Concepts and realisations of flexible exercise design and feedback generation in an e-assessment system for mathematics

Nils Schwinning, Melanie Schypula, Michael Striewe and Michael Goedicke

Paluno - The Ruhr Institute for Software Technology, University of Duisburg-Essen, Campus Essen,
{nils.schwinning, melanie.schypula, michael.striewe, michael.goedicke}@paluno.uni-due.de

Abstract. In computer aided assessment (CAA) and automated tutoring the feedback to a submitted solution is very important for the learning outcome of students. It should help them to improve their performances, which means it is designed to go beyond a classification into right or wrong. In this paper we consider the domain of mathematics and introduce a general framework that allows to localise mistakes precisely and offers many options in exercise and feedback design. We present first experiences with the tool and discuss problems, we would like to work on in the future.

Keywords: automated tutoring, e-assessment systems, feedback generation, OpenMath, computer algebra systems

1 Introduction

Whenever large numbers of students apply for a single university course, it is often not possible to provide a suitable number of face-to-face tutoring sessions. Experiences show that training sessions and feedback to exercises are very useful to promote learning success [6]. For this reason, the University of Duisburg-Essen has set up an e-learning and e-assessment system with mathematics capabilities in order to support several introductory courses at the department of economics[1].

Four major requirements were defined for the system: First, the system is intended to not only judge solutions in terms of right or wrong, but also to provide feedback as detailed as possible. Second, it must allow and encourage students to work on the same exercises more than once, e.g. by permitting different ways of solving tasks or by offering variations of tasks. Third, it must provide flexibility in the design of exercises so that exercises can be added and changed easily. This implies the need for a generic way of representing mathematical content independent of the means of input specific to an exercise (e.g. plain text input or

[1] Work on this project was funded by the Federal Ministry of Education and Research under grant number 01PL11075

input supported by formula editors) as well as of the means specific to grading and feedback generation (e.g. connections to computer algebra systems). Finally, the system is intended to not only cover school mathematics, but to provide all necessary means for input and evaluation of exercises in higher mathematics to cover a wide range of bachelor degree courses.

As a consequence of these requirements, the e-assessment system JACK is used as a baseline for the project since it is designed as a flexible framework for computer supported exercises [8]. It was extended to support a concept of path-based exercises with variations, which allow students to work incrementally on their tasks and receive feedback to each of their steps. As a generic way of handling mathematical content, OpenMath[2] was chosen, which provides a generic XML representation of formulas.

Details on the current progress of the project are presented in this paper in the following order: Section 2 reviews existing work both on e-learning systems for mathematics and generic representation of mathematical content. Section 3 elaborates on the concept of path-based exercises as well as on the techniques of feedback generation used within the JACK implementation. Section 4 digs into the details of the implementation and the integration of different parts of the system by using OpenMath. Section 5 describes the use of JACK in university classes. And Section 6 presents preliminary conclusions and next steps.

2 Related Work

Recently, many e-learning systems have been developed. The e-learning systems ActiveMath[5] and Math Bridge[4] provide mostly the same exercise types as JACK does. Although these systems are in general capable of offering alternative paths in exercises, variable content and extensive stepwise feedback, they do not consequently promote any of these features. Neither provide these tools a general framework for interacting with different computer algebra systems.

MathCoach[1] evolved from ActiveMath after the project with ActiveMath ended. We could not find any hints, that MathCoach uses a semantic representation for the submissions of students. MathCoach has an interface for the communication with R, which, however, is only used for statistic computations and plotting and not for the evaluation of solutions.

Another e-learning system is Lon-Capa[3], which is primarily a course management system. Exercises can be generated with variable content, which means that every student gets an individual task. In contrast to the systems mentioned before, students more consequently get evaluative or descriptive feedback in Lon-Capa, depending on the submission. However, we could not find any hints for different paths through an exercise that depend on the student submission or exercise content.

