Automatic exercise sequencing-based algorithmic skills

Meriem Abdessemed*, Tahar Bensebaa and Takie Eddine Belhaoues

Computer Science Department,
Laboratory of Research in Computer Science (LRI),
Badji Mokhtar Annaba University,
Annaba, Algeria
Email: abdessemed.myriam@gmail.com
Email: t_bensebaa@yahoo.fr
Email: te.belhaoues@gmail.com
*Corresponding author

Anis Bey

Computer Science Department,
Laboratory of Research in Computer Science (LRI),
Badji Mokhtar Annaba University,
Annaba, Algeria
and
Ecole Supérieure des Sciences de Gestion (ESSG),
Annaba, Algeria
Email: anis.bey@univ-annaba.org

Abstract: In any learning systems and especially automated assessment tools, the most common task is to evaluate the students’ performance using training exercise. The selection of the next exercise is generally performed as a static set of exercises or free by students. But, it would clearly be advantageous if this exercise selection process were to be automated based on their previous performances. Therefore, the focus of this paper is the development of a method capable of determining exercise progression and sequencing during a training session based on the students’ past performance. A dynamic planning of algorithmic exercises was developed based on a semantic and pedagogical description to be used in training exercise.

Keywords: sequencing; algorithmic; assessment; learning programming; exercises.


Biographical notes: Meriem Abdessemed is a PhD student at the University of Badji Mokhtar Annaba and she prepared her PhD thesis at the Laboratory of Research in Computer Science (LRI). Her PhD research treats the teaching of algorithmic skills based on the learning by doing method, by adapting the exercises sequencing to a learner profile.
Tahar Bensebaa is a Professor of Computer Science at the University of Badji Mokhtar Annaba. He received his Doctorate in Computer Science in 1992 from INSA Lyon, France. He is currently at the Head of the research group of Technology Enhanced Learning at the Laboratory of Research in Computer Science in the University of Badji Mokhtar Annaba, whose objective is the development of a TEL environment dedicated to teaching algorithms.

Takie Eddine Belhaoues is a PhD student at Badji Mokhtar Annaba University and he prepares his PhD thesis at the Laboratory of Research in Computer Science (LRJ). His PhD research focuses on proposing a semantic description and organisation of assessment resources in the algorithms filed by adapting the meta-data standards for learning objects description (LOM, IMS QTI, Dublin Core …) to obtain specific application profile and then assist actor in the indexing process to facilitate this laboured task.

Anis Bey is currently an Associate Professor at Ecole Supérieure des Sciences de Gestion Annaba, Algeria since 2015 and a member in the research group of Technology Enhanced Learning at the Laboratory of Research in Computer Science in the University of Badji Mokhtar Annaba. He obtained his PhD thesis in 2013 from the University of Badji Mokhtar, Annaba, Algeria. His current research projects focus on automated assessment in programming, MOOCs and learning analytics.

1 Introduction

Without realising it, algorithmic such as math is used in our daily life. We execute algorithms in following a cooking recipe or instructions manual. We also develop and execute them when we specify a path for a lost person, etc. Nevertheless, in education, it is often evoked as a difficult discipline to teach and to learn; some people could even consider it as a very complex field. However, it is the basis of any program or computer system. A study conducted at Badji Mokhtar Annaba University (UBMA) compared the success rate of the students in algorithmic with other modules taught in the same year. The result showed a failure rate of 75% for two consecutive years concerned by the study (Bensalem and Bensebaa, 2010). This failure is also observed in other studies (Lister et al., 2004; McCracken et al., 2001). In front of this problem, this study comes in the framework of a project that aims improving learning algorithmic in UBMA University. It seeks to develop a dedicated system, based on a problem-solving learning, to combat the learner’s stereotypy, passivity and logic of failure (Bey and Bensebaa, 2013; Bouacha and Bensebaa, 2015; Belhaoues et al., 2016).

