Towards Adaptive Generation of Faded Examples *

Erica Melis and Giorgi Goguadze

Universität des Saarlandes and
German Research Institute for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg, 66123 Saarbrücken, Germany
phone: +49 681 302 4629, fax: +49 681 302 2235

Abstract. Faded examples have been investigated in pedagogical psychology. The experiments suggest that a learner can benefit from faded examples. For these experiments a few examples were faded manually. For realistic applications, however, it makes more sense to generate several variants of an exercise by fading a worked example and to do it automatically. For the automatic generation, a suitable knowledge representation of examples and exercises is required which we describe in the paper. Moreover, a user-adaptive system such as ActiveMath can select or dynamically produce faded example and present it to the student in response to her learning activities and adapted to her goals, capabilities, previous learning experience, etc. The structures and metadata in the knowledge representation of the examples are the basis for such an adaptation. In particular, it allows to fade a variety of parts of the example rather than only solution steps.

1 Introduction

Worked-out examples typically consist of problem formulation, solution steps, and the final answer. Empirical studies suggest that exposing learners to worked-out examples in early learning stages and for low-ability students can be more effective than learning by problem solving only. Worked-out examples proved to be learning-effective in certain contexts because they are a natural vehicle for practicing self-explanation. Taking the idea of self-explanation a step further, *faded* examples provide another ground for self-explanation that is slightly more difficult for a learner. Here, faded example means a worked-out example from which one or more parts have been removed (faded) deliberately. Those faded examples are exercises in which the learner has to fill in an equivalent for what has been removed. Recently, faded examples have been investigated in pedagogical and cognitive psychology with promising results [15, 13, 12].

* This publication is partly a result of work in the context of the LeActiveMath project, funded under the 6th Framework Programm of the European Community – (Contract IST-2003-507826). The authors are solely responsible for its content. The European Community is not responsible for any use that might be made of information appearing therein

Faded examples are of interest because they offer an interaction with the student in which

- the working memory load is not as heavy as for totally faded examples – the typical exercises – so there is a gradual transition to 'full' exercises
- the example context might act as a reminder
- an active analysis of the problem solution is necessary to fill in the faded details – a superficial processing of the example is hardly possible
- there is less stress on performing and more on the actual understanding
- fading can be used to anker and stimulate
  - limited problem solving
  - reflection
  - self-explanation.

So far, faded examples have been produced manually. However, for realistic applications, as opposed to lab experiments, it makes sense to generate several variants of an exercise by fading and to do it *automatically*. For such an automatic generation, a suitable knowledge representation of examples and exercises is needed.

A faded example – if faded deliberately rather than randomly – can serve the purposes of misconceptions discovery and deeper learning. In order to target particular difficulties, needs, learning goals the actual fading has to be adapted to the learner's characteristics and her learning history.

In our mathematics seminars we experienced the value of faded examples for learning. We are now interested in generating adaptively faded examples which can then be used in our learning environment for mathematics, ActiveMath. Several steps are needed before the course generator and ActiveMath' suggestion mechanism can present appropriate faded examples to the learner: the knowledge representation has to be extended in a general way, the adaptive generation procedure has to be developed, and finally, the ActiveMath-components have to request the dynamic generation of specially faded examples in response to learners actions. This article concentrates on a knowledge representation of examples and exercises that allows for distinguishing parts to be faded and for characterizing those parts. This is a non-trivial work because worked examples from mathematics can have a pretty complex structure, even more so, if innovative pedagogical ideas are introduced. We describe a suitable knowledge representation based on the semantic XML-representation for mathematics, OMDoc [6, 5]. On top of this, the adaptive fading is considered and we discuss general adaptations of the fading procedure we are currently implementing.

2 Examples

Here, we introduce some worked examples that are taken from the mathematics textbook [1]. Then we discuss how these examples can be faded at several places.

Example 1 p.75, [1] provides a worked-out solution of the problem

$$If \;\; a > 0, \;\; then \;\; \lim_{n\to\infty} (\frac{1}{1+na}) = 0$$

Solution
Step 1. Since $a > 0$, it follows that $0 < na < 1 + na$...
Step 2. ... Therefore, we conclude that $0 < 1/(1 + na) < 1/(na)$, ...
Step 3. ... which evidently implies that

$$\left|\frac{1}{1+na} - 0\right| \leq \left(\frac{1}{a}\right)\frac{1}{n}, \;\; for \;\; all \;\; n \in N$$

Step 4 .... Since $lim(1/n) = 0$, we may invoke Theorem 3.1.10 with $C = 1/a$ and $m = 1$ to infer that $lim(1/(1 + na)) = 0$.