[2] http://www.openmath.org/

3 Path-based exercises and feedback generation

As a general tool for e-assessment and automated tutoring, exercises in JACK are not limited to the domain of mathematics. Instead, exercises may contain multiple choice, fill-in, and drop-down elements for receiving input from the students. Thus a minimal exercise consists of some sort of task description, at least one element that receives input from students, and at least one feedback message. However, JACK offers some options for a more sophisticated exercise design and more detailed feedback messages, in particular for exercises with mathematical content, including the support of LaTeX, input with a formula editor and an evaluation of solutions with a computer algebra system. We will discuss the options in the following subsection.

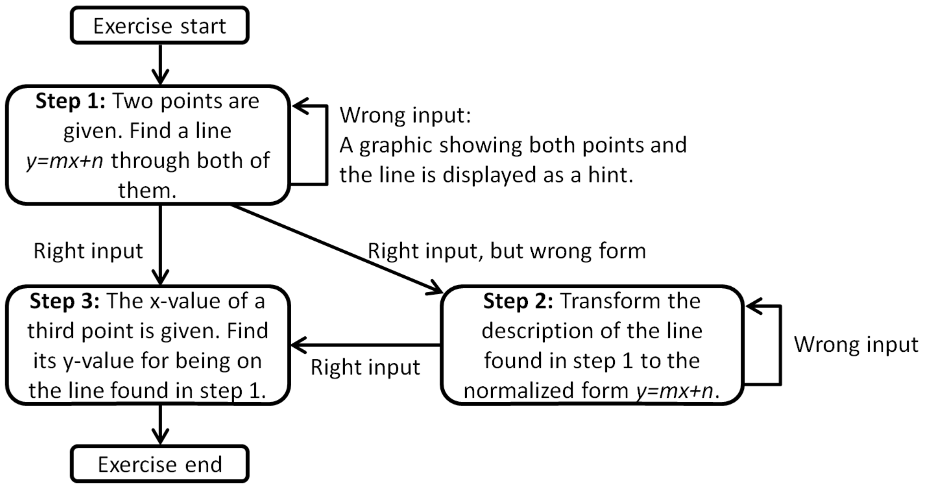
3.1 Design of path-based exercises

As a basic feature of JACK, tasks may be generic by using variables as placeholder. These placeholders are filled in dynamically, so the exercise presents different content to the student every time it is attended. There is no limitation in where to use these variables, so the task description may vary, the number or content of multiple choice or drop-down options may vary, the expected correct results may vary, and so on. While this is of no immediate use for a single visit on a single exercise, it is very helpful when students work with the tutoring system for a longer time. In this case, they can work on the same exercise more than once, receiving different values within that exercise. Thus exercises remain challenging for a longer period of time. Moreover, it possibly helps students to understand the abstract concepts and encourages them to talk to each other about solution strategies instead of plain solutions.

In addition, JACK not only allows for single tasks within an exercise, but also for so-called path-based exercises. A path-based exercise in JACK consists of an arbitrary number of steps, where each step defines a single task. Usually, these tasks are related to each other in some way, e.g. by sharing some context or by reusing values or even inputs from a previous step. A good example of such an exercise is the sketching of a curve, where the determination of each feature of the curve can be seen as a step in the sense of path-based exercises.

Path-based exercises in JACK are not limited to linear sequences of steps, but can be considered as a directed graph. In this graph, each node represents a step of the exercise, and edges are allowed between any two steps. Each step (except for a pre-defined one, called "end"-step, that terminates the exercise) has at least one outgoing edge for the next step. If there is more than one outgoing edge, conditions must be provided under which the system takes the student to the particular step. In this way, not only the values inside a task can vary, but also the sequence of steps can be different every time a student attends the exercise. A sample graph for a path-based exercise is shown in figure 1.

Conditions for choosing an edge can depend on the following parameters:
Variables: As we discussed earlier, exercises in JACK can be generic, e.g. the coefficients of a polynomial can vary every time a student attends the exercise.