In algorithms, solving a problem need being methodical and have a certain intuition, which means to have some experience. These abilities are necessarily acquired through practice. In fact, we noted in our research that the theory of ‘learning by doing’ fosters skill development and learning factual information in the context of ‘how’ it will be used (Schank et al., 1999). “For the things we have to learn before we can do them, we learn by doing them” (Barnes, 1976). This perfectly reflects the borrowed strategy in this project for the Algorithm teaching. Indeed, since this field is constituted not only of knowledge but mainly of know-how, so learning cannot be reduced only to learn this knowledge.
An automatic assessment tool called Algo+ (Bey and Bensebaa, 2011) was the first experience in the project. The tool is centred on a problem-solving learning situation and algorithmic skills assessment. For a given exercise, the system provides a learner an editor to formalise its algorithmic solution that is evaluated automatically. This assessment has a dual role:

- assessing learner’s solution by providing a score and provide feedback
- allowing students to practice programming, because this know-how is acquired by the approach ‘learning by doing’.

In another part (Belhaoues et al., 2016) supported that the best way to learn and to assimilate the algorithmic concepts and skills is practicing on exercises treating the good pedagogical objectives (POs) and corresponding to students’ profiles. To respond to this need, two objectives were exposed.

The first need supported in Belhaoues et al. (2016) concerned the build of an algorithmic exercises-base semantically and pedagogically organised, based on an ontology of algorithmic notions and skills named ‘AlgoSkills’. The domain concepts (notions and skills) and the relations between them are exploited in a formal and explicit structure to enhance the description of each exercise provided to students through Algo+ (Bey and Bensebaa, 2011).

The second need, which is the main of this paper, is to remedy to the traditional proposal of exercises that are given in a logical scenario for all students, without taking into account the knowledge and skills of each one. Some might untie them, others much less. In fact, we found that some students are often discouraged by their inability to develop good algorithms or at least correct and valid ones (Lack of practice, bad use/combination of knowledge and skills …). While others students might get bored during an assessment activity since they have already mastered the skills of the exercise.

In this paper, we present an approach that allows proposing one or several adequate exercises for a learner in a learning situation. Indeed, those skills extracted from the ‘AlgoSkills’ ontology (Belhaoues et al., 2016) are used to describe the cognitive learner profile (acquired, not acquired and being acquired skills), the educational objectives (targeted skills) and the exercise-base (semantically and pedagogically organised).

The presented approach is based on the curriculum sequencing technique. The latter is used to build adaptive learning resources whose main objective “is to provide the students with the most suitable individually planned sequence of knowledge units to learn and sequence the learning tasks to work with” (Brusilovsky, 1999). In this process, it is necessary to take into account the needs and the learning profile of each student to determine the accommodations and adaptation that will be necessary for the student success (Calgary Learning Centre, 2003).

Curriculum sequencing can be seen as a two-part process: deciding on relevant topics based on the current student model, and then selecting the best one. Our work is oriented learning by problem solving; therefore we focus on exercises sequencing technique, in which we propose to determine the relevant exercises, order them (according to a set of parameterised rules) and then select the best(s) one(s) for a given student. Selecting the adequate exercise is based on a set of pedagogical rules. These rules take into account the learner profile, the PO of the learning situation and the exercises-base (Figure 1).
After having introduced the context and motivation of the proposed work, the following parts of this paper will present: in Section 2 some related works of the sequencing learning process in problem-solving systems and Intelligent Tutoring Systems. In Section 3 we introduce the proposed exercises sequencing approach, detail the components of the general architecture, and present the implementation of the sequencing process. Finally, Section 4 concludes this paper and gives some orientations for future works.

2 Related works

We introduce here some related works identified as relevant in the fields: problem-solving systems and personalised learning content systems. At the author’s knowledge, no Adaptive learning systems are offering algorithmic exercises sequencing.

Many systems dedicated to problems solving exist in the literature. We mention among these systems; PROUST (Johnson, 1990), JPie (Goldman, 2003), Iconic program (Chen and Morris, 2005), Exercises DataBase about Algorithms (EDBA) and SQL-Tutor (Mitrovic, 1998, 2003). We focused on these systems to identify their exercises selecting and sequencing techniques. The first three mentioned systems are learning environments for programming mainly devoted to assistance in the program development; return the feedback to learners and proposing new techniques for solutions assessment. However, we note that no system among them offers exercises proposal process.

