TESTING TOOLS OF REASONING. MECHANISMS AND PROCEDURES

Bertil Rolf
Blekinge Institute of Technology, Ronneby, Sweden
School of Management
bertil.rolf@bth.se

How to test reasoning software?

One would expect it to be easy to show beneficial educational effects of computerized reasoning tools in courses teaching reasoning and decision making. As we shall see, the matter is complicated.

Here, I will restrict myself to software support for teaching elementary reasoning skills. There are half a dozen workable software packages of this kind, e.g. Araucaria, Athena, Belvedere and Rationale (formerly Reason!Able).[1]

Common to such general purpose reasoning tools is a graphical interface enabling users to build a kind of tree-like graphs, containing nodes representing premises and conclusion and edges between nodes, representing logical relations. Such reasoning tools support judgment formation and decision making by externalizing and visualizing inner mental processes, enabling stepwise, openly inspectable procedures.

Prima facie, we would expect that such software would be beneficial for enhancing students' logical capacity. The software packages typically encode the graphical tools used in various analytic treatises and textbooks. They put teachers' explanatory tools in the hands of students. If the graphs used by textbook authors and argument teachers have any educational effects, we would expect even greater educational effects if those concepts in software form are put in the hands of student users.

However, in the debate about effects with and without reasoning software, conflicting or problematic claims are made. Cheikes et al. test Toulmin based argumentation structures without software and find effect claims problematic.[2] Van Gelder and Hitchcock claim to have found software effects.[3] Baker et al. have found none.[4] Suthers attempts to link particular software features to particular effects.[5] Braak et al. have analyzed effect claims related to four different software packages. They reject the studies as inconclusive for lack of experimental rigor.[6]

This paper claims that there are two kinds of empirical testing: intercontextual and intracontextual, depending on the intended scope of inductive generalization. Intercontextual tests would need to correct for mechanisms contributing to effects. Such mechanisms are today largely unknown. Intracontextual testing is far safer. But are not its lessons tied to a specific context? It will be shown that, to some extent, its lessons generalize beyond its context.

Intercontextual and intracontextual tests of reasoning software

There are two kinds of inductive testing strategy. One kind of testing strategy aims for intercontextual conclusions, i.e. extrapolations of patterns from one educational context to others. Another testing strategy aims for intracontextual conclusions, i.e. conclusions about which effects are produced by which mechanisms in a fixed educational context or contexts similar to it.

In intracontextual testing and generalization, the idea is to fixate a package of educational mechanisms pertaining to the educational frame, the type of students, their level, program and class, the teaching method, teacher-student communications and student's tasks. Normally, this package of mechanisms is not explicitly identified or disentangled. The

inductive test and inference is valid in relation to the package of educational mechanisms of the context at hand, but perhaps not beyond that.

The two types of induction can be used as complements. For instance, intercontextual induction may assure the teacher that, in principle, the software should work in that course. But s/he would still need intracontextual induction to accommodate the use of software so as to maximize the effects of the package of mechanisms. For most teachers, intracontextual induction can help solve educational problems connected with a certain type of course.

Intercontextual testing – the state of the art

Let us consider the state of the art of intercontextual testing. In a meta-analysis of four attempts to study effects, van den Braak et al evaluate the results of testing four software packages: Belvedere, Convince Me, Questmap and Reason!Able. [7]

The test procedures are evaluated with respect to internal validity and external validity. The "internal validity" of a test is related to the possibility of the test to establish causal effects in the population tested. By the "external validity" of a test, one refers to the generalization of the effects in the tested population to populations not tested. The authors supply a number of criteria for these types of validity:

> Internal validity: at least one control group, random assignments of participants and homogenization of population
> External validity: draw a random sample from a population, use real world setting and stimuli, and replicate the experiment.

These criteria then form the basis of the evaluation of the tests of the software packages.

There are problems about these criteria. First, one cannot, of course, draw a random sample from a population, part of which is in the future. Second, the authors note that present unreliable measurements destroy validity of experimental results. Without reliable measurement of argumentative skills, none of the four tested packages can claim validity.

According to these criteria of validity, there are no significant results. There are no significant results about positive effects of reasoning supporting software. The authors recommend design of future tests so as to be both internally and externally valid. Objective measures for the effectiveness of tools should be formulated. Finally, a stepwise research plan is proposed.

