Computer Assisted Assessment in Elementary Algebra

Denis Bouhineau, Alain Bronner, Hamid Chaachoua, Sophie Mezerette, Jean-François Nicaud

Computer Assisted Assessment in Elementary Algebra

Experiences and points of view from the APLUSIX project

Denis Bouhineau*, Alain Bronner**, Hamid Chaachoua*, Sophie Mezerette* and Jean-François Nicaud*

* Lab. Leibniz-IMAG, Grenoble, France
FirstName.LastName@imag.fr
http://aplusix.imag.fr

** Lab. LIRDEF - IUFM de Montpellier, France
FirstName.LastName@montpellier.iufm.fr

Introduction

Since 2000, we are working on a project for a global computer system dedicated to the teaching and learning of formal elementary algebra at secondary level schools. During a first phase, from 2000 to 2003, most of our work has been employed with the conception and realisation of a microworld for doing actively elementary algebra. This period has lead to the APLUSIX environment [Nicaud 2004], a tool dedicated to the first activities concerning algebra (resolution of equations, factorisation and expansion of polynomials). It was principally a piece of software devoted to students, based on the concept of microworld, which means that a set of objects and relations were defined internally to represent algebraic expressions and represented externally in the most natural way, allowing activities in elementary algebra with direct-manipulation and intuitive advanced edition. Main principles used to develop this environment were, first, to let students freely and easily build and transform algebraic expressions, as they can do on paper, and second, to give permanent but not intrusive epistemic feedback on the syntactic and semantic correction of the reasoning followed by the students. Experiments conducted in classes from K7 to K11 have shown that learning occurs with the use of APLUSIX microworld [Nicaud 2005]. Screen copy of the environment is shown Fig. 1.

During this first phase we concentrate on the microworld paradigm and our main concern were for students as our principal users. The second phase introduces two shifts in our preoccupations. First, besides the fact that we tried to continue to improve the microworld aspect of our environment, we have worked to nested it into some kind of exerciser where activities could be automatically generated, solutions could be automatically find out and work done by students automatically analysed and scored. Second, we have worked specifically for teachers, trying to provide them with tools for managing and improving or facilitating their work, for example, tools for the administration of classes and students, tools for the edition of specific exercises, list of exercises, or richer exercises (for word problem, or problems with many linked sections), tools for statistical analysis of student's results. Many tools linked or not to the microworld have been developed during this phase; many are linked to problematic belonging to the Computer Assisted Assessment domain.

In the following sections we will focus on our work close to the Computer Assisted Assessment domain. We will deal with the followings subjects: automatic generation of

exercises and list of exercises, definition of possible different student's activities related to CAA, automatic production and verification of solution, scoring and statistical analysis of resolution. Notice that those subjects are almost independent one from the other. We will also try to gain some perspective from the last 5 years applied to the development of our system to find some principles according to the evolution we have observed from the initial definition of the previously listed elements to the current state of these elements and we will try to find the main final characteristic of these elements.