In this example, we can fade the assumption $a > 0$ at any or at some occurrences. The purpose would be to (1) focus more attention on the example, (2) make the learner check whether all cases have been considered which is an important reasoning capability for mathematical problem solving, and last but not least to remind the student that dividing by 0 is not defined. Such a faded example may require from the student to re-visit the complete solution and introduce that assumption again.

Another useful fading removes steps. For instance, removing Step 2 could train general *application* skills of the learner who will have to find the bridge between Step 1 and Step 3. Fading Step 3 could stimulate reasoning about the conditions of Step 4 and thus train the *application* skills w.r.t. Theorem 3.1.10, since step 3 prepares the application of this theorem in the next step.

In the Step 4 different parts can be faded depending on the targeted competence level. Fading the assumption $lim(1/n) = 0$ requires the learner to remember the facts, fading the reference to the Theorem 3.1.10 the *comprehension* is requested and fading the additional assumption $C = 1$ and $m = 1$ which are used as parameters in the theorem requests the application of Theorem 3.1.10.

Example 2 p.82, [1] provides a worked-out solution of the problem

$$The \;\; sequence \;\; ((-1)^n) \;\; is \;\; divergent$$

Solution
Step 1. This sequence $X := ((-1)^n)$ is bounded (take $M := 1$), so we cannot invoke Theorem 3.2.2. ...
Step 2. ... However, assume that $a := limX$ exists. ...
Step 3. ... Let $\epsilon := 1$ ...
Step 4. ... so that there exists a natural number K_1 such that

$$|(-1)^n - a| < 1 \;\; for \;\; all \;\; n \geq K_1 \;\; ...$$

Step 1 is, formally seen, not necessary for the solution. But it provides a metacognitive comment showing why an alternative proof attempt would not work.

It would be sensible to fade this step, and request from the learner to indicate valid or invalid alternatives or to fade parts of this step.

In steps 2 and 3 two hypotheses are defined. These hypotheses are dependent. Fading both hypotheses introduces more under-specification, than when fading only one assumption.

Some good textbook authors omit little subproofs or formula-manipulations and instead ask "Why?" in order to keep the attention of the reader and make her think. For instance, in the proof of the current example Bartle and Scherbert say:
... If n is an odd number with $n \geq K_1$, this gives $|(-1)^n - a| < 1$ so that $-2 < a < 0$. (Why?) ...
The proof of those statements is usually easy and trains *application* skills.

Example 3 Consider the work-out solution of a differential equation

$$\frac{dy}{dt} = \frac{4sin(2t)}{y}, \quad y(0) = 1$$

Solution
Step 1. We begin by separating the variables and creating the two integrals,

$$\int ydy = \int 4sin(2t)dt$$

Step 2. The integrals are evaluated giving $\frac{y^2}{2} = -2cos(2t) + C$
Step 3. This is solved for $y(t)$ to give $y(t) = \pm\sqrt{2C - 4cos(2t)}$
Step 4. We evaluate the arbitrary constant C using the initial condition. Because $y(0) = 1$, we take the positive square root. Thus, we have $y(0) = \sqrt{2C - 4cos(0)} = \sqrt{2C - 4} = 1$
Step 5. It follows that the solution is $y(t) = \sqrt{5 - 4cos(2t)}$

By fading the calculation steps, parts of them, the final result, or the verification of the result one can obtain a sensible exercise training knowledge of different concepts or application skills. Different parts can be faded depending on the competences to train.

3 Psychological Findings

Some empirical studies investigated faded examples [14, 15, 13, 12]. Merrienboer [15] suggests positive effects of faded examples in programming courses but those experiments do not clearly indicate whether those effects are due only to working with incomplete examples.

