Equality: A tool for free-form equation editing

Stephen Cummins, Ian Davies, Andrew Rice, and Alastair R. Beresford
Computer Laboratory
University of Cambridge
JJ Thomson Ave, Cambridge, CB3 0FD, United Kingdom
{FirstName.LastName}@cl.cam.ac.uk

***Abstract*—We describe a new tool, *Equality*, for equation entry using free-form layout of components drawn from a palette of symbols. Our approach is designed to enable learners to easily manipulate the structure of their equations, to be functional in both desktop and mobile environments, and to minimize the amount of learning required to use the tool. We present the results of a study comparing a prototype of our approach with Microsoft Equation Editor using a desktop machine. The initial results are promising with participants reporting that the mechanism is easy to learn and an easy way to manipulate their equations. We report the results of the study and the views of the participants and identify how these will inform the future development of *Equality*.**

***Index Terms*—equation entry; equation manipulation, teaching mathematics, teaching physics**

I. Introduction

The teaching and learning of problem solving skills is important to many Science, Technology, Engineering and Mathematics (STEM) disciplines. An important skill in these disciplines is the use of mathematical formulæ as a means to describe and reason about physical and theoretical concepts or problems.

The increasing use of technology in learning means that it is becoming more common to ask students to enter and manipulate mathematical equations electronically. There are a plethora of tools available to support entry of these symbols including LaTeX, Microsoft Equation Editor, and handwriting recognition tools [1]. Online platforms such as Khan Academy provide specially designed symbolic entry interfaces that require a combination of Windows, Icons, Menus, Pointer (WIMP) and keyboard entry modes.

However, these existing tools suffer from a variety of issues. The primary concern is that of the educational process. Pupils do not (and should not) get everything right first time. Therefore, equation entry tools should make it easy to change the structure of an existing answer. As an example, a structural change to an equation might involve changing the scope of a square root, or replacing the multiplication of two expressions with an exponentiation. This is a more sophisticated change than simply editing a single element of the formula in-place. This is obviously particularly difficult for text-based tools such as LaTeX since they are attempting to serialise a two-dimensional layout into a one-dimensional character stream. However, any tool which relies on a structured entry approach will also be affected. The second issue is that of input modality. The rise of mobile devices means that pupils will sometimes have access to a keyboard and mouse, but other times will have only a touchscreen. Similarly, there are circumstances where use of a stylus (for handwriting recognition) is infeasible. Equation entry needs to work well in all of these contexts. The third issue is that of having to learn how to use the language or tool prior to being able to use it properly. Tools which are difficult to learn or frustating to use will simply increase the obstacles already faced by students of STEM subjects.

We have attempted to address these issues with *Equality*[1], a new tool which allows users to construct equations through free-form positioning of elements drawn from a palette.

In this paper we describe our prototype implementation of *Equality* (Section III). We describe a user study comparing *Equality* with the Microsoft Equation Editor (MEE, Section IV). The results (Section VI) show that users found it significantly easier to manipulate their equations with Equality but that the lack of keyboard support in our prototype was a large impediment to fluency. We explain the threats to validity in our study (Section VIII) and highlight our future intentions (Section IX) to meet the needs identified in our study.

The contributions in this paper are:

- a novel equation entry mechanism which we believe is well suited to multi-modal use on modern devices; and
- a demonstration that users find this mechanism easy to learn and an easy way to manipulate their equations.

A. Project background

Equality has been developed as part of a series of tools to be used in the Isaac Physics project.[2] The project is funded by the Department for Education in the United Kingdom and aims to ‘bridge the gap’ between secondary school physics and Higher Education. In particular, the project hopes to improve problem solving skills by providing technologies and content to support physics problem solving. All content produced by the project is available under a Creative Commons license and all software developed (including *Equality*) is open source.[3]

Equality has been produced to support users of Isaac Physics, who may be distance learners, in the entry and manipulation of equations to be submitted as part of their problem solving practice. The target audience is students aged

[1] `http://equality.isaacphysics.org`
[2] `https://isaacphysics.org`
[3] `https://github.com/ucam-cl-dtg/equality`

between 16 and 18 who are interested in Physics, however the *Equality* editor has been developed as a stand-alone module and can be used outside of the physics context.