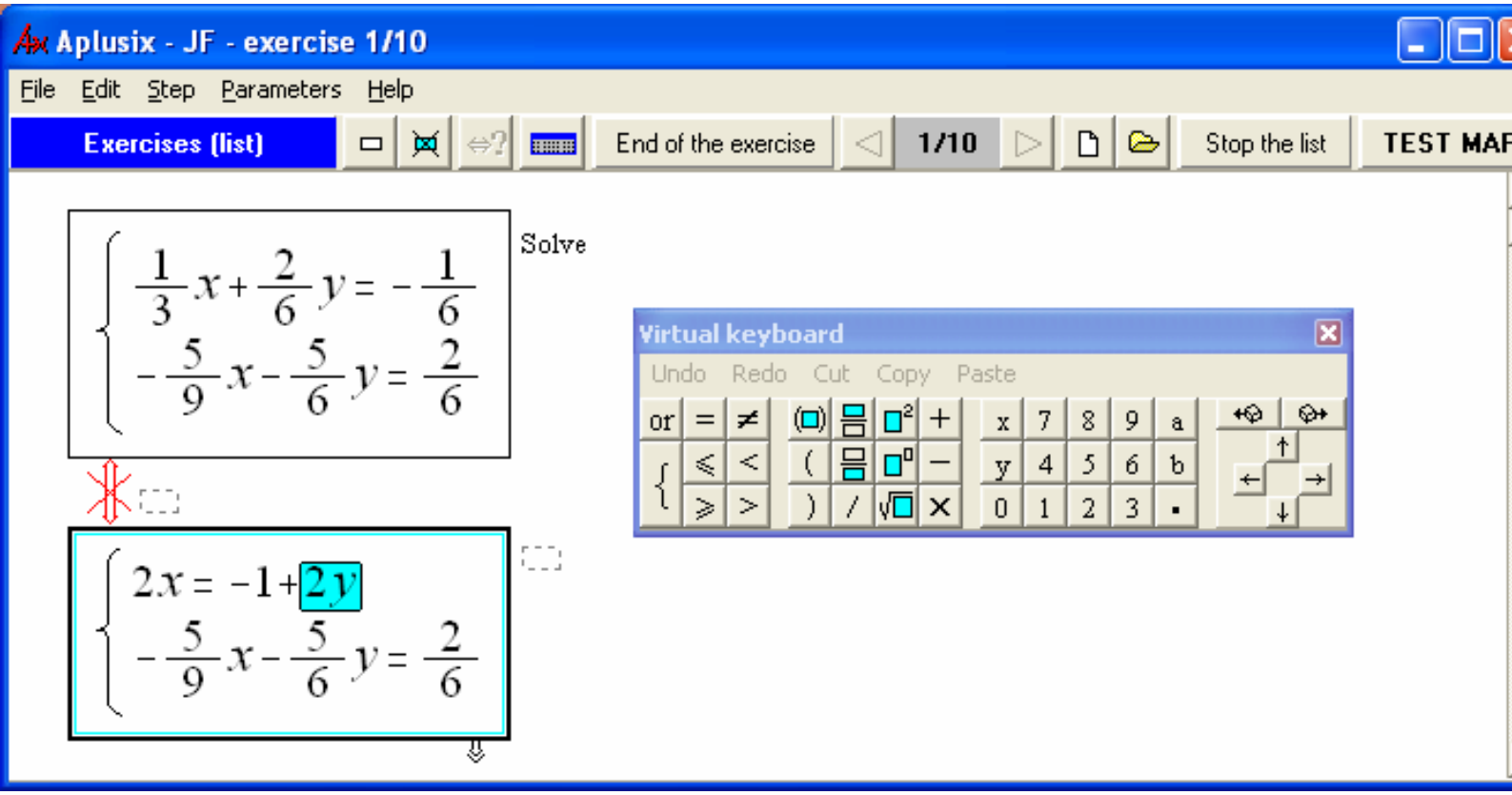

Figure 1. The APLUSIX microworld.

Delivering exercises

When teacher define and organise exercises

The first step in transforming a microworld into an exerciser could be to provide exercises to students. The simplest way consists: first to ask teachers to record exercises in some structure and then to arrange theses exercises in lists. Second, functionality had to be added to the microworld which permits students to work on the list. It was our first realisation and was used for our first experiments.

We work then on a second version with complex scenario combining list of exercises with general system parameters for the microworld (important parameters including didactical parameters) to be merged with student's and classes general parameters for the microworld, and other parameters for the execution of the scenario (time management, order of the exercise, message set of exercises, temporal condition for executing the next series of exercises). It allows us more complex experiments, but was abandoned latter, because of its complexity. Apart from academic experiment, the complexity of that tool was too costly to be used and fit not well with teacher's actual needs.

We came back to a simpler editor of list of exercises, just a bit more elaborated than our first realisation. But we have drawn from that failure three lessons: first not to conceive too complex tools and situations, second two principal situations are to be considered, training situation and test situation, third some system parameters can be set without loss of generality, some can be ignored by the system, masked to teachers and let to the only use of the students. After that episode we continue to develop the exercise editor for teachers such that richer kind of exercise could be proposed: problems with several sections and word problems.