While it is easy to agree with the authors about the absence of conclusive effects, their diagnosis of the situation is too optimistic.

Intercontextual inductive inference depends on constancy of mechanisms

Inductively strong inferences are collections of non-deductive inferences that confer probability or likelihood on their conclusions, given that the premises in the inference are true.[8] When we use induction, we wish to draw conclusions about a population whose members had no chance of being sampled, e.g. future users of medicines or educational tools.

Induction is sensitive to causal mechanisms. Typically, we are interested in the question whether our reasoning software has larger, beneficial effects on reasoning than our textbooks have. Software and textbooks take effect by partly different mechanisms that we might represent as follows:

> Educational effects of software use: S, M1, M2…Mn, S+M1, S+M2,… S+Mn.
> Educational effects of textbook use: T, Mk, Mk+1…..Mw, T+Mk, T+Mk+1,…T+Mn.

Here, we assume that the M's stand for mechanisms not involving software and S+Mi stands for a mechanism involving both software use and other mechanisms. Normally, it is not software as a whole that produces effects but the use of certain functions in the software

packages. So we should not symbolize software effects as a unit but as a vector: (S1……Sj) with different Si representing software functions. The same holds for textbooks. We can here ignore this complexity.

Mechanisms of software use and textbook use can blend, interact or partially overlap. For instance, there can be interaction effects, S+Mi versus T+Mj where effects cannot be traced to Mi or Mj alone. Probably software effects are sensitive to task factors and contribute more to tasks involving complex reasoning with many premises and several layers of argument. Furthermore, reasoning software can employ educational mechanisms that do not exist or cannot be used without software. Software has many functions that textbooks do not. Many of them are technically trivial – such as undo, redo, cut, copy, paste and save. A standardized workspace facilitates student-student and student-teacher communication in problem solving.

All inductive inference relies on constancy of mechanisms producing the outcome. When we extract properties of one population P1 and generalize them to another population P2, we need to assume that the mechanisms generating effects in P1 occur in the same proportion or exercise the same strength in P2. For instance, say that P1 – unknown to us – contains 90% bacterial infections and 10% viral infections whereas P2 contains 10% bacterial infections and 90% viral infections. Suppose that we randomize patients between experiment and control group when testing treatment of penicillin in P1 and find a significant difference between experiment group and control group. Even so, we should not expect the same difference between those exposed and those unexposed to penicillin to occur in P2.

Inductive generalizations about effects become invalid when effects are produced by partly different mechanisms. Interference of unknown mechanisms that the experimenter did not know and had no chance of controlling may invalidate inductive inferences.

Induction is critical in educational matters. There, we often have few clues about which important mechanisms there are and how they in general contribute to which effects.

Intercontextual testing of tools assumes another kind of causal modeling

The design and selection of a test procedure for intercontextual effects needs to be guided by causal modeling. In testing, causal mechanisms need to be specified and controlled for. Test procedures first systematized by R.A. Fisher relied on a causal modeling of interventions in agriculture. Fisher's kind of modeling was proper to agriculture, where we know little about the causal mechanisms producing effects.[9] The mechanisms controlled for were robust and relatively well known, e.g. slope or moisture. Generalizations and their limitations are fairly clear, e.g. experiments on English soil, climate and microorganisms cannot be inductively generalized to Mediterranean soil, climate and microorganisms or, in general, where outcome is determined by mechanisms not controlled for in the experiments.

Can the causal models used to test effects of manure in the agriculture also be used to test the effectiveness of tools, used in education? I doubt this. Models of cause and effect are more precarious in the cultural and social realm than in agriculture. Causes and effects may not always generalize across contexts.

There are causal differences between adding manure to a field, controlling for slope, sun or moisture and having students use software to solve certain tasks. Manure, slope, sun or moisture are natural kinds, i.e. classes where the effects are determined by natural law. The yield per acre of a field of wheat is the outcome of natural forces measured on a ratio scale.

Educational effects are conventionally delimited and produced by artifacts. Reasoning skills are evaluated according to human convention, somewhat varying from one theorist to another, from one teacher to another or from one professional context to another. We may assume that there are family resemblances between various actors showing reasoning skills. But the classification or ranking of skills is largely a social and cultural artifact.

