Computer Science Undergraduates Learning Logic Using a Proof Editor: Work in Progress

J. C. Aczel[1], P. Fung[1], R. Bornat[2], M. Oliver[3], T. O'Shea[4] & B. Sufrin[5]

[1] Institute of Educational Technology, The Open University

[2] Department of Computer Science, Queen Mary & Westfield College, University of London

[3] Learning and Teaching Innovation and Development, University of North London

[4] Birkbeck College, University of London

[5] Oxford Computing Laboratory, University of Oxford

Extended Abstract

Introduction

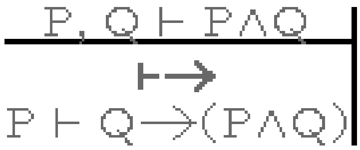
Computer science undergraduates can experience certain difficulties in employing formal reasoning methods (Fung & O'Shea, 1992; Fung et al, 1993; 1996). Although the role of visualisation in assisting the reasoning process in abstract situations appears to be complex (Cox, Stenning & Oberlander, 1995; Cox & Brna, 1995), it is clear that there are software tools - such as Tarski's World (Barwise & Etchemendy, 1992) - that are considered by students to be not only useful for learning the syntax and semantics of first order logic but also enjoyable (Fung et al, 1994). However, the extent to which visualisation tools can assist the learning of heuristics for constructing proofs is uncertain.

The ongoing research outlined here aims to explore the benefits of a particular tool called Jape (Bornat & Sufrin, 1996) that enables students to construct, revise and test formal proofs. The cognitive processes at work when students work on proofs - when using Jape and when using pencil-and-paper - are of particular interest. In doing this it is hoped to increase understanding of the effectiveness of such tools in software engineering. In particular, a contribution might be made to understanding the appropriateness of this approach for different user groups, the features of software engineering tools that best support learning formal reasoning for software development, and the role of visualisation in the reasoning process.

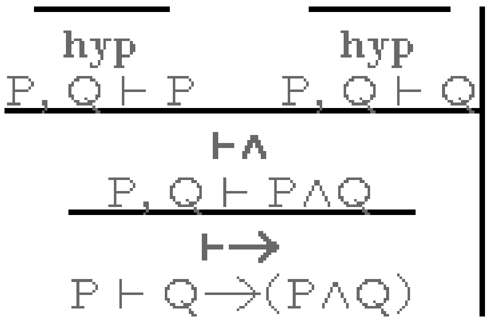
Brief Outline of Jape

Jape takes a description of a particular logic as a system of inference rules. The program has been applied to several logics, including predicate calculus, operational semantics, Hindley-Milner type assignment, axiomatic set theory, a functional programming logic, and a Hoare logic of program refinement. During proofs in these logics the proof tree is directly manipulated simply by clicking on a formula with the mouse. Jape has a tactic language in which actions may be bound to mouse clicks, menu items, and keystrokes, and is used to control the display of proofs and to perform simple searches. The syntax of logical formulae, the form of judgements, the rules used, the entries in menus, the effect of selection and double-clicking are under the control of the person who encodes the logic.

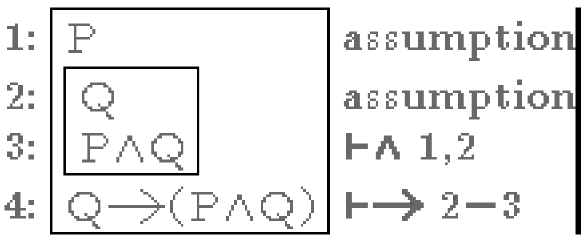
So, for example, in proving $P \vdash Q \rightarrow (P \wedge Q)$ using natural deduction, clicking on the implication-introduction entry in the rules menu changes the proof record to look like this:



The formula on the right hand side of the turnstile is now a conjunction, so one could use the conjunction introduction entry on the rules menu to make the next step. But the designer of this presentation of the logic built in some direct manipulation rules, so simply double-clicking the formula to the right of the turnstile invokes the appropriate proof rule for the topmost connective of the formula, and the proof record changes to look like this:

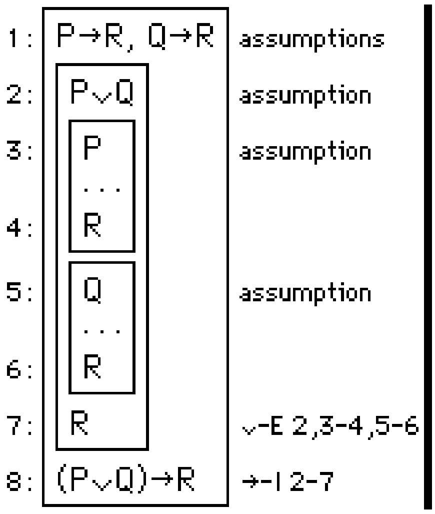


Proofs can also be displayed as boxes:



Description of the Study

It should be stressed that Jape offers great versatility both in the logic being studied and in the construction of a learning environment. However, it is currently only being evaluated with respect to a particular implementation designed for the learning of natural deduction in a term-long course in introductory predicate logic. This implementation uses the Fitch boxes rather than tree display:



a natural-deduction proof in progress

The study involves approximately 170 first-year computer science undergraduates at Queen Mary & Westfield College, University of London. In order to provide a thorough evaluation of the effectiveness of the software tool, a combination of qualitative and quantitative methodologies that have been developed for previous work in this area will be adopted.