In a context in which the subjects have little pre-knowledge Stark investigates faded examples in probability theory courses and shows a clear positive correlation of learning with faded examples and performance on near and medium transfer problems [13]. A positive correlation also occurred for the flexibility of applying the knowledge. He also suggests that in comparison with worked-out examples, faded examples better prevent a passive and superficial elaboration/processing. His experiments included an immediate feedback in form of a complete problem solving step which he thinks is necessary for low-ability students. Moreover, for low-ability students Stark recommends to start with very detailed solutions.

Renkl and others found that *backward fading* of solution steps produces more accurate solutions on far transfer problems [12] – an effect that was inconsistent across experiments in other studies. Backward fading led to a statistically significant effect on accuracy of anticipation. Moreover, the advantage of backward fading cannot be attributed to additional time on task. These studies suggest that mixing faded examples with worked-out examples (with self-explanation) is more effective than self-explanation on worked-out examples only.

We expect that backward fading is superior mostly for low-ability students or novices (rather than experts) because they rely more on backward solution strategies.

4 Preliminaries from ActiveMath

ActiveMath is a user-adaptive, web-based learning environment for mathematics. It dynamically generates learning material for the individual learner according to her learning goals, preferences, and mastery of concepts as well as to the chosen learning scenario [9].

Currently, ActiveMath's user model consists of the history data base, the user's preferences profile, and a database of mastery-levels of the concepts in the domain. The history contains information about the user's activities (reading time for instructional items, exercise success rate, manual changes of the user model). The user profile contains the student's preferences, learning scenario, and learning goals submitted for a session. To represent the concept mastery, the current user model contains values for a subset of the competencies of Bloom's taxonomy [2]: knowledge, comprehension, and application.

The content to be assembled and presented by ActiveMath is separately stored in a knowledge base. It is represented in the semantic markup language for mathematical documents OMDoc [7]. For educational applications we extended OMDoc with educational metadata and structures. In OMDoc, mathematical knowledge is represented as typed items of knowledge together with relations among them, organizing the items into mathematical ontologies. This knowledge representation allows for better reuse and interoperability of content and for displaying dependencies of concepts in a domain.

5 Knowledge Representation

Since ActiveMath relies on OMDoc knowledge representation, the structure and metadata annotation of that XML-language have to be enhanced for generating exercises by fading examples. Since more and more ITSs rely on XML-representations, we think it is useful for the community to explain what the representation of examples (and exercises) looks like and how it is extended.

First, we had to extend OMDoc because its ontology (properties and relations) is insufficient for educational purposes. Therefore, some additional metadata elements are defined, such as `difficulty` and `abstractness` of a knowledge item, `learning-context` of the learner, `pedagogical-level` of an exercise with values 'knowledge', 'comprehension', 'application', and 'transfer' from the Bloom Taxonomy of learning goal levels, etc.[1]

Moreover, the examples in OMDoc do not possess any internal structure and exercises are restricted only to multiple-choice questions.

In this paper we shall introduce a rich internal structure for worked out examples and show the guidelines for adaptively generating interactive exercises out of them.

Another ActiveMath extension of OMDoc refines the micro-structure of interactive exercises, as described in [5]. An interactive exercise is a collection of interlinked steps, called *Interactive Actions*. The amount of these steps is finite, but the amount of steps the learner has to perform in order to complete the exercise is not fixed, and could be potentially infinite. The goal of the exercise representation language is not to describe the completely pre-defined set of solutions, but possibly the plan of the solution, partial results, the final result. The correctness of the step or the strategy of the user might be verified in different ways, since the exercise might be communicating to a computer algebra system, or a theorem prover. The evaluation of each interactive step can consist of estimating the correctness of the step using a Computer Algebra System or any other tool available for the evaluation of the user's input.

We mention this format here, since this is going to be the target format for the faded examples.

5.1 Anatomy of Mathematical Example

Mathematical examples can possess a complex internal structure, depending on the kind of example considered. The solution for an example can be a mathematical proof, calculation, exploration, construction of a model etc.

We concentrate on the knowledge representation of a mathematical example, suitable for being faded.

The knowledge representation we suggest is experimental, mostly based on the experience of few authors and teachers using ActiveMath. The real evaluation of the suitability of the proposed knowledge representation is to come in the future.

Since in a faded example one introduces under-specifications at pre-defined places in a worked-out example, i.e., in its problem statement or its solution, these places have to be marked and annotated with metadata to characterize them. The original information from the example can be used later for evaluation purposes.

[1] For full reference to all metadata extensions made by ActiveMath, see [3].

