Equation Editing in a Mixed-Initiative User Interface

Eric Lu
Harvard University
ericlu01@g.harvard.edu

Sean Hughes
Harvard University
shughes@college.harvard.edu

ABSTRACT

We present a mixed-initiative interface to a computer algebra system enabling user-guided manipulation of algebraic expressions, and compare it to a modified interface that enables only fully automatic transformations, finding that a mixed-initiative interface does not improve user performance on simple mathematical problems over an interface that enables only fully automatic transformations.

INTRODUCTION

Algebraic expressions are often manipulated using computer algebra systems such as Wolfram|Alpha or Mathematica to help simplify expressions or solve equations. Existing computer algebra systems have two main flaws. First, their metrics for what constitutes full simplification can cause simplification to produce expressions that do not match the form desired by the user. Second, they over-generalize or over-complicate the output with conditions that do not match the conditions of the problem, which must be added manually to be taken into consideration.

As an example, consider the following large output of Mathematica's FullSimplify, which arose in the course of a physics problem in classical mechanics:

$$-\sqrt{-\frac{p_r^2 r^2 (p_t^2 + r(-2km + p_r^2 r))}{p_t^4 + k^2 m^2 r^2 + p_t^2 r(-2km + p_r^2 r)}} \sqrt{1 + \frac{p_t^2 (p_t^2 + r(-2km + p_r^2 r))}{k^2 m^2 r^2}} + \operatorname{asec}\left(\frac{kmr\sqrt{1 + \frac{p_t^2 (p_t^2 + r(-2km + p_r^2 r))}{k^2 m^2 r^2}}}{p_t^2 + r(-km + p_r^2 r)}\right) \tag{1}$$

This is poor output because the equation can be further simplified to the following:

$$-\sqrt{-\frac{p_r^2 r^2 (p_t^2 + r(-2km + p_r^2 r))}{k^2 m^2 r^2}} + \operatorname{asec}\left(\frac{\sqrt{k^2 m^2 r^2 + p_t^2 (p_t^2 + r(-2km + p_r^2 r))}}{p_t^2 + r(-km + p_r^2 r)}\right) \tag{2}$$

which is nearly a third smaller. In this case, FullSimplify might have been made to run further by informing it of a number of *ambient assumptions* present in the case of the problem, such as that k and m are both strictly nonzero. However, adding these requires iterated guess-and-check, and can introduce errors.

ACM ISBN N/A.
DOI: N/A

For an example from another field, consider the following problem arising from introductory statistics:

$$f(z) = \frac{1}{2\pi\sigma_X\sigma_Y} \int x \exp\left(-\frac{1}{2}\left[\frac{1}{\sigma_X^2}x^2 + \frac{1}{\sigma_Y^2}(x-z)^2\right]\right) \tag{3}$$

This integral wedges Mathematica, but a student taking this class might recognize that the appropriate steps to take are 1. to complete the square, and 2. to apply a formula for the Gaussian integral. Both of these steps are conceptually simple but technically involved, presenting many opportunities to make mistakes. Neither can be applied directly without writing special functionality for Mathematica.

In neither case is a human user able to communicate high-level insights to the automated tool in order to achieve a better outcome. Common workflows beyond this point include: guessing subexpressions to substitute with different representations in order to "trick" the automation into producing more desirable output; or writing the expression down on paper in order to deal with it directly. Both take little advantage of automatic support. Ideally, we can combine the flexibility of working with paper with the speed and rigor of computer automation.

Conceptualizing the computer algebra system as an intelligent agent providing the service of fully automated transformation in an equation editing interface, the key problems described above are: uncertainty in the goals and needs of a user, and the unavailability of means for the user to refine the results of an automatic transformation. Previous work suggests that a mixed-initiative user interface is an effective way to allow users to collaborate with the computer algebra system. In particular, including affordances for user decision-making significantly reduces uncertainty about a user's goals and attention, permits scaling the precision of automatic actions, and enables refinement of results, while retaining the value of automation [1, 13].

