An Automatic Method for Structuring and Recommending Exercises
In Light of Case-based Reasoning, Knowledge Representation and Error Mediation

Carlos André Zavadinack[1], Fabiano Silva[1], Alexandre I. Direne[1] and Alexander Kutzke[2]
[1]*Department of Informatics, Universidade Federal do Paraná, Caixa Postal 19.081 – 81.531-980, Curitiba-PR, Brazil*
[2]*Sector of Technological and Professional Education, Universidade Federal do Paraná, Curitiba-PR, Brazil*

Keywords: Case-based Reasoning, Error Mediation, Knowledge Representation, Educational Informatics.

Abstract: Case-based Reasoning (CBR) is a method for solving problems with similar retained solutions. CBR demands a knowledge representation that allows the reasoner to find similar cases by a query and the similarity rate is given by a distance in hierarchical tree structure, an ontology. The main goal of this research is to use CBR as a pedagogical tool supported by three pillars: Case-based reasoning, Knowledge representation and Error Mediation in Education. It is considered that the error has a role of importance in the pedagogical development, so it has to be mediated. The error mediation is used as a rule for a quantitative classification of exercises, it takes into account how many times an exercise have been uncorrected answered, the distance between exercises gives the similarity between them. This kind of automatically classification for exercises in a educational support systems is one of the main contributions of this research. This work suggests that the CBR cycle is useful in the designing of a tool for automatic creation of exams.

1 INTRODUCTION

In the last decades, researches and tools have been developed aiming to improve educational support systems. Distance Learning is a type of educational support system that could be helped by those researches. Automatic creation of exercises, introduced by Fischer (Fischer and Steinmetz, 2000), could make distance learning systems reliable and credible. Using a reasoner that could, automatically, generate assorted exercise lists for various students is a solution that looks very helpful because, e.g., it can avoid students sharing exercise lists with their colleagues, what could improve the evaluation reliability.

Generating assorted exercise lists is a trivial activity, a simple algorithm could set groups by some parameters. But to find out a way that certifies the similarity between exercises lists and generates lists with dynamics variables could be helpful. Artificial Intelligence (AI) with its techniques and methodologies has been providing substantial contributions for educational systems. An AI methodology that seems helpful for the task of suggesting exercise lists is the Case-based Reasoning (CBR). Case-based Reasoning is a methodology which solves new problems reusing past experience, taking into account similarity between singular elements in a complex range to find out solutions. This work relies on a quantitative way in an hierarchically organisation of the elements.

According to Kutzke and Direne (Kutzke and Direne, 2015), error is "an integral part of the teaching and learning process". Among the future directions pointed by their research, we highlight: analysis of error recommendation in the learning and knowledge process; research and design of a tool for an automatic creation of exams and collected data analysis; a data structuring that takes into consideration the error is a useful contribution. We suggest that a CBR system could be a tool for a an automatic creation of exams.

This paper shows a research that aims to experiment CBR as a method for recommendation of learning objects, specifically, exercise lists. Two applications were used for the experiment, one is a CBR framework named *myCBR 3*[1] (Hundt et al., 2014); and the other is *FARMA-ALG* (Kutzke and Direne, 2015), an application for computer programming education, where the data for the present tests were taken from. Briefly, there are data with sets of questions and answers from FARMA-ALG, that data were structured in an ontology that is the knowledge representation used in *myCBR 3*.

The results point that a structured data as in an on-

[1]http://mycbr-project.net/

Zavadinack, C., Silva, F., Direne, A. and Kutzke, A.
An Automatic Method for Structuring and Recommending Exercises - In Light of Case-based Reasoning, Knowledge Representation and Error Mediation.
DOI: 10.5220/0006318903550362
In *Proceedings of the 9th International Conference on Computer Supported Education (CSEDU 2017) - Volume 1*, pages 355-362
ISBN: 978-989-758-239-4

tology is a requirement for a good use of knowledge base in a CBR system. An automatic and quantitative way for exercise classification was made taking into consideration the students' mistakes, and we believe that it may be helpful for learning tools. Also, we provide a validation to prove the utility of a CBR automatic tool for exams creation.

