HAL open science

Natural Editing of Algebraic Expressions

Jean-François Nicaud

To cite this version:

Jean-François Nicaud. Natural Editing of Algebraic Expressions. MathUI workshop, 2007, Linz, Austria. online-proceedings. hal-00190029

HAL Id: hal-00190029

https://telearn.hal.science/hal-00190029v1

Submitted on 23 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Natural Editing of Algebraic Expressions

Jean-François Nicaud
LIG, 46 avenue Félix-Viallet,
38031 Grenoble cedex France
Jean-Francois.Nicaud@imag.fr

Abstract
We call "natural editing of algebraic expressions" the editing of algebraic expressions in their natural representation, the one that is used on paper and blackboard. This is an issue we have investigated in the Aplusix project, a project which develops a system aiming at helping students to learn algebra. The paper summarises first the Aplusix project. Second it presents a notion of algebraic expressions, of representations of algebraic expressions. The last section develops ideas about natural editing of algebraic expressions (insertion, deletion, selection, cut, copy, paste, drag and drop) and presents the choices made in the Aplusix system.

1. Introduction

In 2000, our team restarts from scratch the development of a new Interactive Learning Environment (ILE) with two main goals. The first main goal was to create an ILE for algebra allowing the student to freely build and transform algebraic expressions. The environment should provide epistemic feedback that can help in the first stages of the learning of algebra. At that time, there were no ILE for algebra that allowed students to freely build and transform algebraic expressions, as they do on paper, and to follow their own reasoning; the existing ILEs were, and most of them are still, command-based systems that did not allow the student to proceed without applying a command chosen from a menu. See MathXpert, [3] and Cognitive Tutor [7] as good examples of these command-based systems. So we decided to build a system to help students solve exercises in numerical calculations and formal algebra, where they would perform their own calculations by typing the expressions and making the steps they want, as on paper. The system, called APLUSIX, would gives feedbacks, mainly whether the calculations are correct or not, and whether the exercise is solved or not, those feedbacks depending on the semantics of algebraic expressions, and the syntactic form of the expression.
The second main goal was to create an ILE that would be used for real. This goal requires building a usable and useful system. We considered two main points for usefulness: Encompassing a large mathematical domain and having several modes of functioning.

Allowed students to freely build and transform algebraic expressions means building an advanced editor of algebraic expressions. We have developed such an editor from the questions "What do users expect when they do action A?" and not from the question "What is the simplest implementation of action A with the data structure we have?". Of course, this lead immediately to display and edit the expression in their usual two-dimensions representation, but this is just the first step. The other steps consist of having often a structured behaviour (also called algebraic behaviour) but sometimes an unstructured behaviour for flexibility.

This paper shortly describes (section 2) the Aplusix system in its current distributed version, some of its experiments, and a recent module devoted to a tree representation of algebraic expressions. Section 3 develops the notion of algebraic expressions and representations of algebraic expressions. Section 4 is devoted to the editing natural expressions, i.e., algebraic expressions in their usual representation system.

2. Aplusix

2.1. Short description of Aplusix

APLUSIX is organized on the base of two editors, an advanced editor for algebraic expressions, and an editor for algebraic reasoning; these two editors have been constructed as microworlds. They permit numerical calculations and formal algebra activities like expansion, factorisation, and resolution of equations, inequations, and systems of equations with verification of the equivalence of expressions during the reasoning. There is an immediate feedback showing whether two consecutive expressions are equivalent or not. Aplusix also provides solution and score on a sub-domain.

In Aplusix, numbers may be written as integers, decimals, fractions and square roots. Exponents must be integers (positive, negative or zero). The domains of the current version are described in table 1.

Type of exercise	*Domain of Verification of the calculations.*	*Domain of Solutions and Scores.*
Numerical calculation	The expressions must not include variables.	The expressions must not include variables.
Expansion	Polynomial or rational expressions.	Polynomial expressions.
Factorisation	Polynomial expressions of one variable and maximum degree 4 or of two variable and maximum degree 2.	Polynomial expressions of 1 or 2 variables and maximum degree 2.
Solving equations	Polynomial equations of one unknown and maximum degree 4 and rational equations giving such polynomial equations.	Polynomial equations of one unknown with maximum degree 2.
Solving inequations	Polynomial inequations of one unknown and maximum degree 4 and rational inequations giving such polynomial equations.	Polynomial inequations of one unknown and maximum degree 1.
Solving systems of equations	Systems of equations with maximum 10 equations and 10 unknowns.	Systems of equations with 2 equations and 2 unknowns.

