manuscript the reader can make it run.

External resources. As a program of open courseware we also benefit from many other world-wide educational initiatives of this kind. The e-book includes many references to other external resources for the teaching of logic.

Logic in Action is free and open courseware, and will therefore continuously be extended and updated. Especially the electronic part of the book will be extended considerably in the near future. But there is more what we will be working on to extend the scope of the Logic in Action project. In the next section we will outline this in a brief prospectus.

4 Logic in Action Today and in the Future

There are three major directions that the Logic in Action group is currently working on: practical logic training for students of computer science and related subjects such as artificial intelligence and computational linguistics, a full new part on the meta-theory of logic for graduate courses, and finally a series of e-books in the Logic in Action style for pre-academic education for secondary schools.

4.1 Logic in Action at Work

A particularly important development that Logic in Action is working on at the moment is a technical appendix that provides instructive material for learning how to build tools of logic. This additional part is specially meant for students in the field of computer science. The functional programming language `Haskell` is used to instruct students how they can build relatively small applications to let the logic tools really do the work.[4] In this part we will specifically aim at applications for the logics that are taught in the second part of the manuscript: epistemic logics and dynamic extensions of those, which are particularly interesting for the aforementioned group of students.[5]

At this moment this part of the course is incorporated as a technical annex, but in the future it will take a more dominant place within the Logic in Action project. For a large group of students today this type of 'hands on'-training works much better than theoretical education. Another obvious advantage is that the students learn logic and computer programming at the same time.

4.2 Graduate Courses

Another part that we are working on is an extension of our courseware meant for more advanced graduate and post-academic courses in logic. For students of mathematics and theoretical computer science we will add chapters about the meta-theory of logic. The most important themes will be completeness and incompleteness results, complexity theory and more advanced model-theory. Other topics which are planned to be incorporated are non-classical logics such as intuitionistic logic, non-monotonic logic and extended modal logics.

4.3 Pre-academic Education

In secondary schools, especially those for pre-academic education, the mathematics curricula have been extended considerably in the last fifteen years. One important reason for this change in basic pre-academic education of mathematics is the role of computer science. Whereas classical mathematical training aims at the application of mathematics within the traditional natural sciences such as physics, chemistry and biology, today pupils in Dutch secondary schools can

[4] For a textbook which shows the use of `Haskell` in this context see [2].
[5] What we have in mind here is very much in line van Eijck's `DEMO`- system [3].

choose to extend this with alternative mathematical programs for other scientific fields in which computer science and related subjects take a dominant place.

In an earlier logic education project at the University of Amsterdam, which was called 'The Dissemination of Logic' which made part of van Benthem's earlier Spinoza Project (1997–2000),[6] we have published several logic textbooks for these extended programs in Dutch secondary schools [6].[7] Logic in Action is currently working on supplementary electronic editions of these textbooks in English.

5 Conclusions

The open courseware project Logic in Action develops modern course material for the teaching of logic to students with different academic backgrounds. In the Age of information and communication formal logic is one of the most important mathematical tools that these new generations will need. Logic in Action contributes to this in two respects: in content as well as in didactics. In addition to the ordinary classical standards of a course in logic, Logic in Action provides elementary undergraduate teaching of modern logics for modelling communication. As for the didactic part, Logic in Action makes use of animated e-books to facilitate the teaching of logic.

Acknowledgements. We like to thank Johan van Benthem, Hans van Ditmarsch and Jan van Eijck who initiated the Logic in Action project. We also want say thanks to Dora Achourioti, Cédric Dégremont, Nina Gierasimczuk, Tomohiro Hoshi, Lena Kurzen, Fenrong Liu and Stefan Minica who have been using the earlier versions of the Logic in Action manuscript in their logic courses during the last two years and have provided very valuable comments and suggestions on the basis of their teaching experience with Logic in Action.

References

1. van Benthem, J., van Ditmarsch, H., van Eijck, J.: Logica in Actie. Spinoza-reeks Open Universiteit Heerlen. Academic Service, Den Haag (2009)
2. Doets, K., van Eijck, J.: The Haskell Road to Logic, Maths and Programming. Texts in Computing, vol. 4. King's College Publications, London (2004)
3. van Eijck, J.: Demo — a demo of epistemic modelling. In: van Benthem, J., Gabbay, D., Löwe, B. (eds.) Interactive Logic — Procs. of The Seventh Augustus de Morgan Workshop. Texts in Logic and Games, vol. 1, pp. 305–363 (2007)
4. van Eijck, J., Jaspars, J., Ketting, J., Pauly, M.: Denkende Machines: Computers, Rekenen en Redeneren. Exact in Context, vol. 1. Amsterdam University Press, Amsterdam (2001)
5. van Eijck, J., Visser, A.: Bewijzen en Inzien. Exact in Context, vol. 2. Amsterdam University Press, Amsterdam (2007)
6. Jaspars, J.: The dissemination of logic in dutch secondary schools. In: Procs. of The First Conference on Tools for Teaching Logic, Salamanca (2000)

[6] See `http://www.illc.uva.nl/lia/`
[7] For those who read Dutch: See the booklets [4], [5] and [1].

A Teaching Tool for Proving Equivalences between Logical Formulae

Josje Lodder and Bastiaan Heeren

School of Computer Science, Open Universiteit Nederland
P.O.Box 2960, 6401 DL Heerlen, The Netherlands
{josje.lodder,bastiaan.heeren}@ou.nl

Abstract. In this paper we describe a teaching tool for proving equivalences between propositional logic formulae, using rewrite rules such as De Morgan's laws and double negation. This tool is based on an earlier tool for rewriting logical formulae into disjunctive normal form (DNF). Both tools make use of a rewrite strategy, which specifies how an exercise can be solved stepwise. Different types of feedback can be calculated automatically from such a strategy specification. We describe a strategy for constructing expert-like equivalence proofs, and present two techniques for improving the proofs that are generated by the strategy.

Keywords: propositional logic, equivalences, e-learning, feedback.

1 Introduction

The construction of proofs is an important topic in logic courses. Several tools have been developed in the last decades to help students in acquiring proving skills [3,9,10]. Most often, natural deduction is used as a proof system, but also axiom systems are used. When students have to apply logic, they also have to simplify and rewrite logical formulae, and this takes some practice. For instance, a computer science student should be able to simplify a database query, or recognize that two database queries are equivalent. A typical example of applying logic is to simplify the following SQL query (using fewer negations):

```
SELECT s.name
FROM   Students s
WHERE  NOT (NOT (s.subject = math) OR s.startdate = 2010)
         AND s.grade >= 8)
```

Other examples of rewriting logical formulae are the simplification of conditional expressions found in most programming languages, specifying and reasoning with business rules, and turning logical propositions into Prolog clauses.

At the Open Universiteit Nederland, we have been working on several interactive exercise assistants, including a tool to train students in transforming a propositional formula into disjunctive normal form (DNF) [7]. These tools are based on a strategy language [6], in which rewrite strategies for solving exercises

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 154–161, 2011.

can be expressed (e.g., converting a formula into DNF). From such a strategy specification, different types of feedback can be calculated automatically, such as providing hints on how to continue, recognizing an intermediate step submitted by the student, and generating worked-out solutions.

We recognize the importance for computer science students to be able to manipulate logical formulae using rewrite rules. In this paper we discuss how our logic tool can be extended with exercises in proving the equivalence of propositional logic formulae. This paper makes the following contributions:

- We describe a strategy for constructing expert-like equivalence proofs (i.e., proofs that appear non-mechanical). The strategy is illustrated by a number of example proofs that are generated by the strategy. Our approach is general, and therefore applicable to constructing proofs in other areas.
- Our claims are supported by a prototype implementation of the strategy for constructing proofs. We highlight the changes that are needed to the tool for rewriting formulae into DNF.

The remainder of this paper is structured as follows. We first introduce our web-based exercise assistant for rewriting logical formulae into DNF in Section 2, where we briefly discuss our approach for developing interactive exercise assistants based on rewrite strategies. Section 3 then presents a strategy for proving equivalences between formulae and two techniques for improving the proofs. Examples of proofs that are generated by our strategy are given in Section 4. The last two sections discuss related work and draw conclusions.

2 Rewriting Formulae into DNF

We start with an overview of our tool for rewriting arbitrary formulae into DNF. Most of its functionality can be reused for exercises in proving equivalences. Figure 1 shows a screenshot of the interactive exercise assistant[1]. Students have to rewrite a formula into normal form, using a fixed set of allowed rewrite rules. At each point, hints are available about the next step, or an example solution can be shown. More importantly, students also receive feedback when they submit intermediate answers. The tool identifies the rewrite rule that was used, or it tries to recognize a common misconception (also known as *buggy rule*) in case the answer is incorrect. For example, Figure 1 contains a feedback message about the incorrect application of De Morgan (i.e., the buggy rule $\neg(\phi \vee \psi) \Rightarrow \neg\phi \vee \neg\psi$).

Feedback is derived automatically from a strategy specification. Such a strategy describes the order in which rewrite rules have to be applied to solve a particular type of exercise. Strategies for reaching DNF have been reported in [6]. When a student deviates from the strategy, this can be detected and reported by the tool. We currently allow these deviations, also because they may prove to be clever shortcuts. For practical reasons, associativity of conjunction and disjunction is silently performed by the tool. On the contrary, commutativity requires an explicit step by the student.

[1] The DNF tool is available at: `http://ideas.cs.uu.nl/genexas/`

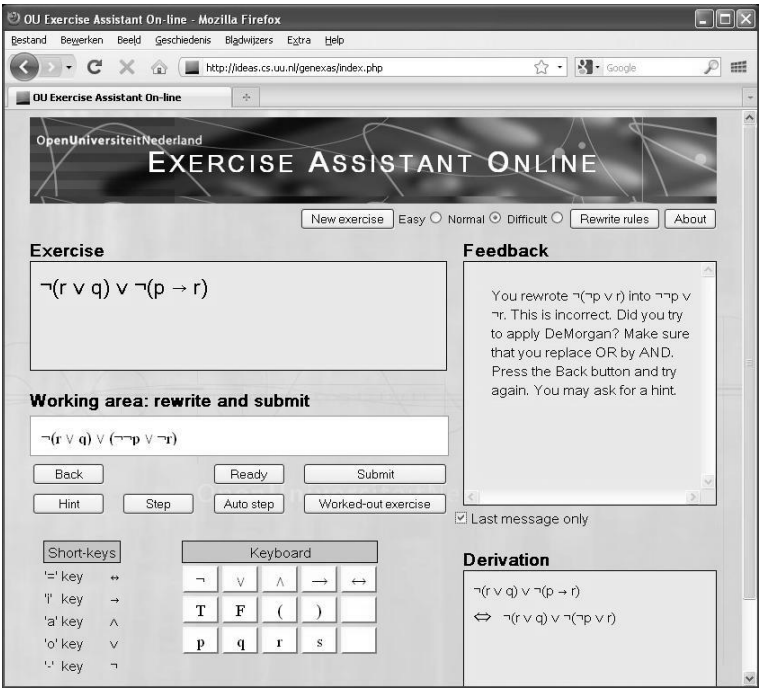


Fig. 1. Screenshot of the Exercise Assistant for rewriting formulae into DNF

The tool has been tested with bachelor's students in a course on discrete mathematics. The results of the test were very promising [8]. The tool helped the students in understanding logical formulae and improving their rewriting skills.

3 Proving Equivalences between Formulae

We will now discuss the design of a tool for practicing the construction of equivalence proofs: the DNF tool is a good starting point for this. The tools operate on the same type of formula, and the same collection of rewrite rules (e.g., commutativity, absorption, and the buggy rules) can be used. Because we base our approach on rewrite strategies, the full machinery for analyzing steps, recognizing common misconceptions, generating hints, and providing feedback is readily available. Changes to the following components are required:

– Modifications to the user interface are needed. Students should be allowed to perform a forward step or a backward step, at all times. Suppose a proof is asked for the equivalence $\phi \Leftrightarrow \psi$. Then ϕ can be rewritten to ϕ' (forward step, giving the task of proving $\phi' \Leftrightarrow \psi$), or ψ to ψ' (backward step, resulting in $\phi \Leftrightarrow \psi'$). The proof is completed when ϕ and ψ have been rewritten to the same formula. The user interface should accommodate for steps in both directions.

- A strategy is needed for proving equivalences in an expert-like way. This is the most significant extension to our tool. The strategy for reaching DNF will play a prominent role in the strategy specification.
- Lastly, the creation of exercises is different. In the DNF tool, we use a random formula generator. In the case of proving equivalences, we restrict ourselves to a fixed set of exercises, such as the ones in Section 4.

3.1 A Strategy for Proving Equivalences

The general idea for proving the equivalence $\phi \Leftrightarrow \psi$ is rather straightforward: rewrite both ϕ and ψ to DNF, and then rewrite these normal forms to equal forms using a given set of rewrite rules. This approach results in proofs that appear mechanical. For example, consider $\neg(p \wedge (q \rightarrow r)) \Leftrightarrow \neg((q \rightarrow r) \wedge p)$. Suggesting to eliminate the implication, or to apply De Morgan's law at one of the sides, is not very reasonable. Therefore, we make several refinements to the strategy. The strategy consists of two parts (explained in sections 3.2 and 3.3):

- Part 1: rewrite ϕ and ψ into normal forms, but search for parts that do not have to be rewritten at each step.
- Part 2: to finish the proof, rewrite the normal forms into equal forms. This is only needed when the normal forms are different.

The strategy is used by the tool to give hints and to provide sample solutions, and these are available at each moment. The strategy alternates between making forward and backward steps, just like a student. In fact, it rewrites a pair (ϕ, ψ) by applying a rule either to ϕ or ψ, leaving the other formula unchanged. Whenever a hint is asked, the next step is computed from the strategy. Although it might be feasible to calculate the *shortest proof* for simple exercises (i.e., calculate the proof once, before a student starts with the exercise), such an approach becomes impractical as soon as students are allowed to depart from this proof. In such a scenario, a shortest proof has to be calculated repeatedly. It is unclear how this can be done efficiently.

3.2 Towards Disjunctive Normal Form (Part 1)

The first part reuses the strategy for rewriting formulae into DNF (found in [6]). Before each step, we first try to perform the following simplifications:

Proof decomposition. Assume we have to prove $\phi \Leftrightarrow \psi$. If ϕ and ψ are both conjunctions, say $\phi = \phi_1 \wedge \phi_2$ and $\psi = \psi_1 \wedge \psi_2$, then check whether $\phi_1 \Leftrightarrow \psi_1$ and $\phi_2 \Leftrightarrow \psi_2$ holds, or $\phi_1 \Leftrightarrow \psi_2$ and $\phi_2 \Leftrightarrow \psi_1$. If so, decompose the proof into two subproofs. Truth tables are used to check the equivalences. In the latter case, use commutativity to exchange ψ_2 and ψ_1. Note that this decomposition is only a mental step, since in the end we are interested in constructing a linearized proof. The effect of this step is that the conjunction will not be rewritten, and that rewriting towards DNF takes place on ϕ_1, ϕ_2, ψ_1, and ψ_2. Follow a similar procedure when ϕ and ψ are both disjunctions, implications, equivalences, or negations. The subproofs can again be decomposed.

Common subformulas. Check whether ϕ and ψ share the same subformula χ (not a proposition letter). Substitute χ in ϕ and ψ by a new proposition letter, and check whether the resulting formulae are still equivalent. If this is the case, treat χ as an atom. No rewriting takes place on such an atom. This substitution is not visible for the student, except that these subformulas will not be transformed by the strategy. Students are still allowed to rewrite it.

3.3 Towards Equal Forms (part 2)

In many cases the normal forms of ϕ and ψ will be equal up to associativity and commutativity, and a simple reordering (applications of the commutative law) completes the proof. Sometimes the differences are more fundamental. In those cases the following steps are performed:

(a) If the set of proposition letters that occur in the normal forms of ϕ and ψ are different, eliminate the letters that occur in only one of the normal forms (by applying simplification rules).
(b) If ϕ contains (up to commutativity) a subformula of the form $p \vee (\neg p \wedge \chi)$ and ψ contains $p \vee \chi$, rewrite $p \vee (\neg p \wedge \chi)$ into $p \vee \chi$, using distribution and true/false rules.
(c) If no other simplifications are possible, extend the normal forms by using true/false rules and distribution to a complete normal form. Each conjunction corresponds to a row of the truth table. A complete normal form is unique (up to commutativity), and thus guarantees the completion of the equivalence proof.

The main purpose of the tool is to improve the skills in applying rewrite rules, and to learn how to recognize simple equivalences. We do not want students to memorize the rewrite strategy, nor do we make it explicit. Its function is only to provide sensible hints and worked-out examples.

4 Examples

We have tested the rewrite strategy for proving equivalences on a set of exercises. We used exercises from textbooks [11,4], and added some exercises to test special cases. We discuss some of the proofs generated by the strategy. The order in which the rules are applied is indicated by numbering the steps.

Example 1. Prove $\neg(p \vee \neg(p \vee \neg q)) \Leftrightarrow \neg(p \vee q)$.

$$
\begin{array}{ll}
 & \neg(p \vee \neg(p \vee \neg q)) \\
\overset{1}{\Longleftrightarrow} & \neg(p \vee (\neg p \wedge \neg\neg q)) \quad \text{De Morgan} \\
\overset{2}{\Longleftrightarrow} & \neg(p \vee (\neg p \wedge q)) \quad \text{double negation} \\
\overset{3}{\Longleftrightarrow} & \neg((p \vee \neg p) \wedge (p \vee q)) \quad \text{distribution} \\
\overset{4}{\Longleftrightarrow} & \neg(T \wedge (p \vee q)) \quad \text{complement} \\
\overset{5}{\Longleftrightarrow} & \neg(p \vee q) \quad \text{true/false rule}
\end{array}
$$

Both starting formulae are negations: because of *proof decomposition*, the top-level negation is kept throughout the proof. The strategy proceeds by rewriting the subformula $p \vee \neg(p \vee \neg q)$ into DNF, resulting in two forward steps (1–2). The right-hand side was already in DNF. The second part of the strategy then rewrites $p \vee (\neg p \wedge q)$ into $p \vee q$. In this case, all proof steps are forward. Note that swapping the starting propositions would give a completely backward proof.

Example 2. Prove $\neg((p \to q) \to (p \wedge q)) \Leftrightarrow (p \to q) \wedge (\neg p \vee \neg q)$.

	$\neg((p \to q) \to (p \wedge q))$	
$\overset{1}{\iff}$	$\neg(\neg(p \to q) \vee (p \wedge q))$	implication elimination
$\overset{2}{\iff}$	$\neg\neg(p \to q) \wedge \neg(p \wedge q)$	De Morgan
$\overset{3}{\iff}$	$(p \to q) \wedge \neg(p \wedge q)$	double negation
$\overset{4}{\iff}$	$(p \to q) \wedge (\neg p \vee \neg q)$	De Morgan

Both propositions have the subformula $p \to q$, and replacing this formula by a new proposition letter (say a) results in propositions that are still equivalent. Hence, the strategy constructs a proof for $\neg(a \to (p \wedge q)) \Leftrightarrow a \wedge (\neg p \vee \neg q)$. More specifically, the common subformula $p \to q$ should not be rewritten (by the strategy). All steps are forward, and belong to part 1.

Example 3. Prove $p \to (q \to r) \Leftrightarrow (p \to q) \to (p \to r)$.

	$p \to (q \to r)$	
$\overset{1}{\iff}$	$\neg p \vee (q \to r)$	implication elimination
$\overset{2}{\iff}$	$\neg p \vee \neg q \vee r$	implication elimination
	$\neg p \vee \neg q \vee r$	
$\overset{11}{\iff}$	$\neg q \vee \neg p \vee r$	commutativity
$\overset{10}{\iff}$	$(T \wedge (\neg q \vee \neg p)) \vee r$	true/false rule
$\overset{9}{\iff}$	$((p \vee \neg p) \wedge (\neg q \vee \neg p)) \vee r$	complement
$\overset{8}{\iff}$	$(p \wedge \neg q) \vee \neg p \vee r$	distributivity
$\overset{7}{\iff}$	$(\neg\neg p \wedge \neg q) \vee \neg p \vee r$	double negation
$\overset{6}{\iff}$	$\neg(\neg p \vee q) \vee \neg p \vee r$	De Morgan
$\overset{5}{\iff}$	$\neg(\neg p \vee q) \vee (p \to r)$	implication elimination
$\overset{4}{\iff}$	$\neg(p \to q) \vee (p \to r)$	implication elimination
$\overset{3}{\iff}$	$(p \to q) \to (p \to r)$	implication elimination

In this particular example, both forward and backward steps are used. The first five steps consist of eliminating the implications. Note that the order of these five elimination steps (two forward steps, and three backward steps) is not fixed by the strategy. In this paper we only show the default order. After 5 steps we have $\neg p \vee \neg q \vee r$ and $\neg(\neg p \vee q) \vee \neg p \vee r$. From this point on, the proof is decomposed into $\neg p \vee \neg q \Leftrightarrow \neg(\neg p \vee q) \vee \neg p$ and $r \Leftrightarrow r$. The latter holds trivially. The remaining steps focus on the former equivalence. After 7 steps we have $\neg p \vee \neg q$ and $(p \wedge \neg q) \vee \neg p$, which are both in DNF. From here on, part 2 of the strategy takes over and completes the proof (steps 8–11).

5 Related Work

A nice overview of the differences between theorem provers, proof checkers, proof assistants, and proof editors on the one hand, and tutorial systems on the other hand, is given by Lukins et al. [9]. In this section we restrict ourselves to other tutorial systems, but we also compare our tool with a computer algebra system.

There are several tools for teaching how to prove theorems in propositional logic, but most teach natural deduction. These tools contain strategies for proving propositions, and use these strategies to provide hints or worked-out examples (e.g. Fitch [2], AProS [10], and Pandora [3]). There are also tools for rewriting exercises in DNF and CNF, such as Organon [5]. Organon is a web tutor for basic logic courses, and is used at a Czech university. A very simple tool for checking equivalences is the Equivalency Checker [1] from Texas A&M University: students can enter two formulae, and the tool checks equivalence (yes/no). DC Proof 1.2 (`http://www.dcproof.com/`) allows students to prove equivalences between formulas using a mix of rewrite rules (De Morgan, double negation, implication and equivalence elimination), and a natural deduction style of reasoning. The student has to choose a rule, which is executed by the tool. Predefined hints and worked-out solutions are only available for a fixed set of exercises. The tool does not contain a strategy to provide help.

The propositional theorem prover of the computer algebra system Yacas (`http://yacas.sourceforge.net/`) uses rewrite rules to simplify a negation of a theorem into false. Since Yacas is not meant for teaching logic, there is no need for a sophisticated strategy. Because Yacas is not a specialized theorem prover, this simple strategy is fast enough. In general, theorem provers are designed to find proofs efficiently, and they typically do not use rewrite rules. As far as we know, the implementation of an expert-like strategy for solving equivalences with rewrite rules is new.

6 Conclusions

We have presented a strategy for proving equivalences between logical formulae. This strategy is based on rewriting both formulae into DNF, with two exceptions: decompose the proof and keep common subformulae, whenever possible. This makes the generated proofs shorter. Afterwards, a strategy for rewriting the normal forms into equal forms is used to complete the proof. The strategy that generates the proofs has been implemented, and we have described how the web-based exercise assistant should be changed to support interactive exercises on the construction of equivalence proofs.

At this moment, the proof generator works as a stand alone application. We intend to integrate it in our existing web-based tool. After that we can test the tool with students: we hope to perform a first test during spring 2011. The DNF tool gives feedback when a student takes a step that deviates from the standard strategy. Since it is not our goal to teach the underlying strategy for proving equivalences, this kind of feedback is no longer necessary. However, we could

compare the lengths of the student solution with the length of the generated proof. If the student needs more steps, we could add a message explaining that a shorter solution exists. Such a message could even be reported as soon as the (possibly incomplete) solution exceeds the expected number of steps. The tool could suggest to go back to the point where the detour started. Proofs that turn out to be shorter than the generated proof are of special interest to us, since they may suggest short-cuts that can be added to the strategy.

The approach followed is not restricted to proofs for logical formulae. Another direction of future research is to apply our ideas to different domains, such as equivalence proofs for relation algebra or set algebra.

Acknowledgements. The authors wish to thank Johan Jeuring, Alex Gerdes, Sylvia Stuurman, and Harrie Passier for their contributions to the exercise assistant. We thank the anonymous reviewers for their constructive comments. Discussions with Daniel Herding about searching proofs for set algebra are gratefully acknowledged.

References

1. Allen, C., Menzel, C.: The Logic Machine: Equivalency Checker (2006), http://logic.tamu.edu/cgi-bin/equivalency.pl
2. Barwise, J., Etchemendy, J.: Language, Proof and Logic (Book & CD-ROM). Center for the Study of Language and Information, 1st edn. (April 2002)
3. Broda, K., Ma, J., Sinnadurai, G., Summers, A.: Pandora: A Reasoning Toolbox using Natural Deduction Style. Logic Journal of the IGPL 15(4), 293–304 (2007), http://www.doc.ic.ac.uk/pandora/runpandora.html
4. Burris, S.N.: Logic for Mathematics and Computer Science. Pearson Education, London (1998)
5. Dostálová, L., Lang, J.: Organon — the web tutor for basic logic courses. Logic Journal of the IGPL 15(4), 305–311 (2007)
6. Heeren, B., Jeuring, J., Gerdes, A.: Specifying rewrite strategies for interactive exercises. Mathematics in Computer Science 3(3), 349–370 (2010)
7. Lodder, J., Jeuring, J., Passier, H.: An interactive tool for manipulating logical formulae. In: Manzano, M., Pérez Lancho, B., Gil, A. (eds.) Proceedings of the Second International Congress on Tools for Teaching Logic (2006)
8. Lodder, J., Passier, H., Stuurman, S.: Using ideas in teaching logic, lessons learned. In: International Conference on Computer Science and Software Engineering, vol. 5, pp. 553–556 (2008)
9. Lukins, S., Levicki, A., Burg, J.: A tutorial program for propositional logic with human/computer interactive learning. SIGCSE Bull. 34(1), 381–385 (2002)
10. Sieg, W.: The AProS project: Strategic thinking & computational logic. Logic Journal of the IGPL 15(4), 359–368 (2007)
11. Thijsse, E.: Logica in de praktijk (in Dutch). Academic Service (April 2000)

Mhy Bib I Fail Logic? Dyslexia in the Teaching of Logic

Xóchitl Martínez Nava

Universidad Nacional Autónoma de México
xochitlzin@gmail.com

Abstract. In this paper it will be shown how dyslexia is a factor that makes learning logic more difficult. It will also provide some methods so the logic teacher can help his dyslexic student to improve his learning process. The text is directed mainly to teachers, but not exclusively, that is to say, we do not deny the possibility that a student with such a disorder can find answers on how to improve the way he learns.

Keywords: Dyslexic, learning difficulty, suggestions on how to teach, learning logic.

1 Introduction

Teaching Classical First Order logic faces a great number of challenges and difficulties that have called the attention of many specialists recently. They have tried to understand these issues, which has resulted in new and better ways to explain and share contents in this area. The effects of these efforts have not been small. However, some of these have been frustrated due to problems that do not allow us to understand the faculties of learning of students and many times we do not wholly comprehend the difficulties they face.

This paper attempts to clarify what was previously presented and to give some general characteristics regarding what dyslexia is and the way this disorder has a role in teaching and learning logic. I will conclude by giving some suggestions that attempt to aid in the dyslexic's development in class.

In most educative contexts, learning disorders do not usually find a way to cope because the consensus seems to be that these disorders are a problem only for those that have them and not the teacher. Nevertheless, radical modifications in the classroom are not necessary to give the dyslexic student an inclusive education (in some contexts this can sometimes be forced or impossible). They also serve as an opportunity to reexamine different didactic strategies and pedagogic focus.

2 The Dyslexic Student in the Classroom

The term "dyslexia" comes from the Greek "δις"(dis) which means separation or disorder and "λεξις"(lexis) which means, among other things, language. Dyslexia is one of the most common disorders in student population (It's difficult to tell the exact quantity, yet there's an estimate of 10 percent [1]). It is characterized as an alteration

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 162–169, 2011.

in the acquisition and use of language, both in the spheres of reading and writing, that is to say, in the comprehension or what is read as well as in the correct execution of writing, also in the listening and speaking aspect. Dyslexia is not an incapacity or an impossibility to possess cognitive skills. People with this disorder can acquire and process information successfully, it is just that they come about to this knowledge in a different way. That is why we should not believe we are dealing with an individual lacking any chance to learn, neither are we with someone that would, unfortunately, never be able to learn logic.

The main characteristics of dyslexia are: deficiency in the use and management of similar signs such as "p", "q", "b", "d", "r", "h", "m", "n", "t", "f", "m", "w", "N", "Z", "M","W", "was", "saw", "body", "doby", etc., and in the case of mathematics "+", "x" among others. However dyslexia is not limited to only reading and writing since it is also common for the dyslexic student to write down something different from what he listens to (for example "god" and "dog") or to read something different from what is written (for example "body" and "boby"). This can be applied to any element that has some distinction between left-right, up-down or any similar phoneme; they all represent an extra challenge for the dyslexic.

The majority of errors a dyslexic student makes are interpreted as a failure to comprehend contents or are not reflected upon by the teacher, simply they are marked off as an incorrect answer. Yet, it is possible to perceive in the mistakes some elements that point to being made by someone with dyslexia (Taking into mind though, that a clinical diagnosis corresponds only to a specialist). These elements are:

- Repetitive and systematic execution of the same or similar mistake in the same homework. Their faults come in patterns, not individual instances.
- Whenever there's a task or assignment supervised by others, the dyslexic student usually exhibits a sense of insecurity and hides his work or tries to pass by unnoticed.
- Constant erasing or crossing out formulas in their homework.
- Memorization results problematic, since the signs that are to remembered are difficult to differentiate. This results in their focus and energy being used only in trying to understand and distinguish elements, not remember them.
- A school record that is below average.
- Constant flaws with their spelling.

It is very important to point out that the previous list is by no means exhaustive. These characteristics do not manifest in the same frequency or degree between any one dyslexic and another. Even more, they are usually accompanied by other peculiarities related with their individual learning profile and their particular context (for example, some languages do not have letters or signs that are similar between themselves). Likewise, we need to emphasize that any single element of that list is not enough for the teacher to suspect he's faced with a dyslexic student. Detection requires a closer observation and much more information that what is put forth in this paper. Hasty judgments should be avoided. A well-made and kind assessment can only be made by a specialist. A rushed accusation can be shaming and unfortunate, even with no ill-will from the teacher. The idea is not for the teacher to try and "treat" or "cure" the disorder. But if there is information to warrant suspicion, the teacher

could experiment with some of the alternative methods that will be pointed out later in this paper. The goal is for the dyslexic student to have the opportunity to learn logic successfully.

