ONTOLOGY-BASED MULTIPLE-CHOICE QUESTION GENERATION

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2015

By
Tahani Mohammad Alsubait
School of Computer Science

Contents

Word Count: 33465

List of Tables

List of Figures

Abstract

Assessment is a well understood educational topic with a really long history and a wealth of literature. Given this level of understanding of the topic, educational practitioners are able to differentiate, for example, between valid and invalid assessments. Despite the fact that we can test for the validity of an assessment, knowing how to systematically generate a valid assessment is still challenging and needs to be understood. In this thesis we introduce a similarity-based method to generate a specific type of questions, namely multiple choice questions, and control their difficulty. This form of questions is widely used especially in contexts where automatic grading is a necessity. The generation of MCQs is more challenging than generating open-ended questions due to the fact that their construction includes the generation of a set of answers. These answers need to be all plausible, otherwise the validity of the question can be questionable. Our proposed generation method is applicable to both manual and automatic generation. We show how to implement it by utilising ontologies for which we also develop similarity measures. Those measures are simply functions which compute the similarity, i.e., degree of resemblance, between two concepts based on how they are described in a given ontology. We show that it is possible to control the difficulty of an MCQ by varying the degree of similarity between its answers. The thesis and its contributions can be summarised in a few points. Firstly, we provide literature reviews for the two main pillars of the thesis, namely question generation and similarity measures. Secondly, we propose a method to automatically generate MCQs from ontologies and control their difficulty. Thirdly, we introduce a new family of similarity measures. Fourthly, we provide a protocol to evaluate a set of automatically generated assessment questions. The evaluation takes into account experts' reviews and students' performance. Finally, we introduce an automatic approach which makes it possible to evaluate a large number of assessment questions by simulating a student trying to answer the questions.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487`), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see `http://www.manchester.ac.uk/library/aboutus/regulations`) and in The University's policy on presentation of Theses

Acknowledgements

Once was a dream, today is lived, finishing this thesis was impossible without the support of many people whom I would like to respect and remember for the rest of my life. The determination and the will to never give up are habits I could only develop with the support of my amazing parents through the different stages of my life and through my different educational milestones. A big thank you to my father, Mohammad, and my mother, Mona. I would also like to thank my husband, Mohammed, for keeping my heart full of love; while my mind was full of ideas to strengthen this thesis. I understand that being away from his home country was not easy for my brave and supporting son. For this and for many other reasons, I would like to thank you Abdulaziz. I should also thank my little daughter, Sarah, for waking me up frequently at night through my final year, which definitely helped me to work for some extra hours at night!

Conducting good research and appreciating one when you see it is something you can only learn from an excellent supervisor, and I was lucky to have two of them! No matter where they are or what they are doing, I was always sure they can find the time to help me! Big thanks to my supervisors Bijan Parsia and Uli Sattler whom their excellence in their supervision can be seen through the achievements of their research students: best thesis, best student paper, doctoral prizes, to name a few. There are no words sufficient enough to thank them for creating such an amazing environment where you can only compete with the best of research students.

I would also like to thank my friends in the IMG group at the University of Manchester for being always supportive and helpful even during the busiest times in the year when we were all chasing deadlines. Thanks to Samantha Bail, Chiara Del Vescovo, Rafael Goncalves, Nico Matentzoglu, Jared Leo and many others. Big thanks also to Sean Bechhofer and Robert Stevens. I would also like to thank Barry Cheetham for being always kind and smiling!

I would also like to thank my country, Saudi Arabia, for being generous in supporting research students and giving me and many others the opportunity to get our degrees from top ranked international universities. Big thanks also to my Saudi friends in Manchester who let me feel home even though we were hundred miles away. Thanks to Amal, Helen, Mona and many others! Your support during the first days after giving birth to my little angel Sarah will never be forgotten.

For all of you, thanks for turning this dream to reality!

Chapter 1

Introduction

"Judge a man by his questions rather than by his answers."
- Voltaire

Asking a question could in many cases be harder than answering one. Indeed, people do ask questions on a daily basis for different purposes. The challenge in forming an appropriate question could be in some cases explained by knowing the purpose of the enquiry. According to Greaser et al. [GRC08], questions can be asked for the following purposes: (i) the correction of knowledge deficits (e.g., to fill a gap in knowledge), (ii) the social coordination of action (e.g., to request a permission), (iii) the control of conversation and attention (e.g., rhetorical questions) and (iv) the monitoring of common ground (e.g., to assess what a student knows about a topic). We focus on the last, i.e., the generation of questions to assess students' knowledge. Generating such questions is challenging as they must accurately measure complex and invisible mental representations of knowledge and yet are expected to provide valid and reliable information upon which important decisions can be made (e.g., college admission).

Economically speaking, it is estimated that the cost of developing one question for a high-stake test, e.g., a nation-wide standardised test, can range from $1,500 to $2,000 [Rud10]. It has also been reported [Ach00] that a large amount of money is increasingly spent on large-scale testing (e.g., US spending doubled from $165 million in 1996 to $330 million in 2000). In addition to its high cost, manual assessment generation constitutes a significant part of the workload of educators. It is expected that the time spent on test preparation could be utilised in better ways. For instance, Randi Weingarten, the president of the American Federation of Teachers, pointed out that "If educators spent less time on test preparation and testing, we could optimise that time for more instruction or for teachers to collaborate and plan lessons" [Wei13].

A question in which a set of plausible answers are offered to the student to choose from is called a Multiple Choice Question (MCQ). Providing a set of plausible answers might or might not make the question easier for the student as we will see in detail later. However, preparing reasonably good answers definitely requires more time and effort from the question designer. Typically, a larger number of MCQs is used in a single test compared to other kinds of questions, e.g., essay questions. This makes MCQ exams hard to construct, despite the fact that they have some advantages such as objectivity and ease of marking. We primarily focus on developing methods to automatically generate this particular type of questions.

It is expected that developing methods for the generation of assessment items can provide an excellent stepping stone to the more general problem of generating instructional content. Moreover, research on the automatic generation of MCQs provides a good baseline for the generation of other kinds of questions such as T/F questions, match questions and fill in the blank.

1.1 Multiple Choice Question (MCQ) generation

1.1.1 The what and why

Assessment is central to (specially modern) education. It it that part of education concerned with measuring achievements in order to help educators to continually reflect on and adjust the teaching and learning process, hence helping to provide better education. There are different families of assessment which can be categorised by their central purpose [WT08]. The so-called *formative assessment* aims at providing feedback both to students and teachers while *summative assessment* is mainly used to certify the achievements of students. *Evaluative assessment* is used to evaluate the quality of the educational institution or program.

Assessment items, i.e., questions, can be classified into two widely used types: (i) Objective (e.g., MCQs or True/False questions) and (ii) Subjective (e.g., essays or short answers) [Gro82]. Each family of questions has its own advantages and disadvantages when considering the different phases of assessment, i.e., setting, taking and marking. On the one hand, objective tests can be used to assess a broad range of knowledge and yet require less administration time. In addition, they are scored easily, quickly and objectively either manually or automatically and can be used to provide instant feedback to test takers. On the other hand, objective questions are hard to prepare and require considerable time per each question [SBD94]. For example, Davis [Dav01] and Lowman [Low95] pointed out that even professional test developers cannot prepare more than 3-4 items per day. In addition to the considerable preparation time, manual construction of MCQs does not necessarily imply that they are usually well-constructed. See for example the study carried out by Paxton [Pax00] who has analysed a large number of MCQs and reported that they are often not well-constructed. For example, she pointed out that some questions were badly worded, ambiguous or contain (misleading) hints to a wrong answer.