For presenting how the research is developed, this paper is organised as follows. The first section introduces the Case-Based Reasoning and myCBR - a CBR system chosen for the present research. Section 3 is about the role of error mediation and show FARMA-ALG, a framework for teaching computer programming. The section 4 brings how the raised data from FARMA-ALG are structured. The experiment and its results are shown in section 5. Finally, section 6 relates the conclusions, contributions from the research and future directions.

2 CASE-BASED REASONING

Case-based reasoning is a methodology introduced by Kolodner (Kolodner, 1992) and Schank (R.Schank, 1989). CBR is a problem solver based on past experiences, named *cases*. According to Kolodner (Kolodner, 1992), the meaning of the therm *case* in CBR is a contextualised piece of knowledge representing an experience that teaches a fundamental lesson for the reasoner achieving a settled goal. Aamodt and Plaza (Aamodt and Plaza, 1994) introduced the CBR cycle, a reasoning model (Figure 1). CBR cycle consists of four tasks: *Retrieve*. *Reuse*. *Revise* and *Retain*.

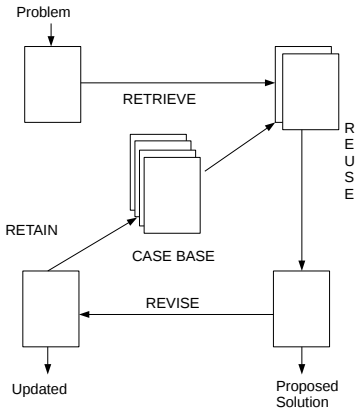

Figure 1: CBR cycle by Aamodt and Plaza (Aamodt and Plaza, 1994).

The *retrieve* process is the matching of features in a settled case, a query with the case features brings similar cases in the case base. The case retrieved and the query are compared, an adaptation of the case for each feature happens in the *reuse* task. The *revision* task is the time that an expert (machine or human) will check if the reuse works. If it works, then a new case is *retained*.

Richter (Richter, 2003) introduced that it is possible to identify knowledge in CBR with four containers (Hundt et al., 2014):

Vocabulary defines attributes and their allowed values.

Cases are the descriptions of past episodes of experience, represented in an ordered pair (*problem, solution*). The cases are stored in a *Case Base (CB)*.

Similarity Measures are functions to calculate the similarity between individual attribute values.

Adaptation Knowledge are represented by rules that can be applied as solution of retrieved cases.

The attribute-value representation consists in a name A; a finite set $\mathrm{DOM}(A)$ with a set of values from A; and a variable x_A. The values allowed are determined in reason of each context, assuming numerical or symbolical values.

The retrieving is a simple operation for CBs and data bases. But, while in the data base a query matches an exact data, at CBR systems the query presents a problem and would retrieve a solution in an inexact match with the problems in the CB. For both of them, the tree structure plays a major role to structure data and cases for an efficient retrieve (Richter, 2003).

Formally, Cimiano (Cimiano, 2006) defines *Similarity Measure as a function sim:* $sim : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \{0,1\}$ *with the following properties*:

$$\forall \mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^n, sim(\mathbf{v}_1, \mathbf{v}_2) = 0 \Leftrightarrow \mathbf{v}_1 \cdot \mathbf{v}_2 = 0$$

$$\forall \mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^n, sim(\mathbf{v}_1, \mathbf{v}_2) > 0 \Leftrightarrow \mathbf{v}_1 \cdot \mathbf{v}_2 > 0$$

$$\forall \mathbf{v} \in \mathbb{R}^n, sim(\mathbf{v},\mathbf{v}) = 1.$$

The first condition means that the similarity between two elements is zero if there are no common features. Thus, the *sim* is greater than zero in case they have at least one common feature. The last property affirms that a vector is maximally similar to itself.

