Generating Multiple Choice Questions From Ontologies: How Far Can We Go?

Tahani Alsubait, Bijan Parsia, and Uli Sattler

School of Computer Science, The University of Manchester, United Kingdom
{alsubait,bparsia,sattler}@cs.man.ac.uk

Abstract. Ontology-based Multiple Choice Question (MCQ) generation has a relatively short history. Many attempts have been carried out to develop methods to generate MCQs from ontologies. However, there is a still a need to understand the applicability of these methods in real educational settings. In this paper, we present an empirical evaluation of ontology-based MCQ generation. We examine the feasibility of applying ontology-based MCQ generation methods by educators with no prior experience in ontology building. The findings of this study show that this is feasible and can result in generating a reasonable number of educationally useful questions with good predictions about their difficulty levels.

1 Introduction

Automatic question generation is a relatively new field and dimension within the broad concept of technology-aided assessment. It potentially offers educators some help to ease the burden and reduce the cost of manual assessment construction. In terms of time, it is reported that assessment development requires considerable time [9, 18, 20]. In terms of cost, it is estimated that the cost of developing one question for a high-stake test can range from $1,500 to $2,000 [19]. More importantly, in terms of quality, as many as 40% of manually constructed questions can fail to perform as intended when used in assessments [11]. This has motivated many researchers to develop automated methods to generate assessment questions. Many of these methods have focused on the generation of Multiple Choice Questions (MCQs) which are typically used in high-stake testing.

Ontologies are machine-processable artefacts that can formally describe the main notions of a specific domain. Recent advancements in ontology languages and ontology tools have created an interest in ontology-based MCQ generation. Various attempts have been made to generate MCQs from ontologies [3, 17, 23, 24]. However, little is known about how useful these MCQs are when used in real educational settings.

We present a new case study for using ontology-based MCQ generation in real educational settings. The purpose of the study is to evaluate the feasibility of using ontology-based MCQ generators by instructors who have no prior experience in ontology development. Rather than using an existing ontology, we

examine the case where a new ontology is required to be build from scratch. We estimate the cost of question generation including the cost of building a new ontology by a novice ontology developer (e.g., the instructor in this case). We also evaluate the quality of the generated questions and the accuracy of the generator's predictions about the difficulty of the generated questions.

2 Background

An MCQ item is an assessment tool which is made up of the following parts: 1) A stem, 2) A key and 3) Some distractors. The stem is a statement that introduces a problem to the student. The key is simply the correct answer. A number of incorrect, yet plausible, answers are called the distractors. The number of optimal distractors for MCQs remains debatable [12].

An ontology is a set of axioms which can be either terminological or assertional. Terminological axioms describe relationships between concepts. Assertional axioms describe relationships between individuals and concepts or between individuals and roles. Description Logics (DL) ontologies have formal semantics [7]. In this sense, an ontology is a logical theory which implies that implicit knowledge can be inferred from the explicitly stated knowledge. For a detailed overview of ontologies, the reader is referred to [7].

3 Related work

Prior to exploring the large body of research related to ontology-based question generation methods, we need to understand the basic/optional components associated with those methods. These components are:

3.1 Source preparation

Before being able to generate questions, a suitable source ontology must be prepared. Gavrilova et al. [10] present a 5-step strategy aimed at developing teaching ontologies. The stages are: 1) Glossary development, 2) Laddering, 3) Disintegration, 4) Categorisation and 5) Refinement. Sosnovsky et al. [21] present a case study for utilising the above 5-step strategy to develop an ontology for the domain of C programming.

3.2 Item generation

The next step of generation is to generate an assessment item or part of it (e.g., distractors) from the developed ontology. For example, Mitkov et al. [16] have developed an approach to automatically generate MCQs using NLP methods. They also make use of ontologies to generate distractors that are similar to the correct answer.

Zitko et al. [24] proposed templates and algorithms for automatic generation of MCQs from ontologies. They generate a random set of distractors for

each questions. Papasalouros et al. [17] constrain the set of distractors to some neighbours of the correct answer in the ontology.