An MCQ consists of a stem, a key and a set of distractors. The stem is the main part of an MCQ which presents a question, a problem to be solved or an incomplete statement to the students. The key is the correct (or best) answer; in contrast to the distractors which are the wrong answers. This structured format of MCQs lends itself to computerised generation. The core challenge in MCQ generation is the generation of good distractors that must appear plausible to a student who does not know the correct answer. For example, Sidick et al. [SBD94] reported that it requires 5 minutes to prepare each distractor. Many guidelines have been proposed to ensure the effectiveness of distractors; however, many major issues are still debatable such as the optimal number of distractors. According to Haladyna and Downing [HD93], the percentage of questions with three effectively performing distractors ranged from only 1.1% to 8.4%. They also concluded that two (functional) distractors may be just the natural limit for human test developers. This indeed strengthen the need to develop alternative automatic test development methods. Moreover, if each distractor takes 5 minutes to prepare, as reported earlier by Sidick et al. [SBD94], then the time needed to construct MCQs can be reduced considerably by limiting the number of distractors. For example, preparing three distractors instead of five distractors per item can save a total of 16 hours of work over 100 questions [SBD94]. MCQs with fewer distractors can also reduce testing time. Alternatively, a larger number of items can be used in a single test. For example, Aamodt and McShane [AM92] have estimated that the number of administered items in a fixed time can be increased from 100 to 112.4 items if we reduce the number of distractors from 5 to 3. This in turn can provide better sampling of content. The theoretical downside of having MCQs with fewer distractors is the increased guessability, however it has been shown that MCQs with fewer distractors are not necessarily less reliable nor less valid [SBD94]. Quality of MCQs is influenced by the quality, rather than quantity, of distractors.

The automatic generation of MCQs in particular and assessment questions in general can help to resolve many issues in students' assessment. For example, constructing a bank of questions of known properties can help to eliminate cheating by facilitating the preparation of different tests with similar properties (e.g., item difficulty, related content). Also, instead of using last years' actual exams as practice-exams, one can generate exams that resemble the original ones.

Developing automatic methods for question generation (QG) can indeed alleviate the burden of both paper-and-pencil and technology-aided assessments. Of particular interest are large-scale tests such as state or nation-wide standardised tests and tests delivered as part of Massive Open Online Courses (MOOCS). Typically, these tests consist mainly of MCQs [SER13b]. In addition, different modes of delivery (e.g., static, adaptive) can benefit from the automatic generation of questions. One of the promising applications is the delivery of questions that adapt themselves to the abilities of test takers to measure their knowledge in a shorter administration time [Urr77, HN10].

1.1.2 The how

Abstractly speaking, a QG system takes, as input, a knowledge source and some specifications describing the questions to be generated. As output, it produces questions which assess someone's understanding of that knowledge and which adhere to the given specifications. These specifications can include, for example, the format of the questions, their cognitive complexity and difficulty.

As for the format of the question, we focus on single response (i.e., one key) multiple choice questions with two or more distractors. This can be done in two phases. Firstly, for each possible stem, all correct and incorrect answers are generated. Secondly, combinations of these answers can be used to generate a set of questions with the desired number of keys and distractors.

Generally, test designers try to generate questions that target a range of cognitive processes (e.g., knowledge recall, reasoning). To achieve this, we do not focus in this thesis on a particular type of questions, but rather try to generate a reasonable variety of questions (w.r.t. their cognitive level). We elaborate on these design options in Chapters 3 and 4.

One of the necessary functionalities of automatic QG systems is the ability to control the difficulty of the generated questions. This can help in generating tests with properly balanced item difficulties. For example, Lowman [Low95] suggests that only a few questions in any exam should be answered correctly by more than 90% or less than 60% of students. In addition, automatic estimations of the questions' difficulty can help to advance research on adaptive assessment systems which usually rely on training data to estimate the difficulty [HN10]. However, generating questions of a certain difficulty is challenging. We address this challenge by developing a novel similarity-based theory of controlling MCQ difficulty. In particular, we examine whether varying the similarity between the key and distractors of each question can vary the difficulty of the generated questions. We elaborate on this in the following sections and further in Chapter 3.

Two alternative sources are typically used for QG: unstructured text and ontologies. In addition, defective questions or old questions in question banks can also be recycled to generate new better items. The QG workshop (2009) identified raw text as the preferred knowledge source for the workshop participants [RG09]. However, a drawback of most existing text-based QG approaches is that they are unable to generate good distractors from text and that they mostly generate shallow questions about explicit information as it is difficult to infer implicit relations using current NLP techniques. Existing attempts to generate questions from text include [Ste91, Fai99, MH03, MAHK06, BFE05, HN05a, LWGH05, SSY05, Hei11, AM14]. Similarly, many ontology-based QG approaches have been developed [CNB03, HMMP05, HMMP06, ZSRG08, PKK08, CT09, CT10, ZPK11, AY11, AY14]. These approaches take advantage of the structured representation of knowledge and the reasoning services offered for ontologies to generate questions about implicit knowledge. However, there are still many opportunities to take ontology-based QG approaches to the next level both theoretically and empirically. In particular, we aim to develop principled methods for the generation of valid assessment questions and control their quality and difficulty. We focus on generating questions from ontologies and justify this design decision in the following section with an illustrative example.

1.2 Ontology-based MCQ generation

1.2.1 The what and why

The term “ontology” is increasingly rising to prominence in educational research. For example, Figure 1.1 shows the increasing number of ontology-related peer-reviewed articles in the ERIC[1] database of educational publications since 1995. Of course, ERIC’s resources capture only a small percentage of online publications and we expect that the overall number of ontology-related educational publications to be proportional to the number of resources in ERIC database.

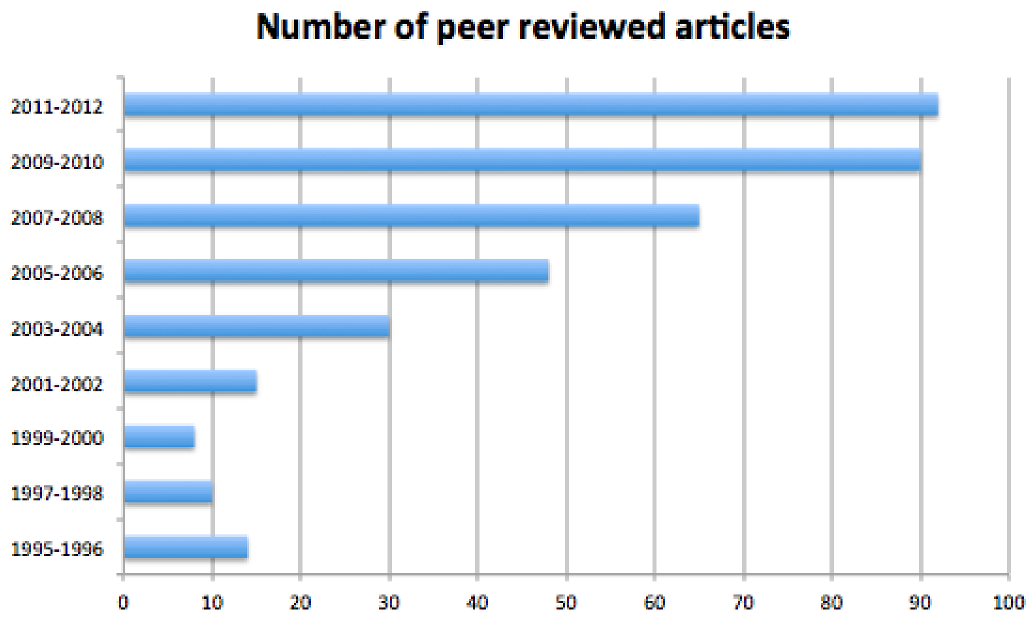


Figure 1.1: Number of ontology-related peer-reviewed articles in ERIC since 1995

So what is an ontology and why can it be fruitful for mining questions? The Web Ontology Language (OWL)[2] is a W3C standard since 2004 and is now part of the Semantic Web stack which includes RDF, RDFS, SPARQL, etc. The prior version of OWL was published in 2009. The current version, which is referred to as OWL2, was published in 2012. An ontology is an engineering artefact which provides formal and machine processable statements about the basic notions of a domain of interest. In OWL ontologies, these statements are referred to as axioms and they describe classes, properties, individuals and any

[1]http://eric.ed.gov/
[2]http://www.w3.org/TR/owl-overview/

interesting relations between them. OWL ontologies are based on Description Logics [BCM+07] which are decidable fragments of first order logic. This makes an ontology a logical theory which allows us to infer implicit knowledge from the explicitly stated knowledge. This reasoning is a key feature of ontologies which makes them superior to other knowledge sources such as raw text. An example of an ontology is presented in Figure 1.2 which is a visual representation of the axioms in Example 1.1.