Table 1. *Domains of the current version of Aplusix.*

Commands for executing certain algebraic actions are available. These commands can be enabled or not, powerful or not, according to parameters set by the teacher, (they have to be adapted to the current level of understanding of the students in order to only present calculations they can do without difficulty). With this feature, such a computer system can provide an introduction to the proper use of a Computer Algebra System. There are 'Calculate', 'Expand and reduce', 'Factor' and 'Solve' commands. See example in figure 1.

Besides the usual way of using Aplusix, called "training mode", with immediate feedback, we implemented a test mode without immediate feedback. At the end of a test (often 10 exercises), or later, students can enter in a self-correction mode where they see their final answer with a score and with feedback, and can correct their errors with the help of immediate feedback. See figure 2.

Last, the microworlds are embedded in a piece of software whose concern is the practical organisation of the learning process (login of the students, organisation of their work according to some prepared scenario, recording of the activities, scoring and statistical analysis of these activities made available for the teacher, tuning of the software with parameters, automated generation of exercises...). A longer description is given in [14].

$\begin{cases} 2x+\sqrt{2}\,y=4 \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 1) Anna duplicates the given exercise and selects $\sqrt{2}y$	$\begin{cases} 2x=4+\sqrt{2}\,y \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 2) She drags $\sqrt{2}y$ and drops it on the right hand side.	$\begin{cases} 2x=4-\sqrt{2}\,y \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 3) She hits the "minus" key.
$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 4) Anna deletes 2 on the left, changes 4 to 2 on the right and inserts 2 as denominator of $\sqrt{2}y$	$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 5) She selects the value of x and makes a copy in the clipboard.	$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ 3x-2\sqrt{2}\,y=-2 \end{cases}$ 6) Then she selects x in the second equation.
$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ 3\left(2-\dfrac{\sqrt{2}\,y}{2}\right)-2\sqrt{2}\,y=-2 \end{cases}$ 7) And makes a paste that produces a substitution (the parentheses are added by the system).	$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ 3\left(2-\dfrac{\sqrt{2}\,y}{2}\right)-2\sqrt{2}\,y=-2 \end{cases}$ 8) Then she selects the left hand side of the second equation.	$\begin{cases} x=2-\dfrac{\sqrt{2}\,y}{2} \\ -\dfrac{7\sqrt{2}}{2}y+6=-2 \end{cases}$ 9) And she applies the "expand and reduce" command using the pop-up menu.

Figure 1. *How to solve a system of equations using* drag&drop *and the command "Expand and reduce".*

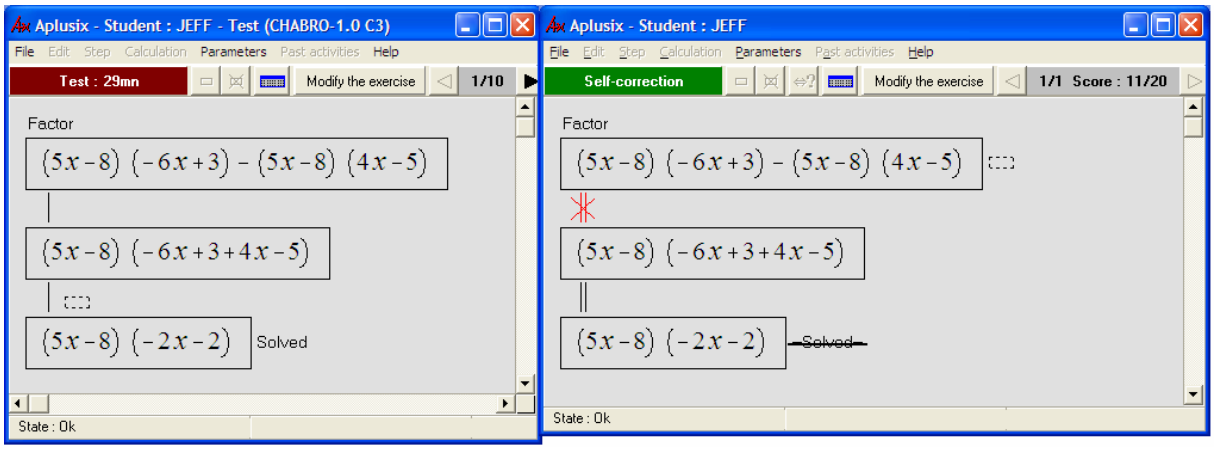


Figure 2. *On the left, Jeff has solved an exercise in the test mode, without feedback. On the right, he is looking at the result in the Self-correction mode. He can see his score (11/20), an incorrect calculation and a stricken "Solved". He can click on "Modify the exercises and correct his error.*