Exercise start
Step 1: Two points are given. Find a line y=mx+n through both of them.
Wrong input: A graphic showing both points and the line is displayed as a hint.
Right input
Right input, but wrong form
Step 3: The x-value of a third point is given. Find its y-value for being on the line found in step 1.
Right input
Step 2: Transform the description of the line found in step 1 to the normalized form y=mx+n.
Wrong input
Exercise end

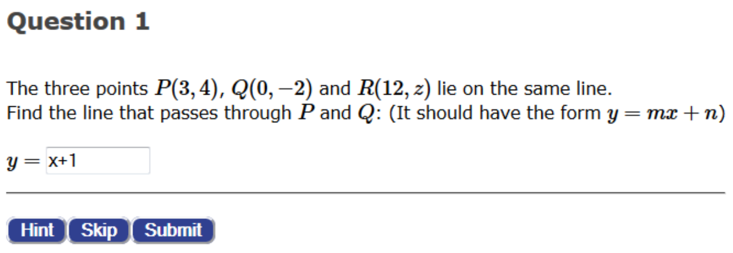
Fig. 1. The graph of a path-based exercise.

Hence it might depend on the variables, whether the polynomial has or hasn't got zeroes. In case the function has got zeroes, the student might have to compute these, while it wouldn't make sense to confront the student with this task when there are no zeroes.
Student's result: The system can react differently to correct and incorrect solutions. For example the student might have to redo the step if he gave the wrong answer, while he may continue otherwise. This approach enables teachers to simulate a strategy often used in oral exams.
Student's input: The student's input from previous steps can be used by the system in the following ones. This enables students to dictate their own pace of work. For example, when students have to solve a linear equation system, JACK is able to accept every correct transformation of the system and provide a new step, where the student has to continue with his input from the previous step.

For students who are not capable of solving a task, authors can define a special succeeding step in case a student wants to "skip" the step. In this case, the student has the option to get to the next step without submitting a solution. In addition to that, the author has to define a message, that is shown to the student instead of the usual feedback message. The possibility of skipping a step can help to prevent students from losing their motivation and thus giving up during an exercise. In summary, path-based generic exercises offer a big variety of possibilities for authors. Using these features, many aspects of an exercise from an oral or a written exam can be simulated.

3.2 Feedback in path-based exercises

As a very important consequence of splitting an exercise into steps, the student may receive detailed individual feedback for each step. A feedback in general


Question 1

The three points $P(3,4)$, $Q(0,-2)$ and $R(12,z)$ lie on the same line.
Find the line that passes through P and Q: (It should have the form $y = mx + n$)

$y =$ x+1

Fig. 2. The student's view of the first step of an exercise in the system.

consists of a score and a feedback message. According to the typology of Tunstall and Gsipps[9], JACK thus provides both evaluative and descriptive feedback: The score is a number in the range of 0 to 100 based on a grading scheme provided by the author. Hence it is an evaluative feedback that provides a judegment and tells the student whether he was right or wrong. The feedback message may contain any content, including dynamically created graphics. In particular, it can refer to the student's input, reuse values from previous steps, and involve any kind of calculation. Hence it is a descriptive feedback that refers to the student's success and may provide guidance on how to improve a wrong solution.

We consider the latter kind of feedback as one of the central features of a tutoring system. It is intended to help the students to comprehend where they made mistakes or where they were correct. To be able to do so, JACK has to understand the semantics of a solution. For this reason, a computer algebra system (CAS) is used to evaluate solutions. On one hand, the CAS can verify the correctness of a solution, even if there are infinitely many correct ones. On the other hand, it is able to locate errors precisely and to compare a student's solution with a standard solution. This enables authors to provide granular feedback, evaluative feedback as well as descriptive feedback.

3.3 Example

Figure 2 shows the first step of the path-based exercise discussed earlier in this section. It was used during an introductory course at the University of Duisburg-Essen in October 2013. The author of the exercise has carved out the following feedback cases:

1. **Conceptual mistakes:** In case a student submits a solution, that does not have the desired form, JACK provides a special feedback for this purpose: (1) If the submitted equation does not depend on x, the system generates a feedback message, telling the student to use the variable x. (2) In case the student uses x and another variable to describe the line, the feedback message recapitulates how a line can be described through the equation $y = mx + n$. (3) If the submitted equation depends only on x, the degree in x of the term on the right-hand side still might be wrong. This means, the student has not computed a line.