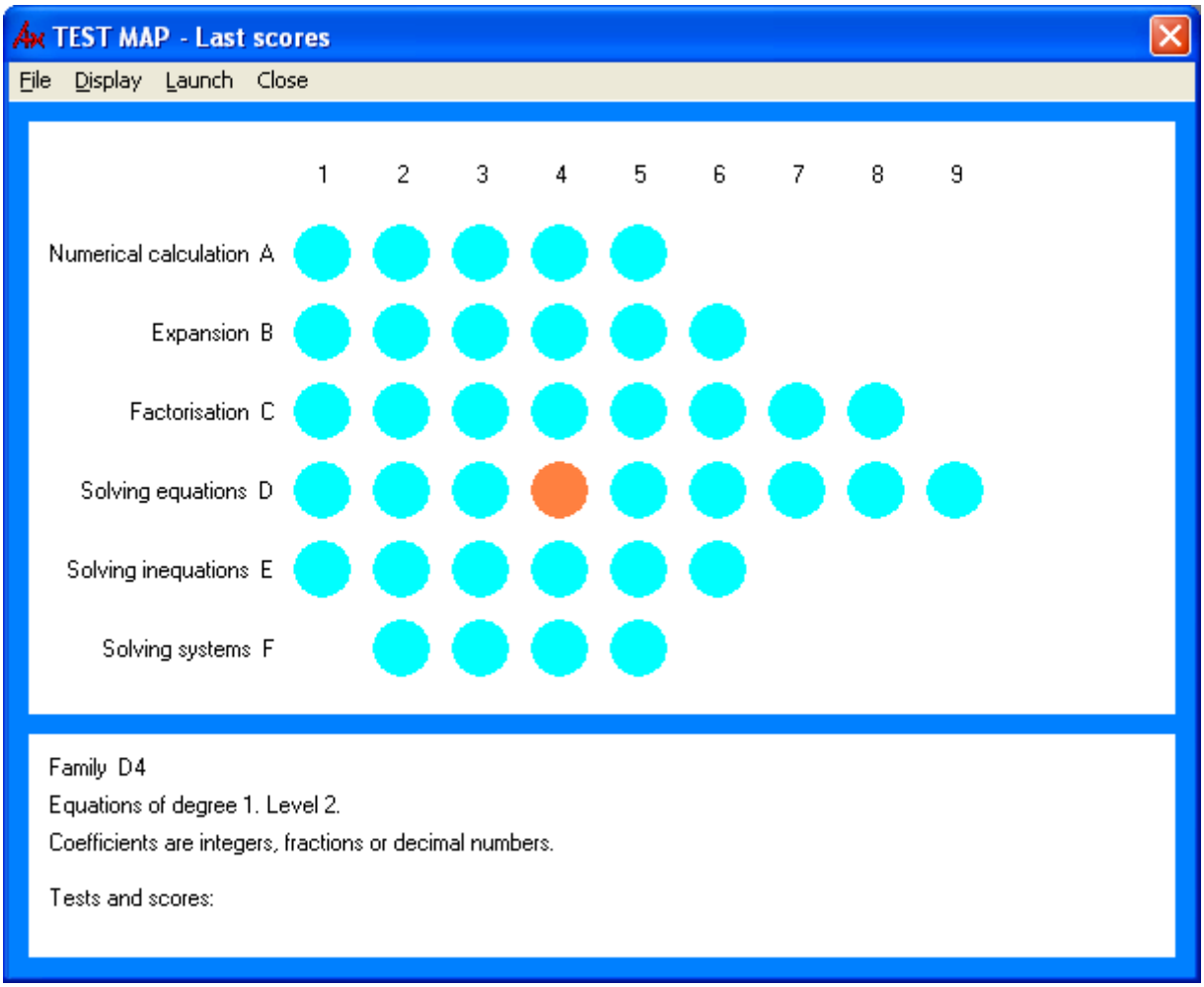

Figure 2. The Map of exercises or test map presents the families of exercises as dots organised in a plane. Each horizontal line corresponds to an exercise type, with difficulty increasing from left to right.

When computer defines and organises exercises

At the same time we began to work on automatic generation of exercises. The goal was twofold, first to have a "ready to use" map of exercises allowing students to work without the need of teachers and second to facilitate the preparation of exercises lists for the teachers. The result is a set of about 40 families of exercises organized by type of exercise and level of difficulty, for example, first degree equations with fractions and square roots. Each family contains about 10 patterns of exercises having variables for the numbers and constraints. The exercises are generated from patterns by random choices of the numbers and verification of the constraints.

The families of exercises and the patterns of exercises used in the map of exercises have been defined by two didacticians in mathematics, with who we have worked to define an ad hoc