We have built a mixed-initiative user interface as a web-based app named DocondDo which applies semantics-preserving transformations according to user input on which transformations to apply. See Figure 1 below for a diagram of the user interface. DoconDo's interface allows for users to maintain control over the overall effect of the applied transformations, while the computer handles low-level rewriting. We hope that this approach will help users more efficiently solve problems involving algebraic expressions.

We hypothesize that this approach 1. enables faster and more accurate completion of algebraic manipulation tasks; and 2. reduces the number of separate queries that need to be made to the tool. We evaluated the efficacy of our approach on a pop-

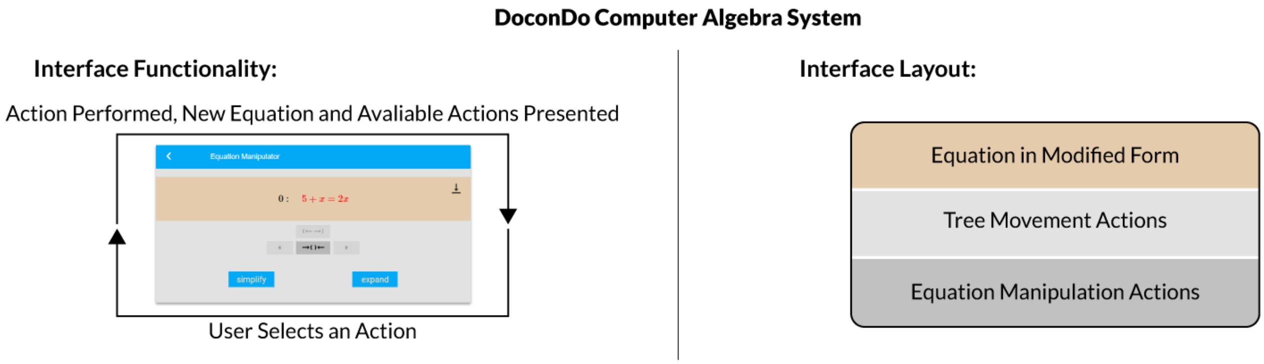

Figure 1. A diagram of the interface presented by DoconDo. The interface shows the current equation, available tree movement actions, and available equation manipulation actions. On user selection of an action, the interface performs the action on the equation, and loads the new equation and list of available actions.

ulation recruited from undergraduate students in quantitative fields.

As discussed in detail in the related work section, few interfaces to date employ a combination of mixed initiative and direct manipulation; we see extant interfaces as lying between semantics-aware systems that exclude human control and human-controlled systems that are semantically unaware. We hope to demonstrate that our interface retains interpretability and accommodates user-set goals and high-level insight, while still improving accuracy and reducing overall workload via use of automation to perform low-level operations.

RELATED WORK

We specifically distinguish between 1. manipulation of algebraic expressions following rewrite rules that preserve semantic equivalence and 2. editing mathematical formulae for the purposes of typesetting. There is an extensive body of work [3456] relating to the latter. In general, these systems do not model the semantics of mathematical expressions; they permit any rewrite, including rewrites that destroy equality, whereas we hope that including semantics will better help reduce error rate as compared to manipulating algebraic expressions on paper. We may incorporate techniques they have established for allowing rapid and umambiguous entry of mathematical expressions; for instance, our prototype relies on MathQuill [12] for equation entry.

Mixed-initiative manipulation of mathematical expressions has previously been implemented for proof assistants [78]. These also respect semantics, but in tactical proof interfaces, lines of text must be entered to perform each operation; furthermore, the supported rewrites are often simple (of the form "$x+0=x$"). Previously, Bertot and others have explored the use of direct manipulation to accelerate operations [79], but these systems supported the same small set of rewrites and were not formally evaluated. We hope to also find that direct manipulation of mathematical expressions will allow operations to be applied in fewer actions. Additionally, proof assistants often make fewer operations available than typical computer algebra systems because they prioritize full formal correctness under a set of logical axioms; like typical computer algebra systems we can relax this requirement to support a broader set of operations while continuing to expect a high assurance of correctness.