Dyslexia plays a very important role in the usual contexts in which logic is taught. Logic involves the use and management of both letters and symbols much more constantly than several other fields of knowledge. It handles many, many elements that become confusing to dyslexic individuals, for example the ubiquitous “p” and “q”. Also “b”, “d”, “m”, “w”, “∀”, “A”, “∃”, “E”, “⊃”, “∈”, “^”,“ v” etc. This is accompanied by a constant evaluation during the ordinary courses, and as result their errors are constantly pointed out, which in turn causes academic grievances.

In order to understand another aspect in which dyslexia makes the learning logic more difficult, we ought to remember that it requires coherence in each and every step in the derivation process and its results. Obviously, simply mistaking a “p” for a “q” halfway with no justification becomes a great error. This is the kind of mistake the dyslexic student is constantly exposed to and can be not only confused as a wrongful execution of a single rule, but also as a more basic execution flaw. And again, due to the nature of the derivative process, it’s likely that the end result of the derivation is totally different from what is expected. Events like this could be another good place for the teacher to check if in the anomalous derivation (or derivations) there is one of those constant and repetitive errors, which is not always evident.

Yet another example of such confusion and accidental errors in logic is visible when formalizing from natural language. Let’s imagine we’re evaluating a typical exercise or exam and we find the following formula that attempts to be the formalization of “If p then q”: “p⊂q” At first glance, this can be seen as an error regardless of the backwards conditional sign. It is usually believed that there was just one mistake and the entire formula was written backwards. The odd formula, however, can be caused by confusion beyond the mere connective. It could be the case that the dyslexic student actually does comprehend the correct use of the conditional between “q” and “p”, but he “mirrored” the connective. This can be judged as incorrect, even though the dyslexic student does indeed possess a correct understanding of how to use the formula. The result is very confusing for the student, because to him the execution and hierarchy of each component are correct; it is just the writing that is him weak point and not something else. Following from that, if the formula “p⊂q” is misunderstood as “If q then p” it could cause the dyslexic to not know which element is to go in which place, believing that the implication must be formalized from necessary to sufficient and not the other way around. If we are not careful, we could be working against very valuable intuitions. Another case in which this may happen is in some systems in which the conjunction and disjunction are vertically symmetrical.

The previous mentions only basic formulas. Another level of difficulty can apply to laws of inference and equivalence. It is such a case in the application of De Morgan’s laws. For the dyslexic, it can become really confusing to derive a conjunction from a disjunction or viceversa (in systems in which these symbols are similar) because if he doesn’t manage to differentiate one from another, it can appear to him that he is going from one connective to the same one (from conjunction to conjunction,

disjunction to disjunction). Thus he either cannot understand what is actually going on, or he could execute it wrongly even if he has a good grasp on it. It is also likely that memorizing this rule goes more slowly compared to other rules or the rest of the classmates. With this rule, we are presented with another chance to check on an error being systematically repeated.

3 The Teacher Faced with the Dyslexic

In a strict sense, treating the dyslexic is not something a teacher is obliged to do, even less so if the teacher does not already have a grasp in psychology or psychopedagogy that can aid in this situation. Then, why should we attend to dyslexics beyond special groups? Why would a logic teacher be interested if chances of having a dyslexic in the classroom are minimal? The answer is not simple and it mostly depends on what each teacher believes should happen in their class, and what he can do. Let us remember that when talking about dyslexia in logic education it is not a demand for the teacher to give an adequate treatment. Not quite, what we're looking for here is to use more elements that aid in the comprehension of the topics seen in class and to avoid frustrations between students and teachers. Many times teachers are not even aware that teaching disorders exist or how they present themselves. It's only in very odd cases that members of an educational institution receive training regarding that is involved in teaching individuals with such disorders and the difficulties this can present. Not having a basic notion of such problems can produce unwanted consequences, from having a high and unexpected failure rate, or well it can cause a strong and deep-rooted anger in students, making them to begrudgingly go through the tasks they are asked to and even to avoid, if possible, doing tasks that evoke that which causes such dislike. Students with strong emotional aversion to particular areas of knowledge are not without cause. Yet they could come from unnecessary difficulties, things that can be easily avoided.

3.1 Suggestions for the Logic Teacher Regarding the Issue of Dyslexia

Currently, a teacher's work is faced with a great quantity of demands from educative institutions. This paper does not want to make their labor any more cumbersome. On the contrary, with the intention to aid them in their performance we give some minimal suggestions as well as some things to reflect on or some considerations to take regarding the activities already occurring in class.

The following suggestions can be developed with no great difficulty in any classroom. This can be done without making a special, secluded group of dyslexic students and in case there was such a group it will be of great help. The suggestions are not about making a teacher take "special" methods towards the dyslexic student; doing so would be ignoring the rest of the group and even cause students who are ahead of the group to feel frustrated by this imposed "slowness". Also, singling out the dyslexic student can alienate him or give him the idea that he will receive special treatment in further contexts (such as work). These suggestions only attempt to give some criteria to the professor so that he may aid a student to achieve self-reliance. That is, for the

student to be aware of his own learning patterns, difficulties and so that he may become responsible for his own learning without depending on teachers.

a. People with learning disorders generally feel insecure due to the great quantity of mistakes they make and constantly doubt about their performance. He is also doubtful about how much he actually comprehends about the topics seen in class. Creating a climate of trust inside the class is fundamental. *Building respect towards the right to commit errors* towards whoever makes them, as well as giving complete confidence that nobody will be ridiculed when he makes a mistake. This applies to everyone, but for someone with dyslexia more so, given that it is difficult for him to detect and understand when he is wrong.

b. *Group activities* encourage bonding and are a chance for the dyslexic to compare his work with others, allowing him to contrast and compare his work without being evaluated by the teacher. This activity is notable by reducing the active participation from the teacher's part and is not all that difficult to put into action. He needs only to write the exercise on the board and let the questions be solved in teams. Copying is allowed! To allow the students to figure out the answers on their own, the teacher should only answer the doubts that come up during the exercise or give indications about how much time is assigned to each question. Another way to encourage teamwork is to suggest students to solve homeworks with each other after class hours.

c. *Try to avoid propositional variables similar between themselves* at the same time in the same exercise: "p", "q", "and m", "w". The vast majority of first order propositional logic languages taught in many educative levels have a large quantity of letters that are not the cause of confusion, and can be used as pedagogic tools. A set of capital letters different than "A" or "E" can be used for symbolizing predicate logic, so that they are not confused with the quantifiers "∀" and "∃".

d. *In using connectives, it is recommended to avoid symbols that can be confused depending which side they are facing*. In case there exists a previous criteria that makes changing them impossible such as using specific textbooks, materials already made, etc., students should be told of the different ways they can symbolize logical connectives. That is, to not limit the set of symbols used during the course but also point out alternatives. I am convinced that most teachers know more than one way to write the different connectives. The previous does not demand a great effort from the teacher, nothing more than writing down at some point the varying options out there. That gives the dyslexic tools that allow him to deal with the confusion in solving the problems and homeworks given in class. Also, it opens up the possibility for the student to ask the professor if he's allowed to use a symbol instead of one that comes up as troublesome for him. It is up to the teacher to allow the student to use such symbols in tests or homework.

e. *Polish or prefix notation* has come to disuse given the difficulty that using it implies. However, its use as a didactic resource is very interesting and fruitful when it comes to helping all kind of students understand the correct way to read, write and formalize connectives. On top of that, they help in teaching the way in which the different elements that compose a formula are arranged, propositions and connectives, without using parenthesis.

Regarding this, I quote: “Interestingly, some students [dyslexics] find one or another notation easier to process than others. So, for instance, one teacher reports that a student who had severe difficulties coping with either mathematical or traditional logic notation was helped by adopting Polish notation. The student had no trouble accurately processing strings of letters and was, with practice, able to parse these strings”[1].

The “Lógica Clara” seminar that has given courses in the Facultad de Filosofía y Letras (Faculty of Philosophy and Letters) in the UNAM, used polish notation as a didactic aid during the first weeks of classes. Thanks to the use of Polish notation, the advances were astonishing as students could accomplish a precise and rigorous lecture of all formulas.

An activity we made, created by Fernando Flores Galicia, (founding member of the seminar) consisted in assigning homework of formulas written in infix notation and it was asked for the same formula to be translated to Polish notation (this activity was only carried out in propositional logic). The exercises were composed of lists between fifteen and twenty formulas arranged (not explicitly) in little groups. Through these groups the number of premises used as well as the number of connectives kept rising. That is, it was made with a progressive difficulty.

My observations, as part of the students giving classes, signal to the fact that in the case of some dyslexics this turns out to be even more clarifying for them than other students. They are looking for ways to formalize that have very few elements that can be confused with each other, and thus they seek alternatives so they are able to describe “what is important”, what is “really happening there”(in the previous activity, sometimes they had to say their formalizations out loud in order to write them successfully). This deepens comprehension in surprising ways. Let us see some examples those exercises:

I.

Infix Notation:	Prefix or Polish notation:
1.- (p v r) ^ (m v t)	KAprAmt
2.- (p ^ r) v (m^ t)	AKprKmt

The formulas in the previous examples have some of the troubling elements mentioned previously. Due to that, on the left side each formula is indiscernible to a dyslexic. Not so in the right column, in this one he could at least distinguish the difference between operators. It’s relevant to say that Polish notation can help the dyslexic to discern connectives, but not always so with propositions. That’s why it’s necessary to also use the previous suggestions, c and d.

Let’s look back at this formalization: “$p \subset q$”. Asking a student to write the same formula but in Polish Notation this time can enlighten us as to whether it is a formalization or writing error. If it’s the latter, then we only have to tell him he is writing it backwards and ask him to be more careful next time. In the former, we can give a more thorough explanation on what is confusing him. The following are three attempts to represent that very same formula:

II.

Infix Notation:	Prefix or Polish notation:
1.- $p \subset q$	Cpq
2.- $p \subset q$	Cqp
3.- $q \supset p$	Cpq

Let's assume that after assigning the previous exercise we find ourselves that the student answers the formalization of the left side with the right side. In the first case we could be dealing with a "mirrored" connective and then the student actually does understand the order in which the elements should be placed. He ought to be told that he's writing the connective the other way around and ask him to watch out for that.

In the second case, we are able to compare both formalizations and observe that not only is there confusion regarding the right way to write the connective, but there also isn't clarity when dealing in which order should things go. In this case a more extensive explanation is required.

In the third case we notice that in the left side the connective is written correctly and it's the propositions that are written incorrectly. Generally speaking, teachers usually just cross off the attempt as incorrect without further reflection, but if as is the case, we ask the student to also write it in Polish notation, then we realize there's not only a careless accident, but that there's not a wholesome comprehension on how to formalize a material conditional.

This proposal does not consist of a radical substitution from one notation to another, but to contemplate using another system as *reinforcement*. By that I mean this is *not to be evaluated* and without using a lot of time in it, since it will only be used to settle and familiarize basic elements when beginning to formalize.

f. Saying the formulas out loud constantly throughout the course before writing them down on the board or just as an exercise. Doing this favors the appropriate writing of such formulas and it helps dyslexics whose main challenge is in the hearing-writing or reading-talking fields. When students are asked to say the formulas out loud, that exercises their understanding of the distinction between symbols in formalization.

A wrongful writing of what is said out loud, or mistaking the name of a connective or propositional variable in a constant and systematic way can help the professor consider if he's dealing with a dyslexic student. If that is the case, he could consider doing more activities like that or using others that do not demand too much time.

The previous listing does not pretend to fill out every action that could favor a dyslexic student. They, however, do attempt to show that making the necessary changes to not exclude or add burdens to the dyslexic who tries to learn logic do not require long and painful structural modifications. Nor do these suggestions necessary add to the difficulty of planning a logic course. Actually some of these suggestions only show that the way in which some activities that already take place (adequate planning, group activities, saying formulas out loud) are helping the dyslexic student. The intention is to allow the teacher to understand what may happen when he doesn't or takes away one of those activities.

4 Conclusions

Understanding what is happening with the student and the process he uses to learn will help everyone involved in the optimal development in teaching. With the

dyslexic we should not think that we're dealing with an odd specimen, nor should we go to the other end of the spectrum and treat him with pity. Both attitudes are not beneficial, for they create rejection or insecurity in the dyslexic and they do not allow him to create the strategies and tools he will need throughout his life, and not just when he's taking classes.

Knowing what dyslexia is and the issues it causes to both teacher and student is necessary to make any action that seeks to make all educative contexts less exclusive. The opportunity to perform any of the previous suggestions (or any other alternative) as well any success in their application will also depend on the aid that different institutions grant. It's undoubtedly crucial that teachers think of these tasks as plausible inside their classroom, and that's why there's so much insistence in how easy and simple these options are. If dyslexia is not seen as something worthy of attention and we do not believe that there are ways to improve how we teach, we can hardly encourage a meaningful change at an institutional level that would in turn make possible for people not just with dyslexia, but with any learning disorder, to learn. Learning new ways to teach from our students reminds us that there really is so much more we can come to know.

Acknowledgements

I'd like to thank the "Lógica Clara" seminar and the PAPIME-UNAM PE403111 "Mejoramiento de la calidad de la enseñanza de la lógica" project. Without either I could have not accomplished this paper. English Translation: Samuel Alejandro Lomelí Gómez. Thanks to: Adrian Caballero, for the first Translation of this paper.

References

1. Jordan, D.R.: La dislexia en el aula, trad., 198 p. Pardal Inés, Paidos (1982)
2. Perkin, G., Tony, C.: The dyslexics student and the mathematics in higher education. Wiley InterScience, Hoboken, http://www.intersciencience.com
3. Amanda, K., David, S., Sally, B., Lisa, E., Rache, E.: Dyslexia and Development, Coordination Disorder in Further and Higher Education- Similiares and Differences. Does the "Label" Influence the support Given? Wiley InterScience, Hoboken, http://www.intersciencie.wiley.com
4. Malmer, G.: Mathematics and Dyslexia- An Overlooked connection (2000), http://www.intersciencience.wiley.com

Information-Theoretic Perspective for Teaching Logic

Ángel Nepomuceno-Fernández

Group for Logic, Language and Information,
University of Seville
nepomuce@us.es

Abstract. This work is about using conceptual tools for teaching logic. Our perspective is based on the information theoretic logic studied by J. Corcoran. Argumentation is referred in terms of a new terminology that is introduced from the concept of information, which is taken as primitive. Trying to show that the understanding of several basic concepts of logic could be facilitated, the notion of information content of a proposition is studied, the concept of logical implication is redefined from this informational point of view and the three kinds of inferences are informationally analysed.

Keywords: Information theoretic perspective, informative content of propositions, kinds of argumentation, logical implication.

1 Introduction

In order to teach basic logic sometimes argumentation theory has been taken as a useful starting point. Then the role of ordinary language may be decisive and some conceptual tools should be used. In this work we propose an information-theoretic point of view based on Corcoran's ideas[1] that could facilitate the understanding of several basic concepts of logic.

The concept of information is taken as primitive, though we can consider the entire information of a given domain of investigation and represent that by means of diagrams or in terms of set theory. So, if $\Im$ represents the entire information, $\emptyset$ represents its complementary information, that is to say, the null information. Then, the complementary, union and the interesection of information are defined in the same way as the corresponding set theory notions: for $A \subseteq \Im$,

1. $A \cup \sim A = \Im$
2. $A \cap \sim A = \emptyset$
3. $A \cup \emptyset = A$
4. $A \cap \emptyset = \emptyset$
5. $A \cup \Im = \Im$
6. $A \cap \Im = A$

[1] Presented in [2] by the author. In [7] a panoramic view of this author is given. A first formal study of such point of view is in [5].

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 170–177, 2011.

The work is organized as follows. First, argumentation is defined in terms of the new terminology. In the central section the notion of information content of a proposition is studied and the concept of logical implication is redefined from this informational point of view, the classical logical implication as well as the relevant one, then the three kinds of inferences, deduction, abduction and induction, are informationally analysed. Finally, in a last section some concluding remarks are offered.

2 Argumentation in Informational Terms

Teaching argumentation can be conceived as a first step in order to pave the way for teaching logic. Then some aspects of the theory of argumentation, and a set of pertinent notions, can be pointed out. In [3] there are interesting suggestions about how to tackle the problem of studying argumentations in several contexts, which can be transfered to the field of logic itself. Argumentation is the rational activity par excellence, which involves information processes. Then we have to pay attention to propositions. To be precise, the approach to information-theoretic logic in [2] atributes information to propositions, but given a domain of investigation the set of propositions to take into account is not the set of "all" propositions but a restricted one, namely the set of pertinent propositions only. So "the set of propositions" should be understood as "the set of pertinent propositions" relative to a given domain of investigation. This restriction permits us to reject spurious syntactic expressions. For example, "the virtue is green" or "this stone is virtuous" are not pertinent propositions when ethics or geology are the domains of investigation, respectively. Whatever the case may be, propositions should be studied with respect to an inferential context, which is the context regulated by an inferential system, which could be codified in terms of classical logic, modal logic, etc.

Despite the diversity of philosophical conceptions of information[2], every proposition has an information content but it is different from the meaning of the sentence by means of which the proposition is expressed. Intuitively, in Fregean terms, a proposition is an objective thought that can be communicated (in particular by linguistical means), but if a sentence is the vehicle to communicate a proposition, the information content of such proposition cannot be reduced to the semantic value of such sentence. Though the only way to achieve the information content of any proposition is by means of the analysis of the sentence used to express it, the information content of a proposition is not directly expressed by a logical or grammatical form. In fact, several sentences can express the same proposition (active or passive voice, etc.) and though a proposition has a unique form and a unique information content, an information content *per se* does not have a form ([2], p. 116).

An argumentation is a process that culminates in a sequence of propositions, ordered according to certain criteria. In a such sequence it can be distinguished

[2] Most of them in [1].

an initial set of propositions, called *premises*. Other intermediate set of propositions follow the former, the so called *chain of reasons*, and a last proposition called *conclusion*. The (ordered) pair of extremes of an argumentation, namely premises and conclusion, is the *argument* of such argumentation (see [3]). The most important aspect of argumentation, and logic, is how the propositions of the sequence are linked, what are the criteria to consider a sequence of propositions as the result of an argumentation, and to avoid arbitrary sequences. The chain of reasons provides information for accepting the conclusion when the premises have been accepted. In theory of argumentation the emphasis is put on the efficacy of the chain of reasons to convince someone of accepting a conclusion, whereas logical theories study formal aspects of the relation between premises and conclusion.

According to the relation between the terms of the corresponding arguments, several kinds of argumentations can be distinguished. Particularly the three kinds defined by Peirce can be defined from this informational point of view. There are argumentations called

1. *Deduction*. The purpose is to achieve a conclusion whose information content is contained within the information of the premises. When the deduction has been completed, the conclusion may be a new proposition but no new information will have been obtained.
2. *Induction*. The information content of the premises is extended in order to obtain the information content of the conclusion. In this case an increase in information is achieved.
3. *Abduction*. The agent must expand on the information content of the premises until the information of the conclusion becomes included. When an abduction is finished, a deduction can be reconstructed.

3 A Treatment of Information Content

Greek letters π, ρ, σ will be used to name propositions and $Inf(\pi)$, $Inf(\rho)$ and $Inf(\sigma)$ represent the information content of such propositions, respectively. For a set of propositions $\Gamma = \{\pi_1, \pi_2, ..., \pi_n\}$, $n \geq 1$,

$$Inf(\Gamma) = Inf(\pi_1) \cup Inf(\pi_2) \cup ... \cup Inf(\pi_n).$$

Given two propositions π and ρ, a domain of investigation whose entire information is $\Im$ and the class Ω of the pertinent propositions, it can be shown that

1. $Inf(\pi) \subseteq \Im$, provided $\pi \in \Omega$;
2. $Inf(\pi) \cup Inf(\rho) \subseteq \Im$ and $Inf(\pi) \cap Inf(\rho) \subseteq \Im$, provided $\pi, \rho \in \Omega$;
3. $Inf(\pi) = \emptyset$ if π has no information content;
4. If π contains all information, then $Inf(\pi) = \Im$;
5. $\sim Inf(\pi) = \Im - Inf(\pi)$.

In general, for every class of pertinent propositions Δ, we defined

$$Inf(\Delta) = \bigcup_{\pi \in \Delta} Inf(\pi)$$

The concept of *logical implication –logical consequence*, or *entailment relation*– can be defined from the primitive notion of information. According to Corcoran's point of view, an information theoretic approach to logic can be characterized by six rules that relate that logical concept to the notion of information content. For a set of propositions Γ, propositions π, ρ and σ,the propositional connectives $\neg$ and $\vee$ and the total information $\Im$, the rules that define the (entailment) relation $\models_I$ are

1. $\Gamma \models_I \sigma$ if $Inf(\sigma) \subseteq Inf(\Gamma)$;
2. $\Gamma \not\models_I \sigma$ if $Inf(\sigma) \cap Inf(\Gamma) \neq Inf(\sigma)$;
3. If a proposition π is a tautology, then $Inf(\pi) = \emptyset$. In general, for every proposition π, $Inf(\pi \vee \neg\pi) = \emptyset$;
4. If π is a contradiction, $Inf(\pi) = \Im$;
5. $Inf(\pi) \cap Inf(\neg\pi) = \emptyset$ and $Inf(\pi) \cup Inf(\neg\pi) = \Im$;
6. $Inf(\pi \vee \rho) = Inf(\pi) \cap Inf(\rho)$;
7. π and ρ are equivalent[3] if and only if $Inf(\pi) = Inf(\rho)$.

Some facts that describe the information-theoretic approach to logic can be derived from theses rules. Among them we point out the following ones. Validity of any argument can be studied in informational terms. The entailment relation, from an informational point of view, represented by $\models_I$, satifies the following known structural rules[4]:

1. Reflexivity. For every $\rho \in \Gamma$, we have that $\Gamma \models_I \rho$
2. Monotonicity. If $\Gamma \models_I \pi$, then $\Gamma^* \models_I \pi$ for any Γ^* such that $\Gamma \subseteq \Gamma^*$
3. Transitivity. If $\Gamma \models_I \pi$ and $\pi \models_I \rho$, then $\Gamma \models_I \rho$

Given a domain of investigation, the total information $\Im$, a set of premises Γ and a conclusion π, the argument $\langle \Gamma, \pi \rangle$ is valid if and only if Γ entails π from an informational point of view, formally

$$Val(\langle \Gamma, \pi \rangle) \text{ if and only if } \Gamma \models_I \pi,$$

which is equivalent to say that

$$Val(\langle \Gamma, \pi \rangle) \text{ if and only if } Inf(\pi) \cap (\Gamma) = Inf(\pi).$$

Of course, given two arguments $\langle \Gamma, \pi \rangle$ and $\langle \Delta, \rho \rangle$, we can determine when both of them have the same content

$$\langle \Gamma, \pi \rangle = \langle \Delta, \rho \rangle \text{ if and only if } Inf(\Gamma) = Inf(\Delta) \text{ and } Inf(\pi) = Inf(\rho)$$

[3] This rule corresponds to a remark in [2] and it can be added as a seventh rule, since every rule is just proposed as a remark.

[4] Permutation and contraction, since the information content of a proposition has been defined in set-theoretical terms, are trivially verified.

According to this conception, for propositions π, ρ and the entire information $\Im$ of a domain of investigation, it is verified that

1. $Inf(\neg\pi) =\sim Inf(\pi)$;
2. $Inf(\pi) \cap Inf(\neg\pi) = \emptyset$;
3. $Inf(\pi) \cup Inf(\neg\pi) = \Im$;
4. Defining both $\wedge$ and $\rightarrow$ as functions of $\vee$ and $\neg$,
   (a) $Inf(\pi \wedge \rho) = Inf(\pi) \cup Inf(\rho)$;
   (b) $Inf(\pi \rightarrow \rho) =\sim Inf(\pi) \cap Inf(\rho)$.

In [5] a modification in clauses 1 and 2 permits to define relevance conditions. In that case the entailment relation from an informational perspective with such conditions can be represented as $\models_{IR}$. In particular, such two first clauses to define $\models_{IR}$ will be[5]

1'. $\Gamma \models_{IR} \sigma$ if $Inf(\sigma) \subseteq Inf(\Gamma)$ and $Inf(\Gamma) \neq \Im$;
2'. $\Gamma \not\models_{IR} \sigma$ if $Inf(\sigma) \cap Inf(\Gamma) \neq Inf(\sigma)$ and $Inf(\sigma) \neq \emptyset$.

This informational point of view may be epistemologically useful for examining scientific practices as long as characteristics attributed to the logic of relevance are included in the modified clauses. In particular, in $\models_{IR}$, the set of premises can not be contradictory, and no tautology can be a conclusin, since information content of premises must be different from the total information of the domain of investigation and the information content of any conclusion must be different from the null information, respectively.

In order to take advantage of this approach, an informational study of the other argument forms is accessible. Let us see first abduction. Given a background theory –a set of proposition Θ– and a fact expressed with the proposition ρ, another proposition π explains ρ if the conjunction of Θ and π entail (in informational sense) ρ. Formally, the corresponding abductive problem can be represented formally as

$$\Theta \cup \{?\} \models_I \rho \text{ or } \Theta \cup \{?\} \models_{IR} \rho$$

where $\{?\}$ represents the gap to be filled. Abduction is a kind of inference in which the conclusion (of a deduction) and part of the premises (of such potential deduction) are the starting point of the inference and the goal is a hypothesis (the 'conclusion' of the abductive inference, so to speak). It can also be seen as a process of searching for a premise to complete the set of premises of a deduction. For a background theory Θ and a proposition ρ, the main abductive rule permits us to complete Θ in order to deduce such conclusion[6]. From an informational perspective the rule could be expressed as

$$\frac{Inf(\rho) \not\subseteq Inf(\Theta);\ Inf(\rho) \cap Inf(\Theta) = A \neq \emptyset;\ Inf(\pi) = A}{\pi}$$

[5] The other clauses are the same.

[6] This is a refined version of what could be called the Peirce's rule for abduction, accordding to which, for "facts" A and C, when the surprinsing fact C is observed, if A were true, C would be a matter of course, then there is a reason to suspect that A is true (in [6], p.231.)

This rule is sound regardless of whether the perspective is $\models_I$ or $\models_{IR}$. So given the abductive problem $\Theta \cup \{?\} \models_I \rho$ –or $\Theta \cup \{?\} \models_{IR} \rho$, when relevance conditions are taken into account–, the process of obtaining the hypothesis π is an inference: the abductive problem constitutes the set of premises and the solution is the conclusion[7]. This is expressible as $(\Theta, \rho) \gg_{Ab} \pi$.

Unlike $\models_I$, in general if we add relevance conditions some structural rules fail. Let π be a proposition, though $\pi \wedge \neg\pi \models_I \pi$, $\pi \wedge \neg\pi \not\models_{IR} \pi$, and $\pi \models_I \pi \vee \neg\pi$ but $\pi \not\models_{IR} \pi \vee \neg\pi$. Moreover, since $\pi \models_{IR} \pi$ but $\pi \wedge \neg\pi \not\models_{IR} \pi$, monotonicity is not verified. On the other hand, $\gg_{Ab}$ does not verify monotonicity either, since an increase in information in the antecedent of the given rule could block the formulation of the corresponding hypothesis.

Finally, inductive generalization can be seen as an inferential process of increasing information from given information. To illustrate that, suppose as given a number of propositions π_1, ..., π_n. Each one of them says "a_i has the property P". The proposition π says "all subjects (of the given context) have the property P". Then the result of an inductive inference can be represented as

$$\pi_1, \pi_2, ..., \pi_n \sqsupset_I \pi$$

provided that

1. $Inf(\pi_1) \cup Inf(\pi_2) \cup ... \cup Inf(\pi_n) \subseteq Inf(\pi)$
2. $Inf(\pi) \neq \Im$
3. $Inf(\pi_1) \cup Inf(\pi_2) \cup ... \cup Inf(\pi_n) \neq \emptyset$
4. If there is a subject different from a_i, $i \leq n$, $\pi_1 \wedge \pi_2 \wedge ... \wedge \pi_n$ and π cannot be equivalent, that is to say $Inf(\pi_1 \wedge \pi_2 \wedge ... \wedge \pi_n) \neq Inf(\pi)$

This inference relation is not monotonic. Suppose $\pi_1, \pi_2, ..., \pi_n \sqsupset_I \pi$. According to previous clauses, it can be shown that $Inf(\pi_1) \cup Inf(\pi_2) \cup ... \cup Inf(\pi_n) \subseteq Inf(\pi)$, but if $Inf(\rho)$ is added[8], for a certain proposition ρ, then the conclusion may change

$$Inf(\pi_1) \cup ... \cup Inf(\pi_n) \cup Inf(\rho) \not\subseteq Inf(\pi)$$

as a result $\pi_1, \pi_2, ..., \pi_n, \rho \not\sqsupset_I \pi$.

4 Concluding Remarks

The information-theoretic approach to logic is not the only way of approaching our discipline. In [2] another point of view is presented. According to this new

[7] The role of some propositions, it should be noted, changes with respect to the corresponding deduction. In abduction, the background theory and the fact to be explained are premises, then the conclusion is a new proposition to fill the gap, which is a premise (with the background theory) of the deduction that justifies the abductive result, meanwhile such fact is the conclusion of this deduction.

[8] Every proposition that represents a particular case of having the property is now a premise of the inductive inference.

perspective, the model theoretic, set theoretic and substitution theoretic approaches to logic could all be understood as *transformation-theoretic* approach. To have a minimal comparison between the two approaches, a short explanation about this new perspective can be given in the following way. First, let τ be a one-one transformation defined from propositions to propositions. This is a function with domain in the set of (pertinent) propositions Ω and range in the same set, such that

1. For every $\pi \in \Omega$, $\tau(\pi) \in \Omega$
2. For $\pi \in \Omega$, $\tau(\neg\pi) = \neg\tau(\pi)$
3. $\tau(\pi * \rho) = \tau(\pi) * \tau(\rho)$, for $* \in \{\wedge, \vee, \rightarrow\}$
4. For $\Gamma \subseteq \Omega$, $\tau(\Gamma) = \{\tau(\varrho) \in \Omega : \varrho \in \Gamma\}$

Since any (pertinent) proposition is true or false, we can associate every proposition with its truth value, a set of propositions is associated with the value "true" if all its members are true, or with "false" otherwise, in symbols, $\tau(\pi) \equiv \top$, or $\tau(\pi) \equiv \bot$, respectively, for any proposition π, and similarly for sets of propositions. Now, for any set of propositions $\Gamma \subseteq \Omega$ and a proposition $\pi \in \Omega$, in transformation-theoretic terms, $Val(\langle\Gamma, \pi\rangle)$ if and only if no transformation τ carries $\langle\Gamma, \pi\rangle$ onto $\langle\tau(\Gamma), \tau(\pi)\rangle$ such that $\tau(\Gamma) \equiv \top$ and $\tau(\pi) \equiv \bot$, that is to say, no transformation carries the given argument onto an argument with (transformed) true premises and a (transformed) false conclusion.

