Integrated Semantic Math I/O in ActiveMath: an Evaluation

Paul Libbrecht, Competence Center for E-Learning
DFKI GmbH, Saarbrücken, Germany, paul@(activemath.org|dfki.de)
Tim Smith, Psychology
The University of Edinburgh, UK, tim.smith@ed.ac.uk

Abstract

The ActiveMath system is a web-based learning environment that integrates static mathematical content and interactive exercises with evaluated mathematical input from learners. Mathematical formulæ in ActiveMath are encoded in OpenMath and presented with regional notations. Users can input formulæ using the same notations via a formula editor or using plain-text input. Input to the editor is assisted by allowing users to copy formulæ from other parts of ActiveMath.

In this paper we will describe how all these components are integrated and work within the system. We will then discuss recent evaluations of the formulæ input methods run within the LeActiveMath project in Malaga and Edinburgh. The results indicate that, even though the assisted input methods provided by the Formula Editor and copy-and-paste are appreciated by users the most popular input method remains the plain text input fields. Proposals are made for how direct input of text can be facilitated and assisted in future formulæ input systems.

Keywords: mathematical formulæ, input, presentation, notations, drag-and-drop, copy-and-paste, evaluation, web-browsers

1 Introduction

Mathematical content platforms are emerging everywhere. Their functionality range from the simple static presentation of content, as provided by most digitial libraries, to rich interactive experiences allowing users to play with mathematical objects in order to explore their understanding. The goal of the EU-funded LeActiveMath project was to develop such an interactive learning environment for mathematics. The LeActiveMath system provides users with the ability to explore their mathematical knowledge by searching through content, engaging in interactive exercises, and examining the system's representation of

their knowledge. Throughout this exploration the user is able to input mathematical formulæ using a variety of techniques that will be described in this paper.

In this paper, we propose a specification for an ideal integrated mathematical presentation and input system and describe a few candidates matching these requirements. We continue with a description the LeActiveMath learning environment, report on evaluations about the mathematical input-and-output and conclude with the hypothesis of changes following the evaluations.

2 Integrated Mathematical User-Interfaces

When designing the ActiveMath learning environment a series of design decisions were made in order to ensure a truly integrated web-based user- interface for the manipulation of mathematical formulæ. These principles are outlined below:

- The platform should be able to present mathematical content graphically with a quality approaching classical print. If the platform is intended for international use, this presentation should be adaptable to the specific customs for mathematical notations within a particular region.

- The mathematical notations should look the same irrespective of where within the platform they appear or whether they are presented on screen or via a printout.

- The mathematical notations should, ideally, be enhanced by interactive features which should support readers' memory about the meaning of graphical constructs. Tooltips and hyperlinks can provide this interactivity.

- All formulæ in the platform should use graphically similar notations. Expressions input by the user should be rendered using the same rendering so that the user understands a common-language between presentation and input.

- Presented formulæ, as much as possible, should be transferrable to input areas following a paradigm familiar to the user and supported by its operating system.

- The mathematical expressions should be procesable by tools offered by the platform. For example, if a function plotter is offered, it should be possible to plot the graph of most functions found within the presented content. Similarly a search tool should match the presented content in a consistent way with the input queries.

The range of actions that can be performed on mathematical expressions defines the depth to which the semantics of the expressions needs to be represented and processed by the platform. It can range from simple presentation to support for each formula including, e.g., type-checking. Orthogonally, platforms can be differentiated with regards to their semantic breadth, that is, the mathematical domain that could be covered by the tools These can range from tools dedicated to a particular task involving special functions to generic repositories which can offer services such as search on any mathematical formula.

3 Integrated Mathematical User-Interfaces in LeActiveMath

The ActiveMath learning environment is an intelligent server software which delivers a rich mathematical experience to learners using contemporary web-browsers. Content in ActiveMath is in `OMDoc` [Koh00], a semantic format for mathematical document with formulæ in OpenMath [BCC+04] and extended by pedagogical annotations and a structure for interactive mathematical exercises. ActiveMath presents documents in Html, xhtml +MathML, and pdf with consistent notations ensured by a declarative format for notations [MLUM06].