2.2. Experimentation of Aplusix

Many experiments have been conducted since 2002 in France, prepared by the researchers in mathematics education of our team. Most of them were carried out in the usual functioning of the class, using the computer lab and supervised by the teacher. Other experiments occurred in Brazil, Canada, Italy, India, and Vietnam. Different goals were pursued, generally two or three simultaneously.

Perception of Aplusix by students and teachers
The general opinion of teacher who used Aplusix is the following: The students work more, gain confident, work with autonomy and acquire knowledge. Teachers save time because of students' autonomy and prepared exercises. They are better able to help students having difficulties with mathematics. See detailed opinions in [2].
Students like Aplusix. Even some students who dislike mathematics used Aplusix with pleasure. See comments of students in [1].

Contribution of Aplusix to the students' learning
Experiments with pre-test and post-test have been conducted, either in learning or remedial situations. Generally there were one or two students per computer. In one case in India, the students were in groups of 4. All the results are very positive; see [13]

Use of Aplusix during all the school year
Some teachers in France used Aplusix each time the subject studied can be exercised with Aplusix. At the end of the school year, they observed better result than previous classes. See [13].

Gathering data for student modelling
As our team also works in automatic student modelling, some experiments had the goal to collect data for later analysis in the lab. See results in [15].

2.3. Distribution

Aplusix has currently publishers in France (since July 2005), UK (since January 2007), and Italy (since January 2007). The distribution will start in Benelux in August 2007. New publishers and distributors will be searched.

2.4. Extension to tree representations of algebraic expressions

In the framework of the European project ReMath [16], a new module devoted to tree representations of algebraic expressions has been developed [4]. The goal is to help students understanding algebraic expressions with a new register which is closer to the definition of algebraic expressions. This extension has been fully integrated to Aplusix (to a prototype): the students can choose between 4 representations and modes at any step, except when the teacher has added constraints to the exercise.

The 4 representations and modes are:
1) The ***usual representation*** (or natural representation) which is the previous representation of Aplusix with its editor.
2) The ***free tree representation*** which displays and edits expressions as trees without scaffolding: students can place in nodes whatever they want.
3) The ***controlled tree representation*** which displays and edits expressions as trees with scaffolding: internal nodes of the trees must be known operators and must have a correct number of sons (correct arity); leaves must be integers, decimals, or variables.
4) The ***mixed representation*** which is the *controlled tree representation* with the possibility to have complex expressions in the leaves in their usual representations. See figure 3.

Two particular types of exercises have been added:
1) Transform a usual representation into a tree representation.
2) Transform a tree representation into a usual representation.

The first experiments of the tree representation will occur in France and Italy in September 2007.

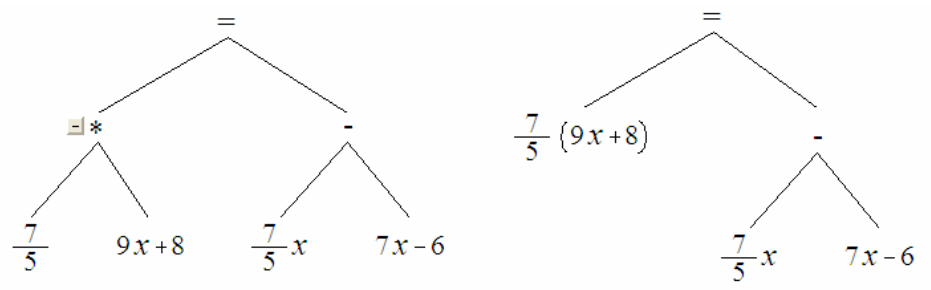


Figure 3. *In the mixed mode, the tree is partially expanded and can be expanded or collapsed using "+" and "-" button which appear when the mouse cursor is near a node. When one clicks on the "-" button near "*" on the tree on the left, one gets the tree on the right.*