The first extension (5.2) targets automatic generation in general. The subsequent extensions (5.3) target the adaptivity in the generation.

5.2 Different Fadable Parts in the Original Examples

Depending on the content and structure of a worked-out example different parts can be faded. At the top-level, either parts of the problem itself (such as a condition), parts of the problem solution, parts of a meta- cognitive Polya framework [11] of the solution can be faded.

How to adopt Polya framework in order to provide structure for the presentation of exercises in specific scenarios of ActiveMath has been described in [10]. This was not based on an extended knowledge representation of exercises. Now we propose knowledge representation, based on OMDoc, which supports Polya structures and allows to fade them.

In more detail, faded parts may include (this list is likely to be incomplete)

- one or several assumptions of the problem
- a full problem solving step or its textual description
- the reason for applying a step, condition of a step (Because ... step ...)
- a sub-proof or sub-solution
- goal statements
- subgoals (e.g., numbers and mathematical expressions in maths solutions)
- parameters of a problem solution or a method application
- explanations and auxiliary information
- the reference to a justification (e.g., a theorem, principle)
- references/links to other instructional items such as similar solutions
- anticipatory information
- meta-cognitive structure and heuristics such as
  - input of Polya-phases
  - headings of Polya-phases

Existing OMDoc *Markup for Proofs* In OMDoc, particular attention is paid to the internal structure of proofs. As described in [7], this representation for proofs is suitable for representing formal proofs in different proof styles as well as textbook proofs.

The `proof` element in OMDoc is briefly a Directed Acyclic Graph (DAG) of steps, connected by cross-references. Each derivation step in the proof can consist of textual content, formal content, it can possess a justification in form of a reference to a derivation rule or method used or a sub-proof. Apart from derivation steps there can occur a number of `hypothesis` elements, containing local hypothesis in the proof. The last step of the proof is called `conclude`.

For meta-cognitive explanatory texts that are not necessarily logical part of the proof, the element `metacomment` is used.

Reusing and Extending OMDoc We generalize the notion of `proof` to the notion of `solution` and allow it as a child element within the OMDoc element `example`.

The difference between proof and solution is that the solution does not always prove some statement, but sometimes calculates the value of some expression or explores the properties of a particular structure (e.g. curve discussion). The whole worked-out solution is now a hierarchy of steps, each of those is potentially fadable completely or partially. In order to enable more intelligent fading of parts of steps, we allow authors to annotate parts of steps to be faded with unique identifiers using the container element `with` for marking.

For representing meta-cognitive explanations of different types, we refine the element `metacomment` by introducing the `type` of `metacomment` with possible values *alternative*, *comparison* and *explanation*. Each of the comments might have more then one type.

Finally, we extend the solution format to represent a meta-cognitive Polya framework. For this, we introduce four meta-steps with following refinements that Polya uses in the respective phases:

1. Understand the problem:
   - What is given?
   - What does it depend on?
   - What is unknown?
2. Devise a plan:
   - Do you know a related problem?
3. Carry out the plan:
   - Can you see/prove that each step is correct?
4. Look back at the solution:
   - Can you check the result?
   - Can you use the result for some other problem?

Understand the Problem The description of the initial problem includes markup elements `situation-description` and `problem-statement`. The first element describes what is given and what it depends on. Dependencies can be provided in the metadata of `situation-description`. The second element encodes the question (statement) of the problem, i.e. what has to be found, proven, etc. These elements prove to be useful not only for faded examples.

Devise a Plan We use slightly modified OMDoc markup in order to simulate the plan of the solution. For this, each step of the solution might not directly contain the actual calculation or derivation, but an intermediate step, containing a brief description of one or more steps of the solution. The `derive` element encodes this step and may contain a child element `solution` for a sub-solution or just group a sequence of steps. We group all the child elements of the `solution` in the entity called `solutionobject` which is analogous to the `proofobject` entity in OMDoc.

Note that not only the plan of the solution can be encoded in this way, but more complex solution plans may consist of sequence of sub-solution plans.

Carry out the Plan The sequence of bottom nodes of the `solution` element is the actual solution. In the encoding of the solution the steps, carrying out the plan, occur inside the corresponding plan steps, in the presentation they can be separated from the plan steps, if wished.

