XMOISE : A Logical Spreadsheet to Elicit Didactic Knowledge

V. Bersagol, J-L. Dessalles, F. Kaplan, J-C. Marze, S. Picault
Ecole Nationale Supérieure des Télécommunications - PARIS - France
E-✉ : dessalles@enst.fr

Knowledge elicitation is a critical problem in computerized learning environments that make use of a knowledge base. Fortunately, contrary to usual expertise elicitation situations, didactic scientific knowledge is quite often well formalized, and authors are used to deal with the logical organization of the domain they teach. We want to propose here an original tool, *a logical spreadsheet* which, if included in an authoring package, will help authors organize concepts and at the same time make both conception and maintenance of didactic knowledge bases much easier.

Keywords : Authoring tool - Self-elicitation - Spreadsheet

The Need for new Kinds of Interface for the Author

In computerized learning environments, information display and simulation programs play important roles. However, in higher scientific education, as soon as we want the machine to be able to generate relevant explanations or to make appropriate error detection and repair, we have to provide it with explicit knowledge. The machine is then capable to perform a much wider range of didactic actions, as shown for instance in [Collins 1976 ; Johnson & Soloway 1987 ; Dessalles 1991, 1995 ; Rätz & Lusti 1992]. Unfortunately, such explicit knowledge is generally complex, it is not directly available and is difficult to elicit.

In previous work we explored the possibility of reducing the knowledge given by the author to a strict logical description of the concepts, leaving the machine in charge of all local didactic initiatives [Dessalles 1991]. The problem is then to provide such a system with this minimal logical knowledge. The logical spreadsheet is an attempt in this direction.

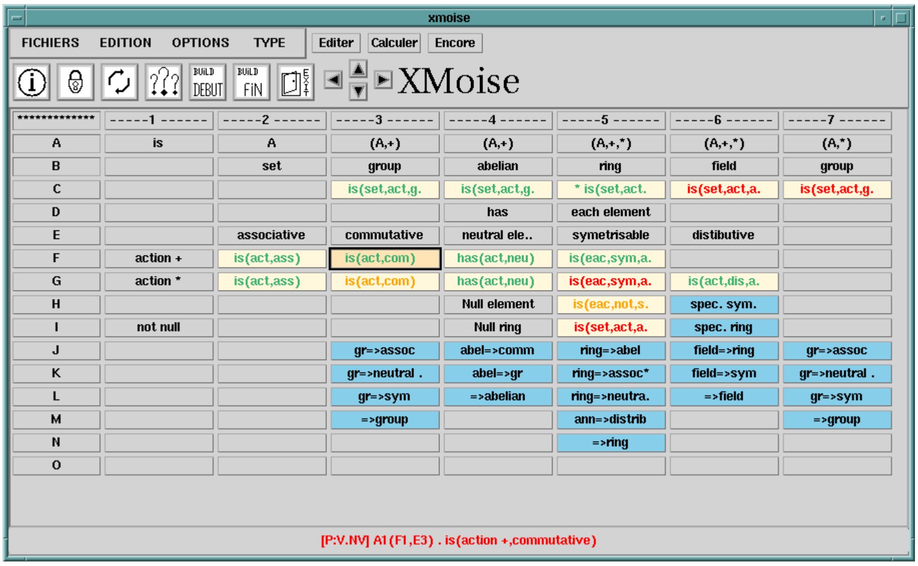
The Concept of Logical Spreadsheet

Usual spreadsheets are devoted to numerical calculus. The basic idea is to use a two-dimensional space in which the user can freely place numbers, formulas and commentaries as if he were using a paper sheet. The main added value lies in the real-time update of numbers defined by formulas. But we are dealing here with concepts and logical formulas, not with numbers. A conventional spreadsheet is thus of no use for our purpose. However, we may preserve both ideas of spatial organization and continuous computation by conceiving a logical spreadsheet in which cells contain logical predicates or logical rules.

The figure shows an example of logical spreadsheet designed with X-Moise. This example is about basic algebra. Cells may contain different types of items :

- a comment (*e.g.*, « (A,+) » in cell *A3*)
- a lexical item : subject, verb or complement (*e.g.*, « associative » in cell *E2*)
- a predicate, as a reference to cells containing appropriate lexical items (*e.g. F3*)
- a logical rule, as a reference to cells containing predicates (*e.g.*, cell *J4*)

In the figure, the current cell is *F3*. It is a predicate cell, making reference to lexical items located in *A1*, *F1* and *E3* (see at the bottom of the screen).