In addition, there are various systems dedicated to personalised learning content or course sequencing, not especially exercises, that close to our architecture. We cited among them, ActiveMath (Melis et al., 2001; Barla et al., 2010), APCL (Salahli et al., 2013; Yarandi et al., 2013; Chaplot et al., 2016).

A description of a few systems is addressed, according to the three fundamental elements; Domain knowledge and exercises-base, learner profile, and adaptation technique.

The first element core of any learning system is the knowledge domain. SQL-Tutor (Mitrovic, 1998) is considered as one the most referenced ITS and it supports students in
learning SQL. The knowledge about the domain is represented as a set of constraints. In ActiveMath (Melis et al. 2001), a generic web-based tutoring system that generates interactive mathematical courses, the appropriate content is retrieved from a knowledge-base. Salahli et al. (2013) use a knowledge-base of topics to be learned associated with a difficulty level. We have noticed that in Chaplot et al. (2016), authors propose a method for selecting practice items of optimal difficulty. The domain knowledge consists of a set of concepts and items (question, quiz…) which are represented in an acyclic pre-requisite graph, denoting the order in which concepts need to be mastered. As regards, the domain knowledge in Yarandi et al. (2013) is a semantic ontology. It contains classes and properties that describe topics of a domain and pedagogical relationships between proposed topics.

For the second element, each system has its own modelling. Bouhinau (2010) do not propose a learner profile. Students are identified by a login and password; they follow a linear path for solving exercises. Chaplot et al. (2016) propose a learner model using an artificial neural network, which receives as input student performance data (item ID, student responses …). In Yarandi et al. (2013) an ontological user model is designed to describe learners’ profiles, in five principal classes (personal information, learning style, preferences, abilities, and performance).

Regarding the adaptation approach, different scenarios are proposed. In Mitrovic (2003) the system proposes appropriate problems to students on the basis of his model. For that, SQL-Tutor examines the student model and selects a problem for constraints that the student is not sure about. Another way to pose a problem is the use of a constraint that has not been used by the student so far. Melis et al. (2001) provides adaptive content and presentation of the course, including exercises, with the appropriate difficulty level, according to a set of pedagogical rules. In the programming environment EDBA (Bouhinau, 2010), exercises in Prolog and Caml language are available to students and are categorised by difficulty levels. To advance to a higher level, the learner has to solve a set of exercises of the current level to accumulate points; otherwise, the highest level exercises remain inhibited. Chaplot et al. (2016), Salahli et al. (2013) and Barla et al. (2010) use the probabilistic model ‘item response theory’ (IRT) to select the appropriate items (K-best questions) with adequate difficulty level, for a particular learner. By using the IRT method, understanding degree of the course topics for each student is estimated (Salahli et al., 2013). For example, when this parameter is low, the system offers the student the learning material with lower difficulty; otherwise, the learning process is continued on the original list of the learning materials proposed beforehand.

3 The proposed approach for sequencing exercises

The approach presented in this paper aims to remedy to the traditional proposal of exercises that are given in a logical scenario for all students, without taking into account the knowledge and skills of each one.

The proposed approach is based on the curriculum sequencing technique, where it is necessary to consider the needs and the learning profile of each student to determine the accommodations and adaptation that will be necessary for the student success (Calgary Learning Centre, 2003).

The context of algorithm learning which promotes the learning by doing led us to reflect in terms of skills. Indeed, these skills extracted from the AlgoSkills ontology
Automatic exercise sequencing-based algorithmic skills (Belhaoues et al., 2016) are used to describe the learner profile, the educational objectives and the exercise-base.

We therefore developed a skills-based algorithmic exercise sequencing approach offering an adequate exercises sequence, depending on:

- learning profile (what s/he knows?)
- learning objectives: final profile (What s/he has to learn?)
- pedagogical strategies (a set of adaptation rules)
- exercises-base described and indexed by the AlgoSkills ontology.