Given this perspective, for teaching basic logic we can distinguish two ways of using the conceptual tools that have been presented in this paper: as set theory analysis and as a diagrammatical representation of set theoretical relations, namely in

1. Scientific and technical teachings, where the set theory language may be more familiar to many students;
2. Humanities and social sciences, where diagrammatical representations of sets and relations may be more intuitive and accessible to many students.

On the other hand, these methods could be used as a "metatheory" for other developed logics (modal logic, epistemic logic, dynamic epistemic logic, etc.). Whatever the case may be, such methods should be applied in context, since each group of students is different and no program of study can be fixed in advance if the context is not well known.

References

1. Adriaans, P., van Benthem, J. (eds.): Philosophy of Information. Handbook of the Philosophy of Science, vol. 8. Elsevier North-Holland, Amsterdam (2008)
2. Corcoran, J.: Information-theoretic logic. In: Martínez, C., Rivas, U., Villegas-Forero, L. (eds.) Truth in Perspective, pp. 113–135. Ashgate Publisihing Ld. (1998)
3. Corcoran, J.: Argumentations and logic. Argumentation (3), 17–43 (1987)
4. Martínez, C., Falgueras, J.L., Sagüillo, J.M. (eds.): Current Topics in Logic and Analytic Philosophy. Cursos e Congresos da Universidade de Santiago de Compostela (2007)

5. A. Nepomuceno: Information and logic. In: [4], pp. 219–232 (2007)
6. Peirce, C.S.: Pragmatism as the Logic of Abduction. In: The Essential Peirce, vol. 2, Edited by the Peirce Edition Project, pp. 226–241. Indiana University Press, Bloomington (1998)
7. Sagüillo, J.M.: Corcoran the philosopher. In: [4], pp. 246–269 (2007)

Teaching Sound Principles about Invalidity

Carlos A. Oller

Departamento de Filosofía,
Universidad de Buenos Aires & Universidad Nacional de La Plata
carlos.a.oller@gmail.com

Abstract. One of the aims of introductory logic courses for humanities students is to help them understand the structure of, and the evaluation criteria for, natural language arguments. In order to show that symbolic logic can help students achieve this understanding the relationship between natural language arguments and formal language arguments has to be clearly, and correctly, elucidated. The notions of logical form and formal (in)validity, and their relation with the (in)validity of natural language arguments, are essential for this elucidation. The purpose of this paper is to show and explain the fact that, notwithstanding James W. Oliver and Gerald Massey's warnings concerning invalidity verdicts, wrong conceptions about logic-based methods for determining invalidity of natural language arguments have a residual existence in some present-day introductory logic textbooks.

Keywords: deductive invalidity, natural language arguments, logical form, logic-based methods for argument evaluation.

1 Introduction: The Asymmetry between Validity and Invalidity

In an until quite recently largely ignored paper James W. Oliver [7] denounced the fact that most introductory logic textbooks of the time assumed, explicitly or implicitly, the following false principle about deductive invalidity:

(1) An argument is invalid if and onlyif it is an instance of an invalid form.

or the weaker conditional:

(2) An argument is invalid if it is an instance of an invalid form.

Oliver conjectured that the authors of those textbooks must wrongly have assumed that (1) could be deductively obtained by replacing "valid" by "invalid" in the following true principle:

(3) An argument is valid if and only if it is an instance of a valid form.

Those authors must have assumed that (1) could be deduced from (3) applying the inference scheme "φ if and only if ψ. Therefore, not φ if and only if not ψ". In fact, what actually follows from (3) is:

(4) An argument is invalid if and only if it is an instance of no valid form.

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 178–182, 2011.

Oliver's objections were further developed and popularized by Gerald Massey [5] [6] who concluded that there exists a radical asymmetry between validity and invalidity: logic is able to provide methods for demonstrating that particular natural language arguments are valid, but there are no logic-based methods for proving invalidity. While the formal validity of a particular natural language argument follows from the existence of at least one valid form of which it is an instance in some logical system, showing that it is not formally valid requires establishing that there is no valid form in any logical system of which it is an instance. Therefore, logic cannot provide conclusive arguments that bad natural language arguments are bad, and invalidity verdicts rest on intuitive judgments altogether unsupported by logical theory. Given that fallacies are a species of invalid arguments, one of the consequences of Massey's draws from his conclusion is that no logic-based theory of formal fallacies is possible.

As a result of the popularization of Massey's asymmetry thesis, most present-day introductory logic textbooks for humanities students include some cautionary remarks against the risk of jumping to conclusions when making verdicts on the invalidity of particular natural language arguments: on one level of analysis, an argument might well be shown to be an instance of an invalid form but if we are not careful enough we may overlook the fact that it is also an instance of a more complex valid form [[9], p.21].

The purpose of this paper is to show and explain the fact that, notwithstanding this awareness about the problems concerning invalidity verdicts, wrong conceptions about invalidity have a residual existence in some present-day introductory logic textbooks.

2 Invalidity and Logical Form

Oliver conjectures that a probable source of wrong conceptions about invalidity lies in the character of the relationship between natural language arguments and the forms which these arguments have or of which they are instances: "[it is] a natural mode of expression to speak of "the form" of an argument, and this way of speaking seems to lead to the view that, for any argument, there is a unique form" [[7], p. 465]. In fact, if the uniqueness of logical form of the natural language arguments is assumed then a variation of principle (1) about invalidity follows. This derivation can be found in Dennis Packard and James Faulconer's introductory logic textbook [[8], pp. 8-16] and proceeds thus:

> (5) An argument is valid if and only if there is no argument of its form (called a counterexample) that has true assumptions and a false conclusion.

(5) (co)entails (6):

> (6) An argument is invalid if and only if there is an argument of its form (called a counterexample) that has true assumptions and a false conclusion.

This derivation, unlike that presented in the Introduction, is a valid one. But, (5) is false if uniqueness of logical form for natural language arguments is not assumed. For, if the fact that an argument can be an instance of more than one argument form is accepted, and (5) is accordingly reformulated as:

(7) An argument is valid if and only if there is no argument of any of its forms that has true assumptions and a false conclusion.

then the following argument can be used to show that (7) is false:

(8) If something has been created by God, then everything has been created by God. Everything has been created by God. Therefore, something has been created by God.

(8) is an instance of the invalid form:

(9) $(\varphi \rightarrow \psi), \psi / \varphi$

which has counterexamples like:

(10) If Philadelphia is the capital of Pennsylvania, then Pittsburgh is not. Pittsburgh is not the capital of Pennsylvania. Therefore, Philadelphia is the capital of Pennsylvania.

But, (8) is nonetheless a valid argument because it is also an instance of the valid form:

(11) $(\exists x Cx \rightarrow \forall x Cx), \forall x Cx / \exists x Cx$

As Oliver observes, the question of whether arguments have or instantiate a unique logical form or many argument forms is a normative one. Nevertheless, the theoretical and practical consequences of adopting the uniqueness notion strongly recommend the adoption of the opposite position. And, in any case, we do not have any generally accepted criteria, and no working methods, by which to determine *the* logical form of a natural language argument.

3 Persisting in Error: A Case Study

In order to show how wrong conceptions about invalidity have a residual existence in present-day introductory logic textbooks I will examine Patrick Hurley's presentation of the counterexample method in his popular manual *A Concise Introduction to Logic* [4]. Hurley acknowledges that there are falsifying instances of principle (2), but immediately minimizes the importance of such cases:

> The fact that some substitution instances of invalid forms are also substitution instances of valid forms means simply that we must exercise caution in identifying the form of an argument. However, cases of ordinary language arguments that can be interpreted as substitution instances of both valid and invalid forms are so rare that this book chooses to ignore them. [[4],pp.56-57]

To reinforce the idea that ordinary language arguments which are substitution instances of both valid and invalid forms are extremely rare he provides the following dubious example:

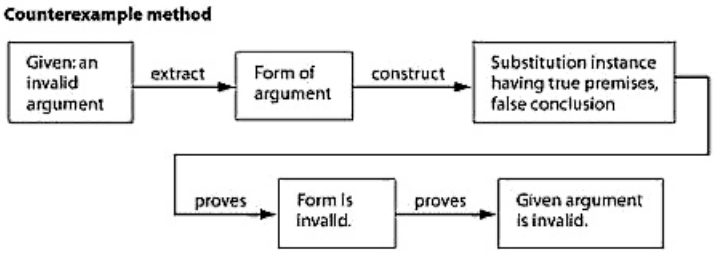


Fig. 1. The counterexample method as schematized in [4] , p.59

(12) All bachelors are persons. All unmarried men are persons. Therefore, all bachelors are unmarried men.

(12) is a substitution instance of the invalid form:

(13) All A are B. All C are B. Therefore, all A are C.

But, because "bachelors" is equivalent in meaning to "unmarried men," Hurley asserts that (12) is also a substitution instance of the valid form:

(14) All A are B. All A are B. Therefore, all A are A.

Of course, given that strictly speaking (12) is not a substitution instance of the form depicted by (14), this contrived example can only reaffirm students' belief that cases of natural language arguments that are instances of both valid and the invalid forms are difficult to find, if existent at all.

Figure 1 depicts the diagram that appears in [4] which further reinforces the notion that the counterexample method provides conclusive verdicts —proofs— of invalidity and that, after all, with negligible exceptions, an argument is invalid if it is an instance of an invalid form.

The diagram also reinforces the associated idea that a natural language argument has a unique form, a notion which as seen in the previous section allows the entailment of a variant of the rejected principle about invalidity. If the multiple argument form position is adopted, the fact that two arguments share one of their forms is not enough to conclude that they are both invalid, should one of them have true premises and false conclusion. But, as pointed out by Bencivenga [2] and Finocchiaro [3], the method of counterexample, like other procedures used to determine invalidity, is a non-deductive technique in which pragmatic considerations play an important role.

4 Conclusions

Typically, one of the aims of introductory logic courses for humanities students is to help them understand the structure of, and the evaluation criteria for, natural language arguments of disciplines such as philosophy[1]. In order to show that symbolic logic can help students achieve this understanding the relationship between natural language arguments and formal language arguments has to be clearly, and correctly, elucidated. The notions of logical form and formal validity

are essential for this elucidation but, as we have seen, wrong conceptions about these notions have a residual existence in some present-day introductory logic textbooks. In spite of what is asserted in these textbooks, symbolic logic cannot provide conclusive arguments to support verdicts of invalidity of natural language arguments.

References

1. Beebee, H.: Introductory Formal Logic: Why do we do it? Discourse 3, 53–62 (2003)
2. Bencivenga, E.: On good and bad arguments. Journal of Philosophical Logic 8, 247–259 (1979)
3. Finocchiaro, M.: Arguments about Arguments. In: Systematic, Critical and Historical Essays in Logical Theory. Cambridge University Press, Cambridge (2005)
4. Hurley, P.J.: A Concise Introduction to Logic, 10th edn. Thomson Wadsworth, Belmont (2008)
5. Massey, G.J.: Are there any good arguments that bad arguments are bad? Philosophy in Context 4, 61–77 (1975)
6. Massey, G.J.: The Fallacy behind Fallacies. Midwest Studies In Philosophy 6, 489–500 (1981)
7. Oliver, J.W.: Formal fallacies and other invalid arguments. Mind 76, 463–478 (1967)
8. Packard, D., Faulconer, J.: Introduction to Logic. D. Van Nostrand Co., New York (1980)
9. Tomassi, P.: Logic. Routledge, London (1999)

Systematic Errors as an Input for Teaching Logic

Gladys Palau[1,2] and Ana Couló[1]

[1] Universidad de Buenos Aires, Departamento de Filosofía,
Puán 480, CABA, Argentina
[2] Universidad Nacional de La Plata
gadi1@fibertel.com.ar, amfernandez62@gmail.com

Abstract. In recent years, the analysis of student's errors in learning Logic is developing. Errors are not solely regarded as mere faults, but often as symptoms of learning development, and sometimes even as necessary steps in the building of knowledge. But these researches have not made an equal impact in everyday teaching and learning. To make this possible, some changes are in order in Logic and Philosophy teacher's education.

Keywords: Logic teaching, teacher education, systematic error, natural logic.

1 Introduction

In recent years, the analysis of student's errors in learning Logic is developing as an instance of the more general studies clinical and educational psychologists and researchers in Mathematics and Science Didactics (i.e. Mathematics and Science Education) carry out with respect to student's errors and misconceptions in different subject matters.

In Logic, for instance, Barker-Plummer et al. [2008] report that "*Students were found (a) to have particular difficulties with distinguishing the conditional from the biconditional, (b) to be sensitive to word-order effects during translation, and (c) to be sensitive to factors associated with the naming of consonants*." Also Barker-Plummer et al. [2009] find that not only is it possible to locate specific systematic errors in Logic learning, but that there are differences in the number of attempts students need to fix them, and, more interestingly, that not necessarily "easier" errors are easier to resolve.

But these researches have not necessarily made an impact in everyday teaching and learning. In school settings, students usually feel that errors are something to be ashamed of, reason for reproach or even sanction. More often than not, they cannot decode their meaning or origin, and attribute them to bad luck or teacher's animosity. On the teacher's side, there is often "common sense pedagogy", that conceives of teaching just as the well structured transmission of information or skills, vaguely reminiscing of Behaviorism. In this light, student's errors are either regarded as the outcome of laziness, or lack of motivation, or, in the best case, a responsible teacher will look upon them as a sign that something must be changed or amended in lesson plans or worksheets [Astolfi et al., 1998].

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 183–189, 2011.

2 Looking at Errors from a Different Point of View

We may draw a partial parallel between the idea that every student's error is their fault and typical interpretations of Wason's task[1] [Wason, 1966a; Wason & Shapiro, 1971] in psychological research. At first, the deviations from normative expected responses were taken to imply the irrationality of human reasoning abilities. But that has not been the unanimous explanation. Stanovich & West [2000] classified a corpus of alternative explanations referring either to performance errors, or to computational limitations, or to diverging interpretations posited by the researcher and the subject, or, finally, to the subjects' construal of the task being at variance from the expected version. For instance, Stenning & Van Lambalgen argue that since grammatical form and logical form are not interchangeable (in the sense that there may be different logical interpretations of a given grammatical form, and viceversa), there are multiple potential interpretations for the expressions in an argument. So, we should first determine the relevant interpretation so as to be able to establish whether performance should or should not be considered adequate. Sometimes, subjects in Wason's task do not give the expected answer, but the answer they do give is compatible with their chosen interpretation. In these cases, subjects are not being irrational, they are failing to employ the researcher's meaning for the task. Therefore, many examples where *performance* on Wason's task has been taken to indicate a departure from rational thought, should instead have been considered as instances of the use of a divergent *interpretation* of it. Correlatively, in a classroom setting, what we consider an ignorant error -for instance, an error related to misunderstanding the truth-tree of the conditional- may be seen as an intuitive mistake -where the student inadvertently interpreted the sentence in modal terms.[2] But, what is an intuitive mistake, and how will we distinguish it from an ignorant or careless one? One way to start discerning intuitive and ignorant mistakes is by distinguishing *systematic* from *unsystematic* or "casual" errors. Teachers discern student's errors in different situations: written work, oral interventions, group or individual tasks, questions asked, etc. Unsystematic errors appear randomly and may be attributed to carelessness or lack of concentration, and are very difficult to anticipate. Systematic errors, on the other hand, appear with a certain frequency, and experienced teachers come to expect them. Charnay & Mante [1990] mention two traits of systematic errors: *co-errance* (i.e. they are recurrent both in individual students and across groups), and *coherence*, (i.e. they are not isolated, but appear in a relationship with others, interconnecting in a system or network of sorts).

From an educational point of view, student's errors across the disciplines may be regarded as the outcome of different sources: conceptual development processes, epistemological obstacles, and social representations. Both Piaget's genetic epistemology and Bachelard's theory of epistemological obstacles have been drawn from to analyze errors and misconceptions, bringing about a rich corpus of research in the Didactics of Mathematics and the Natural Sciences.

---

[1] "It is argued that subjects did not give evidence of having acquired the characteristics of Piaget's 'formal operational thought'." [Wason, 1966b]

[2] In [1960] J. Bruner distinguished what he called "intuitive mistakes" from "ignorant" ones, and required that teachers should be both sensitive to the difference, and ready to give simultaneous approval and correction to the intuitive student.

Piaget (and neo-piagetians) have taken into consideration both the historical and the individual point of view. In Piaget's framework, the growth of knowledge, both personal and theoretical, shouldn't be regarded as a smooth linear course, but rather as a progression of successive configurations and reconfigurations, through a many-layered hierarchy. At each level, knowledge is in a state of dynamical balances and imbalances, and successively becomes the object of equilibrium mechanisms, to be re-stabilized on a higher level. Thus, error becomes a core factor within the theory, as a necessary step in the development process through consecutive levels. Furthermore, the historical-critical approach to the history of science holds that the development of knowledge gathered by humankind through time may be read in some cases as related to some extent to the development of thought in the individual. So, history of science may be regarded as a sequence of stages that develop neither at random or haphazardly, but as a chain where every link becomes both a consequence of prior ones, and a necessary step to subsequent ones. Rolando Garcia and Jean Piaget [García, 1999; Piaget y García, 1982] show how in some cases, the mechanisms of passage from states of lesser knowledge to states of higher knowledge in the individual psychogenesis may be read with reference to the development of those concepts in the history of science.[3]

Genetic psychology and epistemology may be described as *constructivist* in so far as they refer to a conception of knowledge that goes beyond empiricist and apriorist epistemologies. Scientific knowledge is regarded as a more sophisticated and better structured continuation of commonsense intuitions, norms and practices, provided two conditions are fulfilled: internal coherence and consistency, and (in empirical theories) positive empirical testing. Genetic epistemology does not imply any anti-realistic metaphysical position, such as Von Glasersfeld's radical constructivism.

From Bachelard's point of view, the history of conceptual development within the sciences has been marked by the fact that every new concept is built on the basis of overcoming a prior understanding, which has operated as an obstacle. But these obstacles should not be regarded as wholly negative: new knowledge would not be possible without their occurrence. While Bachelard's work is epistemological in a sense concerned in particular with the building of knowledge, since the 70's, G. Brousseau [2007, 2009] has given a pedagogical (didactical) version, distinguishing "*erreur*" from "*faute*" in the teaching and learning of Mathematics.

We do not intend to draw an unequivocal parallelism here between historical scientific development and the building of individual knowledge. But nonetheless, we believe that a careful look at the history of Logic can help us determine obstacles and points of special concern in discerning and interpreting those systematic difficulties, errors and misconceptions which show distinctive patterns and can be found across the work of different students. Also, it may help us tell the difference between mere slips (or "ignorant mistakes") from genuine systematic errors ("intuitive leaps").

For instance, even though we certainly don't mean to establish a strict analogy between Logic's historical development and the individual's cognitive processes, we think this framework gives support to the priority of Classical Logic over Subclassical

[3] Mechanisms of passage from one historical period to the subsequent one may provide insight into the mechanisms of passage from one individual stage to the next, not as a strict analogy, but as a methodologically fertile comparison.

and Supraclassical Logics on the teaching Logic process. It was not mere coincidence that Logic coalesced into a discipline when Aristotle developed Classical Logic. Therefore, as Deductive Logic became the starting point for the building of other Logics, so it is a requisite point in the teaching and learning process both of Classical Logic and of the beginner's understanding of the many diverse logical systems that may be found in current logical literature [Palau, 2010].

3 Systematic Errors in the Learning and Teaching of Logic

Systematic errors, in the sense outlined above, are therefore to be considered as signs of constitutive components of knowledge development, which should not be looked upon as mere lack of knowledge but as a step in the building thereof. It is a peculiar, incomplete, insufficient form of knowledge, but at the same time, an important starting point in the development of new understandings. Success or failure in successive experiences will allow the necessary adjustments, in a spiral progression, rather than in an all-or-nothing perception of learning. There is no learning without error, neither at school nor in the history of science. So, error must be considered as a significant issue not only in psychological research, which has been so in recent decades, but as an essential component of teaching. Good teaching, from this point of view, can be enhanced by careful research and analysis of errors and misconceptions.

But good Logic teaching, and any educational research that takes into consideration systematic error, should not be regarded as mere direct derivations from psychological research. From a psychological point of view, we may, evidently, draw from modern theories of reasoning and human problem solving. But from a philosophical and logical point of view, we believe teachers need take into account contemporary non-classical Logic research. If we assume (as Stenning & Van Lambalgen do) that there are multiple interpretations, derived from different logic frameworks, that a student may employ when performing a task, then identifying and establishing which kind of responses would be considered erroneous is not only a psychological concern but a logico-philosophical and pedagogical one. As we have hinted before, what can be considered a fault within a Classical Logic framework, may not be so in a Non Classical Logic one.

In another text [Palau, 2009] we have stated our belief that "natural logic" may become a true didactic obstacle in learning crucial logical subjects, such as the concept of deduction. Students usually reject the principle that a true conclusion may be inferred from false premises in a valid reasoning schema, even though they may have shown the requisite deductive abilities in mathematical theorem demonstrations. Natural logic seems to reveal at least three main traits that differentiate it from Formal Logic (FL). (i) Natural logic (NL) does not build series of inferences as syntactically complex as those found in FL systems. On the other hand, NL arguments are harder to analyze, since they involve a pragmatic dimension lacking in FL. (ii) NL arguments diverge from FL ones in that they do not follow a step by step inference sequence, but rather progress by leaps and bounds based on lack of information or on unspecified presuppositions. (iii) NL logic inferences are usually tied to speakers' beliefs on the truth-value of sentences concerned, and the common use of the terms involved. In other words, NL inferences may be taken to be "context-dependent".

In an in-progress research with first year students' exams, at Universidad de Buenos Aires, we have found some instances of systematic errors related to the relationship of truth and validity: for instance, having difficulties to tell sentences from arguments (especially in the case of conditionals) and logical laws from valid argument forms; or the validity of an argument from the truth of the premises. For instance, trying to determine a case of validity, a student states: "*The [first] premise is true, the second one ("bread is poison") is false; therefore, the conclusion is false, or else the bread may be poisoned. Which would give us a true conclusion, since we cannot know for certain whether the bread is or is not poisoned.*"

4 Systematic Errors in Logic Teachers' Education

Students will sometimes have a strenuous task in overcoming natural inference biases in order to be able to assimilate new formally correct reasoning strategies and schemas. This sort of conceptual change has been shown to be much harder to achieve than it originally may have been expected to be. So, it rests with teachers and educational researchers to devise didactic strategies apt to make change possible and favor the building of new structures and new logical abilities by means of reflective abstraction [Dubinsky, 1991]. This can be regarded as an almost paradoxical process: human logical competence, Natural Logic, is the genesis of Logic as a science, but at the same time, learning the scientific outlook implies breaking with the intuitive one.

From this standpoint, research on systematic errors may be of particular relevance to those interested in teaching Logic, since they offer pertinent data on the growth of student's understanding, and of those cases where it strays from the expected patterns not as a result of negligence or distraction, but as a step in that growth. Consequently, they may become teachers' and researchers' point of interest in devising strategies to foster or modify such representations.

From a didactical (i.e. teaching) point of view, making explicit a logico-philosophical framework will be crucial in deciding on the corpus of conceptual content to be taught; and the history of Logic may provide insights for that choice. Likewise, choosing a psychological framework will provide both a scaffold and a set of restrictions for effective teaching and learning possibilities.

We subscribe to the idea that teachers should create their own lesson-plans and activities in a professional capacity, aided by results from educational and Logic research, and not become mere technicians, reduced to reproducing what others produce for them. Therefore, to make this kind of teaching possible, pre-service and in-service Logic teachers' education should refer not only to Classical Logic but to non-Classical Logic systems.[4] Not, of course, so that non-classical Logic become part of high-school or college curricula. But enough familiarity with them should give teachers the requisite framework to recognize and analyze intuitive mistakes. Explicitly stating the context in which answers diverging from the expected results were comprehended, and drawing from theories of reasoning and problem solving may also help teacher design richer and more adequate lesson plans and activities that take into

[4] Even a first approach to non-monotonic logic may be attempted, for instance by reference to some chapters of D. Makinson [2005] Bridges from Classical to Non monotonic Logic.

consideration such errors and misconceptions. Also, they may profit from a familiarity with the history of Logic that is, now, quite rare to find in Logic or Philosophy teacher's education programs.

Deciding on the desirable goals and purposes of Logic teaching, and the relevant methodology become therefore the aim of a Didactic of Logic which will draw as much from the history of Logic and logico-philosophical standpoints as from choices pertaining to the best possible teaching strategies. This groundwork should let us design teaching class projects that take into consideration interesting systematic errors.

Acknowledgments. We would like to thank Universidad de Buenos Aires and UBACYT for the granting of Project 124 "Relaciones entre la lógica formal, la lógica informal y la argumentación filosófica: análisis crítico y consecuencias pedagógicas". Also, the useful remarks and suggestions made by two TICTTL anonymous reviewers.

References

Astolfi, J.-P., Peterfalvi, B., Vérin, A.: Comment les enfants apprennent les sciences. Retz, Paris (1998)

Bachelard, G.: La formación del espíritu científico. Siglo Veintiuno, Madrid (1999)

Barker-Plummer, D., Cox, R., Dale, R.: Dimensions of Difficulty in Translating Natural Language into First Order Logic (2009), http://web.science.mq.edu.au/~rdale/publications/papers/2009/EDM2009.pdf

Barker-Plummer, D., Cox, R., Dale, R., Etchemendy, J.: An Empirical Study of Errors in Translating Natural Language into Logic. In: Sloutsky, V., Love, B., McRae, K. (eds.) Proceedings of the 30th Annual Cognitive Science Society Conference, http://web.science.mq.edu.au/~rdale/publications/papers/2008/fp273Barker-Plummer.pdf

Brousseau, G.: Iniciación al estudio de la teoría de las situaciones didácticas. Zorzal, Buenos Aires (2007)

Brousseau, G.: L'erreur en mathématiques du point de vue didactique. Tangente Éducation 7, 4–7 (2009)

Bruner, J.S.: The Process of Education. Harvard University Press, Belknap Press Books, Cambridge, Ma (1960)

Charnay, R., Mante, M.: De l'Analyse d'Erreur en Mathématiques aux Dispositifs de Re-médiation. Quelques pistes.... REPERES-IREM (7) (Avril, 1992)

Dubinsky, E.: Reflective Abstraction in Advanced Mathematical Thinking. In: Tall, D. (ed.) Advanced Mathematical Thinking. Kluwer Academic Publishers, Dordrecht (1991)

García, R.: A Systemic Interpretation of Piaget's Theory of Knowledge. In: Scholnick, E.K., Nelson, K., Gelman, S.A., Miller, P.H. (eds.) Conceptual Development. Piaget's Legacy, Lawrence Erlbaum, Mahwah (1999)

Makinson, D.: Bridges from Classical to Non monotonic Logic. In: Texts in Computing, vol. 5. King College, London (2005)

Palau, G.: ¿Por qué hay que "enseñar" lógica clásica? Su prioridad histórica y epistemological. In: Trabajo presentado en el XV Congreso Nacional de Filosofía – AFRA (Asociación Filosófica Argentina). Buenos Aires (2010)

Palau, G.: La didáctica de la lógica desde una perspectiva cognitiva. In: XII Encuentro Internacional de Didáctica de la Lógica, Universidad de México, Querétaro (2009)

Piaget, J., García, R.: Psicogénesis e Historia de las Ciencias. Siglo XXI, México (1982)
Stanovich, K.E., West, R.: Individual Differences in Reasoning: Implications for the Rationality Debate? Behavioral and Brain Sciences 23, 645–665
Stenning, K., Van Lambalgen, M.: Human Reasoning and Cognitive Science. MIT Press, Cambridge (2008)
Wason, P.: Reasoning. In: Foss, B. (ed.) New Horizons in Psychology, Penguin Books, Harmondsworth (1966a)
Wason, P.: Reasoning about a Rule. Quart. J. exp. Psychol. 20, 273–281 (1966b)
Wason, P., Shapiro, D.: Natural and Contrived Experience in a Reasoning Problem. Quart. J. Exp. Psychol. 23, 63–71 (1971)

The AProS Project: Teaching Logic to Business and Engineering Students

Moris Polanco

Universidad Francisco Marroquín
Facultad de Ciencias Económicas
6a. calle final zona 10, Guatemala, Guatemala
mp@ufm.edu

Abstract. The paper discusses the use of the tools provided by the course *Logic & Proofs*, of the Open Learning Initiative (Carnegie Mellon University), for teaching Logic to freshmen students of Business Administration and Engineering at Francisco Marroquín University (Guatemala) over a period of two years. It is argued that the distinctive focus of the course—"strategic argumentation"—needs to be adapted to the students specific interests in order to be relevant to their education. Polya's "heuristic" approach to problem-solving is proposed as a complement to the course.

Keywords: Introduction to Logic, strategic thinking, automated proof search, Proof Lab, natural deduction, proof.

1 Context

The curriculum of Business Administration at Francisco Marroquín University (Guatemala) includes several humanities courses. Since 2009, Logic has been included as one, and is a required course for around one hundred and ninety students in their freshman year. Before the implementation of the L&P course in 2009, the students were required to take a course of Critical Thinking.

In 2009, a new program was opened in the School of Business Administration. It was specifically designed for students who would receive their degree (*licenciatura*) in Engineering, combined with Business Administration. Based on the profile of the students, it was decided to teach them Logic instead of Critical Thinking, the course taken by Business Administration (BA) students. Many options were explored in 2008 before the academic year started in January 2009, and finally the decision was made in favor of the course *Logic & Proofs* (L&P), of the Open Learning Initiative of Carnegie Mellon University. In 2010, the course of Critical Thinking for Business Administration students was replaced by a new course called "Logic and Critical Thinking," in order to reduce the gap between the two programs (BA and Engineering). While the Engineering students were taking Propositional and Predicate Logic, the BA students were taking Propositional Logic and Critical Thinking. For 2011, the plan is to teach the same course for both programs. Teachers will be required, however, to adapt the content to the interests and specific needs of their students.

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 190–197, 2011.

2 Logic & Proofs

L&P is a fully web-based "introduction to modern symbolic logic. It provides a rigorous presentation of the syntax and semantics of sentential and predicate logic. The distinctive emphasis is on strategic argumentation." [1] It has been developed in the Laboratory for Symbolic and Educational Computability (LSEC) as part of the AProS—Automated Proof Search—Project, which "consists of four separate, but deeply integrated parts, namely, the central proof search engine Proof Generator, the Proof Tutor, the Proof Lab, and the web-based course Logic & Proofs." [2] That project is being directed by Wilfried Sieg.

In order to take the course, students are required to create an account at the Open Learning Initiative of Carnegie Mellon University and pay a fee.