In a data structured tree, the similarity between two attributes is given by their distance. *A Distance measure is a function dist* : $\mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_0^+$ *with the following property*: $\forall \mathbf{v}\ dist(\mathbf{v},\mathbf{v}) = 0$, means that the distance of a vector to himself is 0. The transformation of distance to similarity and the opposite way needs to fulfil the following conditions (Cimiano, 2006):

$$dist(x,y) = +\infty \leftrightarrow sim(x,y) = 0$$

$$dist(x,y) = 0 \leftrightarrow sim(x,y) = 1.$$

Three CBR systems was studied during the research: TUUURBINE, jColibri and myCBR. TUUURBINE[2] is a generic CBR engine for the semantic net able to reason on knowledge stored in RDF (Resource Description Framework)[3] format (Gaillard et al., 2014). We used TUUURBINE in our first attempt, but a technical unsolved problem didn't enable its usage[4]. One of the most significant CBR systems developed is jColibri[5], presented in (Díaz-Agudo et al., 2007) and (Recio-García et al., 2013), a free software with more than 10000 downloads. jColibri and myCBR are compared in (Atanassov and Antonov, 2012), where they bring the conclusion that jColibri is a framework for complex operations, with multiple data bases, requesting a strongly programming knowledge and time for developing an application with it. In light of (Atanassov and Antonov, 2012), myCBR is recommended for non-complex applications and not recommended for applications with a great amount of attributes. The experiment in the present research counts with only five attributes, and, in face of that, we decided for myCBR. There is a brief report about its properties and its knowledge modelling in the subsection 2.1.

2.1 MyCBR 3

MyCBR[6] is a CBR framework developed through a partnership between DFKI[7] and UWL[8]. MyCBR 3 is the latest version known, and it comprehends three knowledge containers: vocabulary, similarity measures and case base. The adaptation task and its rules are not available in the last version, although in (Hundt et al., 2014) is mentioned that a version would be soon available with those properties.

Developed in Java, *myCBR 3 Workbench* has a friendly interface that allows users modelling the knowledge by a set of attributes and its own set of values. The cases are sets of allowed attributes, the case base can be fed by the interface or by the exportation of *.csv* files. *Vocabulary* in myCBR 3 is given by attributes with allowed values by the types: `symbols`, `boolean`, `float`, `integer` and `string`.

Similarity measure functions (SMF) are given within concepts and attribute values. SMFs can be edited by advanced similarity mode, table editor mode and taxonomy editor mode. The advanced similarity mode is recommended for values of the types `integer` and `float`. The table editor mode is the default edition mode, the relations could be settled by the interval [0,1] where the main diagonal are the similarities between the same values $sim(x,x) = 1$. In the taxonomy edition mode, settled by the hierarchy, only the `symbol` type is allowed.

The *retrieve* window is composed of a *query* with predetermined fields that can be filled in only with the allowed attribute values. User can also fill in with unknown (`__unknown__`) and undefined (`__undefined__`) values. The result search correspondent elements with a similarity function and, therefore, among the cases. The retrieved cases are shown in an ordered list by similarity rate.

3 THE ROLE OF ERROR MEDIATION IN TEACHING AND LEARNING COMPUTER PROGRAMMING

In this study we consider error an answer that is not in accord with the expectation for who made the question. Kutzke and Direne (Kutzke and Direne, 2015) bring that, according to the Cultural-Historical Psychology, the relevance of error in the learning and teaching process merges in given differences between the development of scientific and spontaneous concepts. According to Kutzke's and Direne, error has to be mediated within the scientific concepts, making it an object of learning task, a part of concept assimilation process. The role of error mediation is an essential problem for the Kutzke research, it presents a demand for tools that help teachers in the students errors analysis.

Kutzke and Direne (Kutzke and Direne, 2015) bring the specificities of computer programming teaching and learning. Learning computer programming raises that "the student needs to appropriate himself of signs and specific operations of this area in a superior way. It is demanded, thus, to form scientific concepts" (Vygotskiĭ et al., 2012) apud (Kutzke and Direne, 2015). It has to be an empirical apprehension of reality in computer programming learning, even the concept assimilation in its finished form, "the error is certainly part of the process of scientific concept formation (...) and must be *mediated*" (Kutzke and Direne, 2015). Thus, the access to error records is needed and useful, so it is the relation between the errors. Kutzke and Direne, in terms of that,

[2]http://tuuurbine.loria.fr/

[3]http://www.w3.org/standards/techs/rdf

[4]TUUURBINE's web service didn't work properly and we had to interrupt its using.