Figure 2 General architecture (see online version for colours)
Figure 2 presents the general architecture. The latter is structured according to the roles played by the different actors of the sequencing process.

As it is described in Figure 2, we can depict the functional architecture according to information needed for the method and how it works.

3.1 What the sequencing method needs to be functional?

In this section, we present the sequencing algorithm developed in this work. An example of a use case and some screenshots of the sequencing algorithm implementation are also presented. Based on; educational objective, learner model, exercises-base, rules and execution parameters, the algorithm performs a set of operations in order to achieve the desired results.

- Denomination and abbreviation
  
<table>
<thead>
<tr>
<th>Denomination</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>knowledge</td>
</tr>
<tr>
<td>S</td>
<td>skill</td>
</tr>
<tr>
<td>E</td>
<td>exercise</td>
</tr>
<tr>
<td>EO</td>
<td>exercise objective (skills to check)</td>
</tr>
<tr>
<td>AS</td>
<td>acquired skill</td>
</tr>
<tr>
<td>NAS</td>
<td>not acquired skill</td>
</tr>
<tr>
<td>LM</td>
<td>learning model</td>
</tr>
<tr>
<td>LP</td>
<td>learner profile</td>
</tr>
<tr>
<td>DL</td>
<td>difficulty level</td>
</tr>
<tr>
<td>SPR</td>
<td>skills pre-requisite</td>
</tr>
<tr>
<td>His</td>
<td>history</td>
</tr>
<tr>
<td>ET</td>
<td>exercise type (H/T)</td>
</tr>
<tr>
<td>PO</td>
<td>pedagogical objective</td>
</tr>
</tbody>
</table>

We worked with XML databases of learner profiles and learning objectives, and we describe a set of exercises with the indexing tool presented in Belhaoues et al. (2016). We also instantiated the skills and topics in the ‘AlgoSkills’ ontology at the lowest level, since the ontology property ‘ObjectProperty’ must link instances to be used in our algorithm.

3.1.1 A base of algorithmic exercises

An exercise activity allows developing a person’s abilities in a given field, and applying what was learned to be certified, and thus progress. Algorithmic exercises aim to verify the acquisition and mastery of skills and reasoning ability on a given problem as a logical sequence of operations.

The first need is to describe algorithmic exercises and determine properties to be taken into consideration in ordering exercises.

The description of the exercises is supported by the teacher of the domain, one of the main actors occurring in the system. The teacher use Index-Exo tool developed in (Belhaoues et al., 2016), providing an editor for navigating in the ontology, along with the description of each exercise and its instantiation in AlgoSkills. The first version of the
Automatic exercise sequencing-based algorithmic skills

A tool allowed describing each exercise with an identifier ‘ID’, a statement, and a set of knowledge (topics) and skills which are related to it. A graphical view allows the user to visualise the concepts and relationships between them with displaying their definition.

For our needs, we have enhanced the description of the exercises. Table 1 summarises the new description.

**Table 1** Exercise description

<table>
<thead>
<tr>
<th>Exercise</th>
<th>Ontology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td>DataProperty: ID</td>
</tr>
<tr>
<td>Statement</td>
<td>DataProperty: statement</td>
</tr>
<tr>
<td>Related knowledge</td>
<td>ObjectProperty: traite</td>
</tr>
<tr>
<td>Class: topics</td>
<td></td>
</tr>
<tr>
<td>Skills to check</td>
<td>ObjectProperty: Has_objective</td>
</tr>
<tr>
<td>Class: skills</td>
<td></td>
</tr>
<tr>
<td>Pre-requisite skills</td>
<td>ObjectProperty: pre-requisite</td>
</tr>
<tr>
<td>Class: skills</td>
<td></td>
</tr>
<tr>
<td>Type: help/trick</td>
<td>DataProperty: type</td>
</tr>
<tr>
<td>Difficulty level (0 to 10)</td>
<td>DataProperty: difficulty</td>
</tr>
</tbody>
</table>

In addition to the previous descriptors, the new description includes: the related knowledge corresponding to the treated topics in the exercise. Skills to check representing the exercise objective in terms of skills it tends to verify. Pre-requisite skills of an exercise correspond to skills that the learner must possess to be able to solve it. The descriptor ‘type’ specifies context where the exercise will be proposed. For example, exercises with the Type ‘help’ are proposed at a remediation strategy which will be detailed below. The difficulty level of the exercise varies from 0 to 10.