2. **Mistakes in computing:** When the student has understood the concept of describing a line by its slope-intercept-form, he might still make mistakes during the computation of this form. Feedback cases were provided for the following mistakes: (1) If the slope was computed correctly but the y-intercept is wrong, the system generates a feedback message for this case. (2) Inverse slope: When students mix-up numerator and denominator in the formula that returns the slope for two given points, a feedback message is generated for this case. (3) If the line only passes through one of the given points, the feedback message that is generated can be seen in figure 3. (4) When none of the above mentioned feedback cases match, the feedback shows the submitted solution and the points P and Q as shown in figure 3, but with a different text. So the student can see, that the line passes through none of the given points.
3. **Formal mistakes:** Even students who have understood the concepts of computing a line from two given points and make no errors in computing may fail in this step because they do not use the desired form. They could for example use the form $n+mx$ instead or extract some factor. While this is possible from the mathematical point of view, it may miss a didactical point of the exercise. Thus a special feedback and an additional step is defined for this case (see figure 1).

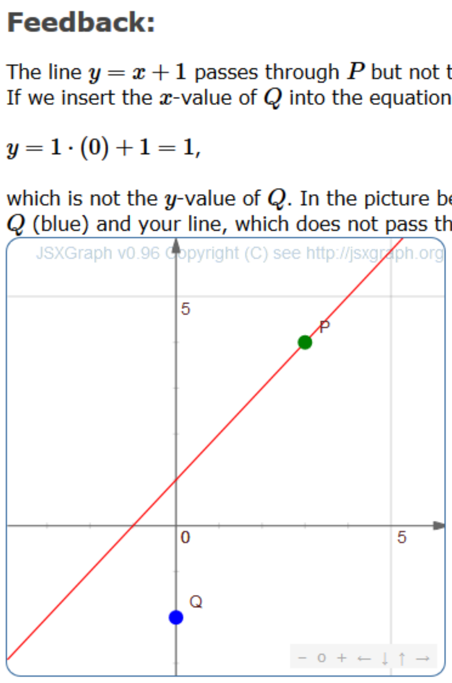
4 Technical realization

In exercises where a solution has to be entered into a text input field, there can be virtually infinite many possibilities to enter a correct solution. The expression $2a + 2b$ could be written as $2a + b + b$ or $a + b + a + b$, for example.

This is why a computer algebra system (CAS) needs to be used for the evaluation of submitted solutions. A CAS is able to compare the student's input with the correct solution and can tell whether they are semantically equal or not. However, most CAS focus on one specific mathematical subdomain. Examples are *Maxima*[3] for analysis and *Magma*[4] for algebra.

In contrast to that, a web-based assessment system for mathematics should be able to cover a big variety of mathematical subdomains, such as algebra, analysis, numerical analysis, statistics and many more. Hence it should be considered to use more than one CAS for the evaluation of solutions. At the same time we cannot expect students to submit their solution in the syntax of the CAS, responsible for the evaluation. In fact, students do not even need to know which CAS is used to evaluate their particular input. Hence it is desirable to have a standardised input syntax. For complex mathematical expressions it even makes sense to support the input with a formula editor, which uses its own representation for expressions. Consequently, an e-assessment system needs to be able to translate between one or more input syntaxes and several CAS syntaxes.

[3] http://maxima.sourceforge.net/
[4] http://magma.maths.usyd.edu.au/magma/



Feedback:

The line $y = x + 1$ passes through P but not through Q.
If we insert the x-value of Q into the equation, we get

$$y = 1 \cdot (0) + 1 = 1,$$

which is not the y-value of Q. In the picture below, you can see P (green), Q (blue) and your line, which does not pass through both points.

Fig. 3. Feedback, in case the submitted line passes through P but not through Q.

This observation is not limited to the input side of systems, but exists in a similar way on the output side: It should be possible to include the output produced by a CAS into the feedback messages displayed to the students. Since the students may not know the output syntax used by a particular CAS, it again has to be translated into a readable syntax like LaTeX. Moreover, the feedback may contain dynamically created graphics as shown in figure 3, so the output from the CAS needs to be translated into a syntax readable by the plotting tool.

For this reason, JACK uses the well-known approach of using OpenMath as a standard representation for mathematical expressions [2], [7] together with a system component or service that translates between the standard representation and a particular syntax, called phrasebook.