The course consists of four parts: an Introduction (one chapter: Statements and Arguments), Sentential Logic (seven chapters: Syntax and Symbolization, Semantics, Derivations, Indirect Rules, Strategy and Derived Rules, Elementary Metamathematics), Predicate Logic (five chapters: Syntax and Semantics I, Syntax and Semantics II, Derivations, Strategies and Derived Rules, Identity and Functions), and Additional Topics, which is an excursion into Aristotelic Logic in one chapter. In our experience at UFM, the material can be covered in 14 to 16 weeks.

Like the rest of the courses of the Open Learning Initiative, *Logic & Proofs* includes a Grade Book and a Learning Dashboard. The Grade Book keeps track of the student's work and scores (which are given automatically) in each of the "learn by doing" sections, practice problems, practice lab problems, homeworks, labs, and exams. The instructors can even recreate any student's work, in order to see the steps he or she took to solve a particular problem. All the data can be exported to an Excel or Blackboard file, or printed out easily. In the Learning Dashboard, instructors get a complete report of how many students have been working on a particular module, how much of the module those students have worked, how much help students need in a module, and how individual students are doing in each module. This is very helpful for instructors. In fact, in my opinion, instructors must avoid the risk of putting too much attention on the scores, while losing sight of the learning objectives of the course.

L&P is complemented by a set of web-based tools that are indispensable in its design as a fully web-based introduction to modern symbolic logic. These tools are the Proof Lab, the Truth Lab, and the Proof Tutor, along with a series of small interactive learning environments along the lessons (called "Learn by Doing").

The Proof Lab, as its creators tell us,

> is a proof construction and student management system. The Lab offers a unique "backward-forward" proof representation through Fitch Diagrams. Students can apply rules backward in the Goal Tree, and forward in the Fitch Diagram. This representation allows students to employ in a very direct way the special strategies developed for the AProS Proof Engine.

> The Proof Lab handles "bureaucratic work" for students, allowing them to focus on the structure of arguments. On the one hand, it keeps track of justifications and applies rules in the Fitch Diagram, after they have been specified by the student. On the other hand, the Proof Lab points out mistakes, when students misapply a rule (for instance, applying elimination rule for the conditional to a conjunction) or mix separate subproofs. It provides the student with information to correct mistakes [3][1].

One of the advantages of the Proof Lab is that it minimizes the working memory load. Sweller [12] demonstrated that a high working memory load can interfere with problem solving.

The Truth Lab is the semantic counterpart of the Proof Lab. It was developed by Dawn McLaughlin, member of the staff of the AProS Project, in 2009. "As the Proof Lab engages the student in the conceptual essentials of proof construction, so the Truth Lab engages the student in the conceptual essentials and techniques that utilize the definition of truth in a fundamental way." [4] In 2010, the Truth Lab was greatly improved, and it now can be used to make truth tables along with truth trees.

Finally, the Proof Tutor, started in 2008,

> is the bridge between Proof Generator and the Proof Lab. It enables students who are stuck on a proof to receive hints, dynamically obtained from Generated Proofs. If a student requests a hint, Proof Generator will construct a complete proof, which the tutor analyzes. The tutor then provides hints to construct the proof using the efficient and natural strategies employed by Proof Generator. The first hint provided at any point in the proof is a general strategic one, and subsequent hints provide more concrete advice as to how to proceed. The last hint in the sequence recommends that the student take a particular step in the proof construction [2].

According to Douglas Perkins, "the goal of the proof tutor is to provide high level advice for students on proofs with the intent that students will learn to incorporate techniques from hints into their own reasoning." [6] The convenience of providing immediate feedback on errors, however, is a matter subject to discussion. Some argue that "the chief benefit of immediate feedback is to reduce time learning substantially (Anderson, Koedinger and Pelletier, 1995). However, others point out that the immediate feedback can interfere with aspects of learning" [8].

In my opinion, the Proof Tutor follows in line with the "general consensus" that "seems to be emerging on the context of advice messages":

> When error feedback is presented, it should generally just signal the error without commenting. This enables the student maximum opportunity to analyze the correct situation. When advice is given, the most

[1] In the AProS home page (http://www.phil.cmu.edu/projects/apros/) there are four videos worth watching, about the course in itself, the Proof Lab, The Truth Lab, and the Tutor.

> cost-effective content focuses on "reteaching" a correct analysis of the situation rather than debugging misconceptions. This correct analysis should be administered in at least three or four stages: (1) a reminder of the problem-solving goal, (2) a description of relevant features of the current problem state and the desired goal state, (3) a description of the rule for moving from the current state to the desired state, and (4) a description of a concrete action to take [8].

What makes L&P more than a simple web-based course is that the developing team is working continually on its development. The surveys provided by hundreds of students from different backgrounds, as well as the instructors feedback, provide them with the information needed to make the necessary adjustments to the content or to the tools[2].

3 The Strategic Approach

The strategic approach is the distinctive emphasis of L&P. Basically, it is "an ordered sequence of tactical steps" [1] that leads the student to an efficient proof construction. The tactical steps that the student is required to take, in sequence, are Extraction, Conversion, Inversion, Division, and Refutation. To apply these steps, the student must have a clear notion of every one of them, which in turn requires an understanding of the notion of "positive subformula."

The authors of the course assure the student that

> If you follow this procedure to the letter you are guaranteed to successfully complete your derivation—eventually—that guarantee being given by virtue of the fact we mentioned at the beginning. . . At first, you might find it a bit tedious to work through the procedure step by step, but if you keep with it, eventually it will become second nature and you will find that you are able to complete derivations with a minimum of effort. [1]

In the 2010 edition of the course, the procedure is nicely illustrated by flowcharts, and is explained in detail as an algorithm. Besides that, the students have the help of the Proof Tutor for some practice problems, which guide them in the application of the algorithm.

In my experience teaching L&P for two years, the potential problems for the strategic approach are at least two:

[2] At the end of the Spring course of 2007, for instance, Douglas Perkins ended a presentation with the following remarks regarding the Proof Tutor: "Where to go from here? The tutors described here show a comprehensive way to use an automated theorem prover to dynamically produce hints for students in proof search. Just how good is this tutor? While it can provide useful hints to students, what will its eect be on the rst few weeks of [working] with proofs? What components of it are the most useful? These questions may be addressed this winter, using logging data from the fall term." [13]

First, many students see the tactics as an unnecessary procedure for solving problems. "Much of the time, students look at the section of a partial proof and see immediately how to finish it." [6] In other cases, they proceed by guessing, and do not care if they take unnecessary steps. The Proof Lab, in this respect, can be a dangerous tool, since the students may arrive at a proof applying rules indiscriminately, taking advantage of the "memory" of the Proof Lab. In this respect, it is of crucial importance that the students learn also to prove arguments using just pen and paper, and compare their procedure with the one they follow using the Proof Lab[3].

Second, the strategic approach can be "too mechanical." According to Gilmore, the "emphasis is not on how well the user achieves the current task goals, but on how well they learn about the nature of that task in some general, abstract way." [7] Perkins acknowledges Gilmore's criticism, and points out that "successful tutoring ought to be responsive to the skills of the student, and this holds for strategic proof tutoring just as it does elsewhere," and even recognizes that "it can be instructive . . . to allow students to use inference rules in valid but unproductive ways . . ." [6]

Some advantaged students may understand the benefits of the strategic approach, once they have also learned its *raison d'être*. Even recognizing that it is an algorithm, they must understand how their creators arrived at it. But that is not easy for the average student.

Douglas Perkins develops an interesting approach in his masters thesis for successful tutoring in propositional logic:

> To account for increasing skill as the student learns, then, there are three distinct tutoring levels or modes: tactical explanation, walking through a proof, and completing a partial proof. . . A tactical explanation is a goal-specific piece of information explaining which tactics can currently be employed, walking through a proof provides an example of how to think strategically, and completing a partial proof provides students with on-demands hints [6].

The main advantage of the "tactical explanation" is that it shows the student that "successful problem solving involves the decomposition of the initial problem state into subgoals and bringing domain knowledge to bear on those goals" [6].

The "walking through" level makes use of the Proof Tutor. Students are required to pay close attention to the hint it produces, and try to understand its justification.[4] The instructor must point out, though, that the hints make sense within the strategic approach that the Proof Lab uses.

[3] "As in any software environment, the efficiency with which the student can solve problems is an important consideration. However, the ultimate and unobservable product is knowledge. As a result, the assessment of educational software ultimately involves a transfer task: how well the students can solve problems working on their own outside the tutoring environment." [8]

[4] The controversy about immediate feedback was outlined in the previous section.

The "completing proofs" level implies making a non-normal proof normal. "Normal proofs are preferred not because non-normal proofs are incorrect, but because non-normal proofs have extra clutter that is both cumbersome to the students cognitive load as well as simply unnecessary." [8] The student is guided by the instructor, with the help of the Proof Tutor, as to how to make a normal proof.[5]

4 Making Logic Relevant

Although the intrinsic value of Logic is quite obvious, teachers usually experience the need to "sell" the usefulness of Logic to students, particularly when they are not students of Philosophy or Mathematics. In the case of the three philosophers and one engineer that teach Logic to Business students at Francisco Marroquín, this is particularly critical.[6] Often, we are faced with the question "What is the use of it?" by students that are eager to know the best techniques to start a business, or how to perform well in marketing or finance. My usual answer to those students is that classes like Logic and Calculus will help them to take their thinking to more abstract levels, probably helping their brains to make more connections, and in that way enable them to be more creative and to "see" more business opportunities. But that is only a guess on my part, and I do not have empirical evidence to support it.

In a report presented in 1990, Wilfried Sieg (currently, the AproS Project Director) and Richard Scheines wrote that

> For our project [the Carnegie Mellon Proof Tutor] it was crucial to have a "theorem proving system" that can provide advice to a student user; indeed, pertinent advice at any point is an attempt to solve a proof construction problem. To be adequate for this task a system must be able to find proofs, if they exist, and to follow a strategy that in its broad direction is logically motivated, humanly understandable, and memorable. [9]

On the other hand, I share Corbertt, Koedinger and Andersons view that

> We are just entering a time when intelligent tutoring systems can have a real impact in the educational marketplace as technology costs decline. . . this will happen only if ITS [Intelligent Tutoring Systems] research focuses on *educational outcomes* as well as AI issues. . . [i]t is important for the field to remain focused on *valid pedagogical principles and educational outcomes* in exploring these areas. [8] (italics added)

[5] Perkins gives a detailed explanation on this topic on Appendix B of his thesis.

[6] In the case of the Engineering program taught at Francisco Marroquín, it is important to point out that it is not "pure engineering," but a mixture of Business Administration and Engineering. The degree that the graduates receive after a four-year study program is "Entrepreneurial Engineering."

In my opinion, we need to make a harder effort to adapt the ITS to valid pedagogical principles. In particular, we need to use our teaching methods to make clear the importance of proofs and sound argumentation. That way, students can better appreciate the value of the "theorem proving system" provided by the Proof Lab of the AproS Project. But besides that, I think that in order to be "humanly understandable, and memorable," the strategy that the system follows still needs to be tuned up.

In order to make the system more humanly understandable and memorable, my proposal is to incorporate Polyas "heuristic approach" [10] to problem solving to the "strategic approach" of L&P.

In *How to Solve it*, Polya [10] suggests the following steps when solving a mathematical problem (these steps are the summary of his "heuristic approach"):

> First, you have to understand the problem.
> After understanding, then make a plan.
> Carry out the plan.
> Look back on your work. How could it be better?

Of course, a logic problem is not a mathematical problem, and heuristic reasoning is not the same as making a formal proof. At the same time, it is important that the students understand the problem, not only in its logic form, but in its context. Examples taken from the LSAT, for instance, can illustrate the importance of logic reasoning for parliamentary debates. It would be a great improvement, in my opinion, if L&P contained more examples taken from everyday experiences, or from scientific problems.

It is true that Polya writes that it is bad "to mix up heuristic reasoning with rigorous proof" [10], since "[F]or a logician of a certain sort, only complete proofs exist. What intends to be a proof must leave no gaps, no loopholes, no uncertainty whatever, or else it is not a proof. Can we find complete proofs according to such a high standard in everyday life, or in legal procedure, or in physical science? Scarcely" [10].

However, Polya also says that "[W]e need heuristic reasoning when we construct a strict proof as we need scaffolding when we erect a building" [10], and that is why I think that heuristic reasoning can be a good complement to strategic thinking. Strategies and tactics only make sense in the context of the discovery.

We all want to avoid the situation described by Barwise and Etchemendy:

> Many students try to construct formal proofs by blindly piecing together a sequence of steps permitted by the introduction and elimination rules, a process no more related to reasoning than playing solitaire [11].

In order to do that, students need to understand the problem and make a plan, and I would argue that even before that, they need to see the problem, in its human or scientific context.

5 Conclusion

L&P is an excellent course for teaching Logic to university students. Its web-based tools are the state of the art in the field. Students can certainly master propositional and first-order logic in 14 to 16 weeks, provided that they are properly guided in their learning process. The web-based tools (the Proof Lab, the Truth Lab, and the Proof Tutor) need to be complemented with traditional teaching (blackboard and "pen and paper") to avoid becoming a device used by the students to deliver solved problems. "Transfer of knowledge to other environments, notably the non-tutor environment, is essential" [8].

On the other hand, I propose that the "heuristic approach" to problem solving as first presented by Polya in 1945 be incorporated to the strategic approach that is at the center of the L&P course. This goal requires the enrichment of the material presented in the course with problems and examples taken from the students everyday experiences, and from sciences that they are studying as part of their regular curriculum.

References

1. Logic & Proofs, http://oli.web.cmu.edu/openlearning/forstudents/freecourses/logic
2. The AProS Project, http://www.phil.cmu.edu/projects/apros/index.php
3. Proof Lab, http://www.phil.cmu.edu/projects/apros/index.php?page=prooflab
4. The Truth Lab, http://www.youtube.com/watch?v=UM1rCHH5wh8
5. Logic& Proofs: The TruthLab, Proof Tutor, http://www.phil.cmu.edu/projects/apros/
6. Perkins, D.: Strategic Proof Tutoring in Logic. M. S. Thesis, Carnegie Mellon (2007)
7. Gilmore, D.J.: The Relevance of HCI Guidelines for Educational Interfaces, http://nth.wpi.edu/classes/cs525t_tutoring_systems/paperforclass/gilmore-relevance%20of%20hci%20guidelines.htm
8. Corbett, A., Koedinger, K.A., Anderson, J.: Intelligent Tutoring Systems. In: Helander, M., Landauer, T.K., Prabhu, P. (eds.) Handbook of Human-Computer Interaction, pp. 849–874. Elsevier Science B. V., Amsterdam (1997); Second, Completely Revised Edition
9. Sieg, W., Scheines, R.: Searching for Proofs (in Sentential Logic), Report CMU-PHIL-21 (1990)
10. Polya, G.: How to Solve It, 2nd edn. Princeton University Press, Princeton (1985)
11. Barwise, J., Etchemendy, J.: Logic, Proof and Language. CSLI Publications, Stanford (2008)
12. Sweller, J.: Cognitive load during problem solving: Effects on learning. Cognitive Science 12, 257–285 (1988)
13. Perkins, D.: Slides (pdf). Strategic Proof Tutoring in Logic, http://dperkins.org/thesis/present.pdf

Using a Learner- and Teacher-Friendly Environment for Turing Machine Programming and Testing

Rein Prank and Mart Anton

University of Tartu, Institute of Computer Science, Liivi Str. 2,
50409 Tartu, Estonia
{rein.prank,mart.anton}@ut.ee

Abstract. The paper describes a learning and assessment environment for Turing Machine programming. The interpreter allows to explore different phases of running a particular test in different working modes: step by step, until a change on the tape, until termination, take a step back, etc. The program enables to display the results of testing with a set of tests together with error messages, numbers of steps, etc. We try to describe small details of the program that are important for learner and for instructor.

Keywords: Turing Machine, Automated testing, Assessment.

1 Introduction

Turing machines (TM), together with proofs in logical calculi, are the most popular topics of computerization in courses of mathematical logic. The main reason for creating exercise environments is quite obvious. The students need external feedback about their solutions. In case of a Turing Machine, the best-known form of feedback is some device for visualization of the work of a TM with given data. However, a good computer environment has other potential benefits as well: syntax-oriented editor, copy-and-modify, reasonable diagnostic messages.

Turing's World [1] could be the most recognized TM simulator, but there are tens of others available. The software for Turing Machines is quite sensitive to small details of concrete definition of TM and, therefore, many teachers have initiated their own projects of computerization. There are successful applications but, in many cases, the use of self-made software has stopped after completion of the respective programming project. This could be due to a variety of reasons: poor visual presentation of the work of TM, inconvenient user interface, insufficient intelligence of the program. Many programs provide a nice exercise environment for students, but the instructors cannot obtain data about students' performance in homework and tests.

In the University of Tartu, we use a TM interpreter since 1988 [3, 4] for exercises and a TM programming test. In this paper we try to describe the tools contained in our Interpreter for programming, for debugging and for teacher's review of student assignments. Section 2 provides a brief overview of the chapter on Turing Machines in our course. Sections 3 and 4 describe the first and second version of our program. Section 5 considers some pedagogical issues and summarizes the results.

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 198–206, 2011.

2 Turing Machines in Our Basic Course of Mathematical Logic

In our course 'Introduction to Mathematical Logic', the chapter on Turing Machines contains three 90 min lectures, two 90 min computer labs with instructor and a 90 min final test on TM programming. The lectures present the definition of a TM, conventions for computing of numerical functions, composition and branching constructs, Gödel numbering of a TM and the first unsolvability results including the halting problem. The last lecture also includes an informal overview of further theorems (including Rice's theorem) and a description of the implications of such results for the theory and practice of programming and testing.

We use the definition of Turing Machine from "Introduction to Metamathematics" [2]. The commands have the form $s_a q_b \rightarrow s_c q_d K$ where s_a and s_c are symbols from the alphabet of a TM (or space), q_b and q_d are states of machine and $K \in \{L,R,C\}$. However, for computation of numeric functions, we use a slightly modified coding of natural numbers on the tape (borrowed from the lectures of A. A. Markov at the Moscow University). The alphabet of a TM contains two nonempty symbols 0 and I, a natural number x is coded by the word 0I…I with x strokes and the n-tuple of numbers has the form 0I…I0I…I…0I…I. The computation of $f(x_1,\ldots,x_n)$ begins with the head of TM placed to the zero of the last argument and should terminate with $f(x_1,\ldots,x_n)$ at the end, with the head placed to the zero of the result. Using two nonempty symbols on the tape helps to avoid unnecessary tricks and makes the machines simpler.

We do not want to force students to program Turing Machines for complex functions. Our aim is to offer them some basic level of experience with Turing Machine programming, enabling them to understand the constructions used in proofs of theorems. We also hope that the experience of programming a wide spectrum of elementary functions gives the students a basic grasp of the Turing-Church thesis.

3 First Version of TM Interpreter

The first version of our TM Interpreter was written in 1988 in Turbo Pascal 3 by the first author of this paper as a part of our package for exercises in Mathematical Logic [4]. The TMs were created in the form of a table. The program ran in DOS ASCII text mode, with the screen containing 25 rows by 80 symbols. Each symbol could have one of 16 text colors and one of 8 background colors. In our program, the tape, the TM table and the areas for runtime data, menu, instructions, etc., were implemented as screen regions with different background colors. In order to enable counting of consecutive spaces on the tape, we used minuses to highlight the spaces.

The program enables to edit the TM table, run the machine with input data from a user, and view the testing history. The machine can be saved in a file and loaded for further editing and running. Figure 1 shows the main screen of the program.

Line 1 contains comments, specifying the function, the author, and the file name.

Lines 2-4 contain the tape with an arrow at the head position and markers of locations of arguments and value to be calculated.

Lines 5-6 are for instructions (like "Enter the arguments") and error messages.

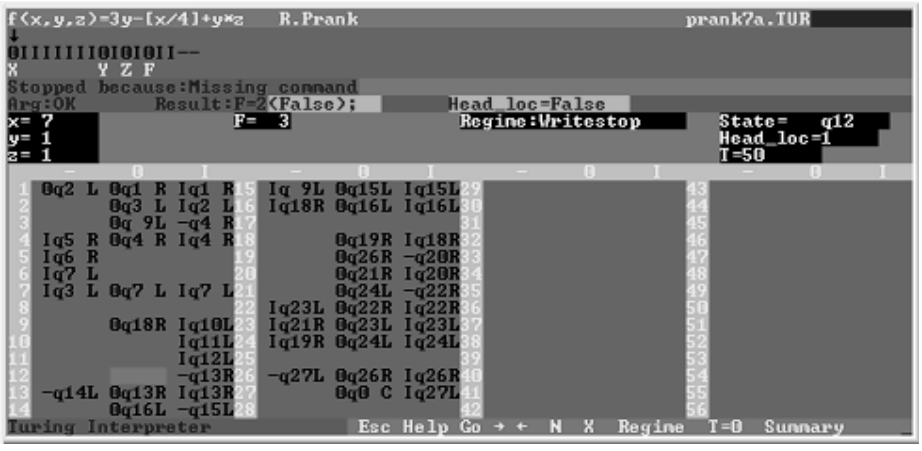


Fig. 1. Main screen of the first version of TM Interpreter. The computation terminated normally at moment T=50 because the head of TM scans the symbol 0, the memory state is q_{12} and the corresponding cell in the table is empty. The arguments are kept in their initial form but the value of function is wrong and the head of TM is not placed at the zero of the result.

Lines 7-9 contain data about the current run: values of arguments x, y, z, the expected correct value of the function, current working mode. The rightmost column displays the state of machine, location of the head on the tape and time (number of executed steps).

Line 10 contains table headings and lines 11-24 contain four columns with the (right sides of) commands of the machine. The maximum number of states is 56.

Line 25 contains the name of the currently working part of the program (Editor or Interpreter) and the corresponding menu.

The student can switch between *Editor* and *Interpreter* (on the same screen) or view the *Summary* of testing (a scrollable table of test runs replaces lines 2-9).

The Editor is syntax-oriented. It allows entering only the syntactically suitable symbols or space at each position. After insertion or deletion of rows in the table, the editor automatically changes the references to the shifted rows.

When the student switches to the Interpreter then the program checks the syntax of the table. However, messages about incorrect commands are treated as warnings. The students usually test every accomplished part of the TM table directly after composing it and at such a moment some commands refer to empty lines or do not contain the number of state.

In order to run the machine, the student enters the arguments and the correct value of the function (as decimal natural numbers). The work of TM can be observed in four working modes: Stepwise (one keystroke = one step), Writestop (stops only before a change of symbol on the tape), Nonstop (continuous animation), and Blind (executes the whole computation and then displays only the final situation).

The menu of the Interpreter contains the following action options:

Help – explanation of the menu items,
Go – start the interpretation,
Arrows – shift the tape (if the tape contains more than 80 symbols),
N – enter the number of arguments (1, 2 or 3),
X – enter new values of arguments and function,
Regime – change the working mode,
T=0 – start with the same arguments from the moment T=0,
Summary – display the table of test history.

The Interpreter executes the TM commands until normal termination (TM goes to the passive state q_0 or the cell for the command to be executed is empty), until user presses the key 'S' (Stop), or until an error situation (the command to be executed is syntactically incorrect, the maximum number of steps has been made, the command requires moving the head to the left from the position 0 or to the right from the position 255).

After abnormal termination or Stop, the Interpreter displays a corresponding message. After normal termination, the program checks whether the arguments are kept on their initial location on the tape, whether they are followed by the word of form 0I…I (only), whether the result on the tape is equal to the correct value of the function, and whether the head is at the zero of the result. In case of inconsistency, the program displays an error message. In all cases a new row is written to the Summary table (entered arguments and correct value of the function together with 'OK', 'Stop', runtime error message or diagnosed faults). The student can then enter and run another test, rerun the same test (in a more detailed working mode if necessary), or switch to the Editor. In the latter event, the program places a border line with words "Editor used" in the Summary table. This means that the student should rerun the tests after editing the table.

We used our program for exercises and for assessment. The last TM task was a test where the students had to program a Turing Machine for computing a function similar to that on Fig. 1. Twenty minutes before the end, we disclosed the instructors' test data (using an overhead projector in the first years). Usually, it contained 7-10 tests. The students could enter and run all tests and correct the machine if there were mistakes and if they had enough time. We also asked them to show us the Summary of testing and collected the files with their TMs. In the first years, this was done using floppy disks, later e-mail. The final grading took place without students.

Some features of our TM Interpreter provided an opportunity for some less conventional tasks. As the Interpreter counts the steps, it was possible to assign a task of programming a function (like $x+y$, $|x-y|$ or $x \cdot y$) and finding the precise function of time complexity in the form of an algebraic expression or spreadsheet formula. Further, it was possible to compare those functions for different strategies of counting the result on the tape (for example, by copying first the strokes from x and then from y or vice versa). Sometimes we also organized homework competitions of programming the TM for computing a given function with minimum time complexity.

4 Second Version of TM Interpreter

We used the first version of TM Interpreter regularly for about ten years. However, the user interface (without mouse functionality) and screen appearance (regions of ASCII text) were clearly out of date by that time. In addition, we had recognized several functional shortcomings. The main problems included the following:

1. We wanted to get rid of the limit of 56 states. Our ordinary programming courses were now much more efficient and the students were able to solve more complex tasks of Turing programming too.
2. In some cases, the maximum length of the usable part of the tape (255 symbols) was also an obstacle. Some tasks with quite reasonable values of arguments required more steps than the maximum element of 16 bit data type Integer.
3. The compressed display of the table did not allow writing comments.
4. The Nonstop working mode – as a means for observing the work of TM in continuous animation – had become unusable, because it ran too quickly.
5. There was no possibility of saving tests in a file for repeated application. While it is useful for the student to invent an appropriate set of tests for each task, it is time-consuming if he/she should enter them again after editing the machine. It was even more tedious for the instructors, who had to enter the same 10 sets of data for each of the 50 students.

The second version of the Turing Machine Interpreter was written by the second author of this paper in 2000-2001 as his Bachelor's thesis. It is a MS Windows 32 bit executable. The problems listed above were solved in following way.

1-2. The new limit of the number of states in the table is 999. The program uses dynamic data structure for the content of the tape. The user can assign a time limit to each test. The maximum value is 999999 steps. Using even larger numbers of steps would have been possible, but then we would have had to think again about establishing a separate limit for the length of the tape. We have not had situations in our ordinary teaching/learning practice where the limits would have been too small.

3. We added a column for comments.

4. It would have been easy to set for the Nonstop mode a constant or user-defined duration in time units. However, we decided to abandon the Nonstop animation and, instead, added new modes of working: until selected command or state.

5. The new version enables to form and save a set of tests and to run them automatically.

The program also allows changing the font size for better visibility.

Five types of numerical and word functions were implemented in the new program:

1) numerical functions with OI…I coding of numbers (as in the first version),
2) numerical functions with binary coding of numbers,
3) numerical functions with decimal coding of numbers,
4) word functions where the argument must be kept on the tape before the result,
5) arbitrary word functions.

The main window of the Interpreter (Figure 2) contains Main menu, Toolbar, Passive and Active tape, Status bar of the Interpreter, a line for task description, Table and Table status line.

The passive (upper) tape contains the desired final situation according to the active test. The active tape displays the current moment in the run of the test. The numbers on the status bar of the Interpreter (under the tape) are: position of the leftmost visible cell of the tape (the tapes can be synchronously scrolled), position of the head, memory state of TM and number of the step. The next line is the description of the function (entered by the student). The lower part of the window contains the table and its status bar. About 30 states can be visible at the same time at usual screen resolution and font size. The status bar shows the number of states in the table, the number of nonempty states, the number of commands, the number of incomplete commands, and time of the last save/load action.

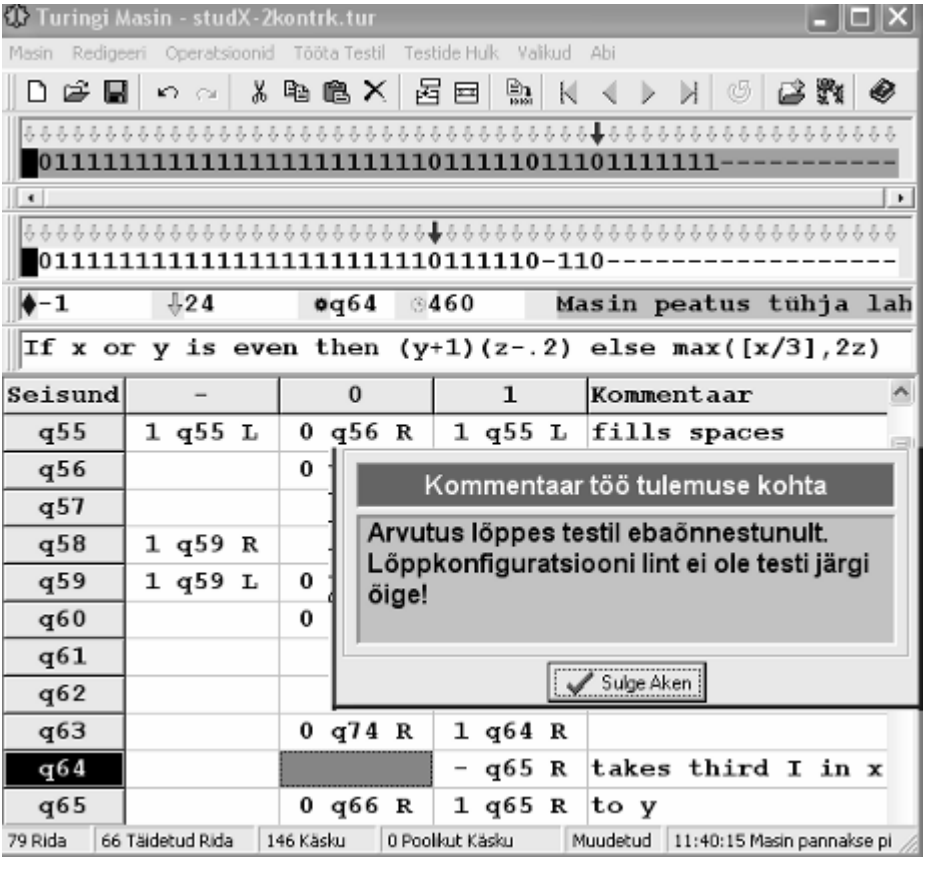


Fig. 2. Main screen of the second version of TM Interpreter after the end of running the test (test 10 in Fig. 3) and with a comment about the result: The tape does not have the correct form

The first submenu Masin contains the usual items of a File menu.

At any moment of the work with a particular Turing Machine, the user can edit the table or undertake an action with the interpreter. The submenu Redigeeri contains the usual items of an Edit menu. In addition, the submenu Operatsioonid contains the table operations for insertion and deletion of rows.