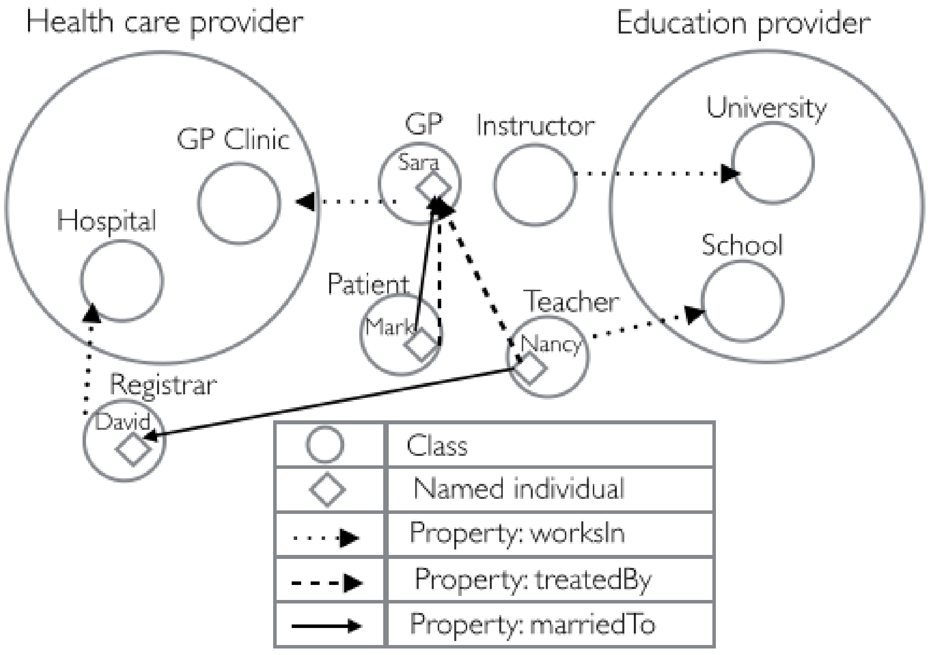


Figure 1.2: An example of a simple OWL ontology

Example 1.1

$$
\begin{array}{ll}
Hospital \sqsubseteq HealthCareProvider, & GPClinic \sqsubseteq HealthCareProvider, \\
University \sqsubseteq EducationProvider, & School \sqsubseteq EducationProvider, \\
Registrar \sqsubseteq \exists worksIn.Hospital, & GP \sqsubseteq \exists worksIn.GPClinic, \\
Teacher \sqsubseteq \exists worksIn.School, & Instructor \sqsubseteq \exists worksIn.University, \\
\multicolumn{2}{l}{LuckyPatient \sqsubseteq Patient \sqcap \exists marriedTo.(\exists worksIn.HealthCareProvider),} \\
Patient(Mark), & Teacher(Nancy), \\
Registrar(David), & GP(Sara), \\
treatedBy(Mark, Sara), & marriedTo(Mark, Sara), \\
treatedBy(Nancy, Sara), & marriedTo(Nancy, David)
\end{array}
$$

Ontologies are usually developed with the help of domain experts who agree on how to describe the main concepts of the domain. Large and rich ontologies can be re-used in different applications or different projects. For example, the average number of projects per ontology in the NCBO BioPortal[3] library of ontologies (as in August 2014 and excluding ontologies with no recorded projects) is 3 with 52 being the maximum number of projects for a single ontology which is the Gene Ontology.[4]

A considerable body of research has been devoted to different ontology-related areas. This includes topics about the theoretical foundations of ontologies as well as empirical studies. A growing body of research is also devoted towards ontology applications. Research in this area can help in making ontologies more mature. The efforts made in this field can help in realising the gaps in the current state of ontologies and ontology services. For example, as part of this research, which considers assessment as a potential application, we observed that there is a need for developing similarity measures that can deal with expressive ontologies. We provide more details on this topic in the following section.

Ontologies with potential educational value are available in different domains such as Biology, Medicine, Geography, to name a few.[5] However, ontologies are typically not designed for educational use. Thus, there is a challenge in generating useful instructional content from them. Consider the first attempt to re-use a knowledge base from one context into another: the GUIDON [Cla83, Cla87] program developed at Stanford University. The basic idea of the GUIDON project is to re-use existing knowledge sources for tutoring purposes. One lesson we can learn from this project is that recycling existing knowledge bases is not always straightforward.

Similarly, we explore the possibility to generate assessment items from ontologies. Of course, there is still a lot to be done in developing ontologies for various domains and topics. This means that in some cases, there is a need to first build an ontology for a specific subject before utilising it for QG. Thus, there is a trade-off between the efforts required to build and maintain the ontology and the overall advantage of single or multiple uses.

[3]http://bioportal.bioontology.org/

[4]http://www.geneontology.org

[5]For a list of ontology repositories, the reader is referred to: http://owl.cs.manchester.ac.uk/tools/repositories/

1.2.2 The how

A reasonable number of questions can be generated from the ontology in Example 1.1. The generation can involve two steps: (1) generating question candidates and (2) transforming the candidate questions to grammatically well-formed questions. The second step is out of the scope of this research. However, for readability, we present below some stems that can be generated from the ontology in Example 1.1 after making any necessary grammatical transformations.

1. Give an example for a health care provider.
2. What is a GP clinic?
3. Where does an instructor work?
4. To whom is Mark married?
5. Which one of the following definitions describe a lucky patient?
6. Nancy to David is as .... to ....?
7. Instructor to University is as .... to ....?
8. Name one of the lucky patients.

The above questions range from simple recall questions (e.g., 1-5) to questions that require some sort of reasoning (e.g., 6-8). For each of the above stems, it remains to specify a key and some suitable distractors. The challenge is to pick distractors that look like plausible answers to those students who do not know the actual answer. For example, for Question 4, the answers are expected to be names of persons. Including distractors of the wrong sort such as names of institutions, would make the correct answer stand out even for a low mastery student. So we need a mechanism to filter out obviously wrong answers.