Until now, two phrasebooks have been developed for the integration of external services to JACK, a Symja[5] phrasebook and a LaTeX-phrasebook[6]. Both phrasebooks are based on OpenMath Version 2.0 and they make use of the official OpenMath schema[7]. The two phrasebooks implement all the content dictionaries (CD) with status "official"[8] but they can easily be extended because

[5] https://code.google.com/p/symja/
[6] Available at http://www.s3.uni-duisburg-essen.de/research/jack/phrasebooks.html
[7] http://www.openmath.org/standard/om20-2004-06-30/openmath2.xsd
[8] http://www.openmath.org/cdnames.html

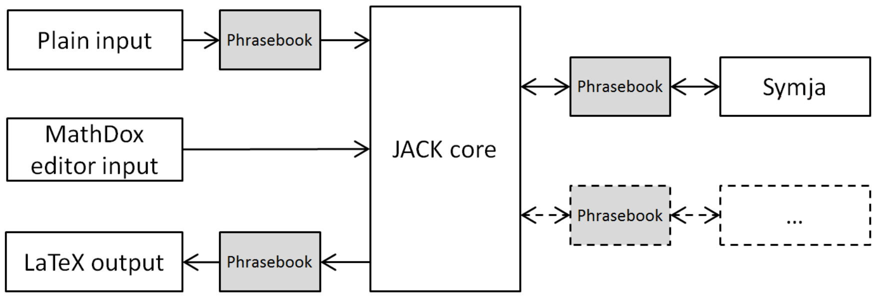


Fig. 4. Sketch of the JACK architecture wrt. data flow for mathematical exercises.

of their structure. To ensure a smooth integration into JACK, both phrasebooks were implemented in Java.

The CAS Symja provides round about 300 functions and shows a certain strength in the field of parsing and comparing expressions. Of course the relatively small number of functions already shows, that Symja is limited in its abilities. For instance, there are no implementations of the inverse hyperbolic functions, which are part of the official OpenMath content dictionaries. This again shows the need for extensibility in terms of computer algebra systems, which is why we have started to work on the integration of Sage[9] into JACK. In addition to that, variables can be determined with the programming language R[10]. This method is already in use in some lectures that work with JACK.

5 JACK in practice

JACK is already successfully in use in some courses at the University of Duisburg-Essen. We started in spring 2013 with microeconomics I, a lecture attended by 722 students. 28 exercises were produced for JACK, which students could work on during the semester. Additionally, students could achieve bonus points in five mini exams, held in JACK. The bonus points were then added to the students' results in the final exam, but only if they had passed the exam. Before using JACK only 48% of the students passed the exam (2011), while this semester (2013) 76% of the students were successful. After the promising start with microeconomics I, we continued using JACK in the lectures microeconomics II in the winter semester 2013/14 and microeconomics III in the summer semester 2014.

Since autumn 2013 JACK is used in classes for statistics, beginning with the lecture descriptive statistics in the winter semester 2013/14. JACK was used in the same way as in the courses for microeconomics. After a successful run with descriptive statistics, JACK is now used in the course inductive statistics as well. We gave workshops for both groups on how to write exercises for JACK, which

[9] http://www.sagemath.org/
[10] http://www.r-project.org/

allows them to provide the exercises for JACK on their own. As you can see in table 1 both groups have produced a significant amount of exercises for JACK until now, which shows that the effort for creating an exercise is not too high.

In addition to that, JACK is used for a preperation course in mathematics as a blended learning concept for the department of economics. Students work on exercises in JACK to refresh their knowledge of school mathematics. Additionally they can ask questions in a forum or in face-to-face sessions. In table 1 you can see the number of exercises and number of students for all courses that used JACK.

term	**course**	**# of students**	**# of exercises**	**# of exams**
summer 2012	prep course mathematics	131	169	10
spring 2013	microeconomics I	722	28	17
	prep course mathematics	177	170	30
winter 2013/14	microeconomics I	415	52	15
	microeconomics II	604	12	11
	descriptive statistics	657	93	23

Table 1. List of all courses using JACK and number of students and exercises.