Specific actions with interpreter are available in the submenu Tööta Testil (Work on the Test): Enter the test, Work until State, Work until Command, Work until Change on the tape, Step Forward, Step Back, Work until Termination, Back to Begin, Refresh Situation. After selection of the first item, the program opens an additional window for entering the values of arguments, the correct value of the function, and a limit for the number of steps. The last item was designed for a situation where the active test revealed a mistake in the table, the student corrected the table and now wants to execute the same number of steps with corrected machine. The other items proceed from the situation created through previous action. Their effect is similar to the effect of a keystroke in the working modes of the first version. However, now it is possible to test a 'module' of TM so that one keystroke brings the computation to the first state of this module but, from then on, the student can inspect the work step by step. If the user edits the table between two execution actions then the program recomputes the previous steps too.

The most important new feature of version 2 was implementation of convenient use of tests. The user can enter different tests, use them repeatedly during programming of different parts of the machine, and save a test file. The submenu Testide hulk (Set of Tests) contains the usual file operations for test files and two specific items: Operations and Add Active test to the Set. After opening a test file or selecting Operations, the program runs all tests from the set and displays the table of results in a special window (Figure 3).

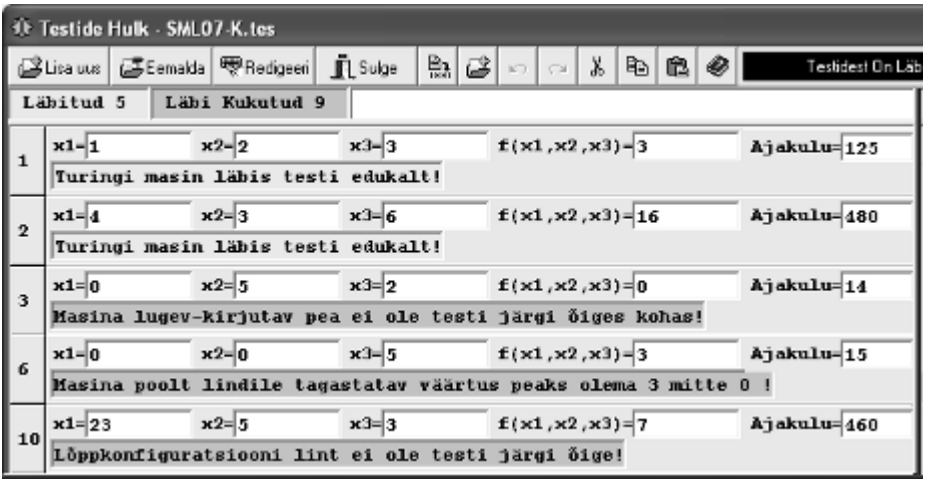


Fig. 3. The machine has passed in 5 and failed in 9 tests. We have presented here only the rows of tests 1, 2, 3, 6 and 10 from a set of 14 tests: arguments, correct value and numbers of steps. Failed tests 3, 6 and 10 display three different error messages: head is in wrong position, the value should be 3 but not 0, configuration on the tape has an incorrect form.

The toolbar of the window enables to perform operations Add new test, Delete test, Edit test; closing the window and switching to the main window: Make the test active, Add active test to the set. After addition or editing a test, the program runs it and displays the result in a table. The item Make the test active is a convenient way to start, in the main window, execution of a test that produced an incorrect result.

5 Using the Second Version

After introducing the second version, we made some trials, attempting to use exercises on functions with binary coding of numbers and word functions. We hoped to give the students some ideas about binary arithmetic but it seems that Turing Machines with one tape are not the right instrument for this. Moving the head requires too much attention. The word functions led us too far from our main direction. We prove the theorems using Gödel numbering and the students should understand the corresponding machines. Therefore, we use today only one of the five types of functions implemented.

Better general skills in programming and a more convenient testing process now enable to investigate a larger set of elementary functions and to implement more complex compositions of such functions. We take extensive advantage of the program's ability to execute quickly any number of tests from the test file. We make available our test files for some exercises (as we did earlier by assessment) and the students have enough motivation to correct their machines and to think about mistakes. On our part, we are able to review the homework of many students quickly.

Some automatic counter data on the screen of version 2 seem to be quite useful. The number of states and the number of commands, displayed on the status line of the main window, enable quick evaluation of the skills of the student. If necessary, we can recommend weaker students to look at some better examples of programming.

The vectors of numbers of steps enable to discover plagiarism even when some parts of the table are exchanged.

Our typical tasks for a final test are now more complex than ten years ago (Figure 2 and Figure 1). Nevertheless, the results of final tests on construction of Turing Machines are very good. For example, in the autumn term 2010, the average result of 50 participants on first attempt was 8.48 points from 10; 31 students received maximum points and only 3 students failed.

What could be added to the version 2? Version 2 works at any moment with data of one Turing Machine and one test file, although it is possible to change one of them, leaving the other unchanged. The teacher could also be interested in an overview of the solutions of a group of students, in an overview of the performance of the student over all tasks, or in test files of different students for some specific function. Such tasks could be solved using a server-based environment and database.

Acknowledgments. The work is financed by Estonian Science Foundation grant SF0182712s06.

References

1. Barwise, J., Etchemendy, J.: Turing's World. Cambridge University Press, Cambridge (1993)
2. Kleene, S.C.: Introduction to Metamathematics. Van Nostrand Company, New York (1952)
3. Prank, R.K.: Turing Machine Editor and Interpreter for PC. In: Second All-Union Conference on Applied Logic, Novosibirsk, p. 197 (1988)
4. Prank, R.: Using Computerised Exercises on Mathematical Logic. Informatik-Fachberichte, vol. 292, pp. 34–38. Springer, Heidelberg (1991)

Using an Argument Ontology to Develop Pedagogical Tool Suites

Chris Reed, Simon Wells, Mark Snaith, Katarzyna Budzynska, and John Lawrence

Argumentation Research Group, School of Computing, University of Dundee, Scotland, UK DD1 4HN
chris@computing.dundee.ac.uk
http://arg.dundee.ac.uk

Abstract. The teaching of argumentation theory, argumentation skills and critical thinking has only very recently enjoyed any bespoke software support for classroom activities. As software has started to become available, it has been characterised by idiosyncratic, incompatible approaches not only to data representation and processing but also to underlying theories of argument. The rise in popularity of the Argument Interchange Format ontology offers a principled solution to this problem, and we describe here three tools (OVA, Arvina and Parley) which use the AIF to provide pedagogical applications, and a sketch is given of how these tools can complement one another and can share resources.

Keywords: Argumentation Theory, Ontology, Argument Interchange Format; Dialogue.

1 Introduction

The study of argumentation, both as an academic discipline, and as a domain of pedagogy, has its roots in antiquity, but has only developed as a vibrant community in the past thirty years or so. It is the starting point, the precursor or the environment in which much logic teaching begins – standard logic texts typically have chapters devoted to the identification of fallacies, the expression of propositional logic in linguistic utterances, and the analysis of natural arguments in propositional, predicate or categorical logics.

Until the late 1990s, however, software support for either scholarly investigation or practical pedagogy was extremely limited. Since that time, many authors and teachers have explored tools that might support their activity in the classroom. Early prototypes were little more than proof-of-concept demonstrators that took theories of argumentation, or in some cases, theory of the pedagogy of argumentation, and showed that it was possible to employ them in the classroom (see, e.g., [8]). Gradually, software tools for manipulating argument resources started to mature and become more robust (for a good snapshot of such systems, see [3] for a review). As it became clear through rigorous analysis that the teaching of critical thinking skills had concrete benefits for students

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 207–214, 2011.

[9], a number of tools were developed to support argument analysis in particular (such as [7] and [5]). Harrell [2] provides a comparative review of many of these systems.

The proliferation of these tools, however, has led to challenges. Each tool is built with ad hoc and idiosyncratic conceptions of argumentation, and there is no scope for sharing and re-using resources between them. This is a serious shortcoming, because collecting and preparing resources for classroom use is a highly labour intensive task. The same problem faced the academic community in argumentation, which was struggling to develop resource sets and standards against which different theories and different techniques could be deployed. This led to a worldwide effort to develop a common language for representing argumentation which was sufficiently general to admit different philosophical conceptions of argument, whilst at the same time sufficiently precise to allow tool development and resource re-use. This representation language (or in fact, a set of languages defined against a common ontology) is now available as the Argument Interchange Format (AIF) [1]. This paper aims to show how the AIF can support not just the development of compatible tools and suites of tools with practical utility in the classroom – along with the generation of reusable learning objects within argumentation contexts, but also how the the AIF can allow the development of innovative tools that support completely new means of offering argument-based learning.

2 The Argument Interchange Format

Descriptions of the AIF are given in a number of places, as are reifications in languages such as RDF and OWL (see, e.g. [1]). We provide here just a very brief summary of the main concepts. The AIF uses a graph-theoretic basis for defining an "upper" ontology of the main components (or nodes) of arguments. Nodes are distinguished into those that capture information (loosely, these correspond to propositions), and those that capture relations between items of information, including relations of inference (which correspond to the application of inference rules), relations of conflict (which represent forms of incompatibility between propositions) and relations of preference (which represent value orderings applied to sets of propositions). The instantiated nature of these relations is emphasised in the nomenclature, so whilst information is captured in Information (I-) nodes, relations between them are captured as Rule Application (RA-) nodes, Conflict Application (CA-) nodes and Preference Application (PA-) nodes. The general forms or patterns that these applications instantiate are given in a second part of the AIF ontology, the Forms ontology. The approach follows in the philosophical tradition of Walton [12] of schematizing stereotypical patterns of reasoning and then extending the tradition into conflict and preference. It is this schematic underpinning which gives the collective name for RA-, CA- and PA-nodes: Scheme (S-) nodes. The AIF upper ontology is designed to allow specialization and extension to particular domains and projects, in an attempt to balance the needs of interchange against the needs of idiosyncratic

development. The original AIF specification has also been extended to handle dialogic argumentation. By this extension, it becomes possible to represent both a dialogue and the connection between a dialogue and the structures it generates such as inference corresponding to a RA-node. A dialogue is described by locution (L-) nodes, which refer to utterances communicated during the dialogue and constitute a subclass of I-nodes; and transition application (TA-) nodes, which refer to the passage between locutions and constitute a subclass of RA-nodes. The TA-nodes are governed by the protocol of a dialogue system, recording, e.g., that a given assertion has been made in response to an earlier question. The connection between a dialogue and the structures it generates is captured by means of illocutionary application (YA-) nodes which link together either L-nodes with I-nodes, or TA-nodes with RA-nodes. For example, an YA-node may represent the relation between a speech act claim(α) with its propositional content α.

3 Critical Thinking and Argument Analysis: OVA

OVA (Online Visualisation of Argument)[1] is a tool for analysing and mapping arguments online. It is similar in principle to other argument analysis tools, including Araucaria [5] and Rationale [10], but is different in that it is an online application, accessible from a web browser, facilitating analysis of online resources.

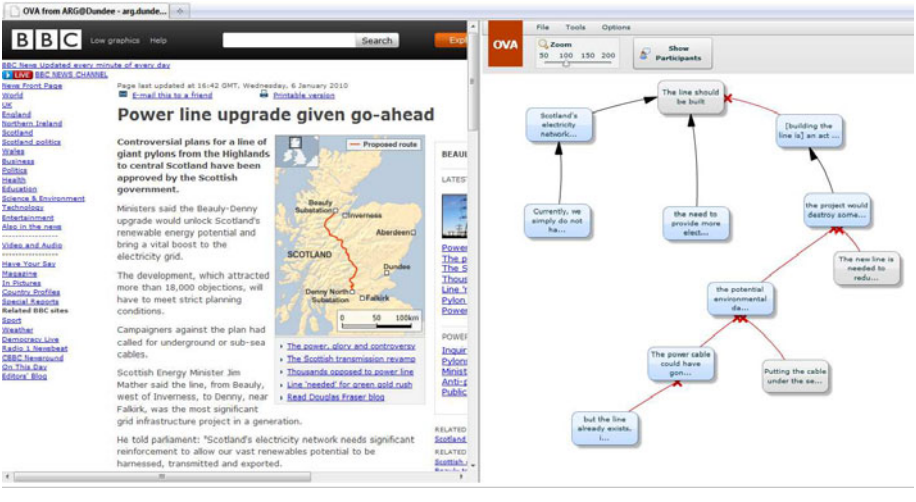


Fig. 1. OVA user interface

A web page or text file is analysed by providing its URL. The page is rendered alongside the main OVA interface, where text can be highlighted and extracted

[1] http://ova.computing.dundee.ac.uk

for analysis [Fig. 1]. The main components of the interface are the analysis canvas (the large, white area on the right-hand side); the web page display (on the left-hand side); and the toolbar at the top (providing tools to manipulate and save the analysis). An analysis is carried out by highlighting text on the web page, then clicking the analysis canvas; this extracts the text into a premise (represented in OVA as a node), which can be used to either support or attack other premises (or indeed, be supported or attacked itself).

OVA supports circular and divergent argumentation, and missing premises (or enthymemes) can also be reconstructed, allowing introduction of information that isn't explicit in the text being analysed. Once an analysis has been carried out, participants can be added. The participants represent the real people who promoted (or uttered) the premises used in the analysis. Finally, the resultant diagram can be exported as a JPEG image or an SVG description. OVA saves its analyses to AIF, either to a local file, or to an AIF repository such as *ArgDB*.[2]

Araucaria [5], which is in some sense a predecessor to OVA, has been downloaded over 10,000 times and is in use in schools and universities in over 60 countries. OVA, released in early 2010, has been trialled in undergraduate courses at Dundee, where it supports a critical thinking class and where early, informal feedback is very positive.

4 Dialogue and Mixed Initiative Argumentation in Pedagogy

Textbooks in critical thinking, of which [11] is typical, focus on the analytical facets of the discipline. That is, students are introduced to techniques that help them to split arguments into their component pieces, identify bias, reconstruct missing premises, identify schematic patterns and ultimately perform evaluative judgements on the quality of the arguments they encounter. Argumentation theory as an academic field has a similar tendency, given its roots in the philosophy of language and epistemology.

The creative, generative aspects are treated more rarely, both in the teaching of the subject and its academic environment more broadly. Inculcating the skills and techniques for producing high quality arguments is, rather implicitly, assumed to follow without further ado from the analytical experience that a student develops. Some other disciplines do occasionally include argument construction in their syllabi – the teaching of rhetoric, though rare, does occur in English programmes in North America and in pure rhetoric programmes worldwide. Vocations such as law and marketing may also introduce some basic techniques for argument construction. But almost without exception, these syllabi cover the creation of written arguments. Pedagogy focusing upon engagement in verbal dialogue is extremely uncommon, and this is surprising for two reasons. First, verbal argument is both very common, and when well executed, highly prized.

[2] *ArgDB* is an online corpus of argumentation, hosted at the University of Dundee and is available at `http://argdb.computing.dundee.ac.uk`

Parliamentary contributors, late night talk show panellists, and figurehead public orators often command significant respect purely in virtue of their rhetorical capabilities. Given its ubiquity and apparent importance, one would expect it to occur very commonly in a wide variety of curricula. There is also a second reason that it is surprising not to find these skills taught more extensively. From antiquity, rhetorical performance has been a central part of a rounded education, right up until the late nineteenth and early twentieth centuries. Cicero and Quintillian both offer treatises that are strongly pedagogical focusing specifically on the ability to create one's own arguments in tandem with analysing and interacting with those of an interlocutor. It is clear from these classical texts that the task is highly demanding (so we should expect to see it respected and prized), so with a strong precedent of teaching a complex and important skill, it is little short of astounding not to see it offered at every university or college. (Of course,verbal argumentation skills do appear in extramural activities quite often: both Europe and North America have strong debating – or 'forensic' – societies aimed primarily at children. But these societies do not involve formal education, and are primarily experientially based).

Technology offers a route to tackling this anomaly, and in particular, recent advances in mixed initiative argumentation offer a very exciting avenue to new pedagogical models.

4.1 Implementing Mixed Initiative Argumentation: Arvina

Arvina is a Google Wave application which builds upon the Google API to offer a rich dialogic interface to argument resources. Arvina's basic dialogue protocol is similar in scope to that offered by Magtalo [6], however using the Wave platform as a base allows a greater interaction between large groups of both virtual and real life participants.

Upon creation of an Arvina wave, a *gadget* is inserted allowing the user to choose a topic from any previously analysed AIF resources. Once selected, the AIF resource is examined to determine the participants involved in the dialogue represented and a new *robot* is added to the wave representing each of these participants. Following topic selection, the user must choose a starting point (an AIF I-node from which the dialogue can progress) and having done so is then given two options: to either ask a question and get the opinion of the artificially represented participants; or to offer their own view by either agreeing or disagreeing with the point being made. Each time a new point is put forward by either a human or a software participant, the wave is updated to show the new point, and to provide controls for interacting with that new point – i.e., to allow the user to challenge it, support it, or ask for views on it from other participants.

Arvina allows for an open mix of both artificially represented participants using knowledge assigned in an AIF resource and live participants. Any real participant may ask questions of any other participant of two forms: "Do you agree with this?" (to which robots will respond with yes or no and supply a supporting reason if one is available in the AIF); or, "Why is that the case?"

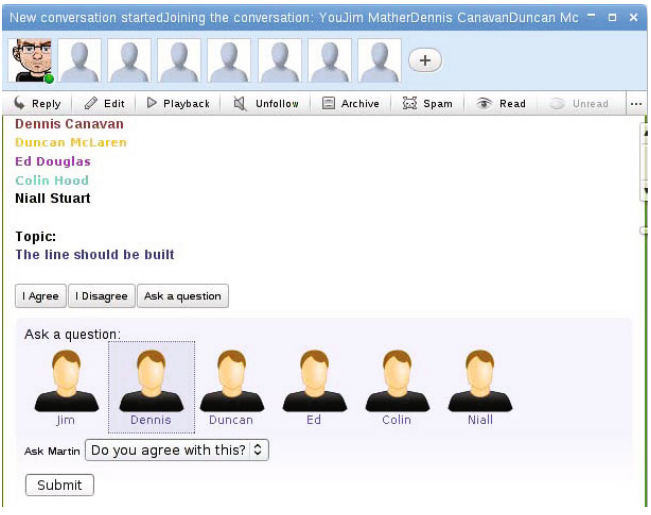


Fig. 2. Asking a question of a virtual participant

(which elicits further supporting reasons) and so uncover, in a natural way, the participants' views [Fig. 2]. This method allows a user to direct the course of the conversation and as such, rather than just being presented with a list of claims, they can instead concentrate on the areas which interest them most. Live participants can also supply their own supporting reasons, allowing the AIF resources to be expanded in a structured way.

This mechanism allows a seamless conversation to take place between live users and those being represented virtually by robots. In this way even a very simple dialogue protocol provides an interface that exploits a naturalistic style of interaction to allow intuitive, user-driven navigation of a complex interconnected web of arguments.

4.2 Implementing Pedagogical Dialogue: Parley

Parley is a prototype networked graphical software tool that supports argumentative interaction between students working in small tutorial groups. By engaging in a dialogue, on a specific topic and according to a carefully defined protocol which governs the kinds of things that can be said at any given point, the students build up a diagram of the dialogue.

The main interface to Parley incorporates a graphical canvas on which the diagram is constructed and a set of tool palettes that provide access to the utilities that manipulate the diagram. The diagram is structured as a tree with a root note representing a central thesis, and responses and responses to responses beneath it. The links record the argumentative relationship between any given node and the node to which it is responding. In the prototype, Parley allows responses to existing statements of two types: support and attack. In this way, instead of diagramming the fine detail of an argument, Parley captures whole

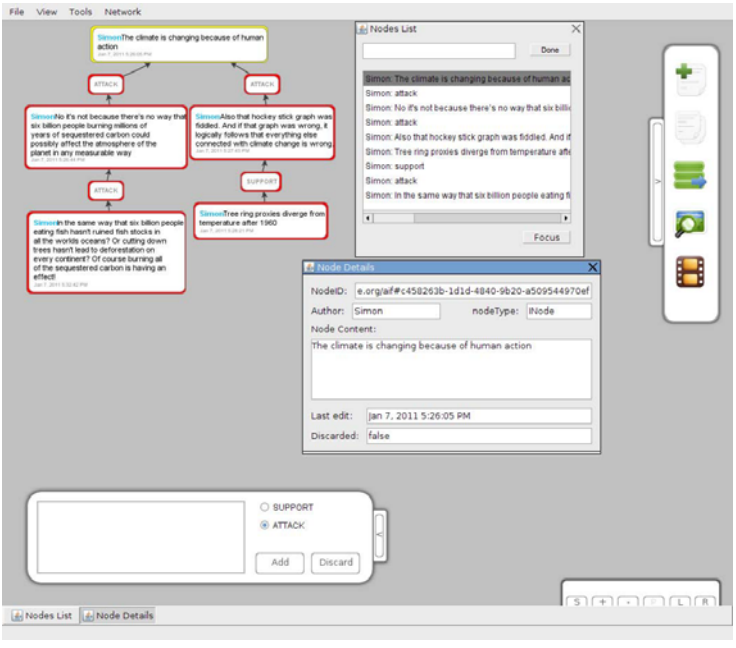


Fig. 3. The Parley dialogue tool

arguments, as they are uttered at the turn level of a dialogue, as individual nodes, showing relationships to other arguments as arrows between the nodes as the students respond to each others' points. By following a series of responses, each branch of the tree becomes a line of discussion within the dialogue, as a student puts forward an argument for or against a given position, which is responded to, and so on, until that line of discussion is exhausted and the students return to an earlier node in the dialogue in order to explore another line of discussion. In this way the dialogue continues until the students run out of things to say. Finally, the node details widget can be used to display information about any individual node within the diagram and identifies the type, author, content, modification time, and status. Currently all nodes have a type which is either an information node, for the content of utterances made by the students, or scheme node, capturing the relationship between given pair of information nodes. These nodes correspond to the equivalent AIF I-nodes and S-nodes respectively and allow Parley dialogues to be exported as AIF documents for reuse in other tools or to allow individual students to maintain their own archive of dialogues.

One of the advantages of the graphical approach used in Parley, as opposed to the text based approach taken in other pedagogical dialogue software like Inter-Loc [4], is that an overview of the whole dialogue can be rapidly gained without having to read the entire transcript and reconstruct the threads of discussion.

5 Concluding Remarks

We have given a brief introduction here to three tools for teaching argumentation skills in the classroom: one that is in the more traditional sphere of close analysis of argument; and two that broaden pedagogy of argument into dialogical systems. The aim has not been to give a detailed description of any of these systems, but rather to demonstrate how they can be used to complement one another in educational settings, and how resources developed for or with one tool can be re-used in very different settings with another. These benefits of compelementarity and re-use arise from the common foundation upon which they are all developed provided by the abstract ontology of the argument interchange format. As more and more tools and datasets are developed that use the AIF, so the potential for educational benefits, both within and between institutions, continues to increase.

Acknowledgements

At Dundee, this work has been supported in part by the Engineering and Physical Sciences Research Council (EPSRC) under EP/G060347/1. Katarzyna Budzynska gratefully acknowledges the support from Polish Ministry of Science and Higher Education under grant N N101 009338.

References

1. Chesñevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S.: Towards an argument interchange format. Knowledge Engineering Review 21(4), 293–316 (2006)
2. Harrell, M.: Using argument diagramming software in the classroom. Teaching Philosophy 28(2), 163–177 (2005)
3. Kirschner, P., Buckingham Shum, S., Carr, C.: Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making. Springer, Heidelberg (2003)
4. Ravenscroft, A.: Promoting thinking and conceptual change with digital dialogue games. Journal of Computer Assisted Learning 23(6), 453–465 (2007)
5. Reed, C., Rowe, G.W.A.: Araucaria: Software for argument analysis, diagramming and representation. International Journal of AI Tools 14(3-4), 961–980 (2004)
6. Reed, C., Wells, S.: Dialogical argument as an interface to complex debates. IEEE Intelligent Systems 22(6), 60–65 (2007)
7. Rolf, B., Magnusson, C.: Developing the art of argumentation: A software approach. In: van Eemeren, F.H., Blair, J.A., Willard, C.A., Snoeck Henkemans, A.F. (eds.) Proceedings of ISSA-2002, pp. 919–926. SicSat (2002)
8. Suthers, D., Weiner, A., Connelly, J., Paolucci, M.: Belvedere: Engaging students in critical discussion of science and public policy issues. In: Proc. of the 7th World Conference on AI in Education, pp. 266–273. AACE (1995)
9. Twardy, C.: Argument maps improve critical thinking. Teaching Philosophy 27, 95–116 (2004)
10. van Gelder, T.: The rationale for Rationale. Law, Prob. & Risk 6(1-4), 23–42 (2007)
11. Walton, D.: Fundamentals of Critical Argumentation. CUP, New York (2006)
12. Walton, D., Reed, C., Macagno, F.: Argumentation Schemes. CUP, New York (2008)

Visual Tools for Teaching Propositional Logic

Aránzazu San Ginés

Departamento de Filosofía 1
University of Granada, Spain
aransangines@gmail.com

Abstract. In this paper, I will outline a diagram-based proposal for teaching propositional logic, as well as the reasons that led me to it. The paper is divided into three sections. In the first section I introduce, and try to justify, the hypothesis that processes like thinking, reasoning or speaking are intimately connected with the process of *constructing* what we see. The second section presents a criticism of the didactic ideas underlying the trend of *modern mathematics* in countries like France and the U.S.A. The final section is devoted to the schematic presentation of the specific diagram-based approach.

Keywords: Propositional logic, didactics, abstract thought.

1 The Hypothesis

The language that we usually use in mathematics is riddled with metaphors. In this way, bridges are built from the realm of the abstraction to the realm of the physical. We say, for instance, that a *sequence escapes us* when it diverges, or that a function with a certain type of discontinuity at a point *jumps at this point*. As a result, we succeed in making an essentially abstruse domain, that of mathematics, more accessible. This phenomenon is not exclusive to mathematics; on the contrary, it seems to appear in all types of contexts, scientific and otherwise[1]. And, of course, logic is not an exception. Expressions such as *the argument does not hold up* or *the conclusion follows from the premises* derive from the close link between the users of logic and their physical environment.

The use of this kind of expressions in domains so apparently away from the physical world, just as logic and mathematics seem to be, raises a question about the possible connections between the abstract and the spatial cognitions. Is the use of 'spatial schemas' [2] a mere ruse of logicians or mathematicians to relate cognitive processes that are essentially different, and so to simplify the specific task at hand, or is there something else to say about them? The answer that will be proposed on this paper is that there is certainly a big deal to say. My position intends to be in keeping with the approach of evolutionary theorists; an approach that can be found, to some degree, in the following text [3] (p.371):

> "A fundamental puzzle in the study of the mind is how evolution could have produced a brain capable of intricate specialized achievements like

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 215–222, 2011.

> mathematics, science, and art given the total absence of selection pressure for such abstract abilities at any point in history. [...] There are precedents for explaining the emergence of novel capabilities in evolution: old parts can be recruited to new uses."

I therefore start from the hypothesis that the extraordinary human ability for generating abstract thought is a product of the adaptation of those parts of the brain which are responsible for perception and manipulation of space. I will actually go further on this line, and give the mechanisms involved in the construction of the visual a prominent role in the construction of the rational. Hoffman [4] (p.1) describes the human as a "visual virtuoso", a "creative genius for vision". This innate talent has achieved such a degree of sophistication in the course of our evolutionary history, that it seems natural to regard it as the expected support of novel cognitive processes.

However, according to Pylyshyn [5] the mental images reported by the great majority of scientists are no more than the epiphenomenal recreation during reasoning of the (symbolic) laws which seem to govern the world. The laws, rather than the images, would then be part of reasoning. Pylyshyn is especially against the pictorial view of mental images, a view that seems to fly over the Shepard's proposal. In his 1978 paper, Shepard [6] equates in importance logical and analogical processes of thought, where analogical process is understood as "a process in which the intermediate internal states have a natural one-to-one correspondence to appropriate intermediate states in the external world" (p.135). From this claim, it could be concluded that the relationship between the processes of perception and reasoning is limited to the manipulation, during reasoning, of mental objects that are analogous to those that are the result of perception. But, how to explain then the way in which Mozart, for instance, said he imagined his compositions during the creative process? See, according to Pylyshyn [5], p.32, the Mozart's letter reproduced in Ghiselin (1952): "Nor do I hear in my imagination, the parts *successively*, but I hear them, as it were, all at once." If the one-to-one correspondence mentioned by Shepard were exact, then it seems that Mozart should have heard the complete composition in his imagination note by note. Yet he did not. He did not imagine each part, each note, one after the other. Mozart said he imagined each of his compositions all at once, in a process which, to my judgement, resembles the way people *construct* the following picture [4]:

Our 'visual intelligence', as Hoffman calls it, constructs from the four black independent figures above a square which was actually never explicitly drawn (a *subjective surface*). In fact, although we are absolutely sure about what we see, if someone asked us about the sides of the square, curiously enough, it would not be easy to give a precise answer, even though having sides is an essential characteristic of being a square. However, at the same time, we would be perfectly

able to outline four appropriate sides to the figure. We have all the information we need to do that, in the same way Mozart counted on the necessary elements to transcribe the composition that he had imagined *all at once*. Both processes seem to have something in common. And it is precisely this similarity what my proposal aims to stress. In this sense, my hypothesis is not only that people reason on the basis of images but, further, that the very processes of thinking, reasoning or speaking are intimately connected with the process of *constructing* what we see.

Now, if we accept that the previous hypothesis is plausible, it should not appear so strange the possibility of producing discovery through the visual.

Giaquinto[7], for instance, argues for the possibility of making discoveries in geometry by visual means. He claims that the experience of visualisation brings about the recovery of certain items that are present in the individual's cognitive state. These items would constitute the pieces in a puzzle which, when fitted together, would enable one to succeed in making discoveries reliably.

My hypothesis differs from Giaquinto's proposal in that, according to my view, the visual contribution is not limited to the activation of items but also concerns the strategy which will eventually enable them to be combined as new beliefs. We can thus see a certain parallel between the mechanisms involved in, respectively, discovery and perceptual construction. Although the two models appear to have different objectives and to function without any apparent connection, they apply very similar rules. In the case of reasoning, these rules could be described as deriving from those governing the construction of perception.

2 A Didactic Plan

If logic has a lot to do with reasoning well, and reasoning is connected to 'vision' as much as I argued for in the previous section, it thus appears natural and even advisable the use, while teaching logic, of didactic resources that involve and exercise actively the visual abilities of the students. This is so because, by making use of those resources, we will ease the process of learning logic, and strengthen the arguing and logical skills of the students.

Making deductions is probably one of the most difficult things to be learnt by the students of logic. They have to learn to deal, with dexterity and harmony, with the rules of inference of specific systems; rules that sometimes do not make any sense to them. Making deductions is certainly not an easy task. Discovering the formulas that can be proved from a sound set of premises is even harder.

Let us suppose that the hypothesis defended along the first section were correct. Then, the use of appropriate diagrams should simplify considerably both deductive and discovery tasks, and at the same time provide adapted training to the kind of mental processes involved. The use of graphic systems would allow the students to strengthen abilities which, in spite of having been proved to be substantial help to the significant progress of science, have been systematically refused and consigned to oblivion for nearly two centuries. Which could be the reason of this contempt for the use of diagrams in science? Let us consider the question for a moment.