[5]http://gaia.fdi.ucm.es/research/colibri

[6]http://mycbr-project.net/

[7]http://www.dfki.de/web

[8]http://www.uwl.ac.uk/

consider that an instrument for the manipulation of this relationship would help teachers' reflection about students' errors.

3.1 FARMA-ALG

FARMA-ALG was developed for helping teachers and students in teaching and learning computer programming. Specifically, FARMA-ALG uses error mediation as a main role in the process of knowledge acquiring. Each question is a computer programming problem, FARMA-ALG compiles andchecks if the answer is correct or not by the comparison of inputs and outputs. FARMA-ALG is well presented in (Kutzke and Direne, 2015), there are some of main application's functions:

Answers' Similarities Computation is the similarity function determined by the expected output and the output obtained, according to test case and answer's source code.

Search of Answers is facilitated using keywords and meta-data such as name of the student, class, time interval etc.

Timeline View where it is possible for teachers and students to access the answers on a timeline representation.

Similarity Graph is a functionality of viewing and manipulating answer records, where is possible to observe and interact with stored answer records.

FARMA-ALG is able to recommend to the teacher answer records and exercises that would be of high pedagogical relevance for a specific group of students through, a semi-automatic recovery of potentially relevant answers. An exercise in FARMA-ALG is a set, basically, with: title, content, identifiers, creation time and programming language(s). For instance:

```
{"Title":  "is_Palindrome?"}
{"Content": "A palindrome is a word, phrase,
number, or other sequence of characters
which reads the same backward or forward,
such as madam or kayak. Write and compile
a program that finds out if a sequence of
characters is a palindrome or not."}
{"Language":  "Pascal, C" }
{"lo_id": c950ded46e4d9e530d61 }
```

The student gives the answer by coding a program that the FARMA-ALG evaluates. Each question has test models of input and output to check automatically if an answer is correct or not.

Kutzke's research was made with 80 students and 23 exercises, 3723 answers were collected. The data was collected and its structure was given by an ontology, the next section brings how part of that ontology was made for the present experiment.

4 STRUCTURED DATA FOR AN ERROR MEDIATION

This section aims to explain how the amount of data that came from FARMA-ALG are exported for a conceptual and hierarchical ontology. First of all, an approach about ontologies in the subsection 4.1. The exportation is explained at the Subsection 4.2, it is a quantitative way for error mediation, this is the conception of the *Knowledge Base* (*KB*) to be used in a CBR system.

4.1 Ontologies

According to Cimiano (Cimiano, 2006), "the term *Ontology* comes from the Greek *ontologia* and means the study of being". Aristotle (384 - 322 BC) shaped the logical background of ontologies, brought notions of *category* and *subsumption*. In Computer Science, accord to Gruber (Gruber, 1995), ontology assumes the meaning as a formal specifications from a conceptualisation, or a representation of conceptual models in a certain domain.

Ontologies allow a general indexation schema that comprehends the knowledge through the similarity between the concepts. Usually, specialists are the humans responsible the creation of an ontology. An ontology representation is given by the individuals, classes and concepts showed in the nodes and the edges could assume values giving relation between the objects.

4.2 Quantitative Error Mediation by Ontology

The FARMA-ALG data model contains relevant classes for a teaching framework, such as: answers, users, teams, comments, questions, etc. The similarity between answers is given in the error mediation by the teacher, a qualitative way for relation between questions, answers and students. The FARMA-ALG data was generated in JSON[9] (Java Script Object Notation), a dictionary structure[10]. This data was exported for a OWL semantic, as seen on the Figure 2.

[9] http://rfc7159.net/rfc7159

[10] A *dictionary* is a data structure that contains, basically, key and its value.