3. Algebraic expressions and their representations

3.1. Definition

Given a set of symbols of operators (e.g., {+, –, *, /, ^, sqrt, =, ≠, <, ≤, >, ≥, and, or, not}), a set of symbols of terminal objects (e.g., the integers, the logical values *true*, *false*) and a set of symbols of variables (e.g., {*x*, *y*, *z*}), we define an algebraic expression as a finite construction obtained from the recursive definition given below, an algebraic expression is:

- a symbol of terminal object,
- or a symbol of variable,
- or symbol of operator applied to arguments which:
    - are algebraic expressions,
    - are in the right number (correct arity),
    - and have correct types.

In this definition, symbols are elements that can be drawn on a paper or a screen; expressions have types (e.g., 67 has the *Integer* type; *true* has the *Boolean* type).

This definition is borrowed from the rewrite rule theory [5]. It only concerns well-formed expressions. Note that as far as editing of algebraic expressions is concerned, a notion of ill-formed expressions is necessary, first, because during the construction and the modification of a well-formed expression, one passes by stages where the expression is ill-formed (e.g., to get 3+x, we have the intermediates stages 3 and 3+, the last one being ill-formed), second because there are situations were we want to allow ill-formed expressions, like in education at some stages.

The definition contains concrete elements (the symbols), the global notion being abstract. It does not tell us how to represent algebraic expressions.

3.2. Representations

There are several ways for representing algebraic expressions. We list here the main ones.

Representation in natural language
Example: *The sum of x and the power of y and 3.*
In this example, the symbols of operators have been changed (sum instead of +, power instead of ^). Such representation is possible only for small expressions; for large one, it is not possible to indicate which part is an argument of which operator.

Functional representations

Examples: `+(x, ^(y, 3))` `sum(x, power(y, 3))` `(+ x (^ y 3))`

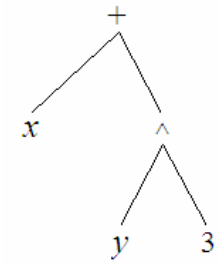
These representations are very close to the definition and have no ambiguity. Large expressions are difficult to understand by human, but not by computers. The last one is the representation used in functional languages like Lisp.

Linear representation

Example: x+y^3

This representation needs priorities between operators, rules for operators of same priority and parentheses. At the beginning, it is the representation used for source code of computer programs. Large expressions are difficult to understand by human, but not by computers.

Tree representation

Example: 

This representation shows very clearly the structure of the expression. It needs neither priority between operators, nor parenthesis, but it needs a large place for large expressions. It can be interesting at some stage of the learning of algebra to help understanding the structure.

Natural representation

Example: $x + y^3$

This is the representation usually used on paper and blackboard. That is why we call it *natural representation* by analogy with *natural language*. It can also be called *usual representation*.

The natural representation is complex: it has hidden operators (like * in *xy* and ^ in y^3), it needs priorities between operators and parentheses.

Latex

Example: $x+y^2$

Latex is a rather old representation used by many scientific communities.

Recent computer representations (MathML, OpenMath)

These representations are computer oriented and based on extensions of XML. They are tree-based representations. Table 2 show an example of representation in Content MathML, Presentation MathML, and OpenMath. They are not supposed to be

Content MathML	***Present. MathML***	***OpenMath***

`<apply>`
`  <plus/>`
`  <ci>x</ci>`
`  <apply>`
`  <power/>`
`  <ci>y</ci>`
`  <cn>3</cn>`
`  </apply>`
`</apply>`
	`<math xmlns=…>`
	`  <mrow>`
	`  <mi>x</mi>`
	`  <mo>+</mo>`
	`  <msup>`
	`  <mi>y</mi>`
	`  <mn>3</mn>`
	`  </msup>`
	`  </mrow>`
	`</math>`
		`<OMOBJ>`
		`  <OMA>`
		`  <OMS cd="arith1" name="plus"/>`
		`  <OMV name="x"/>`
		`  <OMA>`
		`  <OMS cd="arith1" name="power"/>`
		`  <OMV name="y"/>`
		`  <OMI>3</OMI>`
		`  </OMA>`
		`  </OMA>`
		`</OMOBJ>`

Table 2. *Representation of* $x + y^3$ *in Content MathML, Presentation MathML, OpenMath.*

Readability and internal representation

The readability of representations is very different depending the reader is a human or a computer. Table 3 shows readability rates according to the author's opinion.