simple language to design families of exercises and patterns of exercises. Patterns belong to a hierarchic organisation with inheritance. Each pattern is an object with a dozen of attributes; most of these attributes are defined by inheritance, ex: mean time to solve the exercise, exercise type, hints to describe the pattern or general family of pattern, pseudo-frequency to set the number of apparition of the pattern with respect to other pseudo-frequency of the other patterns of the same family. The main other attributes are the pattern itself, the domain for the value of parameters and constraint to be tested out to validate an instantiated exercise.
Parameters are to be chosen in their domain, but it is well known that "parameters produced must be realistic, in that the questions should be neither overly tedious nor trivial" excerpt from [Griffin 2004]. Most of the time, or even always as far as we know, it is the responsibility of the producer to avoid wrong value for the parameters, and it means that developing new questions can involve significant programming effort and tricky formulation. We have chosen another solution to avoid such incorrect instantiated exercises; we have introduced constraint to be respected when patterns are instantiated. As a consequence the instantiation mechanism is a non trivial randomised choice where constraints are to be verified to validate an exercise, and which much deals with problems when not too many instantiations are possible or other problem resulting of the rejection of instantiated exercises. The introduction of constrains to be checked is an innovative aspect of our simple language. Compared to patterns defined for Wims [Perrin-Riou 2004], MacQTeX [Griffin 2004] and others, our approach permit to have a rather efficient, powerful and expressive language.
Obtaining all those patterns (more than 400) has been a fairly nice result, considering that providers were two didacticians smart with mathematics but with almost no skills in computer science. Our goal was to find a way to have didacticians working with computer within a framework not too complicated, but nevertheless powerful enough so that they can work easily. Complex framework like those introduced by [IMS-QTI 2004] and MathML or OpenMath where avoided, even syntax relying on TeX language or similar languages were rejected. We did not believed that bringing the complexity of computer science in the hand of non-specialist would be a success.
Once the exercises have been obtained, algebraic pretty-print process are executed to suppress neutral elements from algebraic expressions, for example (-1)3x=1*3 is transformed into –3x=3 and finally exercises are locally randomly rearranged in the list, and associated such that the global duration (sum of the exercises durations) is close to an arbitrary duration of 30 minutes.

```
{[name                                                FactorDistSTD23Z]
 [KindOf                                                 FactorDistSTD]
 [pattern                                                  <<ax^2+bx>>]
 [domain  ((c integer+ small)(d integer+ small)(e integer* small))]
 [with                    ((<> c 1) (:= a (* d c)) (:= b (* e c)))]}
```

Figure 3. Example of a pattern: x remains abstract variable, a and b are instantiated with products d.c and e.c where c and d are small positive integers, e is a non-null, non-unitary integer. For example, that pattern can produce the following expression $18x^2+30x$, 12x-9x and $14x^2+7x$.

Possible activities related to CAA

At the beginning, recall that our system was a microworld whose goal was to help students solve exercises and problems in elementary algebra. Students could perform their own calculations by typing in expressions and making the steps they want, as easily as on paper. Students were supposed to work by equivalent steps. APLUSIX providing the fundamental feedbacks that indicates whether the calculations are correct or not (equivalent to the initial expression), and at the end, for the final step, whether the exercises are correctly solved or not

with respect to one of the following predefined type of exercise: Calculate (for numerical calculations), Expand, Factor or Solve (equation, inequation or system of equations). The shift from microworld to exerciser has been possible because of the existence of the reduced set of problem type which concerns almost all problems of elementary algebra at secondary school. Apart from these simple exercises, there are problems composed of sections providing long text full of information and asking questions. Answers are algebraic expressions that are compared to the one given by the designer of the problem. For example, a problem can have a text in natural language with a first question asking to input an equation and a second question asking to solve this equation.

In the last versions of our system, we organise student's works within 5 activities: microworld activity, exercises activity, test activity, self-correction activity and observation activity.

Microworld activity
In the microworld activity students have to input the first expression of their sheet and can change that first expression, whereas in the exercise activity and the test activity the first expression is defined by the exercise in the file or produced by the map of exercises and could not be changed. The first expression on the sheet is important; it defines the exercise and what will be the answer.

Training activity
In both the training and the microworld activities the system provides epistemic feedbacks in order to help students learn the domain: verification of the equivalence verification of the correct end of the exercise. These feedbacks are also a way for the students to self evaluate their skill in algebra all along the resolution of the problems. Additionally while solving an exercise, students can ask for their score at any moment. They can also ask to see the solution. These possibilities are disabled for the test activity.

Test activity
The Test activity provides a 30-minute test on exercises chosen from the map of exercises or from a file (with time limitation possibly redefined in the file) without any feedback on the correctness of student's steps or on the end of an exercise and without score and solution.
At the end of a test, APLUSIX gives a score and provides students with an opportunity to correct their solution using a "Self-correction" activity. This is a new kind of activity which reinforces student's self evaluations and allows students to overcome their mistake.