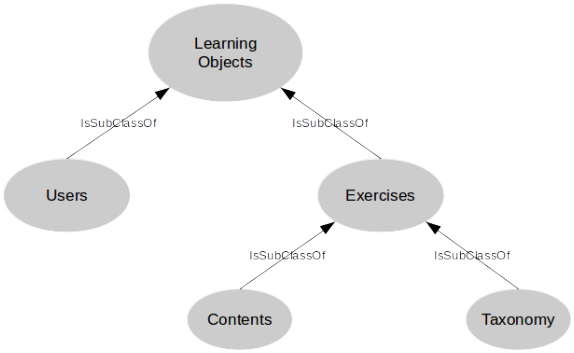

Figure 2: Ontology of Learning Objects.

The root class is *Learning Objects* and its subclasses are: *users* and *exercises*. The exercises class is subdivided in *contents* and *taxonomy*. The class *Content* contains pairs with titles of the exercises and its contents. The class *Taxonomy* is detailed bellow, at the Figure 3.

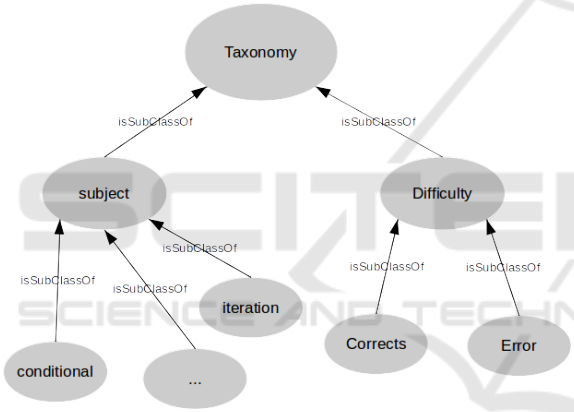

Figure 3: Ontology representing the Taxonomy class.

Taxonomy is a class determined to discern the exercises (or questions) according to *subject* and *difficulty*. *Subject* is about the conceptual content that exercises are in the learning of computer programming, e.g., conditional, iteration, data structures, etc. The subject tags were determined by the teachers as they had created the exercises. Exercises, as instances for the *subject* class could be assigned for more than one concept, for example, an exercise tagged as conditional could also be tagged as iteration. Difficulty is a class that subdivided exercises between *corrects* and *errors* up to the answers. The same exercise could appear in a subclass of *corrects* and in a subclass of *errors*, but must not appear in different subclasses of *errors*.

Differently from the *subject* class, the *taxonomy* class is a dynamic class. The questions in classes *errors* and *corrects* are divided in levels by the times each one had been correctly or wrongly answered. The assignment is binary, "correct"(True) or "wrong"(False), in this first experiment approach, i.e, currently, there is no fuzzy classification. This exercises approximation by errors is given automatically and quantitatively, complementing the qualitative way in the Kutzke's work.

A Python[11] program was designed for the exercises classification. There are two subsections for the error classification, after an average number of errors, a first cut is given: *High* and *Low*. In the *High* class, there are the exercises with more than the average of errors, thus the *Low* class has the exercises with the lesser average or errors. The second cut subdivided *High* and *Low* classes in the same way. The second cut, in the same way, subdivided *High* and *Low* classes. After that, there are four disjunctive classes (see Figure 4): *High_high*, *High_low*, *Low_high* and *Low_low*. With *n* exercises, it is reasonable to use four levels, but the ontology could be more complex and bigger as bigger is the input data. Section 5 brings details about the data.

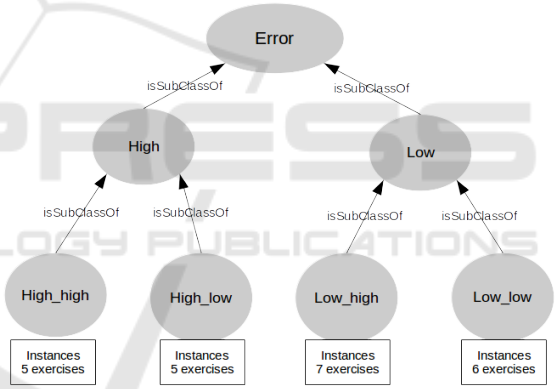

Figure 4: Ontology representing the Error class and its subclasses.