Look Back at the Solution Here, an element `conclude` is used. This element has the same meaning as in OMDoc and is used not only if the solution is the proof of some fact. For example, if the root of the equation is calculated in the solution, in the `conclude` step one should substitute the result found in the solution into the original equation and verify its soundness.

The reference to other problems for which the result of the current problem can be useful, is provided in the metadata record, as discussed below.

Figure 1 shows the internal representation of the Example 2 [2], embedded into the Polya framework. The bold case shows the actual steps of the exercise, italic shows additional steps, introduced for building the Polya framework.

5.3 Adding Metadata

In order to enable adaptive generation of faded examples we need to know what to fade according to capability of the learner and in which learning situation.

The characterization by a learning-goal level is necessary in order to fade adaptively w.r.t. the learning goal of the learner, other properties such as difficulty and abstractness of the step or its part are important parameters to compare to the skills of the learner before fading.

According to the first phase in the Polya framework we have to assign dependencies to the `situation-description` element. For the last phase we also need to provide the references to other problems for which the result of the current one can be useful.

Metadata records are possible for each structural element of the solution. Moreover, metadata can be assigned to parts having `with` containers labeled with identifiers.

A metadata record consists of common ActiveMath metadata elements, such as `difficulty`, `abstractness`, `competence-level` (for learning goal level), `relation` of types *depends_on*, *is_useful_for*, and others.

The described knowledge representation provides the basis for automatically generating faded examples as a type of exercises and for integrating such exercises into a learning material or into a suggestion.

6 (Adaptive) Generation of Faded Examples

Varying the faded places in examples is more interesting and less schematic for the learner. In addition, adapting the actual fading with a specific purpose in mind adds value to faded examples. The adaptivity has at least two dimensions: the choice of the worked example to be faded (e.g., depending on the interests and ability of the learner) and the choice of the gaps to be introduced.

[2] Mathematical formulas in OMDoc are represented in OpenMath format, but in this paper we shorten them due to lack of space

```
<example id="div_seq_1" for="def_divergence">
<metadata>...</metadata>
 <situation-description>
  <CMP>Given the sequence ((−1)^n) </CMP>.
 </situation-description>
 <problem-statement>
  <CMP> The given sequence is divergent. </CMP>
 <problem-statement>
<solution>
 <metacomment id="c1" type="alternative">
   <CMP>This sequence X := ((−1)^n) is bounded so we
    can not invoke Theorem 3.2.2.</CMP>
  </metacomment>
 <derive id="plan_step1">
  <CMP>Assume the sequence is convergent</CMP>
  <hypothesis id="hyp1" discharged-in="last_step">
   <CMP> Assume that a := limX exists. </CMP>
  </hypothesis>
 </derive>
 <derive id="plan_step2">
  <CMP>Come to the contradiction.</CMP>
   <hypothesis id="hyp2" discharged-in="step_1">
     <CMP> Let ϵ := 1 </CMP>
   </hypothesis>
   <derive id="step1">
    <CMP>so that there exists a natural number K_1 such that
    <with id="obj1"> |(−1)^n − a| < 1 </with> for all
    <with id="obj2"> n ≥ K_1 </with> </CMP>
   </derive> ...
   <conclude id="last-step">
     <CMP>Since a cannot satisfy both of these inequalities, the
     hypothesis that X is convergent leads to a contradiction.
     Therefore, the sequence X is divergent. </CMP>
   </conclude>
 </derive>
<conclude />
</solution>
</example>
```

Fig. 1. OMDoc Example enhanced with Polya-structure in the solution

```
<metadata for=”c1”>
 <competence-level use=”meta_cognition”/>
 <relation type=”depends_on”><ref xref=”thm_3_2_2”></relation>
</metadata>
<metadata for=”hyp1”>
 <competence-level use=”knowledge”/>
</metadata> ... (the same for ”hyp2”)
<metadata for=”step1”>
 <difficulty level=”fair”/>
 <competence-level use=”application”/>
 <relation type=”depends_on”><ref xref=”def_limit”></relation>
</metadata>
<metadata for=”obj1”>
 <difficulty level=”easy”/>
 <competence-level use=”application”/>
 <relation type=”depends_on”> <ref xref=”def_limit”></relation>
</metadata> ... (the same for ”obj2”)
```

Fig. 2. Sample Metadata Record for Solution Steps

Choice of Fading The structure of the worked-out example determines the possibilities of fading. The annotation of fadable parts gives rise to a reasoning about the choices depending on the purpose of the faded example.