II. Existing tools

Development of equation entry tools typically involves solving a number of problems including: symbol recognition, segmentation of the symbols, structural analysis (parsing) and ambiguity resolution [2].

Use of the keyboard for input makes the stage of symbol recognition unncessary. Probably the most widely used example of this are the LaTeX math commands. The problem with this approach is that it requires learning a defined set of keywords which together allow description of both the symbols as well as their layout in 2D space.

Many studies advocate the use of handwriting directly onto touch devices as being a natural and efficient mode for entry [3], [4], [5] and manipulation [1], [6] of mathematical equations, whereas our use of a palette instead provides a form of direct symbolic entry. We took this decision since it is uncommon to find a stylus shipped with desktop PCs or mobile phones and mobile devices in particular are often used in circumstances where accurate writing with a stylus is difficult.

Symbolic entry of equations using only the keyboard has been found to be slower and more error prone when compared to other modes (handwriting, speech, handwriting and speech in parallel) [3]. However, it is unclear if the advantage of handwriting based modes remains if one also considers that reliable segmentation and parsing into machine readable form is required for a useable system.

Our hypothesis is that the combination of symbolic entry with free-form layout will provide a better, more flexible solution. Tools that support free-form layout of formulæ and the associated parsing of the mathematical expression are rare but not unheard of [7], [8]. Pollanen et al. introduce the interface for Collaborative Equations (iCE), a tool with similar intentions as *Equality*, however it does differ in its implemention. For a start, the iCE tool introduced requires server connectivity in order to support parsing of equations limiting its offline usability somewhat. *Equality* offers a complete client-side tool so after the initial download of the web page users are able to use it without internet connectivity, which can be particularly intermittent on mobile devices.

From our perspective, we agree with Pollanen et al., that the ability to modify an existing equation is as important as being able to create a new one from scratch. We believe that, especially for novices, working with mathematical symbols is a constructive process and one that requires the medium of entry to be flexible and facilitate easy manipulation of expressions over time. We are not aware of other work which both explicitly considers this goal and also evaluates the approach.

III. Equality Equation Editor

Our primary requirement for *Equality* is that it must support the learning process by being unobtrusive and easy to learn. In order to focus development, a number of design requirements were identified. The final implementation must:

- allow unconstrained entry and manipulation of formulæ,
- allow students to evolve and change their solutions as they learn more about the problem,
- be usable without the need for a keyboard,
- produce machine-readable representations of the entered formulæ,
- require minimal training on first use,
- be suitable for display on mobile as well as desktop devices,
- be general purpose and stand alone from the rest of the Isaac Physics project.

Taking these requirements into consideration, we elected to build a user interface based on drag-and-drop of symbols onto a canvas. No restrictions are placed on the user, they are free to select any symbol from a palette and place it anywhere on the canvas at any size. In many traditional What You See Is What You Get (WYSIWYG) equation editors the user is required to edit a hidden Abstract Syntax Tree (AST), parts of which are sometimes exposed as placeholder boxes. While this can help with accurate entry, it often makes manipulation of formulæ confusing for users without a mental model of the AST. In *Equality*, there is no grid system or hidden AST for the user to explicitly edit, they need only think about the layout of the symbols on the page. The prototype software described here meets most of the requirements above, but still requires some use of the keyboard for entering numbers.

One consequence of allowing free-form symbol manipulation is that an extra parsing step is required. The description below is split into two sections, the first describing the user interface and the second describing the parsing engine built for *Equality*.

User Interface

In order to achieve maximum cross-platform compatibility, the *Equality* user interface is implemented as a web component, written in Javascript and making use of the React framework.[4] This user interface, shown in Figure 1, consists of a main canvas surrounded on three sides by symbol palettes, with an output pane beneath. Symbols can be dragged from the palettes using the mouse or touch and dropped onto the canvas. Once placed, clicking or tapping on a symbol selects it and selected symbols can easily be moved, resized or deleted. There are also facilities for multiple selection and panning the canvas. Figure 2 shows a selected symbol and its associated manipulation tools. Numbers are entered by clicking or tapping anywhere on the canvas and then typing a number.