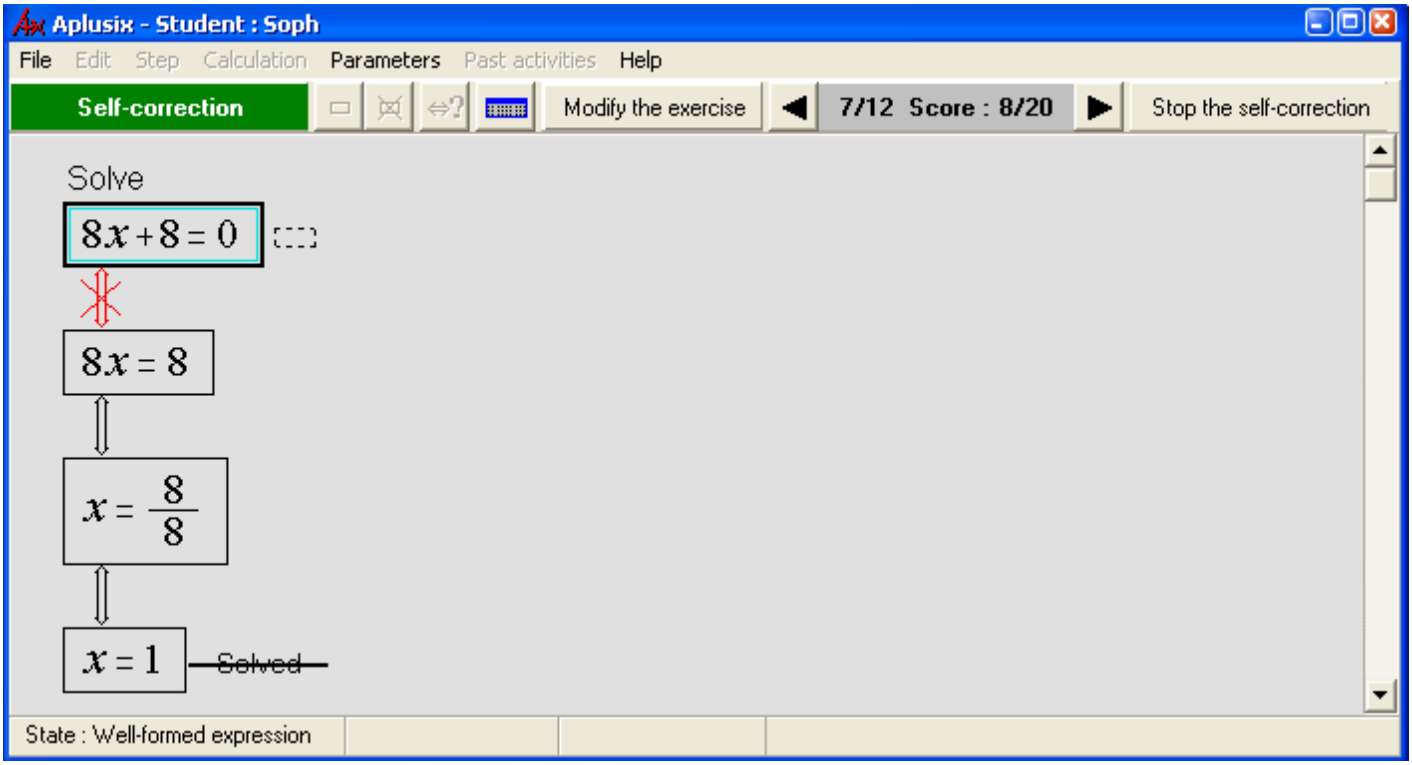

Figure 4. Screen copy of the self-correction activity. The incorrect calculations are marked in red. The word "Solved" is crossed out because there are incorrect calculations.

Self-correction activity

In the self-correction mode (which is available at the end of a test and also by mean of a menu concerning past activities) APLUSIX presents the work made by students in its final form with indications of the correct/incorrect steps, of the correct/incorrect end, and of the score. Students face their errors and then they can modify each exercise to correct their errors with the help of the feedbacks of the training mode (i.e., feedbacks that were not available during the test mode).

Observation activity

The observation mode is a more elaborated version of the visualisation of past activities where students can launch a replay system to see action by action their work. The observation mode was initially intended to teachers and especially didacticians so that they can see exactly what have been done by students, but usage have been reported of teachers working with their students with the replay system to analyse their own work.