6 Conclusions and Future Work

In this paper, we introduced a web-based tutoring system for mathematics in which exercises can be represented as directed graphs. Each node of the graph is a subtask, that needs to be fulfilled to solve the exercise. Authors of exercises have many options for connecting the nodes of such an exercise, which adds a great didactical value to the system. Furthermore the system supports authors with many feedback options, which enables them to respond very precisely to student's mistakes. Feedback options include computations using the input as well as dynamic plotting of functions. Combining both, exercise design and feedback options, the system offers a big variety of tutoring skills, which can serve to improve the student's learning outcome. We have pointed out with the help of some examples, how this can be done.

From the technical perspective, we have worked out that computer algebra systems play a central role for the purpose of providing feedback. Hence we need a semantic representation of mathematical expressions. A first step has been taken with the connection to SYMJA, but we have seen that a more general solution for translations between various syntaxes is required. In this context, the use of OPENMATH has proven its worth for the system, because an easy integration of further CAS is possible by implementing further phrasebooks. Our experiences from the implementation of the SYMJA phrasebook should be useful for this purpose and we will actually try to reuse some parts of it.

The fact that JACK is already in use in some courses at the University of Duisburg-Essen indicates that our approach is heading into the right direction.

The pool of exercises as well as the number of participating groups is constantly growing. However, an analysis has shown that the abilities of JACK are not enough for some courses. This is particularly the case for classes from the field of classical mathematics, such as Linear Algebra or Analysis where exercises go beyond computations and include proof strategies. It is our aim to extend our framework with such exercise types, which requires further research in the field of automated theorem proving and reasoning as well as in other fields like mathematical user interfaces and representation formats for mathematical knowledge.

References

1. Jörg Herter Thomas Köppen Barbara Grabowski, Susanne Gäng. MathCoach und LaplaceScript: Ein programmierbarer Mathematiktutor und eine XML-basierte Skriptsprache für interaktive Übungsaufgaben. In *Tagungsband der Leipziger Informatiktage 2005*, pages 211–218, 2005.
2. A. Cohen, H. Cuypers, D. Jibetean, M. Spanbroek, O. Caprotti, E. Eixarch, D. Marques, and A. Pau. LeActiveMath Integrated CAS with OpenMath. EU-Project Deliverable No D12, June 2005.
3. Gerd Kortenmeyer, Guy Albertelli, Wolfgang Bauer, Felicia Berryman, Jeremy Bowers, Matthew Hall, Edwin Kashy, Deborah Kashy, Helen Keefe, Behrouz Minaei-Bidgoli, William F. Punch, Alexander Sakharuk, and Cheryl Speier. The learningonline network with computer-assisted personalized approach (loc-capa). In *Proceedings of Computer Based Learning in Science Conference*, Cyprus, 2003. CBLIS.
4. Josje Lodder and Johan Jeuring. Math-bridge, bridging the math gap between high school and universities. In *Proceedings EADTU annual conference 2011*, pages 177–185, 2011.
5. Erica Melis, Eric Andrés, Jochen Büdenbender, Adrian Frischauf, George Goguadze, Paul Libbrecht, Martin Pollet, and Carsten Ullrich. Activemath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education*, 12:385–407, 2001.
6. Edna Holland Mory. *Handbook of research on educational communications and technology*, chapter Feedback research revisited, pages 745–783. Erlbaum Associates, 2004.
7. H. Prieto, S. Dalmas, and Y. Papegay. Mathematica as an OpenMath application. *ACM SIGSAM Bulletin - Special Issue of OpenMath*, 34:22–26, June 2000.
8. Michael Striewe, Moritz Balz, and Michael Goedicke. A flexible and modular software architecture for computer aided assessments and automated marking. In *Proceedings of the First International Conference on Computer Supported Education (CSEDU), 23 - 26 March 2009, Lisboa, Portugal*, volume 2, pages 54–61. INSTICC, 2009.
9. Pat Tunstall and Caroline Gsipps. Teacher Feedback to Young Children in Formative Assessment: a typology. *British Educational Research Journal*, 22(4):389–404, 1996.