Williams [23] presents a prototype system for generating mathematical word problems from ontologies based on predefined logical patterns. The proposed method makes use of data properties in general ontologies. The data properties are used to replace certain place holders in the predefined patterns.

3.3 Characterisation

Some methods of ontology-based question generation vary the characteristics of the generated questions (e.g., their difficulty). For instance, Williams [23] proposes to vary the difficulty of mathematical problems by introducing/removing distractor numerical values and varying sentence complexity and length. Alsubait et al. [3, 2, 1] propose to vary the difficulty of MCQs by varying the similarity between the key and distractors.

3.4 Presentation

To enhance the readability of automatically generated questions, Williams [23] extends the use of SWAT[1] natural language tools to verbalise ontology terms which are used in the generated questions. For example, "has a height of" can be derived from the data property "has_height". Similarly, Papasalouros et al. [17] use simple natural language generation techniques to transform the generated questions into English sentences.

3.5 Post evaluation

Mitkov et al. [16] present an evaluation study of automatically generated MCQs in real testing settings. Item response theory (IRT) [15] has been used for the statistical analysis of students results. In particular, they study the following properties: (i) item difficulty, (ii) discrimination between good and poor students and (iii) usefulness of distractors. They also compare manual and automatic methods of MCQ generation and report that that automated methods perform better than manual methods of test items in terms of time without compromising quality.

In an earlier study [3], the authors evaluated a large set of multiple-choice questions which have been generated from three different ontologies. The evaluation was carried out using an automated solver which simulates a student trying to answer these questions. The use of the automated solver facilitated the evaluation of the large number of questions. The findings of this study show that it is feasible to control the difficulty of questions by varying the similarity between the key and distractors. A more recent study [5] in which the authors recruit a group of students in real testing settings confirms the results of the study carried earlier using the automated solver.

[1] http://swat.open.ac.uk/tools/

4 Ontology-based MCQ generation in practice

Introduction to Software Development in Java is a self-study course run by the School of Computer Science at the University of Manchester. It aims to ensure that students enrolled in Masters programs in the school have a thorough grasp of fundamental programming concepts in Java. Topics covered in this course include: object-oriented basics, imperative programming, classes, inheritance, exception handling, collections, stream and file I/O. The course material is delivered online via Moodle. As with any self-study course, students enrolled in this course need a series of self-assessments to guide them through their learning journey.

4.1 Materials and methods

Equipment description the following machine was used for the experiments in this paper: Intel Quad-core i7 2.4GHz processor, 4 GB 1333 MHz DDR3 RAM, running Mac OS X 10.7.5. In addition to the following software: OWL API v3.4.4 [14] and FaCT++ [22].

Building the ontology An ontology that covers the contents of the course has been built by an instructor who has an experience in Java but with no huge familiarity with materials of this course. In this case, the instructor had no prior experience in building ontologies. The online course material covers both fundamental concepts (i.e., terminological knowledge) and practical Java examples (i.e., procedural knowledge). Only terminological knowledge is suitable to be modelled in ontologies. This type of knowledge is typically a vital part of education in general and of assessment in particular. It is regarded as the basic level in Bloom's taxonomy of learning objectives [8]. The development of the ontology has gone through the following steps:

- The instructor has been introduced to basics of ontology development in an initial meeting which lasts for 2 hours. This included a brief hands-on tutorial on using *Protégé* 4 ontology editor. Further online materials [13] were forwarded to the instructor to familiarise herself on building and dealing with ontologies.
- The instructor built an initial version of the ontology. She went through the first 6 modules of the course, extracted and added to the ontology any encountered concepts and finally established links between the added concepts. This took a total of 10 hours and 15 minutes spread over 6 days. This has resulted in a total of 91 classes, 44 object properties and 315 axioms.
- A two-hours feedback session took place to highlight weak points in this version of the ontology. The instructor reported that, as the number of classes and relations increased, it got very hard to maintain the same level of understanding of the current state of the ontology.