The ActiveMath exercise system bases on this presentation system, it can evaluate the user's input semantically or syntactically, navigating through a graph of interactions which allow rich authored feedback [GPE05]. Mathematical formulæ are input either using a textual input syntax, available in several flavours, from the Maple to the Macsyma syntax or using the Wiris input editor [MEC+06] which is derived from the Wiris CAS quoted above but is a self-contained component which has been tuned for the edition of (extensible) OpenMath expressions. The notations of this input-editor are maintained, partially, through the same declarative notations thus striving for a consistent appearance. A screenshot of an exercise with the input-editor can be seen in figure 1.

LeActiveMath is an EU project that started in 2004 with the role of delivering prototypes and evaluating the software around ActiveMath. Among other tools, the LeActiveMath project featured an intelligent tutorial component, a search tool, including searching for mathematical formulæ [LM06], connections to computer-algebra-systems, the Wiris OpenMath Input Editor, and the transfer facility (so-called *mathematical copy-and-paste*) [LJ06].

The LeActiveMath project also featured the development of a complete content for the derivation part of calculus, including foundational material and interactive exercises. The overall platform, made of content and software is often called the LeActiveMath learning environment.

4 Evaluation First Results

The project includes classroom evaluations. The system has been tested in class-based learning in about 10 classrooms in Germany, in the University of Malaga, and in the University Edinburgh during the Fall semester 2006-2007. The objective of the evaluation was to measure the impact and affordance of learning with the learning environment. The complete results of the evaluations are not ready yet and will include measures of the learning impact differences. Quick analyses on the logging data of the usage of the main evaluation server `http://leam-calculus.activemath.org/` has indicated, in the period 2006-2007, a quantity of 31994 formulæ input in linear syntax and 22407 input with input editor.

Optional surveys post-session have estimated a mean quality of formula presentation of 66%, in comparison to 72% quality of texts. It should be noted, that, in order to provide the interactive support on formulæ as described above, the default presentation medium was still HTML+CSS) which explains why formula rendering could be enhanced.

Among the feedback that we obtained, appeared often the *betrayal* of the consistency rule telling that the visual presentation of the formulæ being input should be the same as the re-rendered content: for example $\sin x$, although written without parentheses in rendered formulæ does need the brackets in the input world since the latter needs to know when the argument of sin is finishing. It may seem that this distane between an input language and presentation language is irreducible.

4.1 In-depth Evaluation Tasks

This learning experience was complimented by guided tasks with questionnaires to allow a precise evaluation about specialized facets of the system. The evaluation tasks focussed on various aspects of ACTIVEMATH: access and presentation of content-items, the exercise system, the learner-model, and the tutorial component. We focus on the evaluation tasks of the mathematical input which were taken by 70 students of the University of Edinburgh and the University of Malaga in December (in various undergraduate fields, ranging from mathematics or engineering, to economics).

The main goal of these tasks were to evaluate the ease-of-use and preferences of input of mathematical formulæ in exercises. Subjects were set a series of derivation exercises and asked to answer the exercises using a variety of input methods. They then provided feedback via post-task questionnaires. The formulæ to be input for each exercise were provided in picture in the task descriptions.

The following tasks were given:

1. The first three tasks happen within an easy exercise and require the subject to input the same polynomial $\frac{70}{3} \cdot x^4 + \frac{16}{3} \cdot x$ using a variety of methods. In the first task the subject has to input the polynomial using the plain-text Maple syntax.

2. The second task requires the polynomial to be input using buttons and keys in the input-editor.

3. The third task requires the subject to drag-and-drop the polynomial from the original question into the input-editor (as is pictured in figure 1).

4. The fourth task occurs within an exercise of medium difficulty and lets the student choose which of the three input methods (text, input-editor, or drag-and-drop) they wish to use to input $2 \cdot x \cdot (6 \cdot a \cdot x^2 + b)$.

5. Similarly the fifth task lets the subject choose how to input $15 \cdot \cos(x)^4 \cdot (-\sin x)$ into an exercise of medium difficulty.

6. Finally, the sixth left the students free to input $\cos(7 \cdot x^{11} - 3 \cdot x^3) \cdot (77 \cdot x^{10} - 9 \cdot x^2)$ within a difficult exercise.

Some of the exercises have dynamically generated numbers so the input above may differ a bit.[1]

Due to a sporadic storage-related bug, several appearances of the input-editor (about 10%) were plagued with an error that prevented any input. For this reason, we had to cancel several experiment results of the input-editor. This bug was fixed 10 days later.