Symbol Parsing

Whenever symbols move or change on the canvas, a background worker thread is spawned to parse the new layout. Like the user interface, the parser runs entirely in the web browser. This keeps the system self-contained and avoids all issues of

[4] http://facebook.github.io/react/

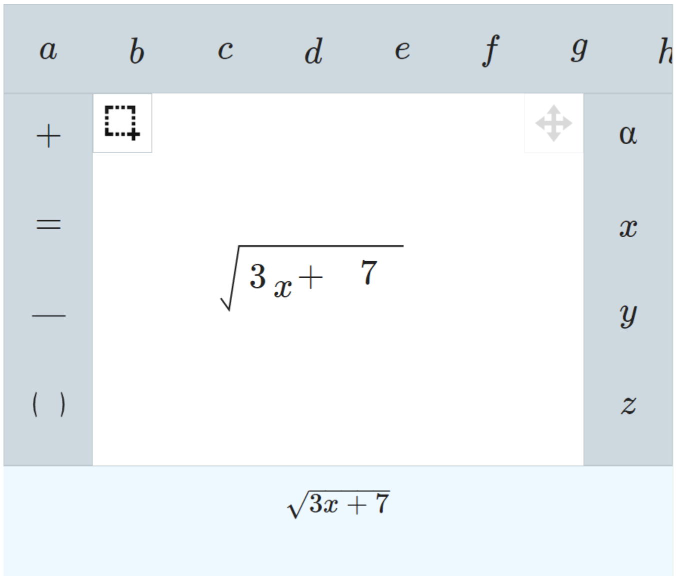

Fig. 1. The *Equality* equation editor. Note that the slightly jumbled symbols dropped onto the canvas have been parsed as $\sqrt{3x+7}$, displayed in the output pane at the bottom.

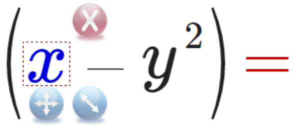

Fig. 2. Symbols on the *Equality* canvas. The x symbol is currently selected and can be manipulated by dragging the surrounding handles. In this case the parsing has failed, producing only $(x - y^2)$ (not pictured). The offending ‘$=$’ symbol is automatically highlighted in red to inform the user that it is unable to be parsed and that, in this case, the expression is incomplete.

deployment, hosting and scaling. The parsing engine is written in ClojureScript,[5] a predominantly functional Lisp dialect with a rich set of immutable data structures and a powerful macro system, all of which make it an ideal environment for implementing complex, recursive algorithms such as this. The Google Closure compiler[6] is used to compile ClojureScript to Javascript for use on the web.

The parsing algorithm uses a breadth-first backtracking approach to combine symbols with appropriate geometric relationships into ever-larger ASTs. For example, an x with a y above and to the right might be combined into an **`exp(x,y)`** AST node. The x and y symbols are then removed from the input and the new **`exp(x,y)`** node is added. This new input is appended to the search queue and the algorithm continues.

This approach allows ambiguous input to be resolved. For example, a horizontal line may be interpreted as a subtraction operator or a separator between fraction numerator and denominator. The backtracking search employed in the parser allows both possibilities to be explored.

The algorithm has two possible termination conditions:

1) The input at the head of the queue consists of a single AST node.
2) No remaining input symbols can be combined by the geometric rules.

In case (1), we have successfully parsed the formula and the single remaining AST node will contain all the symbols on the canvas. In case (2), we were unable to combine all the symbols into one node, so the parsing fails. Depending on requirements for a particular use-case, the parser can be configured to output all possible complete parsing of the symbols, or terminate on the first one that it finds. In the case of a parsing failure, the AST node containing the largest number of symbols is returned, allowing the user interface to highlight un-parsed symbols. The prototype implementation of *Equality* presented here configures the parser to terminate on the first complete parsing, for speed. Symbols that were not parsed are highlighted in red (see Figure 2).