In the first phase of the research, initial profiling of the students is intended to provide a baseline for use in evaluation of the tool. The data has been gathered by adapting instruments developed as part of the previous evaluation of visualisation tools (Fung & O'Shea, 1992). Profiles, although not generated at the time of writing, will be used to inform the analysis of performance indicators taken throughout the course. Students are from a variety of backgrounds: - not all will have A-levels, for example; some will be taking a degree in, say computer science and linguistics, or computer science and business studies; not all have prior programming experience; and motivations may vary from the intellectual satisfaction of learning rigorous methods to anticipated financial rewards. Data is also available from a number of short mathematical and reasoning tasks. The use of control groups was discounted on both ethical and pragmatic grounds. However, an analysis of the comparative effects for different backgrounds of the student population will be possible.

Observation of the students has been undertaken during their weekly practical sessions, and this data is supplemented by students' reflections on their learning. When Jape is being used, a logging mechanism is in operation which provides data on how long each student spends on each proof. It may therefore be possible to find out which proofs appear particularly difficult, and whether the duration and frequency of program usage is associated with better test results. However, it should be noted that some students may be using Jape at home, for which timings would not be available. It has not been possible at this stage to obtain more detailed records of the particular rules applied by students during proving - ideally, a complete record of each student's interactions with the program would be captured, but this may not be logistically possible for this study. So qualitative investigation of student use of Jape includes in-depth studies with a number of students who are being individually observed using the program over a period of time and who are interviewed about their experience of using the tool. Video evidence is available. Paper-and-pencil behaviours in response to natural deduction items for students before and after using Jape may allow an opportunity to demonstrate any transfer of skills - for example to parsing expressions.

Preliminary Results

A number of issues have been raised as a result of the data collection; but the limited analysis that has taken place so far means that the preliminary results are sketchy at the time of writing. It does seem, however, that student have some difficulty in recalling proof heuristics that were developed a week before. Moreover, the relationships between the completed proof, the pencil-and-paper process of constructing that proof, and the Jape-supported process of constructing the same proof appear subtle. This is more than just a conflict between the way the proof is constructed by hand, and the apparent sequential nature of the final product. In supporting students' constructions, Jape seems to encourage a particular way of working that "feels" quite different.

Although there are interface issues involved here - especially to do with which line of the proof is to be clicked in order to apply a particular rule - a more interesting aspect seems to be the steps that students find that they cannot make using Jape, even when these steps might be unhelpful or illegal. Also, several students suggested that they were confused at certain points in particular conjectures about whether, when they selected a rule, it would be applied "forwards or backwards", and it is not at all clear yet whether this is purely an interface issue or whether there a lurks in this a deeper misconception about the nature of the formal reasoning process.

Looking at specific rules, there appeared to be some doubt about whether the rule "Ù -E(L)" selected or removed the left-hand-side of the formula. A similar doubt did not appear to occur for or-introduction. The implies-introduction and implies-elimination rules were apparently applied without difficulty. Meanwhile, and-introduction was found straightforward when the and-formula already existed, but it was sometimes not clear to the students how to create a particular formula - they were seeking a way to select the two components and then "and" them together (which is not at this time possible using Jape). Conjectures without assumptions seem to be viewed as potentially harder a priori.

Suspicion of the or-elimination rule was widespread, to such a degree that students would often prefer to attempt or-introduction from the "bottom up" rather than attempt or-elimination from the "top down". A typical case of this would occur in a conjecture such as $(P \vee Q) \wedge (Q \vee P)$. Mentally checking using informal meanings for the logical connectors that it would be possible to prove a later line from earlier lines was an uncommon strategy. The negation rules were seen as very difficult. The "undo" feature, however, was universally praised as an excellent way to explore the utility of rules to make progress at a particular point in the proof. A number of strategies for judging such utility were noted, including looking at the size and number of scope boxes, the appearance of new propositions (labelled with an underscore) or variables, and the closeness of lines to what is desired.