Activities versus System Parameters

The organisation of work with APLUSIX according to activities has been a solution to reduce the use of the system parameters. A set of system parameters allows customising the system for each class and situation. Parameters continue to exist and can define (few examples only): the mode and scope of the verification of the equivalence, and how the system must manage an incorrect or ill-formed step; the access to the solutions and to the CAS-like commands, the order of exercises obtained from a list (randomised or not), the introduction of strong invitation to students to comment their steps, and so on.

Activities do not set all the system parameters but the most important ones and reduce the number of those which still need to be set. Because of the complexity of the use of system parameters (set of parameters can be assigned to each class, for each session), there was a real need to find a way to have customised version of the system without big effort. Activities

have been our solution, that transformation has followed the same pattern that our different versions for the production of exercises: trying to make the system easier to use.

About solution

After that exercises have been produced, delivered to students, and worked out by students, solutions have to be checked, and if no solution has been found by students, maybe the system should propose one. Two main situations are to be considered, the first one are for 'simple' exercises produced by teachers or by the map of exercises that is exercise with an initial expression and a well defined type of problem between Calculate, Factor out, Expand, and Solve. These exercises rely only on algebra, and solutions can be computed from the initial expression straightforwardly. As a consequence, for the exercises given by teachers, no solution is asked to teachers. More, in the "Exercises" and "Self-correction" activities, the system can give the solution to students on their requests when system parameter for giving solution is enabled.
The second situation concerns complex problems given by teachers (for example word problems and problems beginning with the finding of an algebraic expression representing the problems). For that kind of situation, teachers must provide the desired solution. Additionally, teachers can specify how solutions given by students will be compared to the desired solution. Three comparisons are possible. First comparison succeeds only for identical expressions. Second comparison succeeds for expressions which are semantically equivalent and syntactically close one to the other. That is, expressions must be equals modulo elimination of neutral element, commutative operators and associative operators. Last, teachers can specify if students can have different names for the variables.

But what for the first situation, the 'simple' exercise case? Use of Computer Algebra System (CAS) is a classic solution [Sangwin 2005] but introduces drawbacks, especially for elementary algebra. In particular, the syntactical part and semantics part of the analysis of solutions are not always distinctly defined; except Mathematica, no CAS allow quantifier elimination which is the most rigorous method for assessing the algebraic semantic equivalence and last of that incomplete list, the use of CAS often introduce disturbing problems with the syntax of inputs asked from students and outputs shown to students. Hence we choose, from the very beginning of our project, to avoid as long as we can the use of CAS. We hoped that elementary algebra was a domain not too complex, and it reveals to be in fact not too complex but not that easy. Below is the complete analyse of reasoning. The analysis is separated in two parts: the syntactic part and the semantic part, each one is important.

When is this solved?
First, the reasoning path leading to the solution must be semantically valid. The reasoning is valid when each step in the reasoning is equivalent to the initial one. APLUSIX verify that two connected steps contain equal numerical expressions, equal algebraic expressions, equivalent equations, equivalent inequations, or equivalent systems of equations. In training mode student can see the results of the verification with the drawn link between steps: a double black line means "The expressions are equal" and with arrows added "The equations, inequations, systems of equations are equivalent", a double blue crossed line means "One expression is achieved or undefined", a double red crossed line "The expressions are not equal" and with arrows added "The equations, inequations, systems of equations are not equivalent". In test mode a single black line is drawn between steps, it means that "No verification is done".
Notice that all the algebraic expressions are considered over the set of real numbers and that our system verifies the calculations in the following ways: for the exercise types Calculate,

Expand, Factor, canonical forms of the expressions are calculated, for the exercise type Solve, canonical forms of the sets of solution of the equations, inequations, systems of equations are calculated. Then equivalence is decided when two canonical forms are identical. The calculations of the canonical forms are approached calculation. The equality of number is judged with 10 digits.