According to Mancosu [8] (p.15) "[...] one of the main paradigmatic examples that were used to discredit the role of geometric intuition in analysis [was] Weierstrass' discovery of a continuous nowhere differentiable function." This result seemed to demonstrate that geometrical intuition is deceptive and, therefore, that the most reliable role that can be assigned to diagrams is that of helping in the construction of reasoning, but never that of playing a decisive role in arguments. This is why scientists in general, and mathematicians in particular, decided to relegate the use of diagrams to a merely heuristic role in the 19th and 20th centuries. And this is one of the reasons didactic perspectives that largely reject the informal background of pupils and students have been put into practice for instance in mathematics. In fact, according to Dehaene [9] (p.139), this informal background has been considered "in most math courses [...] as a handicap rather than an asset". The objective pursued was to teach *modern mathematics* to the pupils, or otherwise, to familiarize them, from the very beginning, with a way of doing mathematics consisting in the manipulation of abstract symbols from a solid axiomatic basis. But to achieve this objective, it was necessary that the students made a new fresh start; that they forgot all intuition acquired out of the premises of their school, which is now set up as the great temple of the abstract, formal, reliable and correct knowledge. Few of them would be then the chosen ones that could appreciate the beauty and truth so zealously hidden by mathematics. Therefore, in this sense, the *modern mathematics* proposes a learning plan that, on the one hand scolds pupils for finger counting, and on the other, tries to familiarize them "with the general theoretical principles of numeration before being taught the specifics of our base-10 system." [9] (p.140). Thus, according to Dehaene "believe it or not, some arithmetic textbooks started off by explaining that 3+4 is 2 - in base 5! It is hard to think of a better way to befuddle children's thinking."

Dehaene has not been the only one to react against these modern techniques of teaching mathematics. Other specialists in the last 50 years (for example, Kline [10]) have also considered those to be more a potential risk on the development of the creative abilities of the students than an efficient way to dynamize their argumentative and symbolic skills. In *modern mathematics*, the teacher has the aim to teach the students to see mathematics in the way professionals do. As a consequence, they neglect the fundamental fact that mathematics such as we know them at present have been the result of a long process in which the intuitions today despised played a very important role. But above all, *modern mathematics* appears to forget that human beings have developed these extraordinary abilities which characterize them (thinking logically and mathematically) in the particular physical environment in which they exist. Each new advance should be seen as a valuable ally of our natural capacities, not as a replacement for them. To my mind, only when the syllabi reflect in some degree this desideratum, we will observe some considerable improvement in the education of our students. And maybe then it will be also possible to reduce the feeling of failure and demoralization that tends to overwhelm many of them.

3 A Diagram-Based Proposal for Teaching Propositional Logic

Mine is not at all the only proposal that there has been up to date in relation to the use of diagrams in logic. Hammer [11] (p.129), for example, tells us the following about C.S. Peirce: "From his experience with chemistry and other parts of science, Peirce had become convinced that logic needed a more visually perspicuous notation [...]". Peirce would develop three types of diagrammatic systems associated to propositional logic, logic of predicates, and modal logic, respectively. And Peirce would not be the only one. *Hyperproof* is a more recent example of that. This is a computer program created by J. Barwise and J. Etchemendy for teaching logic in the context of the project *Openproof* at the University of Stanford.

Although it is not the only one, I expect my diagram-based proposal for teaching propositional logic to be the closest to the didactic interest expressed before. With the following proposal I will try to involve, in the deductive and creative process, other brain processes (related to vision) that I have defended before to be specially linked to abstract thought. Let us pass now to briefly describe the system.

The formulas and the rules for derivations for propositional logic will be represented in terms of colored matrices in the following way.

3.1 Formulas

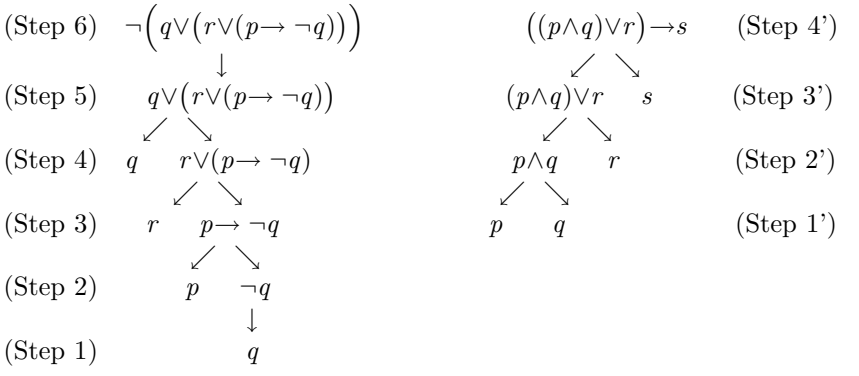
Let p and q be atomic formulas, and let M be a matrix, that is, a square divided into the same number of rows and columns.

- p is represented in M by coloring one entry of the matrix. The color could be any but once is chosen it will identify the atomic formula.
- $\neg p$ is represented in M by coloring one entry with the color used for p and crossing it out: $\not{p}$[1]
- $p \rightarrow q$ is represented in M by alternating the colors of p and q in one chosen entry of the matrix. The movement from p to q is represented as being quicker than the movement from q to p.
- $p \leftrightarrow q$ is represented in M by alternating the colors of p and q in one chosen entry of the matrix. The movement from p to q is represented as quick as the movement from q to p.
- $p \wedge q$ is represented in M by representing p and q in different entries of the matrix. The chosen entries are represented close to each other (contiguous entries).
- $p \vee q$ is represented in M using just one entry of the matrix. The entry will be split in two by the diagonal. Each one of the arguments of the disjunction (p and q in this case) will take up a part of the split entry.

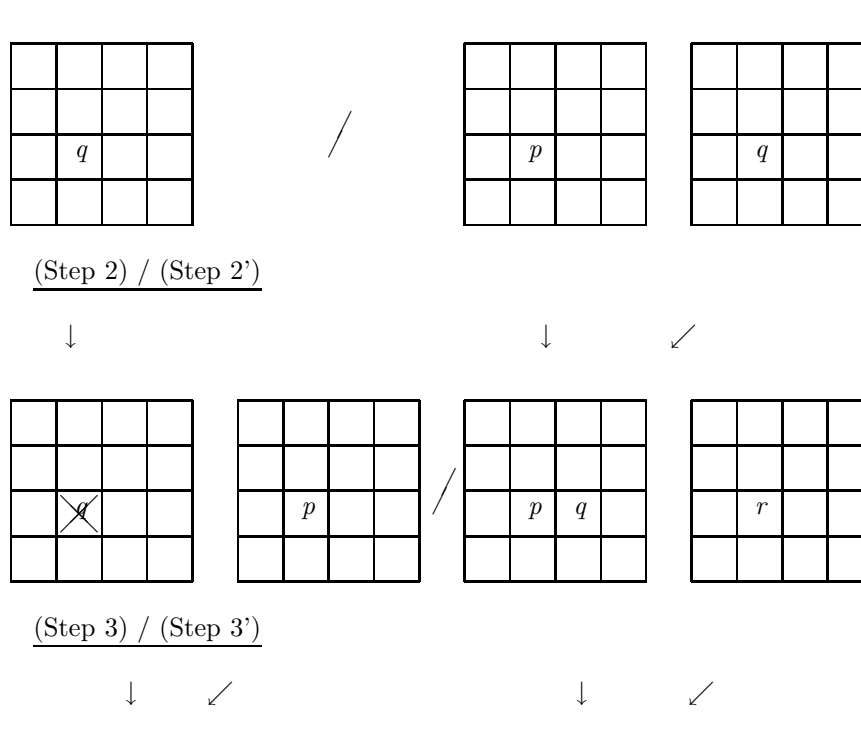
[1] I will not use colors in the examples here but the propositional letters itself. We shall understand any propositional letter in the matrix as if the entry was colored.

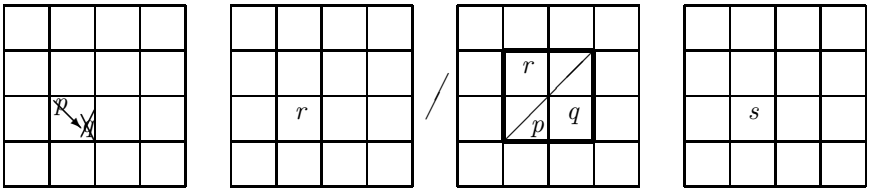
But what if the arguments were not atomic formulas? For example, in the case that we had formulas such as $\neg\Big(q\vee\big(r\vee\ (p\rightarrow\neg q)\big)\Big)$ or $\big((p\wedge q)\vee r\big)\rightarrow s$?

The first thing to do will be then to draw up the genealogical tree of the specific formulas:



Secondly, the steps will be represented from down up:

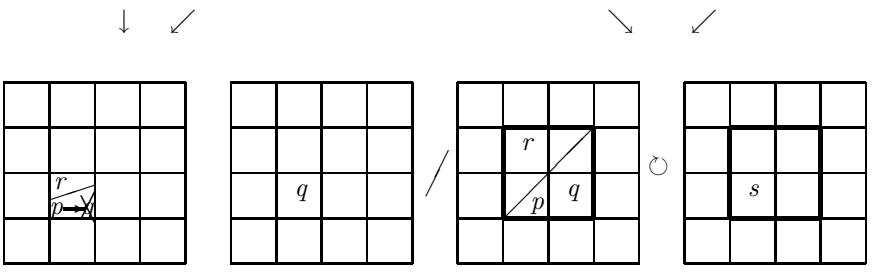




I shall make explicit here the ∨-rule applied in (Step 3'):

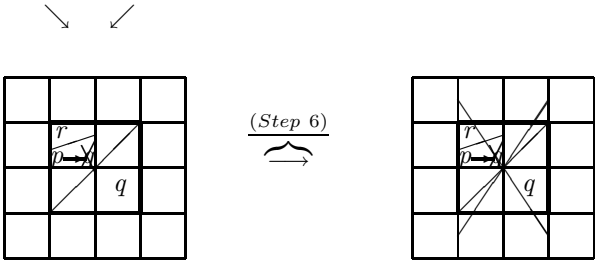
- If each one of the arguments of the disjunction were placed previously (Step 2') in just one (not split) entry of the matrix, then the disjunction will be represented in one single entry. This is not the present case (Step 3'), but will be case in (Step 4): $r \vee (p \rightarrow \neg q)$
- Otherwise, it will be taken the smallest submatrix in which, once it has been split by the diagonal, it will be able to represent each one of the arguments of the disjunction in different parts of the split submatrix. This is the rule we are applying in (Step 3'). The submatrix appears above emphasized.

(Step 4) / (Step 4')



The submatrix in which is represented $(p \wedge q) \vee r$ in (Step 3') will blink in (Step 4') indefinitely from $(p \wedge q) \vee r$ to s.

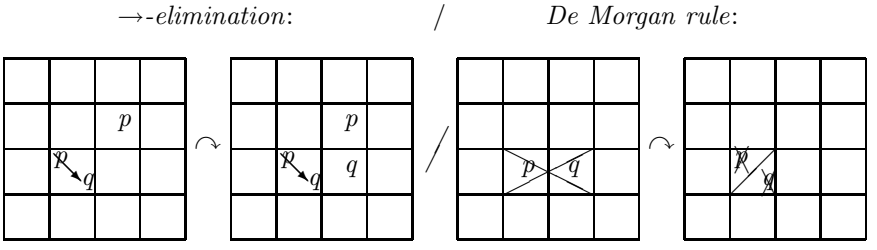
(Step 5)



I do not represent the disjunction in (Step 5) in one single entry because one of the arguments of the disjunction $(r \vee (p \rightarrow \neg q))$ was represented, in the previous step (Step 4), in one split entry. As for the negation, (Step 6), it works by crossing the representation of its entire argument out.

3.2 Rules for Derivations

Imagine we wanted to make a deduction in propositional logic by making use of the matrix representation. The first thing to do would be then to represent the premises in a single matrix M. Following that, we would transform M until getting represented in it the conclusion we are aiming for. Thus the question is: what are the rules which allow us to transform a matrix into another? Here is, to finish, a sample of these rules:



Acknowledgements

This work has been partly supported by the research group HUM 432 of the Plan Andaluz de Investigación, Desarrollo e Innovación (PAIDI) and by the project FFI2010-15704 of the Ministerio Español de Ciencia e Innovación, both of which are gratefully acknowledged. I also thank María José Frápolli and the referees.

References

1. Lakoff, G., Johnson, M.: Metaphors we live by. University of Chicago Press, Chicago (1980)
2. Gattis, M. (ed.): Spatial Schemas and Abstract Thought. The MIT Press, Cambridge (2001)
3. Pinker, S.: Learnability and cognition: the acquisition of argument structure. The MIT Press, Cambridge (1989)
4. Hoffman, D.D.: Visual intelligence: how we create what we see. W.W. Norton & Company, New York (2000)
5. Pylyshyn, Z.W.: The Imagery debate: Analogue Media Versus Tacit Knowledge. Psychological Review 88(1), 16–45 (1981)
6. Shepard, R.N.: The Mental Image. American Psychologist, 125–137 (February 1978)
7. Giaquinto, M.: Visual Thinking in Mathematics. An Epistemological Study. Oxford University Press, Oxford (2007)
8. Mancosu, P., Jorgensen, K.F. (eds.): Visualization, explanation and reasoning styles in mathematics. Springer, Heidelberg (2005)
9. Dehaene, S.: The number sense. How The Mind Creates Mathematics. Oxford University Press, Oxford (1997)
10. Kline, M.: El fracaso de la matemática moderna. Siglo XXI de España editores (1984)
11. Hammer, E.: Peircean Graphs for propositional Logic. In: Barwise, J., Allwein, G. (eds.) Logical Reasoning with Diagrams. Oxford University Press, Oxford (1996)

Logicamente: A Virtual Learning Environment for Logic Based on Learning Objects

Patrick Terrematte, Fabrício Costa, and João Marcos

Federal University of Rio Grande do Norte (UFRN)
Department of Informatics and Applied Mathematics (DIMAp)
Group for Logic, Language, Information, Theory and Applications (LoLITA)
Natal – RN – Brazil
terrematte@ppgsc.ufrn.br, fabriciocscte@ppgsc.ufrn.br,
jmarcos@dimap.ufrn.br

Abstract. Logic is a subject connected to several fields of study, by which it is possible to improve the understanding of information and the reasoning process in many domains. In most courses, it is remarkable how Logic represents a pedagogical challenge for both tutors and pupils, and the recorded number of cases of failures and of discontinuity is often high. One of the reasons for this situation is the gap between, on the one hand, the repetitive aspects of exercises for learning and, on the other hand, the inventive activities of researching or applying Logic to practical situations. Given the need to provide a solid basis for the subject at undergraduate level, and also to focus on inductive learning with creative skills, we propose the project LOGICAMENTE[1], a Virtual Learning Environment (VLE) for Logic composed of a growing collection of Learning Objects combined with the respective learning scripts, expositions, tasks and activities on subjects of Logic. The VLE illustrates fundamental concepts and algorithms from Logic, as well as allows students to conduct interactive experiments involving the understanding of various logical concepts belonging to topics ranging from Theorem Proving to Formal Semantics.

Keywords: Learning Objects, Teaching Logic, Logic.

1 Introduction

"Contrariwise," continued Tweedledee, *"if it was so, it might be; and if it were so, it would be; but as it isn't, it ain't. That's logic."*
— LEWIS CARROLL
Through the Looking-Glass (1871).

Tutors of undergraduate Logic classes are often faced with problems directly linked to causes of failures and discontinuity of their pupils, such as:

– *The laborious application of theoretical subjects*: little visualization is provided for logical concepts, such as the construction of proofs, the recursive properties of language, and the construction of models.

[1] Available at http://www.lolita.dimap.ufrn.br/logicamente

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 223–230, 2011.

– *The difficulty of a creative approach for logical skills*: some very significant aspects in Logic, such as proof search, the use of lemmas, the creation of definitions, and the search for counter-models, are unfairly treated as a repetitive matter.

To address such issues, many tools for teaching Logic are currently available, such as those presented in the comprehensive list compiled by the Association for Symbolic Logic [4]. Many of them are inspiring and widely used, e.g. *AproS Project*, *JAPE*, *ProofWeb*, *Tarski's World*, *Fitch* and *Boole*, among others. Such systems are appropriate to deal with the topics they are aimed at, but they cover, in isolation, very restricted areas of Logic, as only one or two topics are explored by each system. The LOGICAMENTE intends to be a multi-subject Virtual Learning Environment (VLE) for the teaching and learning of the various facets of Logic. The basic idea of the project is to conceive an extensible framework to teach Logic without the following very common problems of e-learning tools:

– *Technological constraints:* Some tools are desktop systems, so they are not available at *'anytime and anywhere'*.
– *Usability of e-learning:* The tools don't have usability patterns to stimulate different learning styles of the users.
– *Integrated learning environment:* They do not offer the possibility of managing the feedback of students, their activities and exercises.
– *Collaborative development:* They do not offer any warranty of continuous development. Big and complex projects, however, must offer collaborative strategies.

A VLE is a computer system designed to support teaching and learning with an educational monitoring mechanism. A VLE should provide a set of tools, such as assessment tools, interactive communication, transfer of content management, student activities, tools, tracking exercises, forums, wikis, and learning objects specified for selected subjects. For the implementation of those features that compose a VLE, the LOGICAMENTE is to employ a module integration with MOODLE, a well-known Learning Management System (LMS) that helps in organizing contents and educational activities. The advantages of using this platform is the possibility of reuse of specified Learning Objects and the management of activities, aiming at a constructive process for teaching and learning Logic.

In the paradigm of object-oriented computing, the compounds are developed to be reused and to solve the same recurring problem as applied to different contexts. An analogy may be employed to define the Learning Objects (LO) as units which provide an epistemological content to stimulate the reflection of a student about a given subject: an LO is any digital entity with a defined educational purpose that can be used, reused or referenced during a learning process [10,9].

The LOGICAMENTE implements the concept of LOs to customize contents and methodologies of Logic courses according to their focus. It also offers learning scripts with expositions, tasks and activities on subjects of Logic, in order to

describe the epistemic content of each LO, guide the teaching and learning of Logic, what is intended to be learned, which skills should be developed, and the use of each LO with its purposes and values. The LOGICAMENTE represents an integrated learning environment which associates a context of activities (tasks or exercises) and a specific content of Logic to a Learning Object.

The LOGICAMENTE is being developed based on three senses of collaboration:

- *Collaborative learning*: We aspire to support the development of LOs that may be operated and reused by different VLE platforms and Learning Management Systems. The LOs are implemented by undergraduate students learning concepts of logic and applying them algorithmically.
- *Collaborative development*: We strive to apply a set of strategies with a development process based on the *open source* methodology, documenting and creating a *Web Service* interface to support the interaction for other tools and projects collaborating with the LOGICAMENTE .
- *Collaborative integration*: We design the LOGICAMENTE to be integrated with other systems and projects. For instance, we develop an LO —the *Theorem Proving Web System*— integrated with the system *ProofWeb*[2].

2 The Virtual Learning Environment: Logicamente

C'est par la logique qu'on démontre,
c'est par l'intuition qu'on invente...
— JULES HENRI POINCARÉ
In Science et méthode (1908).

The first prototype of the LOGICAMENTE was assembled in 2006 as a collection of final assigments for an undergraduate course on Logic Applied to Computing. The goal was to apply a methodology of Problem-Based Learning (PBL) to the implementation of some key concepts of Logic. In the following semesters, the project continued to be developed in collaborative learning through the work of students and volunteers, who had the opportunity to do research that explored the computational aspects of the algorithmic implementations as well as aspects of computer technology applied to education [13,2,12]. Currently, the project involves master's students and undergraduate students in its development. The prototype of the home screen can be seen in Figure 1.

The LOGICAMENTE is developed aiming at exploring the advantages of Web applications, implementing LOs to perform tasks such as: automatic generation of formulas with the desired complexity; configuration of a language and the definition of new connectives; translation between different syntaxes; construction of truth-tables; interactive presentation of formulas in the form of trees; implementation of the resolution method for Classical Logic; search for (counter-)models; use of a proof assistant for the practice with writing derivations in a formal deductive system.

[2] The *ProofWeb* is an *open source* software for teaching Natural Deduction which provides interaction between some proof assistants (*Coq, Isabelle, Lego*) and a Web interface [6]: Check `http://proofweb.cs.ru.nl/`

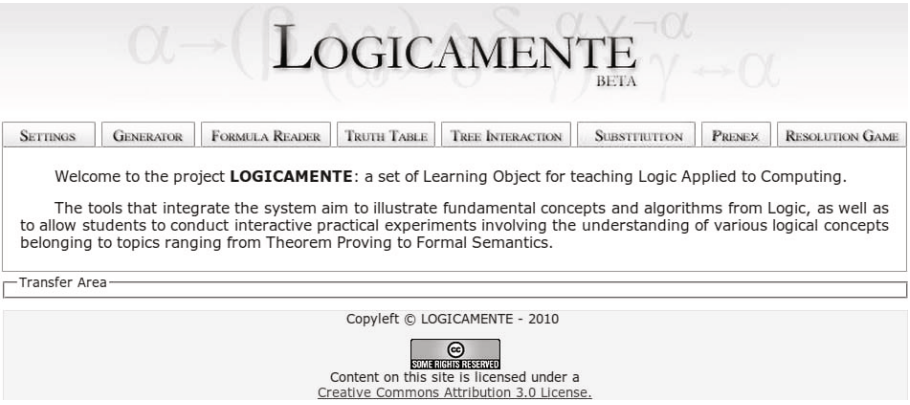


Fig. 1. The current version of the LOGICAMENTE

2.1 Logicamente on e-learning: Learning Objects and Modules

Nowadays, with the advancement of the Web, knowledge is related to the dynamic way in which we have access to it, searching it on-line or collaboratively constructing it in social networks. It is necessary to rethink the way by which we teach and adapt to this new dynamics on e-learning. It is still difficult, nonetheless, to devise a fair assessment method based exclusively on the e-learning paradigm. It is essential thus to identify and work with both the main current educational paradigms, *face-to-face* and *e-learning*. In agreement with Antonia Huertas [7], we believe that teaching activities for logic must be accomplished through both paradigms. Our proposal is to assign practical activities on the e-learning modality and to assign theoretical activities on the face-to-face paradigm. The e-learning is meant to explore the practical abilities of students and even help to emphasize the interactive and creative skills. However, the success of the e-learning modality does not depend only on techniques applied by most VLEs. To take the feedback from the students into account and maintain a high level of motivation for them, the main methodologies that we apply to the project LOGICAMENTE are instrumental, constructionist, conceptual modeling, personalization of the learning process and continuous evaluation process [1].

With the LOGICAMENTE , the tutor handles students, classes, lessons and exercises. Besides, the tutor also has access to activity reports of performances of students, and there is an agent for monitoring the learning modules and students' activities in interacting with the LOs. In turn, the student accesses exercises related to pre-defined learning scripts and is evaluated by his performance with personalized tasks.

The LOs that have been implemented aim at clarifying the relationships between different approaches to Logic, such as Formal Semantics and Theorem Proving. These different subjects have their contents associated to activities and challenges offered in MOODLE, aiming at a constructive process of learning Logic.

The LOGICAMENTE has two internal modules to support the translation between syntaxes and the interoperability between other learning tools and other programming languages used in the development. The following LOs are very basic and aim at helping both in the production of practical activities for the students and in the collaborative development of our tool:

• *Formula Translator*: This module takes care of the syntax for formulas and can represent their constructors using different symbols, images, HTML codes, Unicode or LaTeX. Also with this module it is possible to uniformly translate between formulas of different applications and to provide an interface of Web-services for the flexible interaction between distinct logic tools.

• *WFF Generator*: This module is responsible for generating well-formed formulas according to certain specifications (occurrences of connectives, sentential variables, and others). With this module, sets of formulas are generated according to the desired complexity and they may be used by other modules in producing personalized exercises.

2.2 The *(Non)Sequitur* Super-Module

One of the important contributions of LOGICAMENTE to produce challenging activities for stimulating the learning of Logic is the super-module *(Non)Sequitur*, where several basic LOs are combined in order to implement a learning script aimed at cultivating an important skill that will be described in the following.

In learning Logic one of the great difficulties lies in the distinctions and the between Proof Theory and Model Theory. The enlightenment in these topics is essential to the understanding of the metatheoretical results of soundness and completeness. In verifying whether an arbitrary argument represents a valid deduction in Classical Logic, most students are attempted to try proving the affirmative answer, instead of finding counter-models to the given assertion. But this is unrealistic, because in an arbitrary such a situation a falsifiable argument, with counter-models, is more likely to be detectable than a verifiably correct argument.

In order to improve the conceptual basis on Classical Logic we offer the students a module composed of five LOs —*Resolution Game*, *Theorem Proving Web System*, *Small Counter-Model Builder*, *Interactive Model Builder* and *Semantic Consequence Tester*— where they may choose between proving or finding a counter-model for an argument or a formula, automatically generated by the *WFF Generator*. The following LOs are responsible for constructing proofs:

• *Resolution Game*: Implements the resolution method, assisting the student in eliminating complementary literalsThe resolution method is used for deciding whether a sentence in conjunctive normal form is a theorem. In our interactive version of this method the student may choose which step she wishes to take in converting a given arbitrary formula to the adequate normal form, as well as in applying resolution at her discretion to the resulting collection of clauses.

• *Theorem Proving Web System*: implements a proof assistant in Natural Deduction. This module is developed by integrating the *ProofWeb* [6] to our project.

To analyze an argument semantically, the student can interactively search for counter-models or justify the correction of her own models, using the following LOs:

• *Interactive Model Builder*: Implements a counter-model searcher for propositional arguments, allowing for valuations to be proposed and stepwise justified. The main goal is to allow the user to freely choose her heuristic for argument analysis.

• *Small Counter-Model Builder*: Combines a bounded model checker with a search engine for finite counter-models of *first-order arguments*, receiving as input a limited number of objects, a collection of assumptions, and a goal formula as conclusion. This module partially simulates the functionality of the tool *Tarski's World*, in which the evaluation of formulas is made with a world's state composed by a finite stock of geometric forms, distributed over a finite board.

• *Semantic Consequence Tester*: Here the student may insert sets of sentences and verify step by step a classical entailment relation involving these formulas. The module may help in generating or justifying a counter-model whenever this is the case.

The aim of the super-module *(Non)Sequitur* is to provide a collection of different tools for evaluating arguments. Through the LOs *Resolution Game* and *Theorem Proving Web System*, students might attempt to justify validity using natural deduction arguments, on both a propositional and a first-order level. For falsifiable arguments, the student should choose to evaluate the arguments through the LOs *Interactive Model Builder* and the *Semantic Consequence Tester*, or using *Small Counter-Model Builder* to seek appropriate (counter-)models for given first-order arguments.

The main methodology used in these LOs is to promote the exploration of different strategies. With the *(Non)Sequitur* module, we offer the possibility for the user to try her hand with both the natural deduction rules and the semantic interpretations, and learn from her own choices and mistakes, being free to choose between proving or finding a counter-model. The methodology of learning from mistakes organizes the information in an inductive model of learning, where there are facts and observations given for inferring principles, as opposed to a deductive model of learning, where applications are consequences of given principles. The inductive learners "do not feel comfortable with the 'Trust me — this stuff will be useful to you some day' approach" [5, p. 6], and they play a major role among computer science students, faced with curricula organized along deductive lines.

2.3 The Modeling and Implementation of Logicamente

In many software projects, the main failures are caused by problems related to usability and scope of system, for instance the lack of user involvement with the software and the fact that the requirements implemented are incomplete and unsatisfied. For instance, to ensure that the users will be satisfied, the

(Non)Sequitur module has an interface designed and focused on the user with an inductive learning style in opposition to a deductive and procedural learning.

The LOGICAMENTE is implemented as a Web application programmed in PHP with AJAX technology in order for the system to be available for use anywhere and without requiring a previous local installation. Aiming at the reuse of LOs, the LOGICAMENTE uses the standard IMS Learning Design (IMS-LD) to describe the LOs with their specified content, and to maintain its resources, activities, tasks or exercises [8,3]. The IMS-LD is compatible with MOODLE and is a framework used for designing courses, supporting activities and organizing VLEs.

3 Future Works and Final Remarks

"*La puissance de vision qui fait le poéte,*
et la puissance de déduction qui fait le savant,..."
— HONORÉ DE BALZAC
La Recherche de l'Absolu (1834).

Many computational tools are currently available for the teaching of Logic. Those applications, however, are not organized into Virtual Learning Environments in an infrastructure that is really integrated and customized with Learning Objects. In the present note, we explained how the project LOGICAMENTE was conceived to fit that role, through the collaborative work of students and with a the problem-based approach to the learning of concepts of Logic. We have described some of the main implemented Learning Objects and finally we have also briefly explained our methodological approach to e-learning.

Particularly in the case of tools for teaching Logic, an inspiring example of collaboration is found with the *AproS Project*[3], a project that has been refined for roughly twenty years and developed with the help of colleagues and students [11]. The *AproS Project* implements a proof display and has a Web-based course *Logic & Proofs* for natural deduction proofs associated with the Open Learning Initiative (OLI) and an interactive learning environment. The differences between LOGICAMENTE and *AproS* lie on our initial challenging implementation by students, our focus being purely for the Web, the collaborative development, the integration with MOODLE, and our approach based on super-modules that implement learning scripts based on specific pedagogical proposals. As we have shown, this is the case of the *(Non)Sequitur*, for instance, a super-module designed to take several distinct concepts from Logic into account and provide a creative approach to their contrast and their practice.

A common problem with software development is the lack of collaborators and stakeholders to continue developing and maintaining the systems. To address such difficulties, the LOGICAMENTE is being built incrementally so that new features may be added gradually. The next challenge is to keep the development active in a collaborative way, ensuring the continuous improvement and growth of LOGICAMENTE . With the *open source* methodology, we aim to facilitate the integration with the work of collaborators from elsewhere.

[3] Check `http://automath.org/`

Acknowledgment. The authors have been partially funded by CNPq and CAPES. Their gratitude is also expressed to all undergraduate students of Computer Science and Computer Engineering who have contributed to the project LOGICAMENTE during several semesters of the course of Logic Applied to Computing at DIMAp/UFRN.