After seeing an example to generate questions from a toy ontology, we want to know what is the case in real ontologies which are usually big and rich. For example, the average number of axioms per ontology in BioPortal is 20,532 with a standard deviation of 115,163 and maximum number of 1,484,923 [HPS11]. This suggests that a considerably large number of questions can be generated from a single ontology. We investigate this by generating some questions from a few selected ontologies from BioPortal. The ontologies were selected according to the following criteria: (i) has a fair number of classes (i.e., 500-3000), (ii) recently updated and (iii) actively used by 2 or more projects. This has led us to the following five ontologies: Units of measurement (2507 classes), Sequence types and features (2032 classes), Plant Ontology (1553 classes), Amphibian gross anatomy (700 classes) and Spider Ontology (577 classes). These ontologies were retrieved from BioPortal in late 2011. From each ontology, we generated some MCQs using three approaches. The first approach, which will be referred to as the naive approach, was proposed by Zitko et al. [ZSRG08] and is based on generating the distractors randomly without filtering them. The second approach, which will be referred to as the customised approach, was proposed by Papasalouros [PKK08], and it utilises custom strategies to generate good (but limited) distractors. The third approach, which will be referred to as the similarity approach, is based on generating distractors according to a simple notion of similarity (e.g., sharing common subsumers). Of course, due to the diversity of assessment questions, we cannot aim to generate all possible questions. Rather, for our preliminary investigation, we have generated only two forms of MCQs:

(i) Stem: What is X?, Key: a superclass of X, Distractors: 3 non-superclasses of X.

(ii) Stem: Which of the following is X?, Key: a subclass of X, Distractors: 3 non-subclasses of X.

Table 1.1 shows how many questions were generated from each ontology using the three approaches. Note that these questions may have redundant stems but each question-stem pair has a different set of answers. The table also shows how many questions could not be generated due to lack of a minimum of 3 distractors.

Clearly, a massive number of questions was generated from each ontology.[6] The question that arises here is: are these questions all good? For example, we expect the questions generated using the naive approach to be highly guessable and thus of low pedagogical value. There is a trade-off between generating all the good questions (and possibly some bad questions) and generating only good questions (and possibly decreasing coverage rate). For example, Figure 1.3 shows the number of questions generated from the above ontologies using the similarity

[6]To calculate the number of unique stems generated from each ontology, multiply the number of classes in that ontology by 2 then subtract the number of questions failed to be generated from the ontology using the naive approach (column F-N in Table 1.1)

Ontology	G-N	G-S	G-C	F-N	F-S	F-C
Units of measurement	3.44005E+13	104,100,285,2	228,937,71	887	890	137,1
Sequence types and features	2.32499E+13	503,531,581	287,034,5	204	210	771
Plant Ontology	7.07411E+12	472,131,950	896,546,1	134	140	701
Amphibian gross anatomy	37,045,208,361	531,51	137,48	519	563	585
Spider Ontology	330,712,156,64	102,847	164,99	261	325	376

Table 1.1: The number of questions generated (G) and number of questions failed (F) to generate from 5 BioPortal ontologies using the three approaches naive (N), Similarity (S) and Customised (C)

and customised approaches. As the Figure shows, the mechanism used for filtering the distractors makes a big difference. Questions arising here are: how good are these approaches in filtering out bad questions and can these approaches be used to control difficulty? We elaborate on these issues in the next section.

1.3 Similarity-based MCQ generation

1.3.1 The what and why

To generate pedagogically sound questions, i.e., of controlled difficulty, we need a pedagogically plausible theory for selecting good distractors. Ideally, we want students' performance to correlate with their knowledge mastery (i.e., amount and quality of knowledge). This means that difficult questions are expected to be answered correctly by high mastery students only while easy questions are expected to be answered by both low and high mastery students.

One of the possible ways to control MCQs difficulty is to use some notion of similarity for selecting distractors. The basic intuition is that offering a set of very similar answers makes it difficult to distinguish the correct answer; hence, increases the need to know more about the topic of the question. Also, it decreases the possibility to identify the correct answer by ruling out obviously wrong answers. Thus, to generate a difficult question, we pick distractors with high similarity to the key. And, to generate an easy question, we pick distractors with low

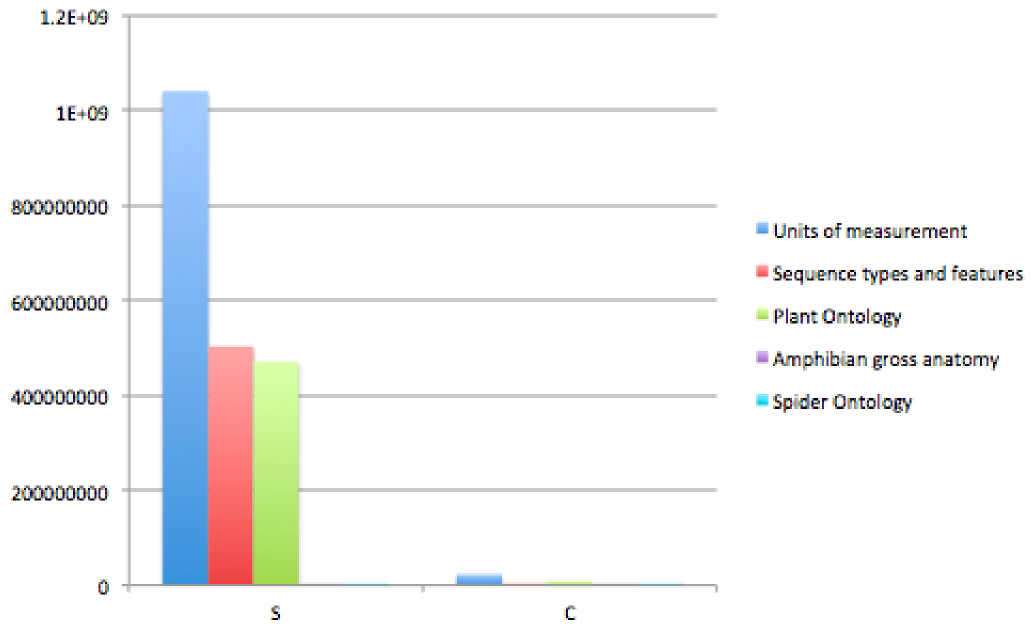


Figure 1.3: Number of questions generated using the two approaches similarity (S) and customised (C)

similarity to the key. In addition, similarity between the key and stem can play a role in controlling the difficulty of MCQs, given reasonable measures of similarity between the key and stem.

As an example, we would expect the difficulty of Question 4 above to increase by providing a list of GP names as distractors since the correct answer "Sara" is also a GP. So, someone who knows that Mark is married to a GP would still need to know the exact name of that GP. This means that a student who knows more about the subject of the question, performs better.

1.3.2 The how

The question that remains to be answered is how can we measure the similarity between the key and distractors? One would expect that there is an impact for using a particular (precise or imprecise) similarity measurement method on the overall quality of the QG method. This is why it is very important to use a "precise" similarity measure that takes into account all the knowledge we know about the compared objects. However, designing a precise similarity measure for ontologies is a big challenge. Looking at existing similarity measures (e.g., [RMBB89, Res95, Lin98, JC97, WP94, ODI07, Jan06, dSF08, dFE05, dFE06, LT12]), we found that no off-the-shelf existing method satisfies our requirements. For example, some measures which we refer to as taxonomy-based measures [RMBB89, WP94, Res95, Lin98, JC97] are imprecise by definition as they only consider atomic subsumptions and ignore any complex subsumptions which can affect similarity computation. Other measures impose some requirements on the ontology that can be used for similarity computation (e.g., low expressivity, no cycles, availability of an *ABox* or external corpus of annotated text).

This has motivated us to develop a new family of similarity measures which can be used with any ontology. The basic rationale of the new measures is that similar concepts have more common and fewer distinguishing features. We introduce our novel similarity measures in Chapter 5 and report on our findings from using the new measures with over 300 ontologies in Chapter 6.

1.4 Topics and contributions of this thesis

The topic of this thesis is interdisciplinary in the sense that it is built upon different areas such as pedagogy, psychology, AI, reasoning and Description Logics ontologies. To increase the readability of the thesis, we choose to localise the background and related work of each topic in the relevant chapters. The thesis is structured around three main topics introduced in the following sub sections.