Currently, for adaptation we consider mastery of the concept, learning history and the learning goal-level (knowledge, understanding, application, and meta-cognition). This information is available in ActiveMath' user model.

The rules we use for fading are still prototypical and not tested with students. They are used for testing generation of exercises from faded examples.

In a nutshell, the reasoning underlying those fading rules includes

- if a concept or rule C is in the current learning focus [3] and if the mastery of C is at least medium, then fade one or several parts which require C as a prerequisite
- if low-ability student, then prefer fading steps backwardly in the solution or fading conditions
- if low-ability student, then prefer fading parts *inside* a problem solving step rather than parts *between* steps
- if the learning-level goal is *knowledge*, then fade parts of problem statement, sub-goals, known assumptions or used facts
- if the learning-level goal is *comprehension*, then fade reference, justification for a step, explanations, or auxiliary information
- if the learning-level goal is *application*, then fade a step, a condition of a step, or a sub-solution, (sub)goal statements

[3] focus concept as defined in [8]

- if the learning-level goal is meta-cognition, then fade links to other instructional items, meta-cognitive structure (headlines), or meta-cognitive heuristics
- start with smaller gaps and enlarge them gradually towards the end of exercising (i.e., depending on the learning history)

The collections of fading 'rules' will be enlarged as soon as we gain more experience with students.

Example Consider Example 2:

$$((-1)^n) \ \ is \ \ divergent$$

`Proof.`

(1) *This sequence* $X := ((-1)^2$ *is bounded (take* $M := 1$*), so we cannot invoke Theorem 3.2.2...*
(2) *...However, assume that a:=lim X exists. ...*
(3) ... Let $\epsilon := 1$...
(4) *...so that there exists a natural number* K_1 *such that* $|(-1)^n - a| < 1$ *for all* $n \geq K_1$*...*

We assume that the example is represented as in the Figure 1 using the metadata records from the Figure 2 and the rules above we discuss the adaptive generation of a faded example.

Part (1) contains reasoning about alternatives. In the extended OMDoc this is characterized as a `metacomment` of a type *alternative* and can be faded, if the learning goal is meta-cognition.

Parts (2) and (3) consist of two `hypothesis` elements. Each of these elements can be faded if the learning goal is knowledge. By fading $|(-1)^n - a| < 1$ in (4) or even the whole step the application of the definition of the limit can be trained. As we see from the metadata records in the Figure 2, fading Step1 results in a more difficult exercise than fading parts ('obj1', obj2').

The result of the fading procedure is an exercise. Each of the derive steps becomes an interaction in that exercise. This interaction has all the information, needed for fading: the place to be faded is marked, the type of interactive element to be placed instead is provided in the `interaction_map` element. The evaluation of the user is described in the `answer_map` element, where the correct answer is compared to the input of the user, and depending on it, the next interaction is introduced.

7 Conclusion and Future Work

One obvious alternative to present different faded examples to students is to select pre-defined / handcrafted ones dynamically. The heuristics for adaptive choice would be very similar to those informing the generation. However, this approach would require to predefine and store all faded examples (in advance) and to characterize each one. Moreover, hand-crafting a variety of elaborate faded examples is a very skillful and laborious work.

In order to improve learning within the learning environment ActiveMath we build on the results of empirical investigations of cognitive psychology about learning with faded examples. Interestingly, a truly user-adaptive presentation of faded examples has not been considered in psychological experiments. Renkl suggests backward fading for every learner rather than adaptively to a learner's problem solving capability or solving strategies. Some of Stark's results in [13] do imply the need for even more adaptivity. He indicates that informal results of the experiments suggest to vary the fading parts in order to increase the motivation of the learner.

Therefore, we investigated *what* can be faded. For this, we extended the annotated semantic XML-representations for mathematical examples and exercises underlying ActiveMath and refined their internal structure. Then, we introduce gaps according to a few variables namely mastery of a concept, general ability, and learning-goal level. The automatic generation can easily be extended because the rules for the procedure are stored separately. Moreover, we include the possibility to manually determine what to fade because teachers/authors have a lot of experience on how to 'fade' worked examples and they might want to be in command.