Direct manipulation for mathematical expressions has also previously been implemented by Avitzur [10] in a simple graphing calculator application. A mixed-initiative interface might support a broader set of operations, including e.g. trigonometry and factorization, so as to be assistive at least on the scale of undergraduate homework problems.

There's extensive existing work in mixed-initiative user interface design. The design of our interface was informed by existing mixed-initiative interface design principles [1]. Mixed-initiative user interfaces have previously been employed for program manipulation [11], where they also allowed users to accomplish high-level goals while using automation to check for errors and make low-level changes. The context of equation editing is similar inasmuch as a given equation has semantics that are preserved under a set of low-level manipulations.

INTERFACE

DoconDo is a mixed-initiative user interface for equation manipulation. We needed to make two significant choices in the design of an interface to enable effective manipulation of algebraic expressions:

1. Given an expression, what set of actions to present (including integration of computer recommendations); and

2. How best to present the feasible action set, which could be large, so as not to impact usability.

Currently, DoconDo allows users to select a subexpression of the overall expression to act upon, and which of a computer-generated choice of specific transformations to apply. Selection and transformation comprise a set of actions at each subexpression; these include actions that e.g. select the next subexpression ("move right") or the subexpression containing the current selection ("move up"). The available actions depend on the selected subexpression and its immediate context only (for instance, “commute right" is available when the selected subexpression is not the rightmost term in a sum or product). Some actions, such as “simplify” and “expand”, are available regardless of the selected subexpression. We present the set of feasible actions as a panel of buttons for user selection.

EXPERIMENT

Does mixed-initiative algebraic expression manipulation benefit user performance on simple mathematical problems over exclusively computer-automated operations? We compared DoconDo’s full mixed-initiative interface over a modified interface resembling existing CASes that comprises only fully automated transformations on measures of problem-solving performance and subjective perception.

In this experiment, we test the following hypotheses:

1. Users will solve problems faster when provided with the full mixed-initiative interface versus the interface that enables only fully automatic transformations.

2. Users will take more actions on the interface when provided with the full mixed-initiative interface versus the interface that enables only fully automatic transformations.

3. Users will find problems less difficult when provided the full mixed-initiative interface versus the interface that enables only fully automatic transformations.

4. Users will be more confident in their solutions when provided the full mixed-initiative interface versus the interface that enables only fully automatic transformations.

5. Users will find the full mixed-initiative interface more confusing than the interface that enables only fully automatic transformations.

6. Users will be find the interface that enables only fully automatic transformations more frustrating than the full mixed-initiative interface.

Tasks

We presented an interface for equation editing which allows the user to enter a starting equation and subsequently displays an equation and a set of actions that apply semantics-preserving transformations to the equation or a selected subexpression. The experimental interface additionally prompts the user to solve a mathematical problem. Problems were specific algebraic manipulation tasks on equations of rational functions, which are ratios of polynomials in several variables; e.g. “solve for x” or “put in the form of a quadratic equation (ax^2+bx+c)”.

For each subject, we measured problem-solving performance under two conditions. In each condition, subjects used one of two versions of DoconDo to solve two problems. Pen and scratch paper were also provided under both conditions. This enables a between-subject comparison of the effect of a mixed-initiative interface on problem-solving performance.

In the control condition, the subject received a modified interface presenting only fully automated functionality, representing the functionality of existing computer algebra systems. Actions presented were “simplify”, which attempts automatic simplification of the equation, and “expand”, which automatically unfolds all subterms of the equation (e.g. distributes products).

In the experimental condition, the subject received the full mixed-initiative manipulation interface, which permitted selection of subexpressions and application of semantics-preserving transformations. Actions presented were “simplify” and “expand” as above, which applied to the selected subexpression, as well as a set of context-dependent transformations including commuting and subtraction or division from both sides of the equation.

Procedure

Participants were recruited in person and online; all participants were known to the researchers ahead of time. Participants performed the experiment on a researcher-furnished laptop computer.