1.4.1 Foundations and methods of QG

The broad aim of this thesis is to advance the state-of-the-art in ontology-based MCQs generation. Prior to the work presented here, most of the work on this topic was unprincipled and lacked theory backing. This is why we devote Chapter 3 for establishing the foundations of QG in general and in the context of ontologies in particular. In the following chapter (i.e., Chapter 4) we describe the landscape of ontology-based question generation and lay out some possible design options. The chapter also justifies the decisions we made as part of this research to design particular methods and algorithms to generate specific types of assessment questions from ontologies.

1.4.2 Measuring similarity in Ontologies

QG methods presented in this thesis depend on measuring the similarity between various ontology concepts. However, we found no precise and widely applicable off-the-shelf similarity measure. There was also a need for characterising existing similarity measures and understanding the possible problems associated with them. Chapter 5 therefore focuses on formulating the problem of measuring similarity in Description Logics Ontologies and presents a new family of similarity measures that overcome the problems of previous measures. In Chapter 6 we empirically examine the new measures with a large number of real ontologies and compare them to existing measures. We introduce the notions of "expensive" and "cheap" similarity measures and examine the new measures to find a cheap but good measure.

1.4.3 Applicability of the developed QG and similarity methods

The aim of the last part of this thesis is to investigate the applicability of the methods presented in former parts of the thesis. The main context for the applicability investigation is, of course, student assessment. We evaluate QG methods in real class settings taking into account data gathered from both experienced test developers and students. We also show that it is possible to automatically evaluate the difficulty of the generated questions.

In addition, we investigate the applicability of the developed QG methods and similarity measures for different ontology-related areas such as ontology comprehension, ontology development and ontology validation.

1.5 Published work

Some of the work presented in thesis has already been published in peer-reviewed workshops, conferences and journals. Below is a list of these publications.

1. [APS12c] T. Alsubait, B. Parsia, and U. Sattler. Mining ontologies for analogy questions: A similarity-based approach. In Proceedings of the 9th OWL: Experiences and Directions Workshop (OWLED2012), 2012.

2. [APS12a] 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.
3. [APS12b] T. Alsubait, B. Parsia, and U. Sattler. Automatic generation of analogy questions for student assessment: an ontology-based approach. Journal of Research in Learning Technology, 20:95-101, 2012.
4. [APS12d] 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.
5. [APS13] T. Alsubait, B. Parsia, and U. Sattler. A similarity-based theory of controlling MCQ difficulty. In Proceedings of the Second International Conference on e-Learning and e-Technologies in Education (ICEEE), 2013.
6. [APS14e] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in ontologies: How bad is a cheap measure?. In Proceedings of the 27th International Workshop on Description Logics (DL-2014), 2014.
7. [APS14b] T. Alsubait, B. Parsia, and U. Sattler. Generating Multiple Choice Questions From Ontologies: Lessons Learnt. In Proceedings of the 11th OWL: Experiences and Directions Workshop (OWLED2014), 2014.
8. [APS14c] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in Ontologies: A new family of measures. In Proceedings of ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference (ISWC 2014), 2014.
9. [APS14a] T. Alsubait, B. Parsia, and U. Sattler. Generating Multiple Choice Questions From Ontologies: How far can we go?. In Proceedings of the First International Workshop on Educational Knowledge Management (EKM 2014), 2014.
10. [APS14d] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in Ontologies: A new family of measures. In Proceedings of the 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2014), 2014.

Chapter 2

Preliminaries

This chapter lays out the basic terminology used throughout the thesis and briefly introduces the core background notions. The chapter is structured around two main topics: (i) ontologies and the logic formalisms that underpin them and (ii) assessment questions and related pedagogical notions.

2.1 Description Logics and Ontologies

The terms below are ordered such that later terms build on top of or use earlier terms. In what follows, we shed a light on some terms used to refer to the basic components of an ontology and some other related notions. We fix both the terms that are usually used in a DL context and the corresponding terms used in an OWL context.

2.1.1 Nomenclature

As a starting point, let us introduce some terms which are commonly used interchangeably although they each have unique meanings (at least in this thesis). The goal is to arrive at an understanding of what is and is not an ontology. The following terms all refer to knowledge representation (KR) approaches aimed at defining a set of shared terms of interest in some domain. The approaches vary in how formal and how expressive they are (with ontologies being the most formal and most expressive).

A **controlled vocabulary** is a collection of terms, possibly with their unambiguous but informal definitions [Gar04, Pid14]. The collection is controlled by a registration authority which keeps it redundancy-free. No relations are defined between the terms of a controlled vocabulary.

A **taxonomy** is a controlled vocabulary that is enriched with generalisation/specialisation relations or the so-called is_a relations [Gar04, Pid14]. A taxonomy can be said to have a hierarchical structure.

A **classification system** is a taxonomy with the principles that underlie the classification of concepts.

A **thesaurus** is a taxonomy that is enriched with more labels, in particular with different kinds of relations (e.g., is_a, associated_with) [Gar04, Pid14]. In this sense, thesauri have network structures.

A **terminology** is a thesaurus that is enriched with a glossary and further explanations regarding concepts' meanings. Those meanings can be defined in different contexts which can help in promoting the consistent usage of terms.

An **ontology** contains concept definitions and general axioms which constrain the possible interpretations of the defined concepts [Gru93]. In addition to concept definitions and general axioms, an ontology can have knowledge about the current state of the world [End01].

In general, a **knowledge base**, regardless of the formalism underlying it, can refer to any of the above terms, e.g., it can be a taxonomy, a thesaurus, an ontology or any other structured representation of knowledge.

Description Logics (DLs) are a family of KR formalisms that are equipped with precisely defined semantics (usually model-theoretic semantics) [BCM+07]. In contrast, historical predecessors of DL knowledge bases (e.g., semantic networks and frames) have no precisely defined semantics [BCM+07]. In this sense, a DL knowledge base is a logical theory which means that implicit knowledge can be inferred from the explicitly stated knowledge. A DL knowledge base (KB), which can also be called an ontology, is defined below.

Many DLs are decidable fragments of first order logic (FOL) [Bor96]. One of the differences between DLs and FOL is the variable free syntax of DLs. Moreover, practical decision procedures for key inference problems have been developed even for expressive DLs.

The **Web Ontology Language (OWL)** is the World Wide Web Consortium (W3C) standard ontology language for the web which was standardised in 2004 [GHM+08]. This standard ontology language exploits DLs. This exploitation is influenced by the need to provide ontologies with semantics and hence helps to make the Semantic Web vision a reality. In this sense, an OWL ontology is a machine processable artefact. An OWL ontology can be mapped to a DL knowledge base. However it must be noted that an OWL ontology can have more components than a DL knowledge base (e.g., annotations, imports).

Different syntaxes [Hor10] have been developed for OWL such as the Manchester Syntax which aims at providing a human-readable syntax [HGR+06]. For example, a DL statement describing the concept of a lucky patient ($LuckyPatient \sqsubseteq Patient \sqcap \exists marriedTo.Doctor$) can be written in Manchester syntax as follows:

Class: **LuckyPatient**
    SubClassOf: **Patient** *and* **marriedTo** *some* **Doctor**

In 2009, a second version of OWL (referred to as OWL2) was published [GHM+08] and later updated in 2012. OWL 2 has different profiles (i.e., language variants) with different expressivities, referred to as OWL 2 DL and OWL 2 Full. The DL which underpins OWL 2 DL is $\mathcal{SROIQ}(\mathcal{D})$ which has a worst case complexity of N2ExpTime [MCGH+12] for core reasoning services. In order to provide more practical reasoning services, subsets of OWL 2 DL were defined, namely OWL 2 DL profiles [MCGH+12]. Each profile allows a restricted set of constructors, hence compromises expressivity, to provide more efficient reasoning. Three main profiles are defined: OWL 2 EL (Existential Language), OWL 2 QL (Query Language) and OWL 2 RL (Rule Language). The OWL 2 EL profile is based on the $\mathcal{EL}$ family of DLs [BBL05] which benefits from tractable decision procedures (PTime) for key reasoning problems [MCGH+12]. The OWL 2 QL profile is based on the DL-Lite family of DLs [ACKZ09] and aims at providing practical query answering (LOGSPACE w.r.t. data size) [MCGH+12]. The OWL 2 RL profile allows to implement scalable reasoning systems using rule-based reasoning engines [MCGH+12]. In this thesis, we refer to OWL 2 ontologies whenever we use the term ontology, unless explicitly stated otherwise.