Representation name	***Example of small representation***	***Human readability***	***Computer readability***
Natural language	The sum of x and the power of y and 3	10%	1%
Functional representation	`sum(x, power(y, 3))`	10%	100%
Linear representation	x+y^3	50%	100%
Tree representation	*See above*	30%	1%
Natural representation	$x + y^3$	100%	1%
Latex	$x+y^2$	10%	100%
MathML and OpenMath	*See above*	1%	100%

Table 3. *Seven sorts of representation and their readability. Readability means capacity to read medium and large representations as there are (e.g., characters and lines on a two dimensions area for the tree representation). The percentages come from the author's opinion.*

Note that the very low computer readability of the tree representation and the natural representation correspond to a situation which is not the most frequent: the case where symbols are written on a two dimensions area like a paper sheet that would be scanned. A more usual situation is a computer screen where the representation is drawn by the computer from an internal representation of the expression. In that case, the computer has not to read the representation, it has just to act on it and use its internal representation to do that.

Algebraic expressions are abstractions

While in classical contexts, like education, algebraic expressions appear as representations of semantic objects (e.g., $3x(x^2 - 4)$ is a representation of a polynomial and $3x^3 - 12x$ is another representation of the same polynomial), the current section shows that algebraic expressions are abstractions having several sorts of representations.

3.3. Syntax and semantics

Above there is a pure syntactical definition. According to this definition, "+" is a constructor of expressions, an operator which can be applied to 3 and 5, giving 3+5, not a function which would have made a calculation and which would have produced 8.

Semantic objects are usually associated to algebraic expressions, e.g., numbers, polynomials, rational fractions, functions, sets (like sets of solutions of equations). Such associations are made by the association of a function to each symbol of operator, for example, for the "+"symbol, it can be the addition of numbers, or the addition of polynomials, or the addition of functions.

The choice of semantic objects introduces a general notion of equivalence: given a set of semantic objects, two algebraic expressions are equivalent if and only if they are associated to the same semantic object.

See more on syntax and semantics in [12].

4. Editing natural expressions

From now, we only consider *natural representations of algebraic expressions* which are called *natural expressions* for simplification purpose. We consider that humans prefer to use this representation system and discuss the way they are graphically represented and how they can be edited.

4.1. The text&box view

Natural expressions are drawn in a 2D (two-dimensions) space and can be viewed as composed of texts (sequences of characters), of boxes with invisible borders and of drawings of particular operators. Box are necessary when the natural representation of an expression cannot be made in one-dimension. Boxes can contain one-dimension expressions and other boxes according to different 2D modes of association. For example, a fraction is composed of a line (drawing of the *divide* operator) with a box above and a box under for the arguments (which are any expressions). We can also consider a surrounding box for a fraction. See figure 4.

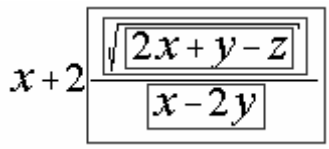


Figure 4. *Text&box view. The borders of the boxes have been drawn.*

There are box operators like *divide*, *power sqrt*, *and* (when it is represented by a left brace "{") , and text operators like +, –, sin, =, <, or.

4.2. Text&box editors

We call text&box editors, editors which follow the text&box view without additional features. This is the case of many 2D editors like Microsoft Equation Editor [11], MathType 5.2 [10] and WIRIS [9, 17] the editor used in the LeActiveMath ILE [8]. Figure 5 shows two examples of expressions displayed by a text&box editor.

These editors allow many sorts of ill-formed expressions: any character can be input at any place of a text part. The only elements which have a clear meaning are the box operators. They are generally obtained through buttons. Often, characters that would benefit to be associated to box operators (like "/" and the *divide* operator) are not.

$$(x/2+5\left(\frac{x}{3}+y\right)+4) \qquad (hello <<< \left(\frac{+-/*\%}{\sqrt{world}}=\right)++$$

Figure 5. Two Text&box expressions that can be built with the Microsoft Equation Editor MathType and WIRIS. On the left, a strange quasi-well-formed expression: the "/" key has been used for x over 2 and the fraction button has been used for x over 3; the parentheses keys have been used at the beginning and at the end, and a parenthesis button has been used inside.