4.2 Results and Discussion

A few conclusions follow directly from the results of this evaluation:

[1] Readers of this paper could access these exercises directly, provided they log-in or register into the server, using the following URLs:
The easy exercise
`http://leam-calculus.activemath.org/ActiveMath2/exercises/run.cmd?exerciseId=mbase://LeAM_calculus/exercisesDiffDeriv/fib_productderiv`
The first medium exercise
`http://leam-calculus.activemath.org/ActiveMath2/exercises/run.cmd?exerciseId=mbase://LeAM_calculus/exercisesDiffDeriv/fib_mediumderiv`
The second medium exercise `http://leam-calculus.activemath.org/ActiveMath2/exercises/run.cmd?exerciseId=mbase://LeAM_calculus/deriv_rules/fib2_diffrule_compose`
The difficult exercise `http://leam-calculus.activemath.org/ActiveMath2/exercises/run.cmd?exerciseId=mbase://LeAM_calculus/deriv_rules/fib4_diffrule_compose` The choice of using the input-editor can be decided any time during the first input of an exercise.

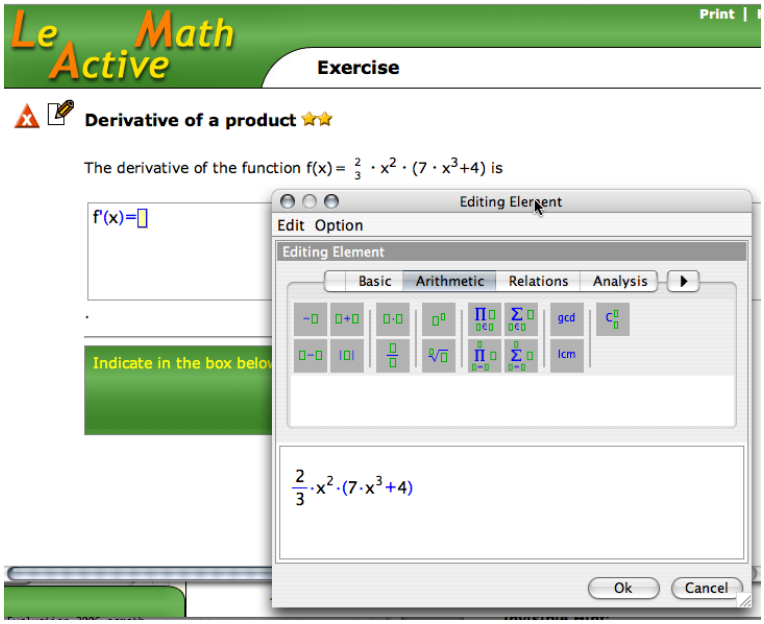

Figure 1: The first exercise for the input-task, using the input-method of first drag-and-dropping the term to be derived then modifying it.

- Most users find the answers simple enough to type into the text-field (68%) and typing into the field is the best solution for all of the free- input tasks: text input was rated as the best input method by 68% of all subjects for task 4, 55% for task 5, and 60% for task 6. The increasing difficulty and length of formulæ across the three tasks which was expected to encourage the usage of the input-editor and drag-and-drop had no such effect.

- Most users believe that the input editor is too complex (63%). However, conflicting with their actual behaviour, they believe that the input-editor or drag-and-drop would be more suited for more complex expressions (63% and 71% respectively).

- Most users knew where to find the buttons needed to input separate symbols ($\tilde{7}0\%$) among the many tabs of the input-editor palettes. However, most users bypassed the buttons and instead typed directly into the input editor, relying on the editor to convert their text input into the appropriate notation.

- Most students found the input-editor's syntax-checking, based on the simple-type-system [Dav00], to be a useful feature. Although, subjects reported that they were often unable to successfully input expressions as the input-editor would reject the expression due to text-input syntax errors without identifying what the error was in a way that would enable

them to solve it. Repeatedly stating ?Input syntax error? does not help a user who cannot self-diagnose their error.

- One of the major failings of the input-editor reported by subjects as extra comments at the end of the evaluation was the problem of sharing the screen space between the main ActiveMath webpage and the pop-up Input-Editor Java applet. In order to successfully back reference of drag content from the webpage to the Input-editor both windows have to be simultaneously visible. This is often not possible due to restricted monitor size, resolution, and the large size of the input-editor. This impracticality combined with inexplicable syntax errors, loading time and storage-related issues meant that subjects experience of using the Input Editor was less than enjoyable.