The growth in popularity of OWL motivated the development of a number of optimised reasoners [GBJR+13] such as FacT++ [TH06], Pellet [SPCG+07], HermiT [SMH08], JFact,[1] ELK [KKS11], MORe [RGH12], jCEL [Men12], Chainsaw

[1]http://jfact.sourceforge.net/

[TP12], Konclude,[2] Mastro [SLL+10], ELepHant [Ser13a], Ontop [Kon14], RACERPro [Haa12] to name a few.[3] Different ontology editing and processing tools and libraries have been developed as well such as *Protégé*,[4] Swoop [KPS+05] and the OWL API [BVL03]. Influenced by what can be referred to as maturity of ontology tools, the interest in ontologies has spread from academia to industry. Many ontology-based projects have been observed both in industry and in non-profit national and international bodies. Examples include NASA's SWEET project (i.e., Semantic Web for Earth and Environmental Terminology),[5] The US National Cancer Institute thesaurus (NCIt),[6] The General Motors variation-reduction adviser [MCG+05] and NCBO BioPortal library of biomedical ontologies [NSW+09].

Atomic concepts, or FOL unary predicates, stand for sets of objects, e.g., *Animals*, *Flowers* and *Courses*. In the OWL context, a DL concept is called a class. We use N_C to denote the set of atomic concepts. This set usually includes the Top concept $\top$ (OWL:Thing) and maybe the Bottom concept $\bot$ (OWL:Nothing). Specifying the exact items of this set should be made clear whenever it is referred to. Throughout this thesis, capital letters A and B (possibly with subscripts) are used to refer to atomic concept names and, for concrete examples, we write concept names in camel case with the initial letter capitalised.

Individuals, or FOL constants, stand for instances of classes, e.g., Tom and $Java101$. We use N_I to denote the set of individual names. Throughout this thesis, lowercase letters a and b are used to refer to individual names and, for concrete examples, we write individual names with the initial letter capitalised.

Roles, or FOL binary predicates, stand for relationships between objects, e.g., *worksIn*, *marriedTo* and *treatedBy*. A DL role is called a property in OWL. For practical reasons, properties are separated into two types: namely object properties and datatype properties. Object properties are merely used to describe relationships between objects while datatype properties relate objects to instances of built-in datatypes such integers and dates. We use N_R to denote the set of role names. Throughout this thesis, the letters r and s are used to refer to role names and, for concrete examples, we write role names in camel case with

[2]http://www.konclude.com
[3]for a list of DL reasoners: http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/
[4]http://protege.stanford.edu
[5]http://sweet.jpl.nasa.gov/ontology
[6]http://ncicb.nci.nih.gov/NCICB/core/EVS

an initial lowercase letter.

Complex concepts or **concept expressions** can be composed from atomic concepts (and possibly roles) with the help of constructors (e.g., conjunctions $\sqcap$, disjunctions $\sqcup$, negations $\neg$, existential restrictions $\exists$, universal restrictions $\forall$). For example, to describe the class of doctors who are also married to doctors we write: $Doctor \sqcap \exists marriedTo.Doctor$. Each constructor can be used to convey a specific meaning. Conjunctions ($\sqcap$) which are interpreted as set intersection are used to describe objects that belong to two or more concepts simultaneously. For example, the expression $Doctor \sqcap Male$ refers to the objects who are both doctors and males. Disjunctions ($\sqcup$) which are interpreted as set union are used to refer to objects that belong to one concept or another (or both). For example, $Doctor \sqcup Nurse$ describes the class of objects who are either doctors, nurses or both. Negations or complements ($\neg$) can be used to describe objects of the complement of a concept. For example, $\neg Male$ describes objects that are not instances of $Male$. The concept expression ($\exists worksIn.Hospital$) can be used to describe objects that have at least one $worksIn$ relationship to some instance of the concept Hospital (i.e., the filler or role successor). Similarly, the concept expression ($\forall treatedBy.Doctor$) can be used to describe objects that only have $treatedBy$ relationships to doctors (including those who have no $treatedBy$ relationships at all). Note that concepts can be defined using different combinations of the above constructors to describe complex concepts (refer to Section 2.1.2 to see a list of constructors supported by each DL). Note also that two concepts might be logically equivalent although they are syntactically different. For example, the concept $\neg\exists worksIn.\neg Hospital$ is equivalent to the concept $\forall worksIn.Hospital$. Throughout this thesis, uppercase letters C and D (possibly with subscripts) are used to refer to (possibly) complex concepts.

Axioms are the basic elements of a DL ontology, i.e., an ontology is a (finite) set of axioms. They are statements that describe the relationships between the different concepts, individuals and roles of an ontology. There are different types of axioms which can occur in different "boxes" as we will see in detail below. General Concept Inclusions (GCIs) are concept inclusions (i.e., implications) that allow complex concepts on both sides while definitions allow only atomic concepts on the left-hand side. For example, $Hospital \sqcup GPClinic \sqsubseteq HealthCareProvider$ is considered a GCI in contrast to the axiom $Hospital \equiv HealthCareProvider$ which can be specifically said to be a definition. GCIs can also be referred to as subsumption axioms (or subclass axioms in an OWL context). Role Inclusions (RIs) are statements that specify relationships between roles (e.g., $hasBrother \sqsubseteq hasSibling$). Equivalence axioms are reducible to inclusion (e.g., $Sibling \equiv Brother \sqcup Sister$ is equivalent to $Sibling \sqsubseteq Brother \sqcup Sister$ and $Brother \sqcup Sister \sqsubseteq Sibling$). Equivalences can happen between concepts and/or role names, however, role equivalences are not allowed in OWL. Two special axiom patterns are $\exists r.\top \sqsubseteq C$ and $\top \sqsubseteq \forall r.C$ which specify the **domain** and **range** of a role r respectively.

If C is the domain of r, D is the range of r, and an individual a is related by the r relationship to an individual b, then a and b are inferred to be instances of C and D respectively. Moreover, an axiom is called a **tautology** if it is entailed from *any* ontology, even an empty one (e.g., $A \sqsubseteq A$).

A **TBox** contains all the axioms which describe relationships between concepts. TBox axioms are said to be *terminological*, hence the name TBox. For instance, the axiom $Hospital \sqsubseteq HealthCareProvider$ states that every instance of the concept $Hospital$ is also an instance of the concept $HealthCareProvider$. Some definitions are said to be cyclic (i.e., contains cycles). For example, one can define Humans as $Human \sqsubseteq \forall hasChild.Human$. Cycles can also be indirect, e.g., $C \sqsubseteq \exists r.D, D \sqsubseteq C$. If the TBox contains no cyclic definitions, it is referred to as an acyclic TBox. A definitorial TBox is one which is acyclic, contains only definitions (i.e., no GCIs) and have unique left hand sides of axioms (i.e., each concept is defined only once).