### 3.1.2 A learner profile

During an educational process where personalisation aimed at strengthening and promotes learning, the learner is the main factor to be taken into account in different aspects. This consideration is important for the tutor in its decisions.

The learner profile can be thought of as an overlay on the domain model, the ontology of algorithmic skills in this study (AlgoSkills). It represents mainly learner’s cognitive state and its evolution during the training exercise. As the student performs a set of exercises through their problem-solving process, student’s profile is updated by the status of knowledge and skills as follows.

- **knowledge « K »:** represents the basic notions usually taught in algorithmic courses
- **acquired skills « AS »:** these are skills progressively acquired by a learner, during the training exercise, and which are thus considered mastered
- **skills in progress « SIP »:** as their name suggests, these are skills that the learner is working on and has not perfectly acquired
- **not acquired skills « NAS »:** are skills that have been used in an inappropriate manner.
In addition, a learner profile also contains the history of the previously solved exercises. This parameter is to be taken into consideration in the sequencing process (for not proposing ever solved exercise). Furthermore, each exercise has a difficulty level ranged from 1 to 10. This value indicates which exercises the student will be able to solve in terms of difficulty.

3.1.3 Base of pedagogical rules

The pedagogical rules constitute the heart of the exercises sequencing approach. The pedagogue is the responsible for defining the set of rules. Two kinds of rules are available. The first kind based on the condition-action rules is used for the selection of the progression approach to be applied (immediate remediation, flexible remediation). The rules of the second kind are based on the first-order logic for the selection of exercises. The rule-base is upgradeable; some rules can be adapted or created and added. Pedagogical rules are made available to the teacher and used to define strategies according to the learning situation.

A first set of customisable pedagogical rules is included in the system. These rules allow the choice of the most suited exercise to each learner. The translation of these rules in predicate logic is presented later in this paper. They are used by querying the exercises-base (OWL1 file) with the query language SPARQL and Jena API.

Here some proposed Rules:

Rule 1 The objective of the exercise must be a subset of the overall objective.

Rule 2 Exercise’s knowledge must be acquired knowledge in the learning profile.

Rule 3 The pre-requisites skills of the exercise must be among the acquired skills in the learner profile.

Rule 4 The exercise difficulty level must be less than or equal to that of the learner.

Rule 5 Do not propose an exercise belonging to history.

R1: \( WObj \ (EO \subseteq PO) \) where \( WObj \) Pedagogical Objective weight

R2: \( WK \ (K.EO \in K.LP) \) where \( WK \) Knowledge weight

R3: \( WSPR \ (SPR.E \in ASLP) \) where \( WSPR \) Prerequisite weight

R4: \( WDL \ (DL.E \leq DL.LP) \) where \( WDL \) Difficulty Level weight

R5: \( WHis(E \notin His.LP) \) where \( WHis \) History weight

3.1.4 Pedagogical objective

Any assessment that does not meet the training objectives is considered inappropriate and demotivating for the learner. The POs are essential for the success of a learning process and represent what we should be able to accomplish at the end of this process. The definition of objectives concerns each module but also, inside a module, each session, each sequence and each exercise (Poitrat, 2011). Defining the objectives is supported by the specialist in pedagogy ‘pedagogue’.
In order to define POs in algorithmic, we worked with a group of experimented teachers of the computer science department at UBMA. POs have been associated to the algorithmic skills that the learner has to acquire throughout its learning. In fact, we categorised POs into three classes; each class is represented by a set of skills extracted from the ontology. Two distinct objectives may share the same skill. This decomposition can be revised by the pedagogue:

- **overall objective 1 (elementary skills):** the first overall objective includes the elementary structures representing the basic skills in algorithmic: read a data, write a data, assign a variable, declare a variable, initialise a variable
- **overall objective 2 (control structure skills):** it encompass the skills on conditional (tests) and iterative (loops) structures: use a simple test, use if_then_else, use loop for, use loop while, use loop do_while, choose between a simple test and multiple tests, choose between loop for and while
- **overall objective 3 (data structure skills):** consists of the skills associated with the handling of data structures: implement, add, lookover, delete.

The selection of the current PO is based on the skills already acquired and specified in the learner profile; indeed, an objective is achieved when all of his skills are acquired. The skill of the current objective is selected by default, corresponding to the first skill not yet acquired in the learner profile.

### 3.1.5 Base of pedagogical strategies

In addition to the description of the exercises, the teacher will have to associate for each situation [a learning model (LM) and a PO stored in their respective bases], a pedagogical strategy (PS) to be applied. The last one constitutes a set of rules and parameters to be informed. Indeed, for each combination of a LM and a PO, the teacher should choose the appropriate progression approach, select and weight the rules adapted to this situation among those previously defined by the pedagogue (condition-action rules).

Each learner profile will close one of the standard models, and according to the current PO, the associated PS will be applied to his profile (Figure 3).

**Figure 3** Appropriate strategy for a learner (see online version for colours)
3.2 How the sequencing process does work?

The main steps that make the sequencing process are described as fellow:

3.2.1 Parameter not acquired skills max

The first step of the sequencing process is to recover from the learner profile, the number of not acquired skills. This number is compared to the not acquired skills max (NASmax) parameter defined by the teacher. Sequencing exercises is done if the number of not acquired skills does not exceed the NASmax. Otherwise, the system may not provide exercises to the learner. The learner profile is then sent to the teacher for review to take the appropriate decision to this case.

3.2.2 Choosing a PS

As mentioned in Section 3, the teacher chooses the appropriate progression approach during the ‘definition of the PS’. These are two proposed approach.

**Progression with flexible remediation**

In the progression with flexible remediation (PFR) approach, the algorithm runs as long as the number of not acquired skills does not reach the NASmax threshold.

**Progression with immediate remediation**

During the learning with immediate remediation, not acquired skills must be supported before continuing the progression (not acquired skills in the learner profile = 0). The application of the progression with the immediate remediation is based on the learners and the POs. Remediation is started first for the most committed mistake (not acquired skill). Indeed, for each non-acquired skill in the learner profile and described in the ontology, the system recovers through the semantic relation ObjectProperty ‘due to’, the skills that are involved. It selects then from the base, exercise type ‘help’, which aims to verify this expertise and whose knowledge and pre-requisites are checked.

\[ \text{WHILE} (\text{NAS} > 0) \{ \text{EO} = \text{NAS}, (\text{K.E} \in \text{K.LP}) \land (\text{SP.R.E} \in \text{AS.LP}) \land (\text{ET} = \text{H}) \} \]

The teacher can consult at any time the learners’ profiles and view the progress of these in the system. His main role is defining the base of pedagogical strategies for learners’ models and POs using the rules-base proposed by the pedagogue. In some cases, the teacher operates to:

1. The questioning of the applied strategy: a strategy may not necessarily match the learner profile. In this case, the teacher can make changes in the settings of one or more rules of a strategy.

2. Making decision on the current state of the learner, for progression/remediation/course. Indeed, teacher consults the concerned learner profile and decides if the learner is able to continue progressing (practicing) or has to be redirected to the courses.
3.2.3 Filtering and assigning weight

Initially, it was defined that exercises may be available when $R_1 \land R_2 \land R_3 \land R_4 \land R_5$ are verified. However, it has been noticed that this general rule is restrictive and very selective. This solution cannot provide results in all cases, and even would not be suitable for some learning situation (example: an exercise may be solved even if its pre-requisites are not perfectly acquired. To have the intention to give exercises previously proposed exercises, etc.). Therefore, we propose to change this restriction by giving variables weights for each rule. These weights are assigned to exercises by addition (weight accumulation) in each step, according to the PS to apply.