References

1. Bannan-Ritland, B., Dabbagh, N., Murphy, K.: Learning Object systems as constructivist learning environments: related assumptions, theories, and applications. In: Wiley, D.A. (ed.) The Instructional Use of Learning Objects, pp. 61–98. AECT, Bloomington (2002)
2. Barros, T.M., Araújo, A.E.F.D., Marcos, J.: A implementação colaborativa de uma suíte de ferramentas on-line de apoio ao ensino de Lógica. VIII ERMAC–R3 (2008)
3. Beauvoir, P., Griffiths, D., Sharples, P.: Learning Design Authoring Tools in the TENCompetence Project. In: Koper, R. (ed.) Learning Network Services for Professional Development, pp. 379–387. Springer, Heidelberg (2009)
4. van Ditmarsch, H.: A comprehensive list of tools for doing Logic. Association for Symbolic Logic (2010), http://www.ucalgary.ca/aslcle/logic-courseware/ (verified at March 21, 2011)
5. Felder, R.M., Silverman, L.K.: Learning and teaching styles in Engineering education. Engr. Education 78(7), 674–681 (1988)
6. Hendriks, M., Kaliszyk, C., Van Raamsdonk, F., Wiedijk, F.: Teaching logic using a state-of-the-art proof-assistant. Acta Didactica Napocensia 3(2), 35–48 (2010)
7. Huertas, A.: Teaching and learning Logic in a Virtual Learning Environment. Logic Journal of IGPL 15(4), 321–331 (2007)
8. Koper, R., Miao, Y.: Using the IMS LD Standard to Describe Learning Designs. In: Handbook of Research on Learning Design and Learning Objects: Issues, Applications and Technologies, pp. 41–86. Information Science Publishing (2009)
9. Polsani, P.: Use and abuse of reusable Learning Objects. Journal of Digital information 3(4) (2006)
10. Ritzhaupt, A.D.: Learning Object Systems and Strategy: A description and discussion. Interdisciplinary Journal of E-Learning and Learning Objects (IJELLO) 6, 217–238 (2010)
11. Sieg, W.: The AproS Project: Strategic thinking & computational logic. Logic Journal of IGPL 15(4), 359–368 (2007)
12. Terrematte, P., Marcos, J., Galdino, T.: O LOGICAMENTE: A implementação colaborativa de Objetos de Aprendizagem de Lógica (IV WAPSEDI). In: SBIE - XX Simpósio Brasileiro de Informática na Educação (2009)
13. Vilela, G., Rosan, M., Marcos, J.: Implementação de um gerador e verificador de modelos finitos para a Lógica Clássica de Primeira Ordem. VIII ERMAC–R3 (2008)

A Framework for Coping with Logically-Minded Arguments in Philosophy

Luis Adrian Urtubey

Facultad de Filosofía y Humanidades, Universidad Nacional de Córdoba, Ciudad Universitaria, Córdoba, Argentina
luis.urtubey@gmail.com

Abstract. I argue that a gentler framework for analyzing logically-minded arguments in philosophical discourse may be given by a twofold structure stemming from the arrangement of two different devices. One standard logical device operates at the formal level of premises and conclusion of an argument-text. The second, is able to deal with intensional steps which are normally performed to support background reasoning. In this contribution I aim at developing a simple general account to deal pedagogically with these two-parallel levels, characteristic to my mind, of many philosophical argument-texts that are embedded in logically valid patterns of reasoning.

Keywords: Logic, philosophy, argument-text, validity, conditionals.

1 Introduction

I address in this article a difficulty which stems from a bewildering confrontation with philosophical arguments that make use of valid patterns of reasoning borrowed from classical logic. Philosophical arguments raise many challenges for teaching elementary logic. They are customarily formulated in natural language, far away from the accurate languages of formal logic. Thus, they remain anchored to a highly interpreted language and material inference. Moreover arguments delivered in philosophical texts are sometimes couched in formal schemas honored by formal logic. Allegedly classical logic is the science of inference and also the science of truth. Alternatively there are some ingredients of an argument other than truth that logical inference helps to preserve, such as evidence and modal force.

It is broadly recognized that logic is normative. It usually means that in a remarkable sense, when an argument is valid, then one somehow goes wrong if one accepts the premises and rejects the conclusion. What is more important yet is that we use arguments deemed as valid to judge inferences. It encompasses the use of deductive inference in the rational assessment of beliefs and theories, arguments and hypotheses. Normativity of logic in that sense is mandatory, even though sometimes it could be rational to violate these norms.

Let me illustrate this point a little further by appealing to a related case where the normativity of logic seems to go astray. There are some venerable

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 231–238, 2011.

problems to confront for anyone who takes logical consequence to be a normative constraint on the acceptance of arguments conclusion. Ancient paradoxes, such as the well known family of slippery-slope puzzles, have badly upset logicians and philosophers for centuries. In some versions of these riddles, such a noble schema of inference as modus ponens seems to be seriously menaced. Here is a version of the paradox in this vein: Lets assume that a person aged 20 is young. Plausibly the following inductive rule holds: If a person aged m is young, then a person aged $m+1$ is also young. Adding just one year to a young person doesn't change the matter considerably. By successive application of modus ponens it turns out that a person aged 60 is also young! Utterly false! In this particular case, it doesn't seem to be a mistake to assert the premises of the argument while denying the conclusion. They cannot be true together and the inference shows this. However, if one has good grounds to reject some implications of modus ponens and one has good ground for each application of this rule, it seems that one has good grounds for sustaining an incoherent collection of sentences. It could happen as [1] has pointed out that the normativity of logical consequence remains, even though in these circumstances it is trumped by other norms.

The paper has the following structure. Sect 2 presents the distinction between object-level and meta-level reasoning, which is motivated by the distinction of a two-level argument-text structure. Sect 3 illuminates the relationship between meta-level and preservation of arguments modal force, which has been recognized as an important ingredient of logical consequence along with the truth-preservation property. In Sect 4 I work out an example in order to apply this proposal. The conclusion gathers some results scattered throughout the paper.

2 Object-Level and Meta-level Arguments

The standard notion of logical consequence is expressed in terms of models or structures. A conclusion is said to be valid if it turns out to be true in all models where the premises are true. A question that is worth considering is whether it is rational to select one model, in order to reason (logically or otherwise) in this model. Characteristically, among alternative logics, nonmonotonic logic is supposed to provide us with several models extending a situation. Moreover, as Daniel Kayser [2] has observed recently, there can be many respectable reasons, aside from any logical framework, monotonic or not, to try several models for a given situation. He perspicuously remarks that observing experts at work reveals that they seem to have a meta-reasoning in progress in parallel with their object-level reasoning. If the latter seems to be stuck, the former has the ability to switch to another model and to import the results they get to help out their reasoning. Sometimes they even seem to reason in parallel on two different models of the same situation. Looking at many logically-minded arguments concerning philosophical issues, we might figure out that they are also structured by intertwining two different models. One standard logical model operates at the semantic level of formal truth-preservation between argument-texts premises and conclusion. The other model deals with an intensional support, which is normally trusted to

a meta-level reasoning. This meta-level is much more akin to the modal force of the argument than to the truth-preservation property. It bears on a more local or material inference and also takes advantage of the preservation of the conclusions modal status. Notably both models seem to reason in parallel on two different levels of the same argument.

The so-called substitutional technique or schematic approach has dominated logic since almost its beginning in the antiquity. It is based in a particular method of working with schemas, which has been used since Aristotle with different purposes. Notably its power lies in the capacity to capture interesting patterns among valid arguments. Characteristically the schematic approach consists in allowing for some terms to remain fixed while others are replaced by schematic letters, yielding schemas. Precisely finding valid schemas is the main task of logic and in view of collecting those schemas logicians incorporate tools which allow them to study systematically the logical properties of valid arguments.

Corcoran [3] has considered schemas to its greatest extent and has pointed out that they may be classed by the syntactic type of their instances as sentence schemas, sub sentential schemas, or argument-text schemas. In its turn an argument-text schema is a schema whose instances are argument-texts. He introduces "argument-text" as a two part system composed of premises and conclusion. An argument is that which is expressed by an argument-text, as a proposition is that which is expressed by a sentence. It will be useful to adopt Corcoran's approach in order to separate argument's schemas from its content. Clear examples of argument-text schemas are valid patterns of reasoning as proof by cases, which I'll be considering below. This valid argument schema corresponds in some systems of natural deduction to the rule known as disjunction elimination. Given a disjunction A or B, as a premise, it is allowed to infer a sentence C, whenever C has been previously inferred from both A and B. Beside proof by cases, proof by contradiction and many types of dilemma figure amongst the most preferred logically-valid schemas, which are found in philosophical arguments.

Logic texts used to employ the expression "sound argument" to refer to an argument that has both valid form and true premises. This terminology is used for instance by long-standing manuals such as Copy and Cohen [5]. Considering philosophical arguments, it turns out to be a rather awkward task to judge them about their soundness, because in this type of arguments truth is problematic. Philosophers have learnt classical logic for centuries and they apply this logic to forge and spell out their own reasoning, but the use of logically valid schemas by philosophers has tended to be rather special.

This last remark also serves to illuminate a concomitant fact concerning the formalization of philosophical arguments by applying elementary deductive logic. Frequently it turns out that it is necessary to insert implicit premises in order to accommodate the argument to calculus deductive requirements. Let alone the issue of translating from natural language to a formal language. It turns out that the stuff usually looks awkward and unconvincing for philosophical audiences.

Turning to my point, in view of having a more adequate account of it, a philosophical argument tailored for a logically valid schema is to be splitted in two levels. At the object-level, such an argument is characteristically served in a truth-preserving argument-text schema. Alternatively, inherent material or local reasoning is justified at a meta-level, by norms of a broader logical rationality. Acceptably formal truth-preservation is the golden rule at the object-level. Valid patterns of inference guarantee sheer consistency with that rule of proof-rationality, as it were. Attributing a broader rationality to human beings, it is easy to understand that the utility of logic in such a controversial matter as philosophy did not depend on that last alternative altogether. Contrariwise, meta-level models are more tolerant and they exploit the less restricted side offered by the preservation of modal force that valid arguments also carries on.

3 Meta-level Arguments and Modal Force

Logically valid arguments have normative force. They compel us to accept the conclusion having accepted the premises. It has been remarked by Gila Sher [4] that one of the most valuable things concerning valid logical inference is that it permits us to extend our system of knowledge without reducing its modal force. Particularly it permits to show that the set of premises is as compelling as the conclusion of a valid consequence. Thus we say that the conclusion of a valid inference preserve the modal status of the premises. There is no lacking in modal power going from premises to conclusion by means of a valid inference. In order to show up the modal status of the elements of a given argument, some informal analysis in terms of possible worlds could be promoted.

Let me assume that the notion of a set of steps being compelling is appropriate for interpreting the notion of modal force of an argument. For the sake of simplicity, let me consider those notions in a modal epistemic sense. Patently full deductive or analytic steps i.e. those based on truth-preserving inference rules-have maximal modal force or modal status. What about the modal status of those instensional steps belonging to the higher-level? They would lack certainly the modal power of truth-preserving steps, but they might be still compelling to a certain extent.

Considering different alternatives to confront this scenario from a logical point of view, it seems that the resource of counterfactual conditionals may be adequate in order to evaluate these meta-level aspects. Possible world semantic has been also applied to the semantic of counterfactuals. Plausibly an account using counterfactuals will be carried out in a meta-level with respect to the argument-text, when it doesn't make much sense to think about the premises as if they were true.[1] Assuming as given an argument valid form, the general rule governing this parallel meta-level would be: If the set of reasons $A_1, \ldots, A_n$ held, then each step of the argument $Q_1, \ldots, Q_n$ would hold. Meaning by "held" that each step is to be admitted.

[1] For an overview on conditionals see [6] and [7].

The pattern of evaluation of sentences of this type complies with the possible world approach to the semantics of conditionals, as developed by R. Stalnaker or D. Lewis.[2] Nonetheless it would be more appropriate to use here the suggestive analysis of Williamson [8] and go some way on the basis of our pretheoretical understanding of counterfactual conditionals in our native language. According to Williamson the process of evaluating the counterfactual conditional requires something like two files, one for the actual situation, the other for the counterfactual situation, even if these situations turn out to coincide. Somehow, one came to know the counterfactual by using her imagination. That sounds puzzling if one conceives the imagination as unconstrained. What constrains should be imposed? Williamson advises that the default for the imagination may be to proceed as "realistically" as it can. Thus the imagination can in principle exploit all our background knowledge in evaluating counterfactuals. It is also difficult to say whether the imaginative exercise can be regimented as a piece of reasoning. We can undoubtedly assess some counterfactuals by straightforward reasoning. We can deduce the consequent from the antecedent. However, as Williamson notes, the treatment of the process by which we reach counterfactual judgments as inferential is problematic in several ways. In particular, not every inference licenses us to assert the corresponding counterfactual, even when the inference is deductive and the auxiliary premises are selected appropriately.

All these reasons motivate Williamson to understand the imaginative exercises by which we judge counterfactuals as not purely inferential. It will be useful then to consider that some kind of simulation is involved and that will mean that cognitive faculties are run off-line.

Williamson does not explicitly specify which kind of formal system his theory relies on. His thesis regards, in fact, only epistemological aspects of our modal judgements and does not aim at providing an alternative semantic or logical framework. At best one can render the process of evaluating a counterfactual conditional by saying that the thinker imaginatively supposes the antecedent and counterfactually develops the supposition, adding further judgments within the supposition by reasoning, off-line predictive mechanisms and other off-line judgments. To a first approximation: if the development eventually leads us to add the consequent, we assent to the conditional; if not, we dissent from it.

Alternatively to reach a negative conclusion, we must in effect judge that if the consequent were ever going to emerge it would have done so by now for example, we may have been smoothly fleshing out a scenario incompatible with the consequent with no hint of difficulty.

4 Working Out an Example

Let me consider by way of example the following paragraph borrowed from Plato's Apology, where Socrates unfolds an inflamed ethical and political discourse:

[2] For a brief exposition of Stalnaker's and Lewis' systems see [9].

> Let us reflect in another way, and we shall see that there is great reason to hope that death is a good; for one of two things, either death is a state of nothingness and utter unconsciousness, or, as men say, there is a change and migration of the soul from this world to another. Now if you suppose that there is no consciousness, but a sleep like the sleep of him who is undisturbed even by dreams, death will be an unspeakable gain. For if a person were to select the night in which his sleep was undisturbed even by dreams, and were to compare with this the other days and nights of his life, and then were to tell us how many days and nights he had passed in the course of his life better and more pleasantly than this one, I think that any man, I will not say a private man, but the greatest king will not find many such days or nights, when compared to the others. Now if death be of such a nature, I say that to die is gain; for eternity is then only a single night. But if death is the journey to another place, and there, as men say, all the dead abide, what good, O my friends and judges can be greater than this? If indeed when the pilgrim arrives in the world below, he is delivered from the professors of justice in this world, and finds the true judges who are said to give judgment there ... that pilgrimage will be worth taking. What would not a man give if he might converse with Orpheus and Musaeus and Hesiod and Homer? Nay, if this be true, let me die again and again! ... Above all, I shall then be able to continue my search into true and false knowledge; as in this world, so also in the next; and I shall find out who is wise, and who pretends to be wise, and is not ... In another world they do not put a man to death for asking questions: assuredly not. For besides being happier than we are, they will also be immortal, if what is said is true [10].

First of all it wouldn't make much sense to worry about the truth of each step of the argument-text, because sentences are hardly true or false in this context. Let alone the insertion of rhetorical questions. What turns out to be important instead is the support that each step obtains from explicit reasons that are given inside the argument. Those supporting reasons have to be evaluated in order to clarify how compelling the set of steps turns out to be. As I have remarked earlier, logic has a special place in our system of knowledge. Analogously our ethical system, as it were, can be consistently expanded by means of logical inference. In a certain Socratic way, ethics gives us knowledge of ourselves and so we are expanding our knowledge system anyway.

Let me analyze now the logical structure of the argument unfolded in the text. It is clearly an instance of the above-mentioned valid inference schema called reasoning by cases. The conclusion Socrates wants to prove is that there is great reason to hope that death is a good. Notably Socrates doesn't claim that it is true that dead is a good. He just claims that there is great reason to hope that. At the beginning the argument announces that its goal will turn out to be seen after reflecting or reasoning. Thus the arguments upholder intends to persuade us of the goodness of death even though he is not able to prove the true. Contrarily,

the logical scheme, which supports the argument, is customarily characterized as valid, because it preserves truth. The argument also has an explicit disjunctive premise which clearly presents the cases:

> ...for one of two things, either death is a state of nothingness and utter unconsciousness, or, as men say, there is a change and migration of the soul from this world to another.

Admittedly one can hardly expect to establish the truth of either case of the disjunction. We may be induced at most to believe that if the alternatives have been exhausted and well founded then the conclusion must hold. Thus the argument proceeds by cases taking each part of the disjunction and developing a sub-argument for each case. Arguably these arguments are hardly deductive in form. Lets consider the argument for the first case, beginning at "Now if you suppose that there is no consciousness ...", which runs in the text until "(...), when compared to the others". Clearly what Socrates intends is to give reasons in order to support the conclusion of the argument, but not to give a subsidiary deductive argument to fulfill his purpose. Incidentally such a cumbersome argument may be hardly cast in deductive form, if any. Similarly the argument for the second disjunct beginning at "But if death is the journey to another place ..." and ending at the last paragraph of the text, doesn't make up either a genuine deductive argument. Nonetheless, both arguments are supposed to make a case for the conclusion "that to die is gain", so that it follows due to the validity of the object-level argument form.

Let's see what happen when the above mentioned general conditional rule is applied to both arguments developed for each case respectively. Thus we have to be prone to accept the argument when each step is admitted after supposing the antecedent and counterfactually developing the supposition. In our example, Socrates partially performs this task by supplying some supporting reasons for each case. The question one has to evaluate is whether his reasons -counterfactually developed- permit to reach the conclusion. Contrariwise, we are not going to accept the argument when the counterfactual rule doesn't hold.

In any event, there seem to be two scenarios that one can forge, incompatible with the consequent. On either scenario which stems from Socrates' tale, the consequent that to die is gain, is held based on the underlying assumtion that our personal identity after dying might not prove problematic. For whom would be deadness a good, otherwise. Notwithstanding, immortality and personal identity might prove problematic after all (see [11]). Lacking personal identity then, either scenario turns out to be incompatible with the conclusion that to die is gain.

5 Conclusion

I have proposed an alternative approach that makes an elementary use of the epistemology of counterfactual conditionals in order to incorporate those features of philosophical arguments reluctant to the application of formal deductive methods. Notably the resulting overall argument looks as a sort of hybrid stemming

from a valid deductive form and a set of reasons that support a higher-level justification of the intensional nexus between premises and conclusion. This mix between logic and rhetoric has caught on since the beginnings on the subject of philosophy. Polluted deductive arguments of this kind may fluster a standard logical analysis. At the best the formal logical schema may be enhanced in such a way that its validity turns out to be evident.

Working out an example, has served to show how the more we are willing to accept the steps and so the conclusion, the more we feel ourselves compelled by the argument's modal force to concede the argument's thesis. To a certain extent, there is a matter of qualitative degrees involved here that has been aptly represented by the model we have proposed, based on counterfactual conditionals. It must be allowed that I have given only a sample and it would be useful to work out other examples in view of trying on other proof schemas pick-up amongst those we have mentioned earlier.

A more conservative approach to a logical analysis of philosophical argument-texts used to appeal to the introduction of "implicit premises" in order to achieve a milder deductive formalization. The main problem with this approach is that to determine what premises have to be inserted and to forge them is frequently seen as a cumbersome and unmotivated issue, at least at the level of introductory logic courses for philosophy. To some extent, this traditional analysis has failed to appreciate the most charming part of philosophical arguments[3].

References

1. Beall, J., Restall, G.: Logical Pluralism. Oxford University Press, Oxford (2006)
2. Kayser, D.: The Place of Logic in Reasoning. Log. Univer. 4, 163–205 (2010)
3. Corcoran, J.: Schemata. In: Zalta, E. (ed.) Standford Encyclopedia of Philosophy, http://www.plato.stanford.edu/entries/schema
4. Sher, G.: Is Logic in the Mind or in the World? Synthese (2010), On line First
5. Copi, I.M., Cohen, C.: Introduction to Logic. Macmillan, New York (1990)
6. van Benthem, J.: A Manual of Intensional Logic. CSLI, Stanford (1988)
7. Read, S.: Thinking about Logic. Oxford University Press, Oxford (1995)
8. Williamson, T.: Philosophical Knowledge and Knowledge of Counterfactuals. In: Beyer, C., Burri, A. (eds.) Philosophical Knowledge; its Possibility and Scope. Grazer Philosophische Studien, vol. 74, pp. 89–124 (2007)
9. Sider, T.: Logic for Philosophy. Oxford University Press, Oxford (2010)
10. Plato: Apology. In: Jowett, B. (transl.)The Dialogues of Plato, vol. 2. Oxford University Press, Oxford (1892)
11. Perry, J.: A Dialogue on Personal Identity and Immortality. Hackett Publishing, Indianapolis (1978)

[3] I want to thank two anonymous referees of this publication for their helpful comments.

A Logic Teaching Tool Based on Tableaux for Verification and Debugging of Algorithms*

Rafael del Vado Vírseda,
Eva Pilar Orna, Eduardo Berbis, and Saúl de León Guerrero

Departamento de Sistemas Informáticos y Computación
Universidad Complutense de Madrid
rdelvado@sip.ucm.es,
{evapilar.orna,eberbis,sauldeleonguerrero}@gmail.com

Abstract. While logic plays an important role in several areas of Computer Science (CS), most educational software developed for teaching logic ignores their application in a more large portion of the CS education domain. In this paper we describe an innovative methodology based on a logic teaching tool on semantic tableaux to prepare students for using logic as a formal proof technique in other topics of CS, such as the formal verification and the declarative debugging of imperative programs, which are at the basis of a good development of software.

Keywords: Logic teaching software, Tableaux, Verification, Debugging.

1 Motivation

Computer Science universities often teach a first year undergraduate course on mathematical logic. The syllabus of the course usually includes syntax and semantics of propositional and predicate logic, as well as some proof systems such as natural deduction, resolution, and semantic tableaux. In some cases, there is also some lecture devoted to explain basic concepts on logic programming and practical work using a *Prolog* interpreter.

Most students find the high degree of rigour required in the learning of these contents daunting. In order to provide learning support to our students, proof visualization tools are always helpful. Many tools for teaching logic have been developed in the last two decades (see `http://www.ucalgary.ca/aslcle/logic-courseware`). Most of these tools concern the construction of proofs in a formal logic using semantic tableaux [2,5,7]. A *semantic tableau* [3] is a semantic but systematic method of finding a model of a given set of formulas Γ. A semantic tableau is a refutation system in the sense that a theorem φ is proved from Γ by getting its negation $\Gamma \Vdash \neg\varphi$.

* This work has been partially supported by the Spanish projects TIN2008-06622-C03-01 (FAST-STAMP), S2009/TIC-1465 (PROMETIDOS), UCM-BSCH-GR58/08-910502 (GPD-UCM), and PIMCD 2010/97 (Project for the Innovation and Improvement of the Educational Quality).

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 239–248, 2011.

While logic plays an important role in several areas of Computer Science, most of the didactic software developed for teaching logic ignores the application of logic in other topics of Computer Science. We believe that our students could obtain more benefits from the techniques they learn with these tools, thanks to the possibility of applying them in a variety of contexts in advances courses. Hence, there is a need for a prototype tool allowing experiments on teaching logic in a more large portion of the Computer Science education domain, where the language and the implementation should be accessible enough and popular to ensure that they will be used into the future, and that they remain available in other courses. These motivated us to write this paper.

The aim of this work is to describe an innovative methodology based on a logic teaching tool on sematic tableaux, called `TABLEAUX`, to prepare our students for using logic as a formal proof tool in other areas of Computer Science, with a special emphasis on the design of algorithms and software engineering. Good algorithm design is crucial for the performance of all software systems. For this reason, an ability to create and understand formal proofs is essential for correct program development.

The major contribution of this paper is the development of new tableau methods that give semantically rich feedback to our students for the formal verification and the algorithmic debugging of programs. In this sense, `TABLEAUX` shows to be a good tool for (a little more) advances students, whose logical skills go beyond the rudiments that the user-level interaction with other logic teaching tools can develop. For instance, our tool is used for logic-based methodologies, such as program derivation, reasoning from specifications and assertions, loop invariants, bound functions, etc. This includes topics in areas whose skills and concepts are essential to programming practice independent of the underlying paradigm, as the analysis and the design of correct and efficient algorithms [4].

2 The Logic Teaching Tool

Solving logical exercises is usually done with pen and paper, but educational tools can offer useful pedagogical possibilities. The role of the educational software is to facilitate the student's grasp of the target procedures of education, and to provide teamwork and communication between teachers and students.

Our logic teaching tool `TABLEAUX` (see `http://www.fdi.ucm.es/profesor/rdelvado/TICTTL2011/`) is a prototype of an educational application based on propositional and first-order semantic tableaux with equality and unification [3] used as a support for the teaching of deductive reasoning at a very elementary university level for Computer Science students. This tool helps our student to learn how to build semantic tableaux and to understand the philosophy of this proof device using it not only to establish consistency/inconsistency or to draw conclusions from a given set of premises but also for verification and debugging purposes as we propose in this paper. Our first year students have learnt tableau calculus in the classroom and this software has helped them to easily understand advanced concepts and to produce their own trees.

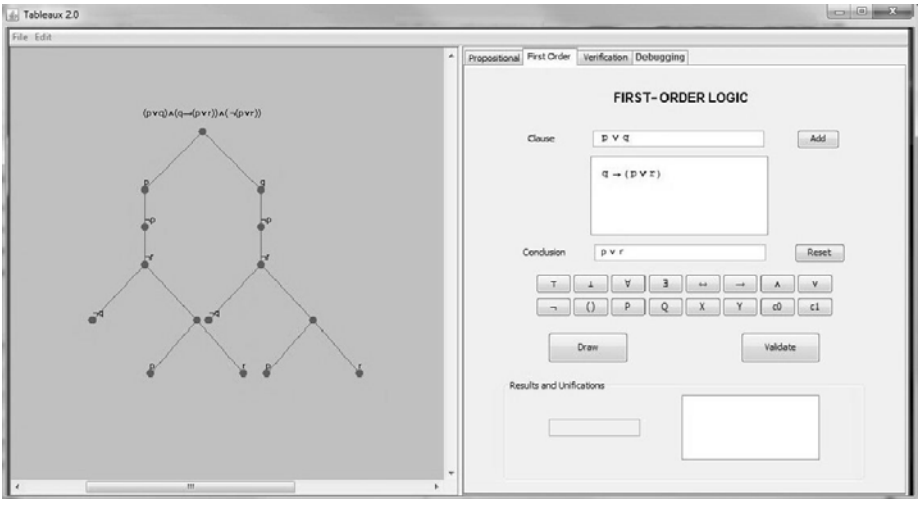


Fig. 1. The logic teaching tool TABLEAUX

2.1 Tool Usage

The tool consists of two main parts: one that produces tableaux and another based on the tableaux method for verification and debugging of algorithms. In both cases, the application possesses a drawing window where the trees will be graphically displayed. The structure of TABLEAUX is shown in **Fig. 1**. The user interacts with the provers through a graphical interface. We have chosen *Java*, but is possible to use *Prolog* to write the provers because its declarative character can give us a natural way to write the operations involved in the implementations (see [3] for more details).

2.2 Tool Implementation

An important design consideration in the tool implementation is that the code must be easy to maintain and extent, guaranteeing its future development and support in a sufficiently large portion of the Computer Science education domain. We have made the choice of an open source *Java* code, facilitating the addition of new features for the verification and debugging of algorithms, and enabling changes to the tableau ruleset to accommodate these new methods and applications. This makes TABLEAUX more interesting for an educator to invest in the application and extension of this tool.

Specific details on the straightforward aspects of a tableaux tool's development are described in [2,5,7]. We have selected the following aspects for a flexible and declarative representation of formulas and tableau rules:

- **Parsing and tokenizing formulas:** We have set up the tool in a declarative way defining all symbols that can be part of a well-formed formula in a

symbol library and a graphical interface (see **Fig. 1.**). The symbol library that is available to create formulas is declarative and extensible. The basic building block of the tool is the formula, represented internally as a parse that holds the formula's syntactic structure. By changing or extending the recursive definition and the symbol library, it is easy to expand the set of symbol strings accepted as well-formed to include *Hoare logic*, which is the basis for program verification.

- **Automatic tableau constructor:** The current implementation of the automatic prover built into the tool is straightforward and similar to other tableaux tools [2,5,7]. The automatic prover checks the rules applicable for a branch in the tableau, and selects the best one using a simple heuristic. Adapting the prover to give new alternative proofs for verification and debugging is explained in the following sections.

3 Verification of Algorithms

The main novelty of the `TABLEAUX` tool is to train our students in the art and science of specifying correctness properties of algorithms and proving them correct. For this purpose, we use the classical approach developed by Edsger W. Dijkstra and others during the 1970s [1]. The proof rules (semantics) of the algorithm notation used in this paper (see [4] for more details) provides the guidelines for the *verification of algorithms* from specifications. We use Edsger W. Dijkstra's guarded command language to denote our algorithms. Algorithms A are represented by functions `fun` A `ffun` that may contain variables (x, y, z, etc.), value expressions (e) and boolean expressions (B), and they are built out of the skip (`skip`) and assignment statements (x `:=` e) using sequential composition (S_1`;`S_2), conditional branching (`if` B `then` S_1 `else` S_2 `fif`), and `while`-loops (`while` B `do` S `fwhile`). This language is quite modest but sufficiently rich to represent sequential algorithms in a succinct and elegant way.

It becomes obvious that neither tracing nor testing can guarantee the absence of errors. To be sure of the correctness of an algorithm one has to prove that it meets its *specification* [4]. A specification of an algorithm A consists of the definition of a *state* space (a set of program variables), a *precondition* P and a *postcondition* Q (both predicates expressing properties of the values of variables), denoted as $\{P\}\ A\ \{Q\}$. An algorithm together with its specification is viewed as a theorem. The theorem expresses that the program satisfies the specification. Hence, all algorithms require proofs (as theorems do). Our tool verify algorithms according to their specification in a constructive way based on semantic tableaux $P \Vdash \neg\, wp(A, Q)$, where $wp(A, Q)$ is the *weakest precondition* of A with respect to Q, which is the 'weakest' predicate that ensures that if a state satisfies it then after executing A the predicate Q holds (see [4] for more details). We can use `TABLEAUX` to mechanize many of the boring and routine aspects of this verification process.