IV. Investigation Method

In order to evaluate the Equality prototype we conducted a user study which compared it to Microsoft's Equation Editor (MEE), included in Microsoft Word 2010. We selected the Microsoft Equation Editor as it is widely available and is reasonably representative of historical modes of equation entry.

The protocol for our user study was as follows:

1) Complete an informed consent form along with a pre-study questionnaire to capture demographic and prior experience data.
2) Familiarisation time with both System α and System β.
3) Use System α to enter the mathematical expressions contained on the pre-printed task sheet.
4) Complete a post-exercise questionnaire for System α.
5) Use System β to enter the mathematical expressions contained on the pre-printed task sheet.
6) Complete a post-exercise questionnaire for System β.
7) Complete a post-study questionnaire comparing both systems.

The familiarisation time started with the investigator demonstrating how to create a simple formula by way of an example before allowing the participant to experiment with both systems. No time limit was imposed on this familiarisation period as we wanted participants to feel comfortable with both systems before the task began, minimising any learning effect.

Each participant was assigned to one of two groups, A or B. In Group A, the participants were first given access to MEE and asked to systematically enter, and then subsequently modify, five mathematical expressions. After they had finished the tasks using MEE, they were given the *Equality* prototype and asked to do the same entry and modification tasks. Group B executed the same protocol except the order of systems tested was reversed, that is the first system under test was the *Equality* prototype and the second was MEE.

[5] `http://clojure.org/clojurescript`
[6] `https://developers.google.com/closure/`

Participant interactions with the systems were recorded using screen capture software[7] as well as software to record the number of mouse clicks and key presses.[8]

Whilst *Equality* has been specifically designed to work on a range of mobile devices, MEE has not, therefore we decided to test both systems on desktop PCs.

A. Research Questions

In order to help focus our investigation, the following three research questions were defined:

1) How quickly were participants able to complete the tasks in System α versus System β?
2) What were the perceptions of participants using System α versus System β?
3) How can we improve the Equality software in general?

V. Context

The user study reported in this paper is our first evaluation of the *Equality* prototype, and we therefore decided to use a sample of first-year undergraduates studying Computer Science or Natural Sciences as participants. This represents a subset of our target audience as described in Section I-A.

We recruited 22 participants (16 male and 6 female) aged between 18 and 21 ($\mu = 18.6, \sigma = 0.79$) from an undergraduate practical class to participate in our user study. No compensation was offered apart from the opportunity to be involved in early testing of the system.

VI. Results

In order to measure the relative efficiency of both systems we recorded the time taken to complete each task as well as the number of keystrokes and mouse clicks made by each participant; the results of this are shown in Figure 3. An analysis of variance showed that it took significantly more time to enter the formulæ in *Equality* ($\mu = 398$ s, $\sigma = 82.6$ s) than in MEE ($\mu = 270$ s, $\sigma = 73.2$ s), $F(1, 40) = 28.51, p < 0.001$.

As expected, the number of mouse clicks required is significantly higher on *Equality* ($\mu = 198, \sigma = 66.5$) than MEE ($\mu = 83.1, \sigma = 34.2$), $F(1, 28) = 35.4, p < 0.001$. This is because it is designed for touch devices. MEE ($\mu = 201, \sigma = 53.1$) understandably requires significantly more keystrokes than *Equality* ($\mu = 41.8, \sigma = 36.1$) to complete the same task, $F(1, 40) = 127, p < 0.001$.

Figure 4 shows a summary of the questionnaire results comparing *Equality* with MEE when rated on a Likert scale from 1 to 5 where 1 is 'Very Difficult' and 5 is 'Very Easy'. In terms of users being able to find symbols, there is no significant variance between *Equality* and MEE, $F(1, 41) = 0.265, p = 0.609$.

There is also no significant difference between the ease of entering new equations when comparing *Equality* and MEE, $F(1, 41) = 0.86, p = 0.36$, although Figure 4 suggests that MEE is perceived as better for equation entry. However, *Equality* ($\mu = 5.14, \sigma = 1.46$) is perceived as being significantly easier than MEE ($\mu = 3.32, \sigma = 1.43$) when it comes to modifying existing equations, $F(1, 41) = 17.18, p < 0.001$.