The causes for producing educational effects are also cultural and social artifacts. In one kind of cultural and social setting, a certain type of teacher-student interaction can produce effects that differ from those that would be produced in another setting. If the students expect and accommodate to one type of tasks and teacher instructions, they will respond in a certain way. But if not, not.

Furthermore, using tools to produce effects is different from bringing about effects by adding manure to a field. When chemicals are added to a field in order to improve on the crop, we assume that the dexterity of the agent adding them plays no causal role. We therefore rightly ascribe causal potency to chemicals, as their effects come about independent of many contextual factors. But we do not ascribe effects to tools in themselves, for the dexterity of the user and factors relating to the aim and the working process play essential causal roles.

Moreover, we do not normally compare, say, the effects of axes versus the effects of saws, controlling for differences in wood, carpenters, aims and work procedures. Axes can be used in indefinitely many ways, laboring with wood and so can saws. In a few cases, their relative effects can be compared. But most often, the user purpose and user process are not quite comparable. Only exceptionally do we ascribe systematic differences in causal potency to the tools themselves.

Finally, in testing educational tools, in particular software based tools, we are testing for what Chomsky called "competence" in distinction to "performance".[10] Competence is the storage of an abstract system of procedures or rules while performance is the output of such a system. Performance is influenced by several other factors such as memory limitations, time pressure, fatigue, lack of processing capacity. Testing for competence merely via actual reasoning performance is an indirect way of approaching the procedures underlying reasoning competence – and a blunt way at that.

Let me sum up these points, applied to education.

- The causal mechanisms connected to educational software and textbooks produce effects by means of social and cultural properties, not by means of robust, natural kinds.
- The causal mechanisms related to the educational software versus textbooks cannot be isolated from other factors, specific for the context. While some of the mechanisms are present in several contexts, the combined effect of their contribution is specific to context.
- There are too many possible educational uses or learning mechanisms related to software versus textbooks. We are interested in comparing them across a large spectrum of tasks, not merely with respect to few, standardized tasks where their relative effects can be compared.
- Education aims for reasoning competence. Competence is not stored in a black box but we have many ways to poke into it. A teacher, designer or tester can rely on information about what has gone into the building of competence. The output of competence can be studied in many different ways.

Given facts about induction and causal modeling, there is very little hope for intercontextual testing. Intracontextual testing is not affected by these objections. Several of the software tests actually performed can better be interpreted as intracontextual, see for instance some tests of Rationale/Reason!Able. When one looks more closely into effect studies, they seem to intracontextual rather than intercontextual.[11] But intracontextual testing brings in other kinds of problems.

Does intracontextual testing make sense across contexts?

In a previous paper, I have shown how Bayesian induction enables intracontextual tests. One can infer the existence of effects from the particular way a software package is used within an

educational context.[12] It may seem, however, that such tests only admit conclusions relative to that context. This would be an error. I will show why, using two examples.

First, we set as target competence the mastery of procedures enabling branch-following oral argumentation. By "branch-following", I refer to argument sequences of a certain type:

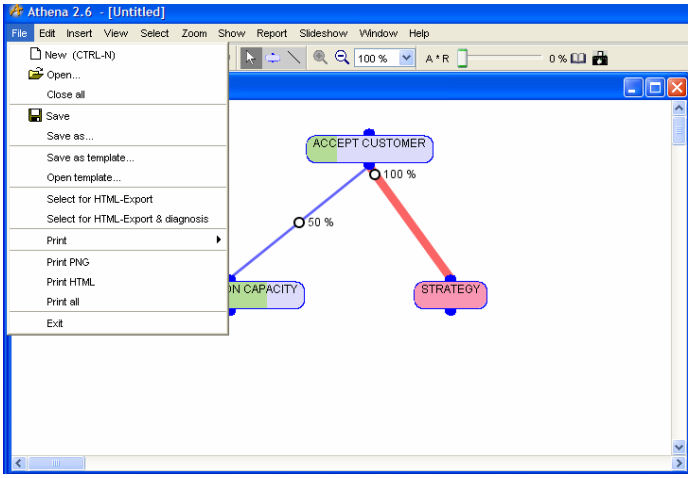
Branch-following oral argumentation	**Non-branch-following oral argumentation**
Proponent: Accept Thesis, because of A. Opponent: Reject A Thesis because of A11 and A12. Proponent: Reject A11 because of A111. Reject A12 because of A121	Proponent: Accept Thesis, because of A. Opponent: Reject Thesis because of B. Proponent: Accept Thesis, because of C.