Formally, in the ontology O of error there is a set C of concepts C := {*High, Low, High_high, High_low, Low_high, Low_low*}. The set R of relations $R := \{$ *isSubClassOf, isInstanceOf* $\}$ and the set A of attributes $A := \{$ *title, content* $\}$. Relations and attributes are assigned, formally, as follows:

$\sigma_R(isSubClassOf) = \{(High, Erred), (Low, Erred), ..., (Low\_low, Low)\}$

$\sigma_R(isInstanceOf) = \{(Id, High\_high), (Id, High\_high)\}$

$\sigma_A(title) = (Id, string)$

$\sigma_A(content) = (Id, string)$

The Knowledge Base is formed by the ontology of error (Figure 4). The experimental *KB* conception is an automatic and quantitative method that uses the

[11] https://www.python.org/

error mediation topic for the classification. This classification method, performed as an assistant tool, can be easily joined to an educational tool as FARMA-ALG. This tool was developed aiming to improve the error mediation in the FARMA-ALG. The question is classified every time it is answered. For example, if a question is not correctly answered, its total number of errors is incremented and the place of the questions in the tree may change.

5 EXPERIMENT AND RESULTS

The data structured, as seen in the previous section, are added to the myCBR's knowledge containers. The vocabulary is composed by the exercise titles. The distances in the Error ontology (Figure 4) consequently give the similarity functions. The Kutzke's experiment (Kutzke and Direne, 2015) brings an amount of 23 questions that were used to make the vocabulary container. The Case Base is a set of exercise lists with those questions.

5.1 Knowledge Containers

Each question in the exercise list is an attribute: $questions = \{question_1, ..., question_5\}$. All the attributes have the 23 exercises for the vocabulary set: $question_n = \{Birthday,\ 1\_rad\_Cosine,\ ...,\ Temperature\_converting\}$. All the values are of type `Symbol` in myCBR. The similarity or distance in the vocabulary set brings an important contribution in this research. The distance between the questions would be helpful to generate similar exercise lists that are given for a same class. A list of questions is structured in a balanced way, i.e. with questions from a varied levels of difficulty and subject.

Each attribute has the same similarity function given by a default table, as seen at section 2.1. The structured graph gives the distance between an element and another. Those distances are used to fill in the table of the similarity function. The distance d between two elements x and y in the ontology is given by the sum of the weights w in the path, given by the formula:

$$d(x,y) = \sum_{i=1}^{n} w_i$$

where i is the given value by each time an edge is crossed, starting with 1 untill the last node, i.e. the y node. For instance, from x to y there are four nodes (we don't count the x as one of them), then $n = 4$. The weight w_i is given in each edge. For the present knowledge, each edge has the weight of 0.125. Trivially, it takes into account how many nodes had been visited and multiplies by the weight. As w has a unique value, the distance can be represented by:

$$d(x,y) = \#visited\_nodes \times w.$$

To find the similarity sim between two elements x and y, it could be possible to use a simple transformation functions, recommended by (Cimiano, 2006):

$$sim(x,y) = k - d(x,y)$$

where k is an appropriate constant. For the present KB in this research, $k = 1$. To find the similarity between the questions 'Prime_Factors' and 'Birthday' as in Figure 5: if the distance d is given by d('Prime_Factors', 'Birthday') = 0.25, then the similarity is given by sim('Prime_Factors', 'Birthday') = 1 - 0.25 = 0.75.

Table 1: A sample Table with an example of the similarity function determined by the ontology in the Figure 5.

	Temp. Conv.	Rep. Nums.	Prime F.
Temp. Conv.	1	0.75	0.5
Rep. Nums.	0.75	1	0.5
Prime F.	0.5	0.5	1

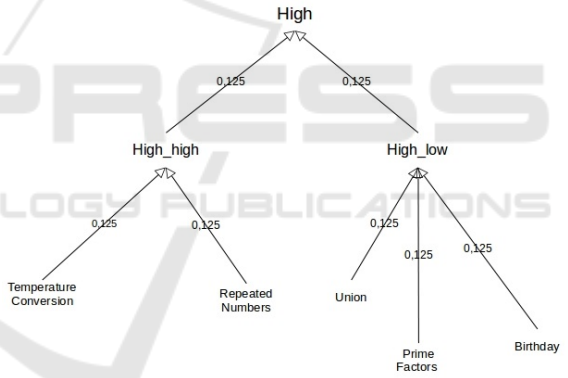

Figure 5: Part of ontology structured with weights between elements and classes.