Storing the original example in the knowledge base and only generating the fading provides a relation between faded-example exercises and their root example which can be very useful in dynamic course generation and for direct comparison.

We presented the results of the first phase of a long-term research on fading examples. Therefore, we want to point out open question and future work.

An improved generation will have to consider the chosen pedagogical strategy as well as

- other characteristics of the learner, e.g., exploratory behaviour, preferred problem solving strategies, and motivation
- more detailed cognitive and meta-cognitive capabilities of the learner, e.g., spontaneous self-explanation.

The domain knowledge and the underlying content model including the dependency of concepts will also influence the fading process.

The rules for adaptive fading are still experimental. Therefore, the next steps are to evaluate them. The suitability of the rules can be evaluated theoretically and empirically and this includes:

- test generated exercises with students (e.g. compared with pure backward fading)
- confront teachers with a student's characteristics and compare his fading with the automatically generated.
- Additional explicit knowledge representation has to be compared with results of manual fading.

Related Work The natural-language part of example generation has been addressed by [4]. It generates a natural language example solution and then introduces gaps into that solution according to the user model's predictions about the mastery of a rule. These gaps are restricted to propositions corresponding to primitive communicative actions of a particular explanation strategy (e.g., Polya-like structuring elements are not planned), not dependent on preferences of the learner or other purposes (such as learning goal), and not as flexible as authors' annotations in given examples. It may be possible to unify this language-based approach with ours that is based on the structure and annotation of given worked example representations.

References

1. R.G. Bartle and D.R. Sherbert. *Introduction to Real Analysis*. John Wiley& Sons, New York, 1982.
2. B.S. Bloom, editor. *Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain*. Longmans, Green, New York, Toronto, 1956.
3. J. Büdenbender, G. Goguadze, P. Libbrecht, E. Melis, and C. Ullrich. Metadata in activemath. Seki Report SR-02-02, Universität des Saarlandes, FB Informatik, 2002.
4. C. Conati and G. Carenini. Generating tailored examples to support learning via self-explanation. In *Seventeenth International Joint Conference on Artificial Intelligence*, 2001.
5. G. Goguadze, E. Melis, C. Ullrich, and P. Cairns. Problems and solutions for markup for mathematical examples and exercises. In A. Asperti, B. Buchberger, and J.H. Davenport, editors, *International Conference on Mathematical Knowledge Management, MKM03*, LNCS 2594, pages 80–93. Springer-Verlag, 2003.
6. M. Kohlhase. OMDoc: Towards an internet standard for the administration, distribution and teaching of mathematical knowledge. In *Proceedings Artificial Intelligence and Symbolic Computation AISC'2000*, 2000.
7. M. Kohlhase. OMDoc: Towards an OPENMATH representation of mathematical documents. Seki Report SR-00-02, Fachbereich Informatik, Universität des Saarlandes, 2000.
8. E. Melis and E. Andres. Global feedback in ActiveMath. In *Proceedings of the World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (eLearn-2003)*, pages 1719–1724. AACE, 2003.
9. E. Melis, E. Andrès, J. Büdenbender, A. Frischauf, G. Goguadze, P. Libbrecht, M. Pollet, and C. Ullrich. ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education*, 1002(4):385–407, 2001.
10. E. Melis and C. Ullrich. How to teach it – Polya-scenarios in ActiveMath. In U. Hoppe, F. Verdejo, and J. Kay, editors, *AI in Education, AIED-2003*, pages 141–147. IOS Press, 2003.
11. G. Polya. *How to Solve it*. Princeton University Press, Princeton, 1945.
12. R.K.Atkinson, A. Renkl, and M.M. Merrill. Transitioning from studying examples to solving problems: Effects of self-explanation prompts and fading worked-out steps. *Journal of Educational Psychology*, 2003.

13. R. Stark. *Lernen mit Lösungsbeispielen.* Münchener Universitätsschriften,Psychologie und Pdagogik. Hogrefe, Göttingen, Bern, Toronto, Seattle, 1999.
14. J. Sweller and G.A. Cooper. The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction*, 2:59–89, 1985.
15. J.J.G. van Merrienboer and M.B.M. DeCrook. Strategies for computer-based programming instruction: Program completion vs. program generation. *Journal of Educational Computing Research*, 1992.