- The fact that errors are properly flagged and resolvable appears to be very important in a learning situation. As an example, we quote one of the experimentation subjects: *It didn't accept my answer as correct, despite clear syntax (checked by the syntax checker) and my answer being correct.* This sentence can certainly show the state of mind of the learner almost loosing confidence in his input capabilities and expression capabilities.

It is not quite clear how these percentage ratios are computed: the authors state that ?60% liked text-base input?, while ?33% preferred using input editor and 39% liked drug-n-drop?. Since these numbers do not add up to 100%, then obviously either the vote scheme for preferable method allowed to choose more than one method or some other statistical scheme was applied. Authors may consider clarifying this part. The same applies to the rating the usefulness of the input method and to the breakdown of task-evaluation results (numbers related to drug-n-drop – last list in Section 5). I guess a simple chart or a diagram may explain the relations of the above numbers well enough.

The usage of the drag-and-drop gesture has been the sole possible *user-initiated transfer* gesture that we could offer within the (untrusted) web-based environment of ActiveMath. Reasons for this are documented in [LJ06]. This gesture is known to be more difficult than a *standard* copy-and-paste that can be done with normal selection highlighting and the platform clipboard.

The task-evaluations have provided the following results:

- 60% users find that drag-and-drop was difficult to use, 60% found highlighting frustrating, and 70% would have not known they could drag a formula.

- 70% of the users find drag-and-drop *a clever way of avoiding the syntax problems* and more than 60% would use it to drag-and-drop from exercise questions, book-pages to exercises, search tools, computer-algebra system. but only 55% would use it to drag to outside applications (such as word-processors).

These result tend to confirm a general wish for transfer facilities but a weak acceptance of non-standard selection and transfer mechanisms as can be found in the current LeActiveMath. Moreover, the fact that drag-and-drop requires the source and target to be almost simultaneously visible is probably an impediment to easy transfers.

5 Other Platforms with Integrated Mathematical User-Interfaces

Quite a few platforms implement, in some way, integrated mathematical user-interfaces.

On the desktop, mathematical composition engines do this kind of work. They often have a close to common feature-set with computer-algebra systems. Examples of composition engines include, Scientific Workplace,[2], XThink's MathJournal.[3] In these first tools, the mathematical expressions are mostly manipulated in presentation format, only when it comes to computing with them, an engine converts the attempted bits to a semantic form, warning the user if this fails, this conversion is rarely extensible. These platforms support keyboard input and palette-based editing, with XThink even supporting stylus input; we believe that it may fail at respecting local notations for such cases as the intervals of the reals.

The classical computer algebra systems such as Maple, Mathematica, or MuPad, all have a similar approach with a stronger orientation of computable expressions; they all support the translation between various encodings of a formula (presentation-2d, TeX, native-source-code, ...). This openness brings fancy surprises such as the acceptance of MathML presentation expressions of the content of figure 2 and its semantic interpretation in Mathematica.

To our knowledge, none of these platforms allow a user to search through libraries of content, especially for mathematical expressions; they can be used for e-learning activities with interactive exercises but the authoring of such content often requires platform-specific programming.

On the Web, few tools try to approach the integrated approach. We are only aware of the Wiris computer-algebra-system[4] which is a complete applet-based computer-algebra-system, with computations relayed to a server; it seems not to try to present content including combinations of formulæ and text.

As far as we know, all these tools put the math-input in-place among the textual content with buttons and palettes far on the boundary of the windows. This seems in line with the result of our evaluation that challenges the usage of an independent formula editor window that is taken out of context such as the one we have used in ActiveMath.

[2]For the Scientific Workplace family of products, see `http://www.mackichan.com/`.
[3]See `http://www.xthink.com/Products.html` for MathJournal.
[4]See `http://www.wiris.com` about the Wiris CAS.

Figure 2: A *difference quotient expression* in MathML-presentation is on the left. For it to be copied one needs to copy its source which can then be pasted into Mathematica which recognizes automatically. However, the semantic interpretation of Mathematica, obtained by pressing the return key, is somewhat surprising! The same procedure in Maple yields an error at interpreting the limit.