– The second version of the ontology took 5.5 hours to build. The resulting ontology has a total of 91 classes, 38 object properties and 331 axioms. The main task was to restructure the ontology according to the received feedback. The decrease in the number of object properties is due to merging those object properties which had very similar meaning but different names. The increase in the number of axioms can be partially explained by the fact that the instructor was advised to assert negative facts in the ontology whenever and wherever possible. In addition, some concepts were re-categorised (e.g., declared as a subclass of another exiting class).
– To ensure that the ontology covers the main concepts of the domain, the instructor was advised to consult a glossary of Java-related terms which is part of the online course material. Adding new terms from the glossary in suitable positions in the ontology took a total of 10 hours over 4 days. The resulting ontology has a total of 319 classes, 107 object properties, 213 annotation assertion axioms and 513 logical axioms. The DL expressivity of the resulting ontology is $\mathcal{ALCHQ}$ which allows conjunctions, disjunctions, complements, universal restrictions, existential restrictions, qualified number restrictions and role hierarchies. For more information and examples, the reader is referred to [7].

Generating questions We follow the same question generation strategies described in [5] to generate multiple choice questions from ontologies. The first step of question generation is to compute the pairwise similarity for all the classes in the ontology using the similarity measures described in [6]. These similarity measures have been shown to be highly correlated with human similarity measurements [6]. The intuition behind using similarity measures as part of question generation is that varying the similarity between the key and distractors can make it possible to vary the difficulty of the generated questions [4]. In other words, increasing the difficulty to distinguish the correct answer among the given answers makes the question harder.

A total of 428 questions have been generated from the Java ontology. Then questions with less than 3 distractors have been excluded (resulting in 344 questions). Questions in which there is an overlap between the stem and the key have been filtered out (resulting in 264 questions). This step was necessary to ensure that there are no word clues in the stem that could make the correct answer too obivous. Previous attempts to generate MCQs from ontologies have identified this as a possible problem [5]. In this study, we filter out questions in which there is a shared word of more than three characters between the stem and key. This does not apply to questions in which the shared word is also present in the distractors. Finally, questions which can be described as redundant and that are not expected/recommended to appear in a single exam were manually excluded (e.g., two questions which have slightly different stems but the the same set of answers or vice versa). This step was carried out only to get a reasonable number of questions that can be reviewed in a limited time. The resulting questions

are 65 questions in total. Among these are 25 easy questions and 40 difficult questions.

Reviewing questions Three reviewers have been asked to evaluate the 65 questions using the web interface shown in Figure 1. All the reviewers have experience in both the subject matter (i.e., programming in Java) and assessment construction. The reviewers have been randomly numbered as Reviewer 1, Reviewer 2 and Reviewer 3 with Reviewer 2 being the ontology developer. For each question, the reviewer is asked to first attempt to answer the question. Next, the reviewer is asked to rate the difficulty of the question by choosing one of the options 1) Too easy, 2) Reasonably easy, 3) Reasonably difficult and 4) Too difficult. Then the reviewer is asked to rate the usefulness of the question by choosing one of the options: (0) not useful at all, (1) useful as a seed for another question, (2) useful but requires major improvements, (3) useful but requires minor improvements or (4) useful as it is. In addition, the reviewer is asked to check whether the question adhere to 5 rules for constructing good MCQs. These rules were gathered from the qualitative analysis of previous reviewer comments in a previous evaluation study [5]. The rules are: R1) The question is relevant to the course content, R2) The question has exactly one key, R3) The question contains no clues to the key, R4) The question requires more than common knowledge to be answered correctly, and R5) The question is grammatically correct.