By filtering, we mean the elimination of the exercises that do not meet the rules. The selected exercises (after filtering) are judged pertinent on which future actions will be performed. Only rules that have no weight are considered by the filtering process. The rest will be used for weighing exercises. Values of weights can take one of the following increasing values $\{0, 1, 5, \text{and } 10\}$ depending on the rule importance.

The algorithm recovers the affected weight: \textit{objective, knowledge, pre-requisites, difficulty level and finally history}; allocated to exercises depending on the validity of the associated rule. Negative weights are assigned if the rule is invalid.

To avoid obtaining an insufficient number of candidate exercises to be ordered after applying filtering, the teacher has to determine a threshold on which a test is performed at each requested filter. The obtained list, after applying the filter, is only considered if the number of exercises it contains is superior or equal to the threshold (Figure 4).

\textbf{Figure 4}  Applying filter on a set of exercises (see online version for colours)

3.2.4 Ranking exercises

After filtering and assigning weights for each selected exercise, following the above steps, a final list of weighted exercises result. A selection sort by maximum is achieved on the latter. Indeed, the most relevant exercises, and obviously the best, will appear first. In some cases, a considerably significant difference could be found between weight’s exercises. Example $(+30)$ and $(-30)$ in Figure 5.

\textbf{Figure 5}  Example of ranked exercises (see online version for colours)

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure5.png}
\caption{Example of ranked exercises (see online version for colours)}
\end{figure}
For each profile, a final selection parameter is assigned by the pedagogue. For an example that a novice learner would need practicing over several exercises to validate a skill, while another type of learner needs a less number or even exercises that aim to check more than one skill at a time. Currently, are selected the first three exercises of the final resulting list. But when the gap between the weight of an exercise with his successor is greater than nine (>=9), the latter is removed (not proposed).

An illustrative example

When the PO is considered important, exercises that do not fit into the objective should not be considered. A first filter is then performed on the entire exercises-base, with regard to the skill of the current overall objective (Rule1 checked (EO ⊂ PO)). If a Skill in Progress is identified in the learner profile, meaning that it is not yet acquired, WSIP weight is recovered, defining that filtering should take into account exercises corresponding to this skill. The latter may change over time into acquired or not acquired skill. In contrast, Rule1 could be weighted, so WObj might take one of these values 1 v 5 v 10. In this case, there is no filtering process, but a weight is given according to the consequence that could have the PO during the customisation of the course.

- We support that we cannot suggest exercises including concepts that have not been yet studied. This is disapproved by learners during the tests and exams, in this case, the rule R2 is considered as a filter. The second filter is then performed if Rule2 is verified (K.EO ∈ K.LP). This filtering is performed, on the subset previously obtained in the first selection or on the entire-base (if Rule1 was not used), depending on the learner knowledge (in the learner profile). However, WK may have the value 0, 1, 5, 10 in another strategy for a given model, and a given learning objective. This numeric weight is then assigned to the exercise satisfying the rule. We remind that for each request filter; a threshold test is done before and after filtering the list of exercises.

- When the filter threshold is reached, the system is no longer accepting filter. Thus, the highest weight will be affected (10) to the relevant exercises. A negative weight (−10) is assigned in the opposite case.

- Assigning a null value for a rule lead to minimise its authenticity. Taking for example, the Rule5 that concerns the restriction of the already viewed exercises belonging to the historic. R5 (E ≠ His.LP), but for a learning situation, it would be allowed even interesting to review these exercises for solving. In this case, they will be not filtered and not disadvantaged.