We first presented participants with a survey of educational background and experience with computer algebra systems, followed by a brief video tour of DoconDo and a description of the task. We then administered two experimental blocks. In each block, participants used one of two versions of DoconDo to solve two problems. The order in which the two conditions were presented was randomized, while the four problems were presented in a consistent order for every participant, so that a participant either solved problems 1 and 2 under the control condition and 3 and 4 under the experimental condition, or solved problems 1 and 2 under the experimental condition and 3 and 4 under the control condition.

After both problems in a block were solved, participants were presented with a survey measuring subjective perceptions of difficulty, confusion, frustration, and solution confidence on a seven-point Likert scale. After the first block, participants were given arbitrary time to take a break before moving onto the second. We offered cookies as a motivation to participate in the experiment.

Participants

We recruited 9 participants among undergraduate students in quantitative fields. Of the 9 participants, eight participants identified as male and one participant identified as female. The average age was 20.6 years, with minimum age 19 and maximum age 23. All participants had taken math at least at the level of algebra. Eight indicated some familiarity with computer algebra systems via coursework.

Design & Analysis

We collected the following demographic measures: gender, age, highest math class taken, highest physics class taken, subjective multivariable calculus experience (seven-point Likert scale), subjective computer algebra system experience (seven-point Likert scale), and previous user study participation.

For each participant, we measured the time taken, correctness of the final solution, number of actions taken, and num-