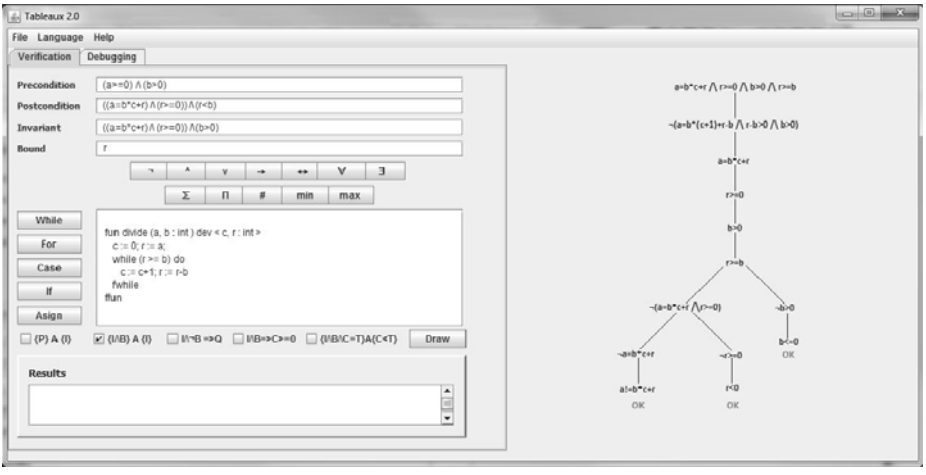


Fig. 2. The logic teaching tool TABLEAUX for verification of algorithms

As an illustrative example, we consider the formal verification of an algorithm to compute the positive integer division (quotient and remainder), specified as:

$\{\, P \;:\; a \geq 0 \;\wedge\; b > 0 \,\}$
fun *divide* $(\, a, b : \mathit{int}\,)$ **dev** $< c, r : \mathit{int} >$
  $c := 0\,;\quad r := a\,;$
  $\{\, I \;:\; a = b * c + r \;\wedge\; r \geq 0 \;\wedge\; b > 0\;,\; C : r \,\}$
  while $r \geq b$ **do**
    $c := c + 1\,;\quad r := r - b$
  fwhile
ffun
$\{\, Q \;:\; a = b * c + r \;\wedge\; r \geq 0 \;\wedge\; r < b \,\}$

Following [4], the verification is based on a *loop invariant* I (supplied by a human or by some invariant-finding tool), a *bound function* C (for termination), and the following five proofs:

- $\{P\}\ c := 0\,;\, r := a\ \{I\}$.
- $\{I \wedge r \geq b\}\ c := c + 1;\, r := r - b\ \{I\}$.
- $I \wedge r < b \Rightarrow Q$.
- $I \wedge r \geq b \Rightarrow C \geq 0$.
- $\{I \wedge r \geq b \wedge C = T\}\ c := c + 1;\, r := r - b\ \{C < T\}$.

Our tool represents each of these proofs as a *closed semantic tableau* (✓). We assume the reader is familiar with the classical tableau-building rules (α and β), equality (=), and closure rules (see [3] for more explanations). We also use the notation $R^{e,\cdots}_{x,\cdots}$ to represent the assertion R in which x is replaced by e, etc. For example, we have the following proof (see also **Fig. 2.**) to verify the preservation of the invariant in the body of the loop $\{I \wedge r \geq b\}\ c := c + 1;\, r := r - b\ \{I\}$:

$I \wedge r \geq b \Vdash \neg\, wp\,(c := c+1\,;\, r := r-b, I) \;\Leftrightarrow\; I \wedge r \geq b \Vdash \neg\,(I\,{}^{c+1,\,r-b}_{c,\,r})$:

$$
\begin{array}{ll}
(1)\;\; a = b*c+r \;\wedge\; r \geq 0 \;\wedge\; b > 0 \;\wedge\; r \geq b & \{I \wedge r \geq b\} \\
(2)\;\; a = b*c+r & (\alpha, 1) \\
(3)\;\; r \geq 0 & (\alpha, 1) \\
(4)\;\; b > 0 & (\alpha, 1) \\
(5)\;\; r \geq b & (\alpha, 1) \\
(6)\;\; \neg\,(a = b*(c+1)+r-b \;\wedge\; r-b \geq 0 \;\wedge\; b > 0) & \{\neg\,(I\,{}^{c+1,\,r-b}_{c,\,r})\}
\end{array}
\quad (\beta, 6)
$$

(7) $a \neq b*c+r$	(8) $r < b$	(9) $b \leq 0$
✓ $(2,7)$	✓ $(5,8)$	✓ $(4,9)$

We can use the tool to guide our students to obtain loop invariants from specifications. For example, if we only provide to our students the postcondition Q, they usually infer only an incomplete assertion $I' : a = b*c+r$ as the loop invariant. Then, when they apply the tool to verify the algorithm, they obtain an *open semantic tableau* (✕) for $I' \wedge r < b \Rightarrow Q$:

$$
\begin{array}{ll}
(1)\;\; a = b*c+r \;\wedge\; r < b & \{I' \wedge r < b\} \\
(2)\;\; a = b*c+r & (\alpha, 1) \\
(3)\;\; r < b & (\alpha, 1) \\
(4)\;\; \neg\,(a = b*c+r \;\wedge\; r \geq 0 \;\wedge\; r < b) & \{\neg Q\}
\end{array}
\quad (\beta, 4)
$$

(5) $a \neq b*c+r$	(6) $r < 0$	(7) $r \geq b$
	⇓	
✓ $(2,5)$	✕	✓ $(3,7)$
	⇓	

We need to insert $\boxed{r \geq 0}$ in I' to close the tableau

From the open branch, our students learn to complete the invariant with $I'' : a = b*c+r \wedge r \geq 0$. However, they still have an open tableau for $\{I'' \wedge r \geq b \wedge C = T\}\ c := c+1;\ r := r-b\ \{C < T\}$:

$$
\begin{array}{ll}
(1)\; a = b*c+r \;\wedge\; r \geq 0 \;\wedge\; r \geq b \;\wedge\; r = T & \{I'' \wedge r \geq b \wedge C = T\} \\
(2)\; a = b*c+r & (\alpha, 1) \\
(3)\; r \geq 0 & (\alpha, 1) \\
(4)\; r \geq b & (\alpha, 1) \\
(5)\; r = T & (\alpha, 1) \\
(6)\; r-b \geq T & \{\neg\,(C < T)^{c+1,\,r-b}_{c,\,r}\} \\
(7)\; b \leq 0 & (=, 5, 6) \\
\Downarrow & \\
\times \Rightarrow \text{We need to insert } \boxed{b > 0} \text{ in } I'' \text{ to close the tableau} &
\end{array}
$$

Finally, they learn to insert $b > 0$ in the assertion I'' to complete the loop invariant I. If they apply the tool again, all the tableaux remain closed and the formal verification session finishes.

4 Algorithmic Debugging

Debugging is one of the essentials parts of the software development cycle and a practical need for helping our students to understand why their programs do not work as intended. In this section we apply the ideas of *algorithmic debugging* [6] as an alternative to conventional approaches to debugging for imperative programs. The major advantage of algorithmic debugging compared to conventional debugging is that allows our students to work on a higher level of abstraction. In particular, we have successfully applied our tool based on semantic tableaux for the algorithmic debugging of simple programs to show how one can reason about such programs without operational arguments. Following a seminal idea from Shapiro [8], algorithmic debugging proposes to replace computation traces by *computation trees* with program fragments attached to the nodes. As novelty, in this work we propose to use computation trees as semantic tableaux. As an example, we alter the code of the previous algorithm with two mistakes:

$\{ P \ : \ a \geq 0 \ \wedge \ b > 0 \}$
`fun` *divide* $(a, b : int)$ `dev` $< c, r : int >$
  $c := 0 ;$ `r := 0` ; ←-- *wrong code* !
  $\{ I \ : \ a = b * c + r \ \wedge \ r \geq 0 \ \wedge \ b > 0 \ , \ C : r \}$
  `while` $r \geq b$ `do` `r := r - b` `fwhile` ←-- *missing code* !
`ffun`
$\{ Q \ : \ a = b * c + r \ \wedge \ r \geq 0 \ \wedge \ r < b \}$

If we try to verify this erroneous algorithm, we can execute again the tool. Now, `TABLEAUX` displays an open tableau $P \Vdash \neg I$ for debugging $\{P\}\, c := 0; r := 0\, \{I\}$, instead of $P \Vdash \neg(I^{0,0}_{c,r})$. However, the weakest precondition $I^{0,0}_{c,r}$ is built from (5) and (6), step by step, to identify erroneous parts of the code in open branches:

(1) $a \geq 0 \wedge b > 0$ $\{P\}$
(2) $a \geq 0$ $(\alpha, 1)$
(3) $b > 0$ $(\alpha, 1)$
(4) $a \neq b * c + r \ \vee \ r < 0 \ \vee \ b \leq 0$ $\{\neg I\}$

$(\beta, 4)$

(5) $a \neq b * c + r$ | `c := 0` | (5) $a \neq r$ | `r := 0` | (5) $a \neq 0$ ⇓ $\times$

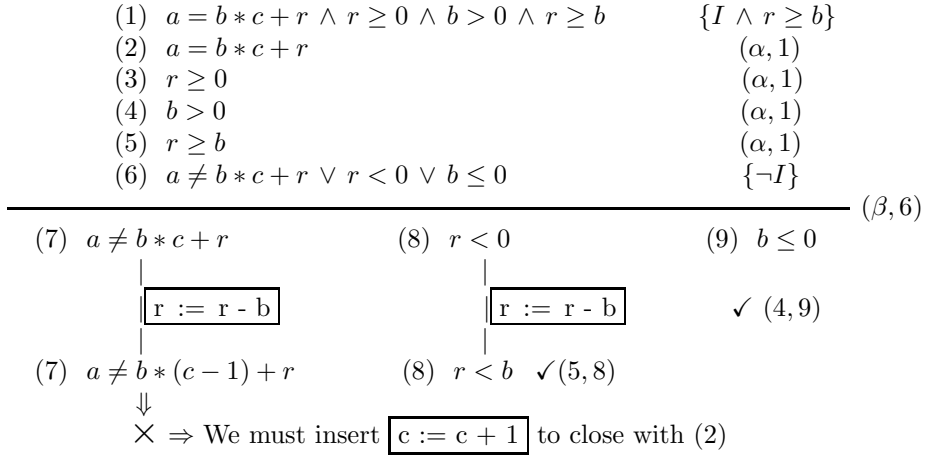
(6) $r < 0$ | `r := 0 (or a)` | ✓ $(3, 7)$

(6) 0 $(or\ a) < 0$ ✓ or $(\checkmark(2, 6))$

(7) $b \leq 0$

⇒ We must replace r := 0 by `r := a` to close the tableau

After this correction, we obtain a closed tableau. Now, we can execute again the tool to perform the algorithmic debugging of $\{I \wedge r \geq b\}\ r := r - b\ \{I\}$:



To close the open branch, we infer that we need to insert new code. This particular incompleteness symptom could be mended by placing $c := c + 1$ in the body of the loop. If we apply again the tool, no more errors can be found and the five tableaux remain closed. The debugging session has finished.

5 An Educational Experience with TABLEAUX

The prototype of the educational tool TABLEAUX is available for the students of the topics *Computational Logic* and *Methodology and Design of Algorithms* in the Computer Science and Software Engineering Faculty of the Complutense University of Madrid through its Virtual Campus. The following results are based on the statistics from the 186 students who took the course in 2009/2010.

5.1 Design of the Experiences

We have carried out two educational experiences:

- One **non-controlled** experience: All the students may access the Virtual Campus and participate freely in the experience: download and use the tool, and answer different kinds of tests.
- One **controlled** experience: Two groups of students must answer a test limited in time and access to material.

With respect to the **non-controlled** experience, the students may freely access the Virtual Campus without any restriction of time or material (slides, bibliography, and the tool) and answer the questions of several tests. For each of the following topics in Computer Science and Software Engineering we have provided a test that evaluates the knowledge of our students applying different kinds of semantic tableaux. The students may use these tests to verify their

understanding of the different concepts. The questions are structured in three blocks: *propositional and predicate logic*, *specification and verification* of algorithms, and *debugging and derivation* of imperative programs. The resolution of the tests by the students is controlled by the Virtual Campus with the help of an interactive tutoring system. In the **controlled** experience we try to evaluate more objectively the usefulness of the tool. In particular we have chosen the application of `TABLEAUX` for the verification and debugging of simple searching and sorting algorithms [4]. We have chosen two groups of students answering the same questions: approximately half of the students works only on the slides of the course and the books at class; and the other half works only with the tool at a Computer Laboratory.

5.2 Results

Non-controlled Experience: We outline here the main conclusions from the results of the **non-controlled** experience. With respect to the material the students used to study, as long as the exercises were more complicated the use of the tool (simulations, cases execution, and tool help) increased considerably. Better results were obtained in the verification and debugging of searching and sorting problems (linear and binary search, insertion and selection sort). The tool helped our students to visualize array manipulations in array assignments. In the rest of the algorithms (slope search and advanced sorting algorithms) they used only the class material or bibliography. When answering the tests questions, the students were also asked whether they needed additional help to answer them. In the case of linear and binary search they used the tool as much as the class material, which means that visualization of their own proof tableaux were a useful educational complement. We can conclude that the students consider the tool as an interesting material and have used it to complement the rest of the available material.

Controlled Experience: The **controlled** experience was carried out with 59 students. We gave 32 of them only the slides of the course and the books of the bibliography [3,4]. The rest were taken to a Computer Laboratory, where they could execute the `TABLEAUX` tool. We gave the same test to both groups, consisting of 18 questions, 12 of them on specification aspects of the algorithms (inference of invariants and bound functions), and the rest on their verification and debugging from the code. In **Fig. 3.** we provide the means and the standard deviations of correct, errors, and *don't knows* answers. First, we observe that students using the `TABLEAUX` tool answer in mean more questions than the other ones. In addition, they make less errors than the others. This is due to the fact that most of the students of the *tableaux/tool* group perform the analysis of the algorithms directly from the corresponding semantic tableau, while the *slides/book* group had to hardly deduce it directly from the code. All the students who used the `TABLEAUX` tool indicated the benefits of using tableaux to understand the code of the algorithms from their specifications. Therefore, we can conclude that the methodology proposed in this work constitutes a good

	correct		errors		don't knows	
	mean	σ	mean	σ	mean	σ
slides/books	9.36	2.35	6.23	2.37	3.21	2.82
tableaux/tool	11.82	2.97	4.81	2.10	1.22	1.73

Fig. 3. Means and standard deviations (σ) of the controlled experience

complement to facilitate the comprehension of the design and analysis of programs. In addition, the methodology based on tableaux has helped us to detect in the students difficulties applying the formal techniques to derive correct and efficient imperative programs from specifications.

6 Conclusions

We have presented an educational prototype tool based on semantic tableaux for a specification language on predicate logic. This is the first step towards the development of a practical reasoning tool for formal verification and declarative debugging of algorithms. We have systematically evaluated the proposed method to confirm that a tableaux tool is a good complement to both the class explanations and material, making easier the visualization of proofs in the reasoning needed for the design of correct and efficient imperative programs.

References

1. Dijkstra, E.W.: A Discipline of Programming. Prentice-Hall, Englewood Cliffs (1976)
2. van Ditmarsch, H.: Logic software and logic education. Conta-ins a Comprehensive List of Educational Logic Software (2005)
3. Fitting, M.: First-Order Logic and Automated Theorem Proving. Graduate Texts in Computer Science. Springer, Heidelberg (1990)
4. Kaldewaij, A.: Programming: The Derivation of Algorithms. Prentice-Hall International Series in Computer Science (1990)
5. Lancho, B.P., Jorge, E., de la Viuda, A., Sanchez, R.: Software Tools in Logic Education: Some Examples. Logic Journal of the IGPL 15(4), 347–357 (2007)
6. Naish, L.: A Declarative Debugging Scheme. Journal of Functional and Logic Programming 3 (1997)
7. van der Pluijm, E.: TABLEAU: Prototype of an Educational Tool for Teaching Smullyan Style Analytic Tableaux, University of Amsterdam (2007)
8. Shapiro, E.Y.: Algorithmic Program Debugging. MIT Press, Cambridge (1983)

Designing an Introductory Course to Elementary Symbolic Logic within the Blackboard E-learning Environment

Frank Zenker[1], Christian Gottschall[2], Albert Newen[3], Raphael van Riel[3], and Gottfried Vosgerau[4]

[1] Lund University, Sweden
frank.zenker@fil.lu.se
[2] University of Vienna, Austria
[3] Ruhr-University Bochum, Germany
[4] University of Düsseldorf, Germany

Abstract. We report on the design of a blended-learning course in elementary symbolic logic. Challenges and solutions pertaining to the Blackboard e-learning environment (Blackboard Academic Suite Release 8.0) and a customized Gentzen-style proof checker are described. The purpose is to provide orientation for those in the planning stage of similar projects.

Keywords: Blended-learning, Blackboard, proof checker, calculus of natural deduction, online assignment, automatic grading.

1 Introduction

Social, political and economic factors have forced an increase in flexibility and efficiency of higher education. Students regularly pursue part-time employment. Additional constraints arise from early family situations. Others experience limited mobility because of handicaps or long-lasting illness. Under these constraints, a self-paced and internet-based learning model is often perceived as a quasi-natural solution. For an overview of computer-assisted learning solutions, see [6] and, particularly with respect to logic, [4].

When it comes to teaching logic online, best praxis examples are rare. Off-the-shelve solutions are either unavailable or not readily compatible with the e-learning system favored by one's university. To avoid "reinventing the wheel," we briefly report experiences gained when designing an introductory course to elementary symbolic logic at Ruhr-University Bochum, Germany. The course was adapted to the Blackboard e-learning environment as well as a customized natural deduction proof checker. It currently "runs" in the third year, and may be considered stable.

2 Design Issues

2.1 Symbolism

The most important factor in the successful deployment of course-material is the display of logical symbols, especially for online tests. Students will access material

P. Blackburn et al. (Eds.): TICTTL 2011, LNAI 6680, pp. 249–255, 2011.

from various locations and machines. Normally, operating systems and software vary; special characters tend to be variably interpreted and displayed.

To guarantee a correct display without creating image-files, we employed both standard symbols and pursued a "type-writer solution" (below). The script, available in PDF-format, used standard symbols to train students in the most common set of logical symbols. For other formats (HTML for tests or ASCII in chats and email), we used the type-writer solution. Conveniently, it relies on symbols which are *cognitively available*, i.e., printed on the keyboard.

Table 1. Logical symbols

Standard Symbol	Typewriter Symbol	Meaning	Example
$\wedge$	&	Conjunction	*p* & *q*
$\vee$	v	Disjunction	*p* v *q*
$\neg$	~	Negation	~ *q*
$\rightarrow$	->	Conditional	*p* ->*q*
$\leftrightarrow$	<->	Biconditional	*p* <->*q*
$\forall$	A	universal quantification	*Ax F(x)*
$\exists$	E	existential quantification	*Ex F(x)*
$\equiv$	=	Identity	*x* = *y*
$\vdash$	|-	is derivable from	*p* |- *p*
$\models$	|=	entails semantically	*q* |= *q*

Besides the cognitive advantage, this solution saves coding time. Moreover, in large part, the symbolism could also be used to code proof exercises (see Section 2.3).

2.2 Training, Testing and Grading of Online-Assignments

Each week, students took a test within the e-learning environment, consisting of eight to twelve test items. The test had to be submitted within four days after assignment and could be taken only once. The correct solutions, along with the automatically calculated grade, became available to students only after the submission deadline.

For each test, a "pre-test" featuring a roughly equal number of similar items was made available (training runs). This was slightly easier and could be taken *ad lib*. Users received pre-coded feedback upon completion. Such feedback included the correct solution and an explanation why it is correct. In multiple choice tasks, the false choices were also explained.

Design-wise, severely debugging a test before deployment is indispensable, as each faulty test item which goes unnoticed will require additional work to correct grades.[1] As a rule of thumb, the test-designer is least likely to find *every* mistake. In our case, three colleagues provided independent bug reports.

Pre-tests were reportedly useful for appreciating the *kinds* of questions on the test. In fact, we received complaints when tasks differed in kind (which was the exception). Generally, students were unlikely to under-perform on the test when compared to their pre-test score. We see a positive (learning) effect here, insofar as taking the pre-test successfully should reduce anxiety when taking the test.

Test-templates native to Blackboard are of fairly limited use. They seem to be conceived primarily with natural language manipulation in mind, and have to be adapted creatively. For example, the *free text gap* template can be useful as a truth table assignment (place the variable which codes the gap in a table cell), or a semantic proof (with the letters T and F, for 'true' and 'false', to be filled in). Likewise, the *pull-down menu* template allows construction of simple formula derivations, transformations and, generally, any ordering task. The *Yes/No* template proved helpful when testing the mastery of definitions. Expectably, designing *multiple choice*-items is easiest in this environment.

Challenges arose when having formulated the task in an unclear manner, moreover from users' spelling errors and, in complex formulas, from users placing spaces in an unsystematic manner. Notoriously, Blackboard can neither "ignore" spaces nor automatically respect commutativity. For example, in a formula input-task such as 'State the formula described by the following truth table, using only ~ and & next to the propositional variables p and q', the designer will have to code not only '$\sim p \& q$' and '$q \& \sim p$' as correct solutions, but also "blanked variations," such as '$\sim p$ & q' (with spaces between '$\sim p$' and '&', '&' and 'q'), etc.

Blackboard is moreover limited to a maximum of 20 correct solutions per test item. Therefore, we often explicitly demanded *not* to use spaces (making the input less readable) or resorted to coding with pull-down menus. On the (de)merits of Blackboard, see [1].

If a test is well conceived, Blackboard's grading function reliably indicates a student's test performance. When required, the scores automatically calculated by the system can be overridden manually. Generally, grades have to be entered manually for essay questions and natural deduction tasks (which were the minority of the tasks we prepared; see below).

2.3 Automated Check of Gentzen-Style Proofs

Students were required to compose derivations within a Gentzen-style calculus of natural deduction, e.g., 'Derive q from $\sim p$ and p v q' (according to defined introduction

[1] We can report two such occasions at the end of the first third of the course, upon which debugging by a third party was adopted. In the error case, the course designer made the current test unavailable for users that had not taken it already, created and deployed a second version (V2), then corrected the students' grade book entry for the buggy version and retained them in the grade book alongside the revised version. If the "buggy" V1 shall be deleted without having V1 students retake the test, the designer will have to manually enter their V1 grades into the V2 test results – time-wise, another black hole.

and elimination-rules). We introduced an adapted version of the calculus described in [2]. Automatizing this task proved to be the project's foremost technical challenge. It could only be met incompletely after custom-fitting an interface between the proof checker application and Blackboard had to be excluded for budget reasons. The proprietary Blackboard code was the most important factor hindering a full integration, because the prospect of having to backward-engineer it made the extent of the project unforeseeable.

The solution we developed mimics integration. It uses *building blocks*, a Blackboard software-enhancement. This allows sending a time and a user ID along with the data that users enter into a form on the input page of (what we refer to as) the "proof checker." This page was requested from the server by means of a Blackboard internal link; the derivation task is coded as part of this link.

For example, the snipped '…task=~p+%26+~q,~%28p+v+q%29…' generates the test-item: 'Prove that *~p & ~q* is derivable from *~(p* v *q)*'. The conclusion is written first. What follows after a comma is a premise. '+' codes a space, '%26' codes '&' while '%28' and '%29' signify a left and a right bracket. Conveniently, a subset of the typewriter symbols (see table 1, above) can be reused both for coding the derivation task and for completing it.

The proof checker's results were not available to students. They could only be accessed by their tutors through a password protected area. Based on, but not determined by the program's responses (e.g., "Not derived in a rule compliant manner, error in line X"), grades were assigned and manually entered into the Blackboard grade-book. A natural future development is to fully integrate the proof checker, so results are automatically transferred to the grade book.

A noteworthy feature is the *syntax validation function*. It is implemented as a button reading 'check formula' below the proof-checker's input form. The form consists of ten rows of four free-text fields (dependent premises, formula, derivation rule applied, lines used); clicking a second button generates additional rows, line numbering is automatic. Before submitting a proof, students can have their input checked to "weed out" mistakes.

For example, in response to a syntactically faulty line such as '*p ~* v *q*', the proof checker reports a general error, e.g., 'Syntax error in line 3'. This extends to less obvious mistakes, e.g., 'The Rule for OR introduction cannot be applied in this way. Note that the order of lines to which the rule is applied is relevant'. Through this feature, we could keep submissions from containing mistakes, while allowing errors.

The proof checker is a custom-adaptation of a program running on Christian Gottschall's website "Gateway to logic" [3]. It performed extremely well. Unlike the blackboard native input fields (see above), the proof checker does ignore spaces.

2.4 Programing Considerations

The proof-checker program has been around since 1992. Based on the calculus of Lemmon [5], it reads a text file containing a derivation, then reports whatever it finds wrong within the respective derivation. Wrapped in a CGI layer, the program later became part of what is now the "Gateway to Logic".

Adapting the program to the course's calculus was a fairly simple affair. However, in the future, we would prefer not to code the parser manually, but to use a "compiler

compiler" (e.g., yacc or javacc). This would have significantly accelerated changes in syntax. Further, using a higher-level programming language (i.e., higher than C) – which, for performance reasons, was not an option at that time – will considerably reduce the time needed for adapting and testing.

On the programming side, the main challenge was the seamless integration of the proof checker into Blackboard's user-flow and user-experience, while minimizing the possibility of manipulation. After all, the proof checker is used for submitting homework, and can be used for exams. We could re-use the existing CGI wrapper from the "Gateway to Logic," although the level of required adaptation was higher than with the proof checker.

In contrast to the "Gateway" proof-checker, the Blackboard version required the implementation of statefulness. States were: (1) the input form with the default number of blank proof lines; (2) the filled-in input form with an additional proof line (being requested by the user); (3) the form after a quick initial input check; (4) the checked, and commented, proof (for practice runs); and (5) a confirmation of the successful submission of a derivation. Furthermore, at each stage transformation, user credentials, and lifetime information for the respective request had to be generated, signed, and properly passed over.

For projects without legacy software, we recommend using as high-level a programming language as possible. Especially explicit list and set processing capabilities come as a plus, and do not necessarily require (things as fancy as) Prolog or LISP. For a number of younger projects, we can report excellent results obtained with Java.

Further, it is recommendable to delegate as much routine work as possible (here: mainly CGI issues) to some open middleware (e.g. class libraries), rather than addressing these issues "by hand". This holds especially for language processing: Do define the syntax of the logical language, and the calculus as a whole, in some established meta-language (BNF, EBNF), then have it parsed automatically. With recent Java-based projects, JavaCC was our tool of choice.

2.5 General Considerations

When designing the course material, it is good to keep in mind that at least one entire session will be "lost" explaining to students the details of using the e-learning tools. Especially for a pure online version of the course, it is advisable to meet/contact students in advance. Currently, at introductory level, the majority of students are unacquainted with e-learning environments.

During online-based tutorials, questions related to these tools arose frequently, thus reducing the time allocated to discuss the session's content. Fortunately, this did not result in grave problems. The loss of time, it seems, was counterbalanced by the fact that students were enabled to find answers themselves, using audio/video-recordings of lectures, as well as pre-tests, an online glossary, and the course script.

2.6 Final Exam

The final exam was a classic pen and paper test. However, most of the exam-questions were in the style of the online tests. Conducting a final exam online did not appear viable for various reasons. Generally, tests should not be taken at home, as this

not only encourages cheating, but also increases the risk of computer system or internet connection failures. Thus, a room with a sufficient number of workstations connected to a server strong enough to handle a high number of quasi-synchronous users is required. These constraints, however, could not be met.

In the future, we plan a mixed procedure by using EvaExam.[2] The program allows generating a paper exam featuring barcodes and a defined layout. The completed exams can be scanned and automatically evaluated. This, of course, only works for multiple-choice tasks. Other types of tasks will have to be evaluated manually. The grade is noted on the paper and then read out by the program. Thus, the tasks from the online tests can be largely (re)used for a pen and paper exam and be evaluated automatically.

3 Lessons

Although the majority of students reported no problems with handling Blackboard, we had to learn that our enthusiasm about the e-learning project was not generally shared. In fact, some students were downright adverse to new media, and dropped the course as soon as they noticed it employed e-learning techniques. Principal criticism also arose, because some argued that e-learning offered an excuse or pretext for not hiring additionally staff, thus reflecting a false development of the university system.

During the first two weeks, we catered for students' differential affinity to new media by providing ample opportunity for individual help with anything from obtaining a user name to arranging a wireless network connection on campus. Fortunately, few needed it, but we think that some might have performed worse without it. Therefore, in the first few weeks, we offered one special online tutorial dedicated to technical questions only.

As stated, course material was made available as a script, and supported by a video recording of the weekly plenary meeting. Nevertheless, students benefited greatly from the weekly online tutorials (one tutor for 15 students). With this important instrument the online course ran rather smoothly. The only part of the course that students truly needed support with was the calculus of natural deduction. So, we resorted to introducing the convention that an online-course (which normally knows no real "face-time") features three meetings. The first is used to introduce the technical background. After two thirds of the semester, one meeting is spent on introducing the rationale of the calculus of natural deduction, before presenting its details online. Finally, the last meeting is used for the exam. With this strategy the course has proven to be extremely useful.

We recommend collecting material from previous courses well ahead of time. Much of it will prove useless, because it cannot be transferred online without major change. Therefore, seek lots more than you think you will need. Having a script ready is a plus. However, it will likely have to be revised to accommodate the online exercises. Generally, most assignments which were formerly graded by humans need substantial revision. Most will have to be built from scratch.

Expect at least one full hour for building one test-item, no matter how simple it looks. Allow two hours for the first twenty items. On those, half your time will be

[2] EvaExam by Electric Paper, http://www.electricpaper.biz/

spent learning what you cannot do. If one subtracts a final exam and perhaps the first session, then, for a 14 week course featuring one weekly pre-test and one test with, on average, 10 items, you face 140 hours of build-time – merely for implementing the tests, not for finding your material. As stated above, building is one part, severely testing the other. One hour of debugging is reasonable per ten test items and reviewer, on pains of spending thrice the amount on corrections.

If reviewing what goes online may count as essential, performing back-ups should count doubly so. Blackboard comes equipped with an easy to use back-up tool. This generates a compressed file of the entire course or parts (e.g. tests) for local storage. In the worst case, a course can be completely restored from this file. A weekly backup should be the minimum standard. As it takes five minutes but might save your neck, a daily backup can be recommended.

Readers interested in details are encouraged to contact the corresponding author.

Acknowledgements. We thank our tutors, Natalia Eberle, Uwe Hunz, Hans-Joachim Höh, David Neugebauer, and Kathrin Braungardt and Jens Riedel for tech-support.

References

1. Coopman, S.J.: A critical examination of Blackboard's e-learning environment. First Monday 14(6) (2009), http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/issue/view/291
2. Forbes, G.: Modern Logic. In: A Text in Elementary Symbolic Logic. Oxford University Press, Oxford (1994)
3. Gottschall, C.: Gateway to logic. A collection of online logic tools (2011), http://logik.phl.univie.ac.at/~chris/gateway/formular-uk.html
4. Huertas, A.: Teaching and Learning Logic in a Virtual Learning Environment. Logic Journal of the IGP 15(4), 321–331 (2007)
5. Lemmon, E.J.: Beginning Logic. Chapman and Hall, London (1987)
6. Mayadas, A.F., Bourne, J., Bacsich, P.: Online education today. Science 323, 85–89 (2009)

Author Index