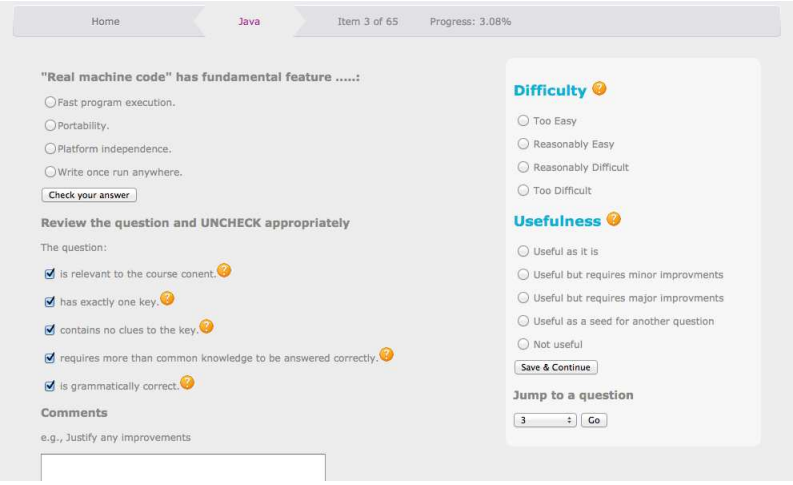

Fig. 1: The reviewing web-interface

4.2 Results and discussion

Total cost We report on the cost, in terms of time, of the three phases: 1) ontology building, 2) question generation and 3) question review. The ontology

took around 25 hours to be built by an instructor who has no prior experience on ontology building and no huge familiarity with the course material used in this study. This cost could have been reduced with an appropriate experience in building ontologies and/or higher familiarity with course content. The generation of a total of 428 questions using the machine described above took around 8 hours including the time required to compute pair-wise similarities. Finally, Reviewers 1, 2 and 3 spent around 43 minutes, 141 minutes, and 56 minutes respectively. We exclude any question for which more than 15 minutes were spent. This indicates that the reviewer was interrupted during the review of that question. In addition, Reviewer 2 reported that she was taking side notes while reviewing each question. For this reason and for other reasons that could interrupt the reviewer, the cost of the reviewing phase should be regarded as a general indicator only.

In terms of cost, it is interesting to compare between two possible scenarios to generate MCQs. The first scenario is where the questions are manually constructed whereas the second scenario utilises ontology-based question generation strategies. The cost of manual generation is expected to be lower than the cost of developing a new ontology added to the cost of question generation and review. However, a few points should be taken into account here. First, in the second scenario, the ontology is expected to be re-used multiple times to generate different sets of questions. Second, the aim is to generate questions with highly accurate predictions about their pedagogical characteristics which has been shown to be possible in the second scenario. Third, no particular skills/creativity for MCQ construction are required when utilising ontology-based question generation strategies.

Usefulness of questions Figure 2 shows the number of questions rated by each reviewer as: not useful at all, useful as a seed for another question, useful but requires major improvements, useful but requires minor improvements, or useful as it is. As the figure indicates, a reasonable number of questions have been rated as useful by at least one reviewer. More precisely, 63 out of the 65 questions have been rated as either useful as it is or useful with minor improvements by at least one reviewer. And 50 questions have been rated as either useful as it is or useful with minor improvements by at least two reviewers. Finally, 24 questions belong to the same category as rated by all three reviewers. As a concrete example of a question that was rated useful by all 3 reviewers, we present the following question:

Q: ..... refers to "A non-static member variable of a class.":
(A) Loop variable
(B) Instance variable (Key)
(C) Reference variable
(D) Primitive variable

Quality of questions Adherence to the 5 rules for constructing goof MCQs indicates the quality of the generated questions. Figure 3 shows the number