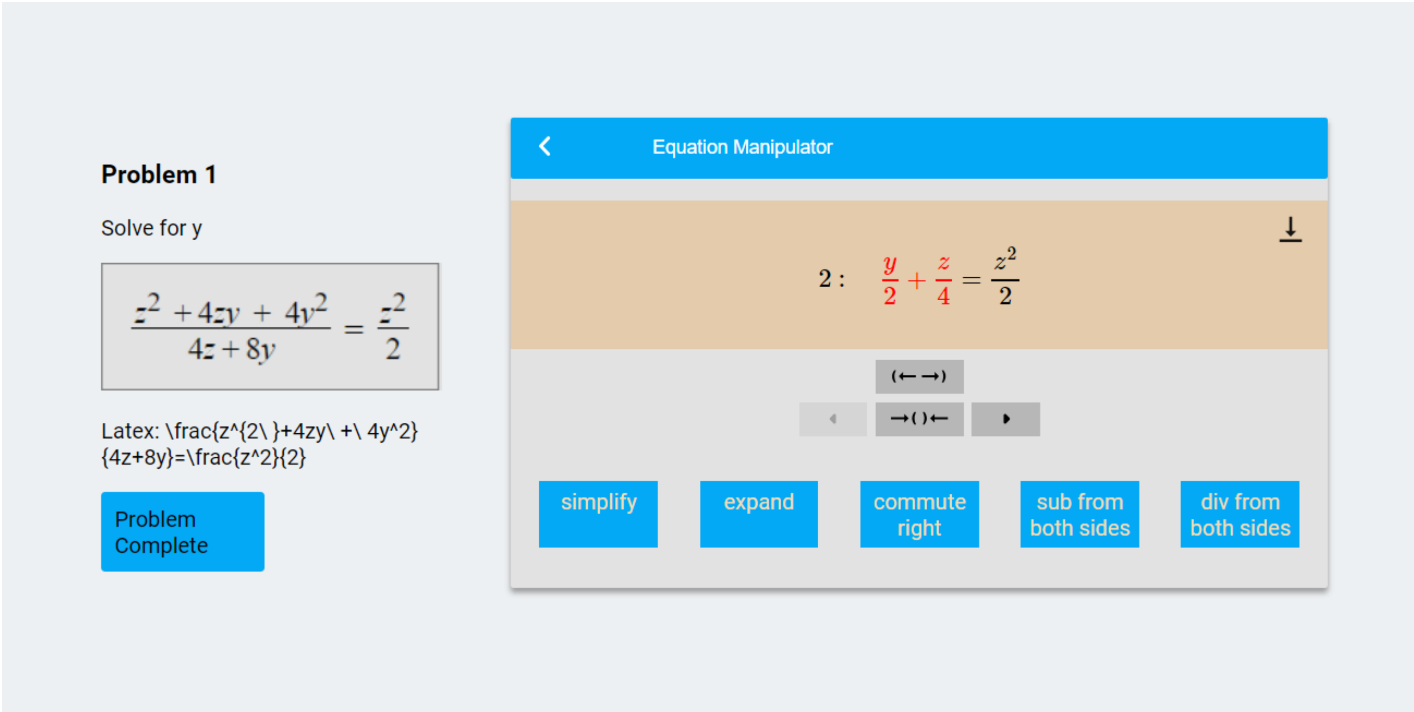

Figure 2. Experimental setup: The area on the left shows the problem given to the user. The area on the right shows the DoconDo interface. Currently displayed is the state of DoconDo under the mixed-initiative condition after the user has entered an equation to manipulate. The current equation state is visible in the beige box, with the current user selection highlighted in red. The range of available movement and manipulation actions is displayed below the equation.