Table 1 shows a few examples of how the default similarity function is settled for myCBR. The distance d between the exercises "Prime Factors" and "Repeated Numbers" as seen in Figure 5 is given by, d('Prime_Factors', 'Repeated_Numbers') = 0.5, then the sim('Prime_Factors', 'Repeated_Numbers') = 0.5. This similarity rate is given twice in the table, at the 3rd row with the 2nd column and at the 2nd row with the 3rd column.

The experimental queries were composed according to three parameters as critical success factors:

Balance is a boolean variable, it is considered balanced an exercise list in which at least three exercises are elements from different subclasses in L.

Amount of F values in a query.

Figure 6: An query example from *Group 1* in myCBR.

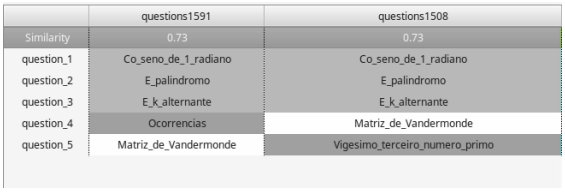

Figure 7: A retrieve example from the *Group 4* in myCBR.

Unknown values in the query.

The Case Base for the experiment is composed of 2000 exercise lists, each list is a case. It is determined a set $F = \{$*'É k-alternante', 'Co-seno de 1 radiano', 'Matriz de Vandermonde', 'É palindromo?'*$\}$. There are four values in the set F, each one is picked from the levels subclasses $L = \{$High_high, High_low, Low_high, Low_low$\}$ at the ontology to provide balanced lists. F is included randomly in each list of case base, i.e. all the cases in the case base have the F elements. That was made because the order of the elements is relevant for myCBR.

An amount of 30 queries were made in six different groups. Each group has a quantity of queries, quantity of elements of the set F, the presence of an unknown value (__unknown__) and unbalanced queries. An unbalanced query is a list with exercises of a same level, what is very important to observe because in a trustworthy CBR system for this context would not find cases with a high similarity. The groups were divided:

Group 1: 12 queries with no elements of F;

Group 2: 3 queries with one element of F;

Group 3: 3 queries with 4 elements of F;

Group 4: 8 with an unknown value (__unknow__);

Group 5: 4 queries where all the exercises are instances of a same subclass in the difficulty level.

A query in myCBR is predetermined as the features structure in the vocabulary container, as seen in the Figure 6.

A list of cases in order of similarity is given at myCBR retrieve, the Figure 7 presents a sample of the cases retrieved, the darker the cell, the bigger is the rate of similarity. A table ordered by the highest similarity for each group of retrieving is given in the Table 2. Obviously, the more elements of F are in the query, the bigger is the similarity rate. In the *Group 1*, myCBR retrieved results with a similarity rate of 0.77, it is good considering the fact that there is no element of F in the query. And this proves the myCBR capacity in retrieving distincts exercise lists by the similarity.

Table 2: Table ordering the groups of queries by the similarity rate in the retrieving.

	Highest Sim.Rate
Group 3	0.89
Group 2	0.77
Group 1	0.77
Group 4	0.73
Group 5	0.63

Relevant points were verified in this research: case retrieving, knowledge representation and similarity relation between elements. There is no adaptation on myCBR 3, what is a another point that could be verified for the stated research.

6 CONCLUSION

This work considers that finding creative solutions for structuring and retrieving learning objects in learning tools are relevant for educational informatics. Case-based reasoning method seems useful as an exercise lists retriever. CBR systems can be used in educational solutions as learning object recommender and automatic creation of exercises. In this research, the context is the computer programming teaching, but it can be applied to others disciplines.