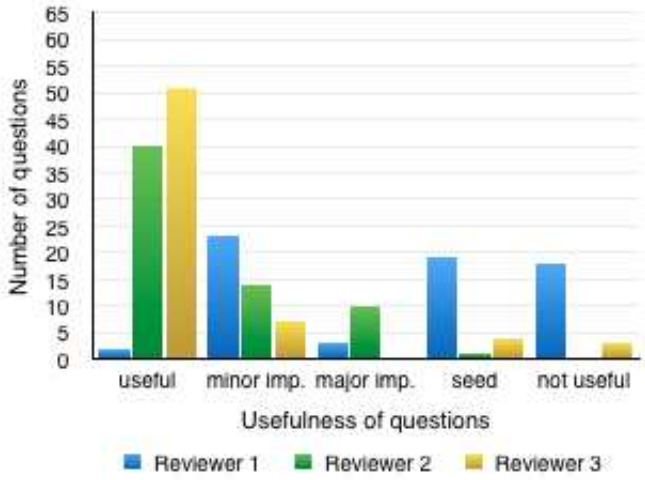

Fig. 2: Usefulness of questions according to reviewers evaluations

of questions adhering to each rule as evaluated by each reviewer. In general, a large number of questions have been found to adhere to Rules R1, R2 and R4. It can be noticed that only a few questions violate Rule R4 (i.e., no clues rule). Recall that a lexical filter has been applied to the generated questions to filter out questions with obvious word clues. This has resulted in filtering out 80 questions. This means that the lexical filter is needed to enhance the quality of the generated questions. The grammatical correctness rule (R5) was the only rule which got low ratings. According to reviewers' comments, this is mainly due to the lack of appropriate articles (i.e., the, a, an). Dealing with this issue and other presentation/verbalisation issues is part of future work.

Difficulty of questions according to reviewers' ratings Part of the objectives of this study is evaluate the accuracy of predictions made by the questions generation tool about the difficulty of each generated question. To do this, we compare difficulty estimations by each reviewer with tool's predictions. Recall that each reviewer was allowed to select from four options of different difficulty levels (too easy, reasonably easy, too difficult, reasonably difficult). This is to distinguish between acceptable and extreme levels of difficulty/easiness. However, tool's predictions can take only two values (easy or difficult). To study tool-to-reviewers agreements, we only consider the two general categories of difficulty. That is, the four categories of difficulty estimations by reviewers are collapsed into two categories only (easy and difficult). Figure 4 shows the number of questions for which there is an agreement between the tool and at least one, two or three reviewers. As the Figure shows, for a large number of questions (51 out of 65 questions) there has been an agreement between the tool and at least one reviewer. To understand the causes of disagreements, we further

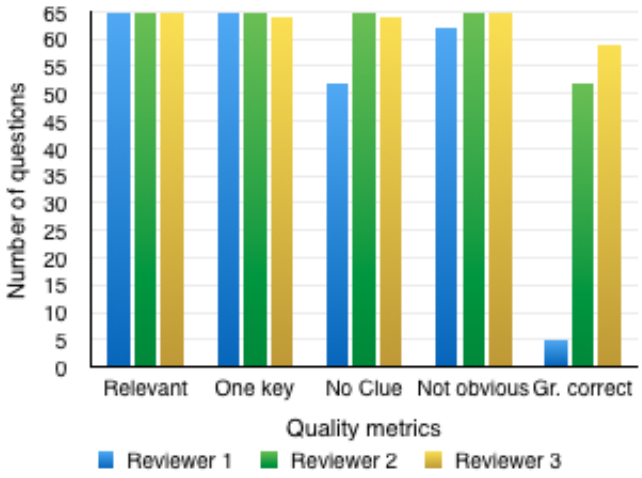

Fig. 3: Quality of questions according to reviewers' evaluations

categorise the agreements according to the difficulty of questions. Table 1 indicates that the degree of agreement is much higher with easy questions reaching 100% agreements with at least one reviewer. This could mean that the generated distractors for difficult questions were not plausible enough. This has been discussed with the ontology developer because we believe that better distractors could be generated by enriching the ontology. In particular, the ontology developer has indicated that many classes in the ontology have been assigned to a single superclass while they could possibly be assigned to multiple superclasses. Restructuring and enriching the ontology is expected to increase the ability of the tool to generate questions at certain levels of difficulty.