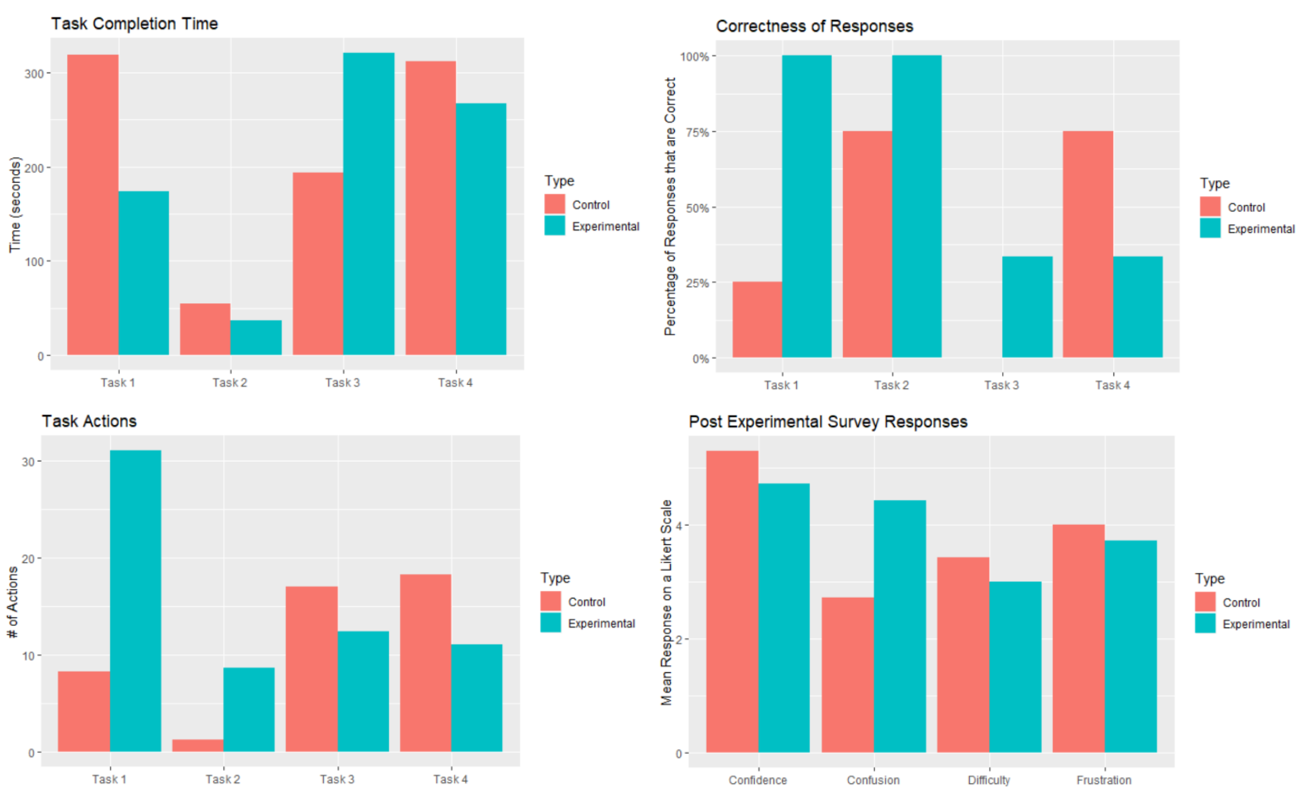

Figure 3. Summary of statistics. In completion time, actions, and correctness, type indicates which condition was used for tasks 1 and 2, so tasks 3 and 4 under the control-first condition were done with the mixed-initiative interface.

ber of equations entered for each of the four problems. After each block, we also measured subjective perceptions of problem difficulty, confidence in the solution, how confusing the interface was, and how frustrating the interface was on a seven-point Likert scale. Our experimental design enabled between-subject comparison of problem-solving performance and within-subject comparison of subjective perceptions. Summaries of data means are presented in Figure 3. The total problem-solving time across all four problems was 845 seconds on average.

RESULTS

Out of 9 participants, we collected complete data from 7. 2 participants did not adhere to the experimental workflow, and their data was omitted from analysis. One participant failed to solve any problem correctly and another avoided using the interface; we included their data in the analysis as these participants were still subject to our complete experimental protocol.

Considering the time taken, correctness, number of actions taken, and number of equations entered on the four problems as multivariate measures dependent on the order of presentation of the two conditions, we ran two-group four-variable MANOVA on each measure to identify the effect of the interface. MANOVA failed to identify a significant effect of condition on time taken ($F_{(1,4)} = 0.597$, $p < 0.7037$), correctness of solution ($F_{(1,4)} = 1.214$, $p < 0.4983$), number of actions taken ($F_{(1,4)} = 1.212$, $p < 0.4988$), and number of equations entered ($F_{(1,4)} = 0.612$, $p < 0.6971$). Running t-test on each univariate measure independently across problems also fails to identify any significant effect on any measure for any problem.

We ran heteroscedastic paired t-test on subjective measures of problem difficulty, confidence in the solution, how confusing the interface was, and how frustrating the interface was to identify overall effects of condition on subjective perception, failing to identify significant effect on difficulty ($t_{(6)} = 0.701, p < 0.5098$), confidence ($t_{(6)} = 0.536, p < 0.6112$), confusion ($t_{(6)} = -1.492, p < 0.1862$), and frustration ($t_{(6)} = 0.603, p < 0.5686$). To check for correlations with problems solved under each condition, we ran two-group two-variable MANOVA, which failed to identify significant effect of condition ordering on difficulty ($F_{(1,2)} = 0.780$, $p < 0.5175$), confidence ($F_{(1,2)} = 5.349$, $p < 0.0741$), confusion ($F_{(1,2)} = 1.146$, $p < 0.4041$), and frustration ($F_{(1,2)} = 0.006$, $p < 0.994$) though the suggestion of an interaction between condition ordering and solution confidence is noted, where presentation of the experimental condition second may raise solution confidence in both conditions.

User Feedback

Our survey of subjective perception also included a free response field for the user to express thoughts about the interface. We report anecdotes summarizing common responses from all nine participants in the experiment.

Six users expressed frustration with or a perception of deficiency in the control interface. For instance, one user responded “This interface definitely had less capability than the last one, which was frustrating"; another responded “The tool kept converging to the same answers which were not that helpful".

Two users indicated that simplify did not do what they wanted during the experimental condition. Two users explicitly requested an “undo” button, which was missing from our interface. Per prior work on mixed-initiative user interfaces, this was a major deficiency; mixed-initiative interfaces present opportunities for undoing undesirable effects of automation and for expanding working memory with a log of recent states [1].