6 Outlook: an Ideal Input Method ?

Following these results, one could propose an ideal system should allow direct text input into a normal text box embedded in the webpage that directly renders the formula in the graphical format, possibly in an embedded java applet below the text box. The applet should not distract from the exercise, or other task, in anyway and must be embedded in the same page. Drag-and-drop or copy-and-paste should be permitted into the text box with immediate rendering below and the rendering could also highlight syntax errors. These errors must have detailed feedback and suggested corrections.

The integration of buttons (i.e. notations templates) with such a system is tricky as it might rob screen space from the main window. One solution would be a pop-up menu with subcategories similar to those already used in the current input editor. This could be navigated direct from the text box and overlaid on the main window that does not distract as it would only be visible when the user calls for it.

An alternative way to achieve the same notations templates function of the input editor with such an approach would be to rely more on the transfer mechanism, possibly making such notations templates' buttons appearing more like a *book of notations* which should be contributed to by books being browsed, by domains already discovered, and maintained by the learners.

Such a system would avoid most probably the dominant opinion that an input-editor is too complex for small tasks, but would provide the same direct-action immediacy of the latter. It could avoid slow error reporting cycles by the immediate display of a partial formula as well as the error-highlight in the expression, one of the main critiques to the text-input. The simplicity plain-text nature, both being input by the user (when typing) and by the computer (when dropping or pasting), an important aspect as noted in this evaluation, would be honoured.

7 Acknowledgements

The development of the tools described here as well as the evaluation have is partly a result of work in the context of the project, funded under the 6th Framework Program of the European Community – (Contract IST-2003-507826). The author is solely responsible for its content. More information about the project can be read from `http://www.leactivemath.org/`.

The components described here have been implemented by Maths4More Inc. (for the input editor), Ian Tsigler (the editor integration), Alberto Gonzales Palomo (the text input-fields' parsers), and Dominik Jednoralski (the drag-and-drop transfer) and the ActiveMath group. Evaluation leaders included Sonia Caro and Marianne Moormann.

References

[BCC+04] Stephen Buswell, Olga Caprotti, David Carlisle, Mike Dewar, Marc Gaëtano, and Michael Kohlhase. The openmath standard, version 2.0. Technical report, The OpenMath Society, June 2004. Available at `http://www.openmath.org/`.

[Dav00] James H. Davenport. A small openmath type system. *ACM SIGSAM Bulletin*, 34(2):16–21, 2000. (see also `http://www.openmath.org/standard/sts.pdf`).

[GPE05] G.Goguadze, A.González Palomo, and E.Melis. Interactivity of Exercises in ActiveMath. *Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences Sharing. Good Practices of Research, Experimentation and Innovation.*, 133:109–115, nov 2005. also published in icce05.

[Koh00] M. Kohlhase. `OMDoc`: Towards an OPENMATH representation of mathematical documents. Seki Report SR-00-02, Fachbereich Informatik, Universität des Saarlandes, 2000. See also `http://www.omdoc.org/`.

[LJ06] Paul Libbrecht and Dominik Jednoralski. Drag and Drop of Formulae from a Browser. In *Proceedings of MathUI'06*, August 2006. Available from `http://www.activemath.org/~paul/MathUI06/`.

[LM06] Paul Libbrecht and Erica Melis. Methods for Access and Retrieval of Mathematical Content in ActiveMath. In Nobuki Takayama, Andres Iglesias, and Jaime Gutierrez, editors, *Proceedings of ICMS-2006*, number 4151 in LNCS. Springer Verlag GmbH, september 2006. Available from `http://www.activemath.org/pubs/bi.php?id=Libbrecht-Melis-Access-and-Retrieval-ActiveMath-ICMS-2006`.

[MEC+06] Daniel Marquès, Ramon Eixarch, Glòria Casanellas, Bruno Martínez, and Tim Smith. WIRIS OM Tools a Semantic Formula Editor. In *Proceedings of MathUI'06*, August 2006. Available from `http://www.activemath.org/~paul/MathUI06/`.

[MLUM06] S. Manzoor, P. Libbrecht, C. Ullrich, and E. Melis. Authoring presentation for OpenMath. In Michael Kohlhase, editor, *Mathematical Knowledge Management: 4th International Conference, MKM 2005, Bremen, Germany, July 15-17, 2005, Revised Selected Papers*, volume 3863 of *LNCS*, pages 33–48, Heidelberg, 2006. Springer.