Any first-order logic knowledge base (without functional symbols) can be represented using X-Moise. Rules are expressed as negative clauses, *i.e.*, incompatible sets of predicates. For instance, the definition « *A group with a commutative operator is abelian* » will be expressed by the incompatible set : {(G,+) is a group ; (G,+) is not abelian ; + is commutative }. This rule is stored in cell *J4* in the above example. The label « abel⇒comm » is just a mnemonic. The deep content of this cell is the formula [+C3-C4+F3] which means that the predicate located in C3 (*G with + is a group*), the negation of the predicate of C4 (*G with + is not an abelian group*) and the predicate of F3 (*action + is commutative*) are together incompatible.

How does a Logical Spreadsheet Function ?

At specific times (or each time the content of a cell is modified by the user), X-Moise is able to compute truth values or to detect inconsistencies. The user may specify truth values for some predicates (true or false) and leave the others as « unknown ». Since each logical rule can be understood as a constraint on the set of possibilities, there may be few or even zero possibilities left. The user can visualize all the possibilities the system could find using the « Encore » (Next) button. Each possibility (valuation in proposition logic) is displayed by different colors. Truth values given by the user and by the system appear with different brightness. When unexpected situations are displayed, the user may become aware that he forgot to mention a rule. When developing the above example, the rule *(G,+) abelian ⇒ (G,+) group* had been forgotten in the first place. This became rapidly apparent when testing possibilities.

The most interesting situation is perhaps when there is no possibility at all, *i.e.*, when the logical system defined by the user is overconstrained. In such a case, one of the rules responsible for the blocking is highlighted. X-Moise may then help the user make the situation consistent by suggesting some change in the truth values initially assigned. The mere fact of highlighting a blocking rule is a real help : the rule may be false. If not, X-Moise can invert one term of the rule, revealing another blocking rule, and so on until the user understands why the situation is inconsistent.

Why Use a Logical Spreadsheet ?

Besides this dynamic help in the elaboration of a knowledge base, the use of a spreadsheet offers several advantages. First, it imposes no *a priori* order, neither among rules nor among predicates in a rule. Since the logical formalism used is totally declarative, such an order would be artificial. The user is free to organize rules and predicates in the two-dimensional space, without caring about any sequence of action. This lack of order is even more relevant here than in a conventional spreadsheet where computations often must follow a specific sequence.

All cells are *a priori* equivalent. The user is thus free to organize his sheet as he wants, either grouping rules together, or grouping subsets of knowledge. In our example, there is a growing complexity from left to right, going from group to ring and then to field algebric structures. Rules were grouped in the lower part of the sheet.

A further argument to justify the use of a spreadsheet is that authors are used to such interfaces. The editing facilities are especially useful when different, but similar, predicates or rules have to be introduced.

Perspectives

We want to develop this tool further before experimenting with it as a component in an authoring environment. From a logical point of view, X-Moise allows to express first-order logic, but it performs calculations on propositional logic only. This extension will be done, but it needs some care to avoid infinite loops. As far as the spreadsheet is concerned, we want to explore the relevance of usual operations like translation, transposition, relative vs. absolute cell reference, etc.

We do not know yet if X-Moise will prove to be ergonomic and natural enough to be spontaneously used by authors. We may think of self-elicitation tools that leave the author unaware that he is manipulating logical rules. However, in a higher scientific education context, it seems relevant to offer a tool which stays halfway between a purely verbal elicitation and a direct manipulation of logical formalisms.

References

Collins, A. (1976). Processes in Acquiring Knowledge. In Anderson, Spiro & Montague, *Schooling and Acquisition of Knowledge*. Erlbaum Assoc..

Dessalles, J-L. (1991). Conversation Assisted Learning: The SAVANT3 Dialog Module. In E.N. Forte, *Proceedings of Calisce'91*. Lausanne : Presses Polytechniques et Universitaires Romandes, 159-165.

Dessalles, J-L. (1995). Generation of relevant didactic explanations by the computer running a simulation for itself. In D. Donoval, *Proceedings of CAEE'95*. Bratislava : Slovak Technical University, 218-225.

Johnson, W. L. & Soloway, E. (1987). PROUST: An Automatic Debugger for Pascal Programs. In G.P. Kearsley, *Artificial Intelligence & Instruction*. Menlo Park, USA : Addison-Wesley Publishing Company, 49-67.

Rätz, T. & Lusti, M. (1992). Explanation Strategies: realization in a tutor for database normalization. In P. Brezillon, *Proceedings of the ECAI-92 Workshop on Improving the Use of KBS with explanations*. Paris : Rapp. LAFORIA 92/21.