Anecdotally, during the experiment, more users verbally indicated confusion that certain automated actions like “simplify” did not result in any change (which occurs when the computer algebra system perceives the expression to simplify as being fully simplified already), and more users expressed a desire for an undo button, typically after an action produced an undesired result.

Discussion

We have not observed a significant difference between conditions with respect to problem-solving performance or subjective perception. Thus, none of our hypotheses were supported. We enumerate a number of reasons that may explain this outcome.

The number of samples analyzed was 7 in total; this sample size may not have been large enough to expose any but the largest effects of the interface. However, several trends, including in problem-solving time, number of actions taken, confidence, confusion, difficulty, and frustration were in the direction hypothesized.

Our experimental design caused interactions between problem and interface type to be important; one of the strongest effects noted arose from condition ordering. We also note that the overall fraction of correct answers is low in tasks 3 and 4, suggesting that these problems may have been too difficult to engage users. A better design may have randomized problem order to reduce variance in between-subject comparisons of problem-solving performance.

Out of the subjective measures, the largest trend was toward increased confusion when presented with the experimental interface. This trend is in the direction of our hypothesis; we expected higher confusion with the mixed-initiative interface due to unfamiliarity. After conducting the experiment, we anecdotally expect that the effect was likely also exacerbated by unintuitive behavior: some tree elements are not exposed by highlight, so sometimes movement did not result in apparent change in the subexpression. Additionally, users anecdotally found the effects of manipulation actions confusing, or found that a desired manipulation was missing.

FUTURE WORK

The current interface leaves large space for improvement. The interface could be improved with at least: more manipulation features, better availability of manipulation features, a log of recent states and an undo button, and more intuitive presentation of equation structure. To identify what users find missing, most helpful, or most confusing, it may help to conduct a review of the design with user interviews following each task.

ACKNOWLEDGMENTS

We would like to thank Prof. Elena Glassman for her time. We consulted with Prof. Elena Glassman about our project and the HCI space in general.

REFERENCES

[1] Eric Horvitz (1999). Principles of Mixed-Initiative User Interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*.

[2] Andrew Head, Elena Glassman, Gustavo Soares, Ryo Suzuki, Lucas Figueredo, Loris D'Antonik, Bjorn Hartmann (2017). Writing Reusable Code Feedback at Scale with Mixed-Initiative Program Synthesis. *Proceedings of the Fourth ACM Conference on Learning @ Scale*.

[3] Pollanen, M., Kang, S. and Cater, B. (2017). Manipulation of Mathematical Expressions in Collaborative Environments.

[4] Dooley, S.S. (2007). MathEX: A Direct-Manipulation Structural Editor for Compound XML Documents.

[5] Pollanen, M., Hooper, J., Cater, B. and Kang, S. (2014). Towards a Universal Interface for Real-Time Mathematical Communication. *Conference on Intelligent Computer Mathematics Workshops*.

[6] Pollanen, M. and Reynolds, M. (2009). A model for effective real-time entry of mathematical expressions.

[7] Luth, C., and Wolff, B. (1999). Functional Design and Implementation of Graphical User Interfaces for Theorem Provers. *Journal of Functional Programming* 9(2), 167-189.

[8] Bertot, Y. (1999). *The CtCoq System: Design and Architecture*. *Formal Aspects of Computing* 11, 225.

[9] Bertot, Y. (1997). Direct manipulation of Algebraic Formulae in Interactive Proof Systems.

[10] Avitzur, R. (1998). Direct Manipulation in a Mathematics User Interface. In: *Kajler N. (eds) Computer-Human Interaction in Symbolic Computation*. Texts and Monographs in Symbolic Computation (A Series of the Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria). Springer, Vienna.

[11] Andrew Head, Elena L. Glassman, Bjorn Hartmann, Marti A. Hearst (2018). Interactive Extraction of Examples from Existing Code. *Conference on Human Factors in Computing Systems*.

[12] MathQuill. http://mathquill.com

[13] Deterding, S. (2017). Mixed-Initiative Creative Interfaces. *Conference on Human Factors in Computing Systems*.