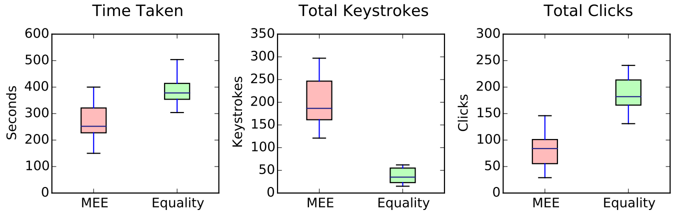

Fig. 3. Observed performance. These metrics were extracted from screen recordings of our user study. All three differences are statistically significant ($p < 0.001$).

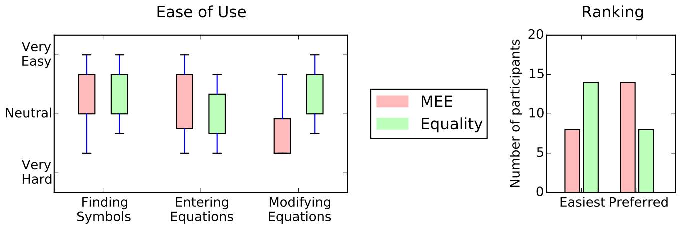

Fig. 4. Questionnaire results. Participants reported that modification of formulæ was significantly easier in *Equality* than MEE ($p < 0.001$). None of the other differences are statistically significant.

Figure 4, shows that participants perceived *Equality* as being easier to use than MEE, but also that they ultimately preferred MEE to Equality for equation entry. An exact binomial test revealed no significant difference in the ease of use ($p = 0.286$) or preferred system ($p = 0.286$) between MEE and *Equality*.

VII. Discussion

We shall now discuss our results with a view of exploring the research questions presented in Section IV-A.

How quickly were participants able to complete the tasks in System α versus System β?

Interestingly, despite our analysis demonstrating that *Equality* is consistently slower than MEE when it comes to formulæ entry, at least one participant perceived it as being faster. This is despite our data indicating the opposite for that participant. They said that they *"... felt less annoyed while using it and it seemed to be faster as well"* when asked to comment on why they thought their chosen system (*Equality*) was easier to use. This could be explained by the flexible approach we have adopted which allows users select and place symbols anywhere on the canvas rather than forcing the user to prematurely commit to the structure of the equation as in MEE. Perhaps this flexible approach altered the perception of how long the tasks took for some participants.

What were the perceptions of participants using System α versus System β?

Interestingly, despite our analysis demonstrating that *Equality* is consistently slower than MEE when it comes to formulæ entry our participants have agreed that *Equality* is easier to use.

[7] `http://camstudio.org`

[8] `http://www.blacksunsoftware.com/mousotron.html`

Conversely, it seems participants agreed that they prefer MEE rather than *Equality* as a system overall. This result was surprising when compared to the aforementioned ease-of-use result but was clearly explained by some of the free text responses to our questionnaires. A number of participants commented that they would prefer to be able to enter formulæ using the keyboard or LaTeX commands as it is quicker, with one participant saying *"The mouse is required for all operations - whilst it is nice to be able to place all symbols, the hand does tend to get tired!!"*. Being able to do both is a sensible feature and will be added to future versions of *Equality* so as not to penalise those who are already familiar with existing techniques of equation entry.

Exploring the questionnaire data further, it appears as though modifying existing formulæ is perceived as being significantly easier in *Equality* when compared to MEE. *"Editing equations was easier because I could drag elements around With microsoft you had to put in the format (e.g. exponent) before actually putting in variables/numbers which was irritating"* This reinforces our aim of providing a tool that supports construction and manipulation of formulæ.

How can we improve the Equality software in general?

The most frequent suggestion we received was to allow *Equality* to accept a greater range of keyboard commands to improve the speed of data input. This feedback is likely the result of some of our participants being already familiar with other equation entry tools such as Microsoft Excel and LaTeX.