Branch-following involves pursuing arguments of the second and higher orders related to a topic, before moving to a new, major topic or branch. It is desirable in order not to create cognitive overload in hearers (sometimes also speakers). Argumentation of the other, non-branch-following type is far more common in non-professional contexts. It is possible for a teacher-instructor to design instructions for oral argumentation along with reasoning tools that will improve student capacities to pursue the desired branch-following oral argumentation. Such instructions and software features may not give the desired effect the first time they are tested. Several generations of instructions and software features may need to be designed and tested.

Observe the following. The target competence consists in specific procedures of oral argumentation. They are not likely to be discovered in a standard test on critical thinking. The competence of students is theoretically described. An expert observer can identify the exercise of branch-following procedures. But a lay observer would perhaps notice only the energy of pursuit and the clarity of communication without being able to identify or diagnose which procedures are exercised.

What can be generalized across contexts here? By letting students execute certain externalized procedures, documented in instructions, tasks and software, one can help students internalize procedures of branch-following oral argumentation. Branch-following is a surprisingly complex, social task, demanding metacommunication between actors mutually controlling the interchange. It is a general and informative fact of human learning capacities that complex procedures of cognitive reasoning paired with social competence can be installed via simple externalized procedures.

Another example of intracontextual design and testing is as follows. In a course where Athena was used, the teachers noted that the students did not mobilize all their knowledge to construct arguments. From their previous courses, many arguments could be drawn of relevance to the issue of debate. The teachers therefore designed a new software function and a corresponding intermediate task: to build an argument template. Students given that extra task were found to improve in breadth or argument, i.e. the desired effect was achieved. By first constructing a general template that can be saved and reopened before proceeding to specific applications of the template, students were able to mobilize previous knowledge – see below:



The target competence involved top-down procedures of inventing arguments, tied to a certain case. The target competence could be reached via interposing a template task and using template software features together with instructions.

Here, a feature of the software and a new task are combined to achieve highly specific target competence. The original software functions are extended to serve a new educational function to solve a new educational task. Of course, student learning effort and time spent in education is not identical so a strict comparison of effects of new tools versus old would not be fair.

When we think of software and textbooks as tools to be tested, we often aim for a very specific target competence. The teacher/designer wonders whether such outcome procedures can result from the exercise of such externalized teaching and learning procedures. Experiments will make this clear. The outcome procedures and the teaching procedures normally have details far beyond the precision of standard tests for critical thinking.

The key feature of the causal models underlying such intracontextual testing is that one type of externalized procedures is used to bring forth competence, stored as procedures in human actors. Many procedures are specific to the context, relying on a multitude of educational features. But while the sentences, so to speak, are specific to the context, the words making them up are not. The causal effects are produced by interacting complexes, the elements making them up are transferable to new context and can be put to good use there. The generalizable conclusion is that such tools in the hands of such craftsmen, using such techniques can bring forth new features of reasoning competence.

How to improve on intracontextual testing?

I propose that intracontextual can be strengthened in several ways. A first recommendation for intracontextual testing is that one sticks to a narrow and operationalizable definition of which parts of reasoning competence one would like to improve. These parts can be analyzed into a class of more elementary procedures. Next, one attempts to find a simple way to form a package of tasks, instructions, necessary background knowledge and software features so that the execution of these procedures is facilitated. This is, in principle a complex, causal model of which procedural mechanisms together could produce the desired storage of reasoning procedures. Finally, one tests whether the outcome when these procedures are used is an improvement towards the desired objectives relative to what is normally achieved without the package. There is an implicit reference to the course context where most of the other features are held constant.

A second recommendation is to document the complete packages used to achieve effects. At present, the documentation of testing conditions is faulty. For intercontextual testing, this is disastrous, for intracontextual testing, this is a missed opportunity of learning

across contexts. The learning involves not a generalization of conclusions about effects but about which procedures can bring forth which competence.