Table 1: Accuracy of difficulty predictions for easy and difficult questions

	At least 1 reviewer	At least 2 reviewers	At least 3 reviewers
Easy questions	100%	88%	52%
Difficult questions	65%	35%	2.5%
All questions	78.5%	55.4%	21.6%

Difficulty of questions according to reviewers' performance Each reviewer has attempted to solve each question as part of the reviewing process. Interestingly, non of the reviewers has answered all the questions correctly, including the ontology builder who answered 60 questions correctly. The first and third reviewers have correctly answered 55 and 59 questions respectively. This can have different possible explanations. For example, it could be possible that

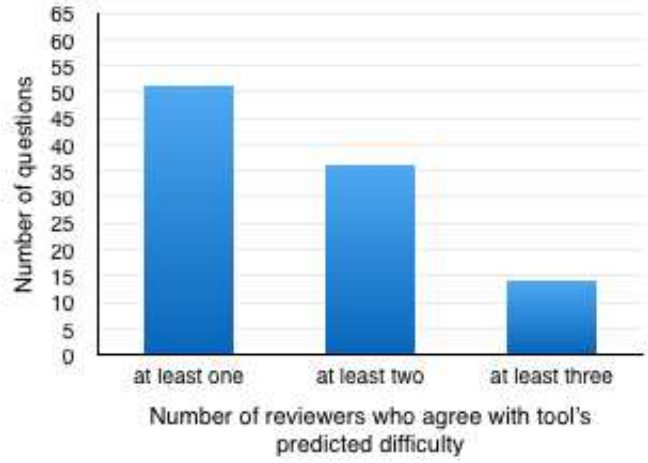

Fig. 4: Difficulty of questions according to reviewers' evaluations

the reviewer have picked a wrong answer by mistake while trying to pick the key. This has actually happened with the first reviewer who has reported this by leaving a comment on one question. Note also that the third reviewer has reported that in exactly one question there was more than one possible correct answer, see Figure 3. This means that if a reviewer picks an answer other than the one identified by the tool as the correct answer then his/her answer will not be recognised as correct. Figure 5 shows the number of questions answered correctly by at least one, two and three reviewers.

In exactly one question, none of the reviewers answered the question correctly, raising a question about the validity of this question as an assessment tool. The stem part of this question was "Which is the odd one out?". To required task to answer the question is to distinguish between the answers which have a common link (the distractors) and the answer which cannot be linked to the other answers (the key). Although all the reviewers have rated this question as "useful", we believe that it is too difficult and not necessarily very useful as an assessment item.

5 Conclusion and future research directions

We believe that ontology-based MCQ generation has proved to be a useful method for generating quality MCQs. The cost of generation is still considered to be high but is expected to be reduced over continuous uses of the same ontology.

As future work, we aim to administer the generated questions to a group of students in order to see how useful the questions are for the purposes of self-assessments. We also aim to add a verbaliser to the MCQ generator to enhance language accuracy. Finally, we believe that there is a potential in using the

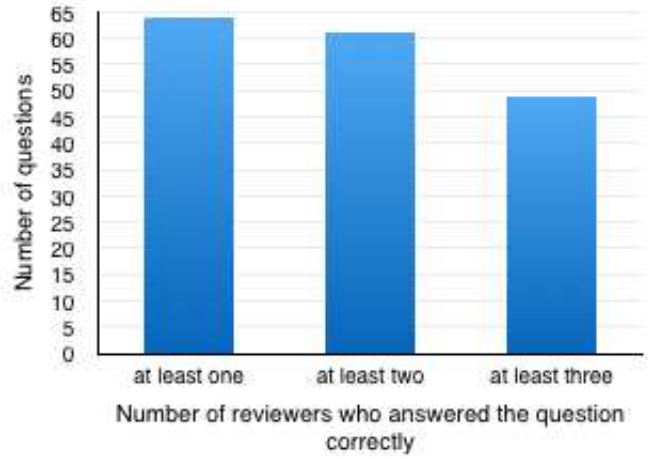

Fig. 5: Difficulty of questions according to reviewers performance

developed MCQ generation methods in application other than assessments. For example, we are interested in exploring the applicability of these methods for ontology evaluation and comprehension purposes.