VIII. Threats to Validity

Our user study involved a relatively small sample of a population that is expected to have some experience with entering equations on desktop computers. This could lead to a bias in favour of existing tools, however equally the novelty effect of using a new system could affect our results.

Since we recruited participants by asking for volunteers, there is a possibility of selection bias. This is because participants who volunteer for the study may also have a deeper interest in mathematics than the wider population.

IX. Future Work

As we did not evaluate the *Equality* tool on mobile devices in this study, we are unable to demonstrate its effectiveness for this medium. This is a subject for future work and is next on our road map for investigation.

We are in the process of designing *Equality* to improve its ease of use and look and feel. The new version will be integrated into the Isaac Physics platform and further studies will be conducted to evaluate its effectiveness with a more varied sample from our target user group.

Another area of investigation is to consider how *Equality* can be used as an aid to problem solving and constructive thinking in addition to being a tool for manipulating machine readable formulæ.

X. Conclusion

Equation editing tools are an important part of technology-assisted learning. In order to best support the needs of learners we believe these tools should allow learners to easily edit the structure of their equations whilst working effectively on a wide range of devices.

We have described our approach to meeting these needs through our tool *Equality* which provides free-form editing of equation elements drawn from a palette of options. Participants in our user study using our prototype reported that it was easy to learn and provided an easy mechanism for editing equations. The need for keyboard support was clearly identified and we shall be implementing and evaluating this as future work.

We developed *Equality* as part of the Isaac Physics project but we believe it is more widely applicable and so we are releasing it as open-source software in order to make it available to other learning platforms.

Acknowledgments

The authors would like to thank Prof Mark Warner, Dr Lisa Jardine-Wright, Dr Nicki Humphry-Baker, Dr Michael Conterio and the rest of the Isaac Physics team for their comments and support. We also thank the Department for Education (United Kingdom) for their support in funding this work through the Isaac Physics project.

References

[1] S. Smithies, K. Novins, and J. Arvo, "A handwriting-based equation editor," in *Proceedings of the 1999 conference on Graphics interface '99*. Morgan Kaufmann Publishers Inc., Sep. 1999, pp. 84–91.

[2] K.-F. Chan and D.-Y. Yeung, "Mathematical expression recognition: a survey," *International Journal on Document Analysis and Recognition*, vol. 3, no. 1, pp. 3–15, Aug. 2000.

[3] L. Anthony, J. Yang, and K. R. Koedinger, "Evaluation of multimodal input for entering mathematical equations on the computer," in *CHI '05 extended abstracts on Human factors in computing systems - CHI '05*. New York, New York, USA: ACM Press, Apr. 2005, p. 1184.

[4] S. L. Oviatt and A. O. Cohen, "Toward High-Performance Communications Interfaces for Science Problem Solving," *Journal of Science Education and Technology*, vol. 19, no. 6, pp. 515–531, Apr. 2010.

[5] J. J. LaViola and R. C. Zeleznik, "MathPad 2," in *ACM SIGGRAPH 2007 courses on - SIGGRAPH '07*. New York, New York, USA: ACM Press, Aug. 2007, p. 46.

[6] B.-Q. Vuong, Y. He, and S. C. Hui, "Towards a web-based progressive handwriting recognition environment for mathematical problem solving," *Expert Systems with Applications*, vol. 37, no. 1, pp. 886–893, Jan. 2010.

[7] M. Pollanen and M. Reynolds, "A Model for Effective Real-Time Entry of Mathematical Expressions," *Research, Reflections and Innovations in Integrating ICT in Education, FORMATEX*, vol. 3, pp. pp1235–1512, 2009.

[8] M. Pollanen, J. Hooper, B. Cater, and S. Kang, "A tablet-compatible web-interface for mathematical collaboration," in *Mathematical Software ICMS 2014*, ser. Lecture Notes in Computer Science, H. Hong and C. Yap, Eds. Springer Berlin Heidelberg, 2014, vol. 8592, pp. 614–620.