4.3. Syntactical basic enhancement

Here are four features in order to improve the basic editing of natural expressions:

1) Operators recognition: "/" is recognized as *divide* and produces a fraction, "sqrt" is recognized as *square root* and produces a square root, "and" is recognized as *and* and produces a left brace "{", etc.

2) Parenthesis recognition: "(" and ")" are recognized as parentheses and associated to existing parentheses when possible (see figure 6)
3) Arity representation: when an operator does not have enough arguments, places for the missing arguments are drawn (see figure 7),
4) Incorrect type indication (see figure 8),

Figure 6. *From Aplusix: On the left, there is an unbalanced "(". When a ")" is typed, it is associated to this"(" and the parenthesis size is adapted to the content.*

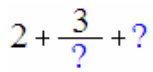


Figure 7. *From Aplusix: "?" represents a missing argument, both in box operators like* divide *and in text operators like* plus.

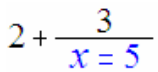


Figure 8. *From Aplusix: "x=5" is drawn in blue to indicate an incorrect type.*

4.4. Enhanced Backspace and Delete

In tex&box editors Backspace with an insertion point inside a box and placed on the left of the content does nothing (Microsoft Equation Editor) or suppresses the box (MathType), see figure 9. This can be improved by deleting the operator. For a unary operator, one can delete the operator and keep the argument, as WIRIS does for the square root in a text way (figure 10) and as Aplusix does in a structured way (figure 11). For a binary operator, one can keep the main argument or the two arguments. See in figure 12 how to suppress common denominators with Aplusix. Similar behaviour can be implemented for the delete key.

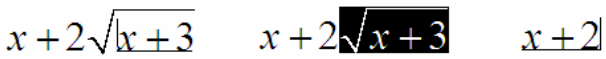


Figure 9. *From MathType. The insertion point is inside the square root, on the left. Hitting backspace selects the square root and its content. Hitting backspace again suppresses the square root.*

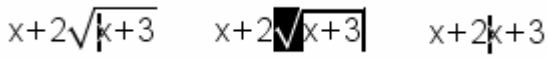


Figure 10. *From WIRIS. The insertion point is inside the square root, on the left. Hitting backspace selects the square root (without the content). Hitting again backspace suppresses the square root and keeps the content in a text way.*

Figure 11. *From Aplusix. The insertion point is inside the square root, on the left. Hitting backspace suppresses the square root in a structured way that makes parentheses appear.*

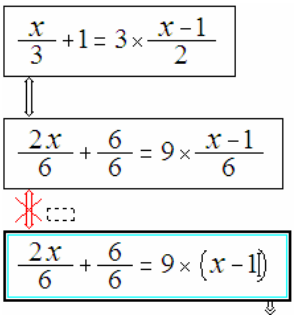


Figure 12. *From Aplusix. The student put all the terms to the common denominator 6. Then, he/she wants to suppress the 3 denominators. He/she has only to place the insertion point on the right of each denominator and hit twice backspace (first, the system replaces 6 by ? and second it deletes the denominator and keeps the numerator in a structured way, that is why parentheses appear).*

4.5. Algebraic selection

Text&box editors allow the selection of parts of the expression regardless the structure (figure 13). The selection is not linked with the notion of sub-expression.

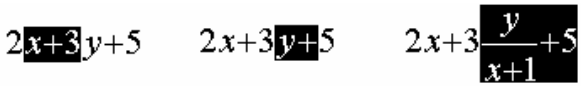


Figure 13. *From Microsoft Equation Editor, MathType and WIRIS: Three examples of selection.*

When places for missing arguments are drawn, it is possible to implement a selection which respects the structure, i.e., the sub-expression. For example, in $(2x+y)(z+1)-2$, let us drag left to right over *x*, *x* is selected; when we continue over +, the selection becomes $2x+1$, when we continue over "(", the selection becomes $(2x+y)$, when we continue over "(" the selection becomes $(2x+y)(z+1)$.
Furthermore, when a selection is present, *crtl-click* can be implemented to extend the selection for associative operators (figure 14) as in many systems.
We call *algebraic selection* this selection mechanism.