The algorithm proceeds similarly during the various stages. Weights can take either numerical value, where exercises accumulate the assigned weights, or be removed during a filtering application. The ordering of filters is important while the order of assigning weights has no consequences (Table 2).
### Table 2  
Exercises filtering and weighting example (see online version for colours)

<table>
<thead>
<tr>
<th>Rules</th>
<th>Weights</th>
<th>Action</th>
<th>Exercises</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule 1</td>
<td>Filtering (exercise-base)</td>
<td>EX2</td>
<td>EX5</td>
</tr>
<tr>
<td>Rule 2</td>
<td>Filtering (sub-set rule1)</td>
<td>EX2</td>
<td>EX5</td>
</tr>
<tr>
<td>Rule 3</td>
<td>Weighting</td>
<td>(sub-set rule2)</td>
<td>EX1</td>
</tr>
<tr>
<td>Rule 4</td>
<td>Weighting</td>
<td>(sub-set rule2)</td>
<td>EX2</td>
</tr>
<tr>
<td>Sorting</td>
<td></td>
<td></td>
<td>EX2</td>
</tr>
</tbody>
</table>

#### 3.3 Some interfaces of the implementing system for automatic exercises sequencing

We implemented our exercises sequencing algorithm, as distinct dedicated interfaces for each actor of the process, thus some inputs on which our work is centred, among others the visualisation of the ontology as a tree structure, the exercises-bases, etc. The resulting exercises’ set of the algorithm execution represents the list of the suitable exercises in a problem solving learning situation through Algo+ (Bey and Bensebaa, 2011) assessment system. An interface designed to display the outcome of the algorithm execution is proposed as a sequencing result Tab. We use the Jena API for the access to ontology (OWL file), JDOM for XML and the SPARQL language for querying the base OWL.

### Pedagogical strategies tab

The “pedagogical strategies tab” (Figure 6) is devoted to the definition and update of pedagogical strategies for a particular learning situation by the teacher of the domain. The latter visualise the features of each LM and PO in their respective tabs; which are defined by the pedagogue. So, for each LM and PO, the teacher should afterward:

- choose one of the proposed progression approach [PFR/progression with immediate remediation (PIR)]
- define the threshold of the not acquired skills (NASmax)
- define the threshold filter
- finally select the appropriate rules with a drop down list, viewing their definition (goal, knowledge, pre-requisites, history, difficulty level) in the desired filter order (the order of assigning weight has no consequences)
Figure 6  Definition of a PS (see online version for colours)

Sequencing result tab

The interface ‘sequencing result’ (Figure 7) is used to view the outcome of the automatic proposition of exercises for a given learning situation. Indeed, it recovers the set of setting options (the applied PS). It also displays the features of the learner profile and the current PO. The interface demonstrates the result of the filtering and/or weighting steps of exercises in the different levels, and the sorting of the final list. Adapted exercises are ranked first.

Figure 7  Example of exercises sequencing result (see online version for colours)
4 Conclusions and future works

In this paper, we have introduced an automatic sequencing exercises approach in the domain of programming and algorithms. The purpose of this study is to implement an approach that supports a tutoring system in providing an intelligent sequence of exercises to be used in training (one or several adequate exercises for a learner in a learning situation). The intelligence used in the process of sequencing is based on an ontology of algorithmic notions and skills ‘AlgoSkills’. The domain concepts (notions and skills) and the relations between them are exploited in a formal and explicit structure in order to describe the cognitive learner profile (acquired, not acquired and being acquired skills),
the educational objectives (targeted skills) and the exercise-base (semantically and pedagogically organised).

There is no doubt that dynamic generation of an intelligent list of exercises for training can provide most benefits by building a personalised sequence for every learner. At the same time, this needs more effort from instructor and pedagogue.

For the future work, we plan to integrate into the learner profile information the learning style. This will give more adaptability of the proposed exercises. And to validate the proposed method we are experimenting the developed tool in realistic conditions.

References


Automatic exercise sequencing-based algorithmic skills


Notes
1  https://www.w3.org/TR/owl-features/.
2  https://jena.apache.org/.
3  http://www.jdom.org/.
4  https://www.w3.org/TR/rdf-sparql-query/.