An **ABox** contains all the axioms which describe relationships between individuals and concepts or between individuals and roles. ABox axioms are said to be *assertional*, hence the name ABox. As an example, the axiom $Doctor(David)$ states that the individual $David$ is an instance of the concept $Doctor$ and the axiom $treatedBy(Mark, David)$ states that the individual $Mark$ is in a $treatedBy$ relationship with the individual $David$.

An **RBox** contains axioms that describe relationships between roles. For example, the axiom $hasSister \sqsubseteq hasSibling$ states that any instances that are in a $hasSister$ relationship are also in a $hasSibling$ relationship.

A **DL knowledge base** $\mathcal{K}$ is a tuple: $\mathcal{K} = \langle \mathcal{T} \cup \mathcal{R}, \mathcal{A} \rangle$ where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is an RBox and $\mathcal{A}$ is an ABox. It must be noted that the separation of KBs into TBoxes, RBoxes and ABoxes has no logical consequence. Rather, it is meant to make it easier to understand or re-use the KB. Note also that the axioms of a

DL KB are said to be asserted axioms, compared to axioms that can be inferred from the asserted axioms. For example, if the following axioms were asserted in a KB ($A \sqsubseteq B, B \sqsubseteq C$), then it is possible to infer that ($A \sqsubseteq C$).

Subconcepts are all the concepts that syntactically occur in a (possibly complex) concept. For example, the concept $Male \sqcap \exists marriedTo.(Doctor \sqcup Nurse)$ contains the following subconcepts (in addition to the concept itself): $Male$, $\exists marriedTo.(Doctor \sqcup Nurse)$, $Doctor \sqcup Nurse$, $Doctor$ and $Nurse$. Subconcepts can also be defined for an ontology $\mathcal{O}$ as all the concepts that syntactically occur in any axiom of $\mathcal{O}$.

The **signature** of a concept, axiom or set of axioms is the set of terms (i.e., concept, individual and role names) which occur in the concept, axiom or set of axioms, respectively. The signature of an ontology is the union of the sets N_C, N_R and N_I, i.e., the sets of concept, role and individual names that are used in $\mathcal{O}$. The symbol (Σ) is usually used to refer to a signature. A signature of an ontology $\mathcal{O}$, axiom α, concept C can be denoted as $\widetilde{\mathcal{O}}$, $\widetilde{\alpha}$ and $\widetilde{C}$, respectively. As an example, consider the small ontology $\mathcal{O} = \{Doctor \sqsubseteq \exists worksIn.Hospital, Patient \sqsubseteq \exists treatedBy.Doctor\}$. The signature $\widetilde{\mathcal{O}}$ in this case is $\{Doctor, worksIn, Hospital, Patient, treatedBy\}$.

In OWL, a **declaration** is a statement that specifies that a specific term is a class, object/data property or individual name; just as a programmer would declare a variable in a program. **Annotation** properties are statements that contain information about a resource (e.g., an ontology, concept or individual). For instance, the following annotation properties are predefined by OWL: owl:versionInfo, rdfs:label, rdfs:comment, rdfs:seeAlso, rdfs:isDefinedBy. Indeed, ontology engineers can define new annotation properties in their ontology as desired. **Imports** are one type of annotation properties which are usually specified in the header of an ontology. They allow us to include other ontologies as part of the current ontology [HPSMW07]. This allows us to effectively manage separate fragments of one ontology or make use of previously coded knowledge. The statement <owl:imports rdf:resource=ImportedOntologyURI> is used to import the ontology which is specified by the URI in the rdf:resource parameter. The **imports closure** of an OWL Ontology is $\mathcal{T} \cup \mathcal{R} \cup \mathcal{A}$.

2.1.2 Basic Description Logics

As highlighted above, DLs are a family of logic-based formalisms. A specific DL language is characterised by the set of permitted constructors and axiom types. Accordingly, DLs can be classified into lightweight DLs (e.g., $\mathcal{EL}$), highly expressive DLs (e.g., $\mathcal{SROIQ}$) and some DLs in between. There is sometimes a tradeoff between expressivity and tractability of reasoning [LB07]. On the one hand, lightweight DLs have limited expressive power but they have polynomial time worst case complexity for core reasoning services [BBL05]. On the other hand, some DLs with high expressive power are intractable [BBL05].

2.1.2.1 Syntax and semantics

The DL language $\mathcal{ALC}$ (attributive language with complement) [SS91] is the smallest DL which is propositionally closed (i.e., can express all boolean set operations such as intersection, union and complement). We describe the syntax and semantics of the DL $\mathcal{ALC}$ as an example and introduce other DLs in the following section. The following concepts are well-formed $\mathcal{ALC}$-concepts:

$\top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \neg C \mid \exists r.C \mid \forall r.C$

where $A \in N_C$, C and D are $\mathcal{ALC}$-concepts and $r \in N_R$.

The semantics of $\mathcal{ALC}$-concepts can be defined using interpretations. An interpretation $\mathcal{I}$ is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}}$ is the interpretation domain and $\cdot^{\mathcal{I}}$ is the interpretation function. The interpretation domain $\Delta^{\mathcal{I}}$ is a non-empty set of interpretation elements (i.e., instances or objects). The interpretation function $\cdot^{\mathcal{I}}$ maps each concept name A to an extension $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, maps each role name r to a binary relation on $\Delta^{\mathcal{I}}$ and maps each individual name to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Then $\cdot^{\mathcal{I}}$ is extended to complex concepts according to Table 2.1.

An interpretation $\mathcal{I}$ satisfies an axiom $\alpha = C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. In this case, we say that $\mathcal{I}$ is a model of α, denoted $\mathcal{I} \models \alpha$. An interpretation $\mathcal{I}$ is said to be a model of $\mathcal{O}$ if it satisfies all the axioms of $\mathcal{O}$, denoted $\mathcal{I} \models \mathcal{O}$. We say that an axiom α is entailed by an ontology $\mathcal{O}$, denoted $\mathcal{O} \models \alpha$ if $\mathcal{I} \models \alpha$ for every model $\mathcal{I}$ of $\mathcal{O}$.

The name and syntax of each $\mathcal{ALC}$ constructor along with the corresponding model-theoretic interpretations are provided in Table 2.1.

Structural equivalence is a notion in OWL which captures that the order of operands used to define a concept is not important, hence would have no logical consequence. For example, the concept $Doctor \sqcap Male$ and the concept $Male \sqcap$

Constructor	Syntax	Semantics
Top concept	$\top$	$\Delta^{\mathcal{I}}$
Bottom concept	$\bot$	$\emptyset$
Concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Concept intersection (conjunction)	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Concept union (disjunction)	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Existential restriction	$\exists r.C$	$\{x \mid \exists y.\langle x, y\rangle \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
Universal restriction	$\forall r.C$	$\{x \mid \forall y.\langle x, y\rangle \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$

Table 2.1: The syntax and semantics of $\mathcal{ALC}$ constructors

Doctor are said to be structurally equivalent (denoted $Doctor \sqcap Male \equiv_s Male \sqcap Doctor$). Similarly, the concept $\exists marriedTo.Doctor \sqcup \exists marriedTo.Nurse$ and the concept $\exists marriedTo.Nurse \sqcup \exists marriedTo.Doctor$ are equivalent and would be considered repetitions. Also, we say that two ontologies $\mathcal{O}_1$, $\mathcal{O}_2$ are structurally equivalent if they contain only structurally equivalent axioms (denoted $\mathcal{O}_1 \equiv_s \mathcal{O}_2$).

Logical equivalence refers to the property which states that two sets of axioms have the same models. For example, the single axiom $A \sqsubseteq B \sqcap C$ is logically equivalent to the two axioms $A \sqsubseteq B, A \sqsubseteq C$. To denote two logically equivalent ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, we write $\mathcal{O}_1 \equiv \mathcal{O}_2$.