Figure 14. *From Aplusix. On the left, 4x is selected. A ctrl-click on 3 or x or 2 on the right part produces the selection displayed on the right in order to have a selection of sub-expression.*

4.6. Operations on a selection

Input on a selection

In *Text&box* editors insertion over a selection of a character (e.g., *x*, 4, =) replaces the selection (this is the general mechanism of text editors) while a click on a button corresponding to a box operator applies the operator to the selection (figure 15).

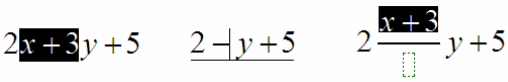


Figure 15. *From MathType: What happens with a selection when one hits "–" (in the middle) or clicks on the fraction button (on the right).*

This can be made more homogenous this way: (1) When the input is an operator, a box operator or not, obtained with a hit of a key or a click on a button, applies the operator to the selection as main argument; (2) In the other cases, replace the selection by the input. In the case of the minus sign, it can

be improved by considering a change of sign, i.e., if the selection is –3x, getting 3x instead of –(–3x), see figure 16.

$$2x^2 \boxed{-3x} + 5 \qquad 2x^2 + \boxed{3x} + 5$$

Figure 16. *From Aplusix. A hit of "–" over a selection of the form "–A".*

Paste over a selection

When an editor implements an algebraic selection, paste over a selection can be performed according to an algebraic way which correspond to a substitution, see figure 17.

$$\begin{cases} z = 3 - 2x + 3y \\ 4x + y + 2\boxed{z} = 4 \\ x + y + z = -1 \end{cases} \qquad \begin{cases} z = 3 - 2x + 3y \\ 4x + y + 2\left(\boxed{3 - 2x + 3y}\right) = 4 \\ x + y + z = -1 \end{cases}$$

Figure 17. *From Aplusix. Paste when z is selected and 3-2x+3y is in the clipboard. This operation is viewed as a substitution and parentheses are added when necessary.*

4.7. Paste on the insertion point and Drag&drop

In texts, paste and drop on the insertion point consists of placing a copy of the clipboard at the insertion point side to side with the text of this place. This has meaning because concatenation has meaning for texts. For algebraic expressions, the link between expressions is made by operators. If we want to combine 4 and 12, we choose to do that with "+" or "/", etc. Placing side to side *x* and *y*, which gives *xy*, has meaning because there are implicit operators (times in this situation). A general behaviour for an editor would be to let the user choose the operator. But this may be a bit heavy. Another way would be to provide a way to easily change the operator used for paste. In the case of Aplusix, we have implemented the second choice, limiting the operators to the operators of variable arity and of the right type, see figures 18 and 19.

$$5x + [3y \qquad 5x + \boxed{12} + 3y \qquad 5x + \boxed{12} \times 3y \qquad 5x + \boxed{12}3y$$

Figure 18. *12 is pasted when the insertion point is before 3. Aplusix pastes 12 with the "+" operator (because it is the upper numerical operator of variadic arity at this position). A hit on the "AltGr" key changes the paste operator to "*".Another hit on the "AltGr" key changes the paste operator to the composition of numbers.*

$$5x + 3y = 6[\qquad \begin{cases} 5x + 3y = 6 \\ \boxed{x + y = 1} \end{cases} \qquad 5x + 3y = 6 \text{ or } \boxed{x + y = 1}$$

Figure 19. *x+y=1 is pasted when the insertion point is after 6. Aplusix pastes x+y=1 with the "and" operator (possible logical operator of variadic arity at this position). A hit on the "AltGr" key changes the paste operator to "or".*

This is a structural, or algebraic, way of performing paste and drop.

Besides this structural drag&drop, there is also an equivalent drag&drop which consists of moving a sub-expression inside a global expression and preserving the equivalence of the global expression. Such functionality is implemented in the GraphingCalculator [6]. This software allows to manipulate the expressions algebraically with the mouse, according to the author's concept "the calculator preserves equality" during these manipulations.

Moving arguments of a commutative operator is the first natural form of equivalent *drag&drop* (e.g., moving $3x^4$ to the left in $2x^3 + x^2 + 3x^4$ provides $2x^3 + 3x^4 + x^2$ if the drop is before $3x^4$ and $3x^4 + 2x^3 + x^2$ if the drop is before $2x^3$). In this particular situation, the structural *drag&drop* does the same thing if the drop is made with the operator.
A second natural form of equivalent *drag&drop* consists of moving an additive sub-expression from a side of an equation (or inequality) to the other side, changing its sign (e.g., in $x^2 + 5x = -6$ moving -6 to the left provides $x^2 + 5x + 6 = 0$).
A third natural form of equivalent *drag&drop* consists of moving a multiplicative sub-expression from a side of an equation (or inequality) to the other side (e.g., in $5x = -6$ moving 5 to the right provides $x = -\frac{6}{5}$). Other forms are proposed in table 4. They are based on factorisations or reductions.