A third recommendation is to try to isolate which desired effects on reasoning procedures can be achieved by which combinations of software feature, background knowledge, tasks, instructions and feedback. For instance, it seems likely that argument diagramming can be used to improve students' ability to analyze argument structure. But does software for argument diagramming contribute anything over paper and pencil?[13] A more finely grained approach will enable teachers to maximize desired effects by minimizing the costs of means.

Finally, in testing of software versus textbooks, one needs to be clear about which questions are interesting to ask and promising to try to answer. Intercontextual testing is neither, I believe. The questions asked depend on a peculiar view of mechanisms and functions of tools. The answers cannot be extracted and applied across contexts. The ambition of intercontextual testing is, however, laudable in its aim for generalizable knowledge. I have suggested how to design intracontextual testing in ways that makes aspects of tool use interesting across institutional and educational contexts.

The answers from tests of educational effects do not generalize across contexts. Only the questions, the methods and a stance of conscious, critical use of reasoning tools do.[14]

---

1. For an overview, see Maralee Harrell, "Review Article: Using Argument Diagramming Software in the Classroom," *Teaching Philosophy*, Vol. 28:2 (2005): 163–177 and Cris Reed, Douglas Walton, & Fabrizio Macagno, "Argument diagramming in logic, law and artificial intelligence," *The Knowledge Engineering Review*, Vol. 22, Issue 01 (2007): 87–109.

2. Brant A Cheikes et al. "An empirical evaluation of structured argumentation using the Toulmin argument formalism," Technical Report MTR 04B0000074, (Bedford, MA: MITRE, Center for Integrated Intelligence Systems, 2004). http://www.the-mitre-corporation.net/work/tech_papers/tech_papers_04/04_1032/04_1032.pdf (accessed July 23, 2007).

3. Tim van Gelder, "The efficacy of undergraduate critical thinking courses. A survey in progress 2000," http://www.philosophy.unimelb.edu.au/reason/papers/efficacy.html (accessed July 23, 2007). David Hitchcock, "The effectiveness of computer-assisted instruction in critical thinking," http://www.humanities.mcmaster.ca/~hitchckd/effectiveness.pdf (accessed June, 2005).

4. Michael J. Baker et al., "Designing a computer-supported collaborative learning situation for broadening and deepening understanding of the space of debate," *Proceedings of the Fifth International Conference of the International Society for the Study of Argumentation*, (Amsterdam: Sic Sat Publications, 2002): 55–62.

5. Daniel D. Suthers, "Representational Guidance for Collaborative Learning", *Artificial Intelligence in Education. 11th International Conference on Artificial Intelligence in Education*, ed. Heinz U. Hoppe et al. (Amsterdam: IOS Press, 2003), 3–10.

6. Susan van den Braak et. al., "A critical review of argument visualization tools: Do users become better reasoners?," *Workshop Notes of the ECAI-2006 Workshop on Computational Models of Natural Argument (CMNA VI)*, ed. Floriana Grasso et al., (Riva del Garda, Italy, 2006): 67–75.

7. See previous note.

8. Brian Skyrms, *Choice & Chance: An Introduction to Inductive Logic* (Belmont: Wadsworth, 2000), 17.

9. Gerd Gigerenzer et al., *The Empire of Chance: How Probability Changed Science and Everyday Life* (Cambridge: Cambridge University Press, 1989), 3.2.

10. Noam Chomsky, *Aspects of the Theory of Syntax* (Cambridge: The MIT Press, 1965), §1.

11. See for instance Charles R. Twardy "Argument maps improve critical thinking." *Teaching Philosophy*, Vol. 27:2 (2004): 95–116.

12. Bertil Rolf, "Testing reasoning software: a Bayesian way", *Selected Papers from the European Computing and Philosophy Conference ECAP 2005*, ed. Gordana Dodig-Crnkovic and Susan Stuart, Vol. 4: 2, (2005): 328–332.

13. A question raised by Maralee Harrell "Using Argument Diagrams to Teach Critical Thinking Skills", draft, http://www.hss.cmu.edu/philosophy/faculty-harrell.php (accessed june 2007).

14. This work has been granted support from the Swedish Environmental Protection Agency. A previous version has benefited from comments and questions by Anders Törnqvist and Marvin Croy.