Another important point explored in the present work is the quantitative error mediation by an automatic classification of exercises. That is a creative and role of error way to classify how hard may an exercise be, that shows the importance of error in education. This dynamic way would help to understand how an exercise can be harder than another in a deversified class of students. Example, a distance learning system is used for teaching Math. The teacher's class brings 300 exercises. They are divided in: basic operations, geometry, algebra, etc. Different exercise lists are given for the students. After they have answered this lists, the exercises are classified up to the difficulty. This classification are used in the test elaboration for that class. And a certain quantity of different exams can be automatically generated and applied.

This research were developed in the context of the project *"Pesquisa de redes sociais em nuvem voltadas par objetos educacionais"* of the Brazilian Ministry

of Education at the C3SL[12] laboratory of the Federal University of Paraná. The learning objects generated can be integrated with the digital repository of this project.

6.1 Contributions

It is possible making myCBR a tool that automatically creates, distinct exams for distinct students. This work suggests that the CBR cycle is useful in the designing of a tool for automatic creation of exams. Another relevant contribution is, in light of error mediation, an automatic creation of a knowledge base that considers how many times a question has been mistakenly answered by a group of students.

6.2 Future Directions

Studies that experiment reasoning methods for automatic creation of exams seem to be very useful. To trust the CBR as an interesting method for that, an adaptation task should be tested.

In the current representation knowledge area, greater repertories of programming exercises will be welcomed, because a big amount of data can be structured in a more complex ontology, what helps the variety of lists and also makes retrieves better with more accuracy at the similarity.

Adding other variables such as the chronological way that the questions had been solved, the user level and a fuzzy classification of errors will be very helpful for a better exercise list build and its recommendations.

ACKNOWLEDGEMENTS

We would like to acknowledge the Brazilian Funding Agency (CAPES) for the financial support of this work.

REFERENCES

Aamodt, A. and Plaza, E. (1994). Case-based reasoning, foundational issues, methodological variations, and system approaches. In *AI COMMUNICATIONS*, volume 7, pages 39–59.

Atanassov, A. and Antonov, L. (2012). Comparative analysis of case based reasoning software frameworks jcolibri and mycbr. *Journal of the University of Chemical Technology & Metallurgy*, 47(1):83–90.

[12]*Centro de Computação Científica e Software Livre* - http://www.c3sl.ufpr.br

Cimiano, P. (2006). *Ontology learning from text*. Springer.

Díaz-Agudo, B., González-Calero, P. A., Recio-García, J. A., and Sánchez-Ruiz-Granados, A. A. (2007). Building cbr systems with jcolibri. volume 69, pages 68–75. Elsevier.

Fischer, S. and Steinmetz, R. (2000). Automatic creation of exercises in adaptive hypermedia learning systems. In *Proceedings of the eleventh ACM on Hypertext and hypermedia*, pages 49–55. ACM.

Gaillard, E., Infante-Blanco, L., Lieber, J., and Nauer, E. (2014). Tuuurbine: A generic CBR engine over RDFS. In *Case-Based Reasoning Research and Development - 22nd International Conference, ICCBR 2014, Cork, Ireland, September 29, 2014 - October 1, 2014. Proceedings*, pages 140–154.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5):907–928.

Hundt, E., Reuss, P., and Sauer, C. (2014). Knowledge modelling and maintenance in mycbr3?

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artif. Intell. Rev.*, 6(1):3–34.

Kutzke, A. R. and Direne, A. I. (2015). Farma-alg: An application for error mediation in computer programming skill acquisition. In *Artificial Intelligence in Education*, pages 690–693. Springer.

Recio-García, J. A., González-Calero, P. A., and Díaz-Agudo, B. (2013). jcolibri2: A framework for building case-based reasoning systems. volume 0.

Richter, M. M. (2003). Introduction. In *Case-based reasoning technology: from foundations to applications*, volume 1400. Springer.

R.Schank, C. R. (1989). *Inside Case-Based Reasoning*, volume 1989. Erlbaum, Northvale, NJ.

Vygotskiĭ, L. S., Hanfmann, E., and Vakar, G. (2012). *Thought and language*. MIT press.