Expression	Selected sub-expression	Place of the drop	Result	Type of action
$3x^2 - 1 + x^2$	x^2	over $3x^2$	$4x^2 - 1$	Reduction
$(y-1)(x+x^2)$	first occurrence of x	Between)(	$(y-1)x(1+x)$	Factorisation
$(xy)^2$	x	Before (	x^2y^2	Factorisation
$\sqrt{4+x}$	4	Before sqrt	$2\sqrt{1+\frac{x}{4}}$	Factorisation

Table 4. *Examples of equivalent* drag&drop *with basic operators.*

5. Conclusion

There is a representation which has been built by humans during several centuries and which is called here *natural expression*. If, at some learning stage, it is good for students to use the natural language representation and the tree representation, and to perform changes of semiotic systems to better understand the notion of algebraic expressions, the rest of the time there is no reason for human to use another representation system.

Software developers are going little by little to fully manipulate natural expression. First, many of them took care of the presentation. Second, some of them took care of the editing in a text&box framework. The last step consists of introducing manipulations which correspond more to the user needs. Many answers can be found by asking the *right question* which is: “As a user, what would I like to get when I do that with the editor?” instead of considering that “With my internal representation the answer to this action is that”.

In this paper, we only considered input with mouse and keyboard. Input with a pen and hand writing introduces new facilities which have already been investigated by some projects.

6. References

[1] Aplusix: Comments of students. http://aplusix.imag.fr/en/ *(Information / Comments (students))*

[2] Aplusix: Opinions of teachers. http://aplusix.imag.fr/en/ *(Information / Opinions (teachers))*

[3] Beeson M. (1996). Design Principles of Mathpert: Software to support education in algebra and calculus, in: Kajler, N. (ed.) *Human Interfaces to Symbolic Computation*, Springer-Verlag.

[4] Bouhineau, D., Chaachoua H., Nicaud, J.F., Viudez C. (2007). Adding new Representations of Mathematical Objects to Aplusix. *Proceedings of the ICTMT-8 conference.* Hradec Králové, Czech Republic.

[5] Dershowitz N, Jouannaud J.P. (1989). Rewrite Systems. *Handbook of Theoretical Computer Science, Vol B, Chap 15*. North-Holland.

[6] Graphing Calculator, http://www.PacificT.com

[7] Koedinger, K.R., Anderson, J.R., Hadley, W.H. and Mark, M.A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education 8*, (pp. 30–43).

[8] LeActiveMath project. http://www.leactivemath.org/

[9] Marquès D., Eixarch R., Casanellas G., Martínez B., Smith T. (2006). WIRIS OM Tools a Semantic Formula Editor. Porceedings of MathUI 2006.

[10] MathType 5.2. http://www.dessci.com/en/products/mathtype/

[11] Microsoft Equation Editor. http://office.microsoft.com/en-gb/word/HP051902471033.aspx

[12] Nicaud, J.F., Bouhineau, D. and Gélis J.M. (2001). Syntax and semantics in algebra. *Proceedings of the 12th ICMI Study Conference.* The University of Melbourne.

[13] Nicaud J.F., Bittar M., Chaachoua H., Inamdar P., Maffei L. (2006). Experiments With Aplusix In Four Countries. *International Journal for Technology in Mathematics Education, Volume 13, No 1.*

[14] Nicaud, J.F., Bouhineau, D., Chaachoua H. (2004). Mixing Microworld and CAS Features for Building Computer Systems that Help Students to Learn Algebra. *International Journal of Computers for Mathematical Learning 9, p. 169-211*.

[15] Nicaud, J.F., Chaachoua H., Bittar M. (2006).Automatic calculation of students' conceptions in elementary algebra from Aplusix log files. Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS2006), LNCS n° 4053, Springer-Verlag.

[16] ReMath project. http://remath.cti.gr/

[17] WIRIS editor. http://www.wiris.com