Second, the final expression must be syntactically checked.
An exercise of the type "Calculate" is solved when the expression is a number written in a canonical form (integer, decimal, rational or irrational).
An exercise of the type "Expand", with an expression that does not contain square roots, is solved when the expression has no parentheses (neither explicit nor implicit like a sum in the numerator of a fraction) and is simplified.
An exercise of the type "Expand" with an expression that contains square roots is solved when the expression is a canonical polynomial form or when the expression has no parentheses (neither explicit nor implicit like a sum in the numerator of a fraction) and is simplified.
An exercise of the type "Factor", with an expression representing a polynomial of degree 1, is solved if the constants have been factored.
An exercise of the type "Factor", with an expression representing a polynomial of degree greater than 1, is solved if the expression is a product of simplified prime polynomials (prime polynomials are first-degree polynomials and second degree polynomials without roots. Note that constant factorisation in an expression representing a polynomial of degree greater than 1 are not necessary.
An exercise of the type "Solve", except in the special cases described below, is solved: for an equation, when the expression is "x=a", where a is a simplified writing of a number, or where "x=a1 or x=a2", a1, a2 are simplified writings of distinct numbers; for an inequation, when the expression is "x<a" or "x>a" or "x≤a" or "x≥a", where "a" is a simplified writing of a number; f or a system of equations, when the expression is "x=a1 and y=a2", where a1and a2 are simplified writings of numbers.
The special cases for "Solve" are first when there is no solution. In this case, no particular form is expected and students have to click on the "End of the exercise" button and to choose "No solution". Second, when any number, for each variable, is a solution, no particular form is expected and students have to click on the "End of the exercise" button and to choose "Any number is a solution". Third, when a system of equations with N unknowns leads to less than N equations, but more than zero, it must have a form in which some variables are expressed in function of others, for example, for the system x+y=3 and 2x+2y=6, one can give the expression x=3-y.

Scoring and statistics

Both students and teachers need to have synthetic analyses of the work done. For students, the first basic solution is to provide a score about the work. For teachers, computations about scores obtained by students can be done to give a statistical or global view of the classes. One can imagine some more complex analysis, and we are in fact working on some computerised didactical analysis of the student's works, but it is for the perspectives. Just before the description of our scoring process, we have to mention out that according to the french legislation score automatically computed (without human intervention) from exercises done by students on computer can't be the used to evaluate students.

Scoring

Scores are calculated considering the progress of the reasoning made with correct calculations. When nearly in solved form, with correct calculations, the score is close to the maximum. When a solved form has been given but "Solved" has not been indicated, the score will not be the maximum.
If there are incorrect calculations, APLUSIX considers the situation before the first incorrect calculation. If the score is not high, APLUSIX looks at the calculations after the incorrect one and may increase the score.

Statistics
The Statistics window displays a table and a graphic (histogram or time curve) about selected period, selected activities, selected population with information (total, average, standard deviation) about: number of attempted exercises, number of well-solved exercises, number of calculation errors, scores, time spend. The table can be sorted by clicking on the head of the column to be sorted. This action has repercussions on the histogram. When students are working, the window is updated each 30 seconds so that teachers can view the progress of student's works.