With a mid-term test approaching, nearly all students reported that they would be "learning the rules" from the lecture notes in order to prepare for the test, rather than using Jape. There is much data to analyse from these sessions, but it is possible to hazard a guess as to why the students would be using lecture notes at this stage to address the knowledge gaps that they have perceived rather than Jape. The main perceived advantage of Jape, as expressed by several students, was that it allowed experimentation in order to work out proof strategies. The two main disadvantages appeared to them to be firstly that it was often difficult to work out how to apply a given rule to given formulae, and secondly that it was not easy to work out what the "difficult" rules did, or how or when they might be useful. This latter point appeared corroborated after later work in which progress with conjectures involving quantifiers appeared slow in the early stages. A number of students commented that it might very well be useful to be able to check using the computer a proof already constructed without the computer; and that is was certainly very useful to be able to explore using the computer the range of possibilities when an impasse has been reached in a proof being constructed without the computer; but that it was difficult to use Jape without at least some sort of "grasp" of the rules.

The Role of Visualisation

In addition to exploring the above issues and hypotheses, the analysis of the data also needs to consider different aspects of the role of visualisation. Three uses of the term "visualisation" are to be distinguished for the purposes of this research. Firstly, in the colloquial sense, simply using the screen record of proof steps may act as a "visual" support in constructing a proof while using the software. Nevertheless, it may be the case - and this clearly needs to be investigated further - that students are able to make progress with Jape without a good understanding of the principles involved in proving a conjecture. They may be supported in proof construction using Jape, and even learn that only certain constructions are appropriate in certain situations; but it is possible that they learn little more than this.

Secondly therefore, the intention of the software is clearly to promote proof-generation skills that are transferable to situations where the computer is not available. In this sense, students might be supported in the cognitive task of learning to visualise a proof strategy. In the same sense, it might also be that students in such situations may, as a by-product of work with the software, get better at the execution of known syntactical rules applied to strings - a relatively mundane but crucial chore that was handled by the computer when the student had the software available. Of rather more interest would be if, when they came to planning software, students were somehow to draw on their experiences of studying logic. For example, they might make implicit use of notions of "rigour" or "completeness" or "logical structure". Again, students would somehow have been supported in the task of learning a skill - this support could be characterised as the development of strategic theories (Aczel, 1998) which may or may not be image-based in character.

Thirdly, separate from the encouragement of any specific strategies for manipulating expressions, constructing proofs, or planning software, students' perception of the structure and nature of expressions, proofs and programs may be affected, "perception" in this context being understood as a form of "visualisation". Any effect which takes place as a result of this enhanced perception may be indicated not just by the use of more efficient heuristics but also by the language used to talk about expressions, proofs and programs, and the recognition of the more significant aspects of particular expressions, proofs and programs for different purposes. Efficiency itself may be explicitly recognised as a relevant aspect of proof-generation. Do, for example, students question whether there is more than one way to prove a conjecture, let alone whether there are more efficient methods? Of the three senses of visualisation considered here, this third appears to be the most difficult for which to collect evidence.

References

Aczel, J. C. (1998) "Learning algebraic strategies using a computerised balance model", *Proceedings of the 22nd International Conference for the Psychology of Mathematics Education*, Stellenbosch, South Africa

Barwise, J. & Etchemendy, J. (1992) *The Language of First-Order Logic*, Cambridge University Press

Bornat, R. & Sufrin, B. (1996) "Animating the formal proof at the surface: the Jape proof calculator", Technical Report, Department of Computer Science, Queen Mary & Westfield College, University of London, ftp://ftp.dcs.qmw.ac.uk/jape/papers/

Cox, R. & Brna, P. (1995) "Supporting the use of external representations in problem solving: the need for flexible learning environments", *Journal of Artificial Intelligence in Education*, 6 (2), p. 293-302

Cox, R., Stenning, K. & Oberlander, J. (1995) "Graphical effects in learning logic: reasoning, representation and individual differences", *Proceedings of the 16th annual meeting of the Cognitive Science Society*, Atlanta, Georgia, p. 237-242

Fung, P. & O'Shea, T. (1992) "Using software tools to learn formal reasoning: a first assessment", CITE Report No. 168, Open University

Fung, P., O'Shea, T., Goldson, D., Reeves, S. & Bornat, R. (1993) "Computer science students' perceptions of learning formal reasoning methods", *International Journal of Mathematical Education in Science and Technology*, 24 (5), 749-760

Fung, P., O'Shea, T., Goldson, D., Reeves, S. & Bornat, R. (1994) "Why computer science students find formal reasoning frightening", *Journal of Computer Assisted Learning*, 10, 240-250

Fung, P., O'Shea, T., Goldson, D., Reeves, S. & Bornat, R. (1996) "Computer tools to teach formal reasoning", *Computers in Education*, 27 (1), 59-69

A Unix/Linux version of Jape is available from http://www.comlab.ox.ac.uk/oucl/users/bernard.sufrin/jape.shtml

An Apple Macintosh version of Jape is available from http://www.dcs.qmw.ac.uk/~richard/jape