References

1. T. Alsubait, B. Parsia, and U. Sattler. Automatic generation of analogy questions for student assessment: an ontology-based approach. In *ALT-C 2012 Conference Proceedings*, 2012.
2. T. Alsubait, B. Parsia, and U. Sattler. Mining ontologies for analogy questions: A similarity-based approach. In *OWLED*, 2012.
3. T. Alsubait, B. Parsia, and U. Sattler. Next generation of e-assessment: automatic generation of questions. *International Journal of Technology Enhanced Learning*, 4(3/4):156–171, 2012.
4. T. Alsubait, B. Parsia, and U. Sattler. A similarity-based theory of controlling mcq difficulty. In *Second International Conference on e-Learning and e-Technologies in Education (ICEEE)*, pages 283–288, 2013.
5. T. Alsubait, B. Parsia, and U. Sattler. Generating multiple choice questions from ontologies: Lessons learnt. In *The 11th OWL: Experiences and Directions Workshop (OWLED2014)*, 2014.
6. T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in ontologies: How bad is a cheap measure? In *27th Inernational Workshop on Description Logics (DL-2014)*, 2014.
7. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. (eds.) Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, second edition, 2007.
8. B. S. Bloom and D. R. Krathwohl. *Taxonomy of educational objectives: The classification of educational goals by a committee of college and university examiners. Handbook 1. Cognitive domain*. New York: Addison-Wesley, 1956.

9. B. G. Davis. *Tools for Teaching*. San Francisco, CA: Jossey-Bass, 2001.
10. T. Gavrilova, R. Farzan, and P. Brusilovsky. One practical algorithm of creating teaching ontologies. In *12th International Network-Based Education Conference NBE*, pages 29–37, 2005.
11. T. M. Haladyna. Developing and validating multiple-choice test items. *Hillsdale: Lawrence Erlbaum*, 1994.
12. T.M. Haladyna and S.M. Downing. How many options is enough for a multiple choice test item? *Educational & Psychological Measurement*, 53(4):999–1010, 1993.
13. M. Horridge. A practical guide to building OWL ontologies using Protégé 4 and CO-ODE tools, edition 1.3. http://owl.cs.manchester.ac.uk/tutorials/protegeowltutorial/ [accessed: 18-04-2014], 2011.
14. M. Horridge and S. Bechhofer. The OWL API: A Java API for working with OWL 2 ontologies. In *In Proceedings of the 6th International Workshop on OWL: Experiences and Directions (OWLED)*, 2009.
15. M. Miller, R. Linn, and N. Gronlund. *Measurement and Assessment in Teaching, Tenth Edition*. Pearson, 2008.
16. R. Mitkov, L. An Ha, and N. Karamani. A computer-aided environment for generating multiple-choice test items.cambridge university press. *Natural Language Engineering*, 12(2):177–194, 2006.
17. A. Papasalouros, K. Kotis, and K. Kanaris. Automatic generation of multiple-choice questions from domain ontologies. In *IADIS e-Learning 2008 conference*, Amsterdam, 2008.
18. M. Paxton. A linguistic perspective on multiple choice questioning. *Assessment & Evaluation in Higher Education*, 25(2):109–119, 2001.
19. L. Rudner. *Elements of adaptive testing*, chapter Implementing the Graduate Management Admission Test computerized adaptive test, pages 151–165. New York, NY: Springer, 2010.
20. J. T. Sidick, G. V. Barrett, and D. Doverspike. Three-alternative multiple-choice tests: An attractive option. *Personnel Psychology*, 47:829–835, 1994.
21. S. Sosnovsky and T. Gavrilova. Development of educational ontology for C-Programming. In *Proceedings of the XI-th International Conference Knowledge-Dialogue-Solution, vol. 1, pp. 127132. FOI ITHEA*, 2006.
22. D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR)*, 2006.
23. S. Williams. Generating mathematical word problems. In *2011 AAAI Fall Symposium Series*, 2011.
24. B. Zitko, S. Stankov, M. Rosic, and A. Grubisic. Dynamic test generation over ontology-based knowledge representation in authoring shell. *Expert Systems with Applications: An International Journal*, 36(4):8185–8196, 2008.