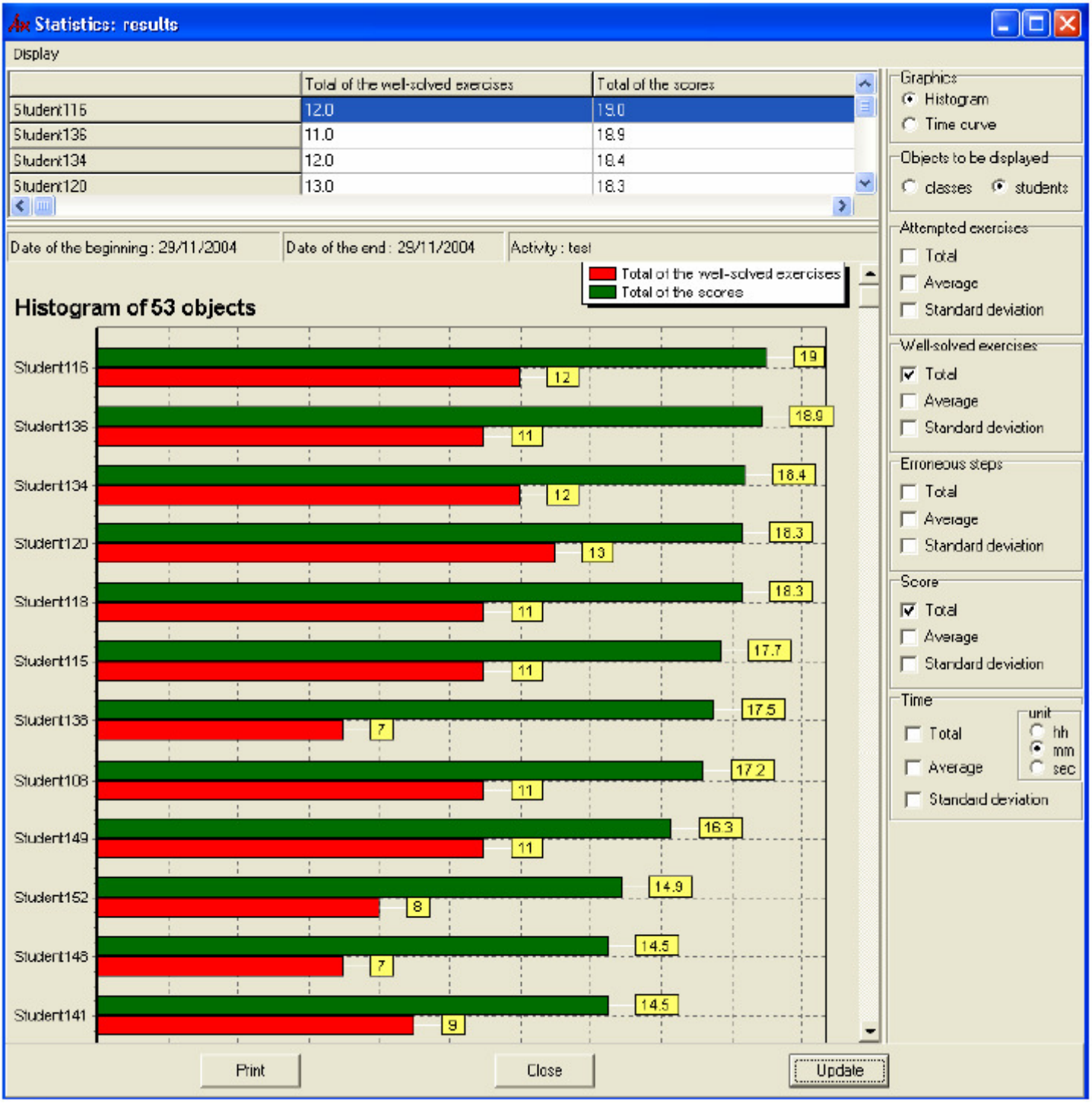

Figure 5. Screen copy of the statistic window.

Conclusion, Result and Perspective

Our choices have been for simplicity. We have tried to simplify our system for students (K7-K11, so that their work deals with mathematics and not with computer science), for teachers (who are not always fond of computer) and for us computer scientist and math didacticians. With the evolution of our system, from the first version which was a microworld to the last version in which the microworld is embedded within an exerciser, we have added components and modify previous component to favour simplicity. The evolution itself can be seen as a search for simplicity. So that teachers can work with our system easily.

As a consequence, we do not have use CAS, nor IMS-LD, nor MathML; we have introduce different activities, some are particularly devoted to CAA; we have forged a map of exercises which can produce exercises for almost 40 families of situation (i.e., for almost all the curricula in elementary algebra); we have added components to score student's productions and to give teachers an overview of the work done in his/her class.
Next step could be to add some Artificial Intelligence components to give students and teachers some higher level information and help. We are already working on a remediation tutor for helping students with difficulties, companion for providing examples of pedagogical solutions to every students and didactical analysis of student's works for teachers which could give them a good, global and understandable high level diagnostic about student's conceptions about elementary algebra.

References

[Griffin 2004], Griffin F. "*MacQTeX Randomised Quiz System for Mathematics*" in Maths CAA Series, LTSN Maths, Stats & OR Network, University of Birmingham, UK, January 2004.

[IMS-QTI 2004] IMS Question and Test Interoperability: "*Item Implementation Guide*" & "*Item Information Model*", V2.0 Editor Steve Lay, University of Cambridge, http://www.imsglobal.org/question/qti_item_v2p0pd written in June 2004

[Nicaud 2004] Nicaud, J.F., Bouhineau, D., and Chaachoua H. "*Mixing microworld and CAS features in building computer systems that help students learn algebra*", in International Journal of Computers for Mathematical Learning, Volume 9, Issue 2, Springer-Verlag, 2004.

[Nicaud 2005] Nicaud J.F., Bittar M., Chaachoua H., Inamdar P., Maffei L. "*Experiments of APLUSIX in four countries*", proceedings of the International Conference on Teaching Mathematics with Technology, ICTMT'7, Bristol, UK, July 2005.

[Perrin-Riou 2004] Perrin-Riou, B. "*Programmation d'exercices OEF (open exercise format)*" http://wims.auto.u-psud.fr/wims/ written in January 2004.

[Sangwin 2005] Sangwin C.J. "*Making Mathematical Distinction in CAA with Computer Algebra*" proceedings of the International Conference on Teaching Mathematics with Technology, ICTMT'7, Bristol, UK, July 2005.