2.1.2.2 Naming conventions

The DL $\mathcal{AL}$ is a restricted version of $\mathcal{ALC}$ which disallows full complements (i.e., it allows complements of atomic concepts only) and full existential restrictions (i.e., it allows limited existential restrictions which only allow $\top$ as a filler, e.g., $\exists r.\top$) [BCM+07, GPFLC04].

For historical reasons, early DLs such as the frame-based $\mathcal{FL}$ family of languages regard universal restrictions as an important part of the language as it corresponds to the slots of the old style frame structure. For instance, the DL $\mathcal{FL}^-$ can be constructed from $\mathcal{AL}$ by disallowing limited complements [BL04, LB07]. And the DL $\mathcal{FL}_0$ further disallows limited existential restrictions [GPFLC04]. However, both DLs allow universal restrictions; just as the DL $\mathcal{AL}$ does. Later, it has been observed that many well-situated medical ontologies (e.g., Gene Ontology (GO) [ABB+00], SNOMED [SC97] and GALEN [RNG93]) do not use universal restrictions but rather use existential restrictions. It has been also noted that these ontologies do not use disjunctions (which imply non-determinism). Therefore, the $\mathcal{EL}$ family of languages [Bra04] has been introduced. This lightweight family of DLs disallow the use of universal restrictions and disjunctions (and thus negation); while allowing conjunctions and existential restrictions only. This family of DLs, including $\mathcal{EL}^+$ [BLS06], are characterised by the polynomial complexity of standard reasoning problems.

Each DL language is named such that the name itself roughly describes the capacity of the language (i.e., permitted constructors and axiom types). The use of one or more of the following letters indicates this:

- $\mathcal{S}$: is an abbreviation for the DL language $\mathcal{ALC}^+$ [Sat96, HST00a] (i.e., $\mathcal{ALC}$ extended with transitive roles; see below for an example of transitive roles).

- $\mathcal{U}$: indicates that concept unions (i.e., disjunctions) are allowed in a DL.

- $\mathcal{E}$: indicates that full existential restrictions (i.e., existential restrictions, such as $\exists r.C$, with fillers C other than $\top$) are allowed in a DL.

- $\mathcal{C}$: indicates that full complements (i.e., complex concept negations) are allowed. Note that a DL which allows concept complements and intersections is semantically equivalent to a DL that allows concept unions (e.g., $\neg(\neg C \sqcap \neg D) \equiv C \sqcup D$, by De Morgan's Law). Similarly, a DL which allows concept complements and universal restrictions is semantically equivalent to a DL that allows existential restrictions (e.g., $\neg\forall R.\neg C \equiv \exists R.C$). For this reason, the DL which allows full complements, disjunctions, conjunctions, universal restrictions and full existential restrictions is named as $\mathcal{ALC}$ rather than $\mathcal{ALUE}$.

- $\mathcal{N}$: indicates the availability of number restrictions (i.e., at least and at most constructors). For example, the concept $\geq 5 hasFriend$ describes the class of individuals who have at least 5 friends. Similarly, the concept $\leq 2 hasFriend$ describes those individuals who have at most 2 friends. Note that nothing is being said about the properties of those friends.

- $\mathcal{Q}$: indicates that qualified number restrictions are allowed. These restrictions are an extended version of number restrictions which further allow us to specify the type of the role filler. For example, $\leq 2 hasFriend.Doctor$

specifies those individuals that have at most 2 friends who are also doctors. Note that the concept $\geq 1 hasFriend.Doctor$ is semantically equivalent to the concept $\exists hasFriend.Doctor$.

- $\mathcal{H}$: indicates that role hierarchies are allowed. These are referred to as subproperties in the context of OWL (denoted rdfs:subPropertyOf). For example, $hasHusband \sqsubseteq hasSpouse$ indicates that the property $hasHusband$ is a subproperty of $hasSpouse$. A logical consequence of this can be explained as follows: if $hasHusband(Eve, Adam)$ has been asserted, then $hasSpouse(Eve, Adam)$ can also be inferred.

- $\mathcal{I}$: indicates that inverse roles (e.g., $hasHusband^{-} \equiv hasWife$) are allowed. This allows us to use one role in both directions. For example, $\exists hasHusband.Doctor$ refers to individuals who have doctor husbands, while $\exists hasHusband^{-}.Doctor$ refers to those individuals who are husbands of doctors.

- $\mathcal{R}$: indicates the availability of limited complex role inclusion axioms, reflexive and irreflexive roles, and role disjointness (e.g., as in the DL $\mathcal{SROIQ}$ [HKS06]). For example, the following role inclusion axiom $hasParent \circ hasRelative \sqsubseteq hasRelative$ indicates that a relative of a parent is also a relative. Reflexive roles allow us to specify that a role must relate an individual to itself (e.g., $similarTo$ as an individual is similar to itself but it can also be similar to other individuals). On the contrary, an irreflexive role must not relate an individual to itself (e.g., $hasSibling$ as one cannot be a sibling of oneself). Examples of disjoint roles include $hasHusband$ and $hasWife$ as one cannot be a husband and a wife of another individual at the same time. Examples of the uses of such constructors in the context of OWL can be found in [Hor11b].

- $\mathcal{F}$: indicates that functional roles are allowed. For example, to indicate that a role (e.g., hasHusband) is functional, we can use the following axiom: $\top \sqsubseteq \leq 1 hasHusband$ which means that an individual can be in a $hasHusband$ relation to one individual only. If an ABox contains the two assertions $hasHusband(Nancy, David)$ and $hasHusband(Nancy, John)$, then it can be inferred that $David$ and $John$ are the same individual.

- $\mathcal{O}$: indicates the availability of nominals which allow us to construct concepts out of individuals. For example, $\{SaintMarysHospital\}$ is a concept which has only one instance, namely $SaintMarysHospital$.
- ($\mathcal{D}$): indicates that datatypes are allowed (e.g., boolean, integer, float, date, time).
- $+$: indicates the availability of transitive roles [Baa91] (e.g., $hasFriend \circ hasFriend \sqsubseteq hasFriend$ which means that a friend of a friend is also a friend).

For example, the DL $\mathcal{SIN}$ [HST98] extends $\mathcal{ALC}^+$ with inverse roles and number restrictions. The DL $\mathcal{SHIF}$ [HST00a, Tob01, HPS03] extends $\mathcal{ALC}^+$ with role hierarchies, inverse roles and functional roles. The DL $\mathcal{SHIQ}$ [HST00b, HS04] extends $\mathcal{ALC}^+$ with role hierarchies, inverse roles and qualified number restrictions. The DL $\mathcal{SHOIN}$ (which underpins OWL) [HST99] extends $\mathcal{ALC}^+$ with role hierarchies, nominals, inverse roles and number restrictions. The DL $\mathcal{SROIQ}$ (which underpins OWL2) [HKS06] extends $\mathcal{ALC}^+$ with limited complex role inclusions, reflexive and irreflexive roles, role disjointness, antisymmetric roles, negated role assertions, nominals, inverse roles and qualified number restrictions.

2.1.3 Reasoning problems

2.1.3.1 Standard reasoning problems

A **reasoner** is a piece of software which, given a set of asserted axioms, is able to provide a yes or no answer to the following questions, which are considered standard reasoning problems:

- **Entailment**: Is an axiom α entailed by an ontology $\mathcal{O}$, denoted $\mathcal{O} \models \alpha$? By definition, the answer is no if there exists a model $\mathcal{I}$ of $\mathcal{O}$ where $\mathcal{I} \not\models \alpha$ (this case is referred to as a **non-entailment**). Given a DL $\mathcal{L}$, the **deductive closure**, denoted $\mathcal{O}^*_{\mathcal{L}}$, is the set of all axioms in $\mathcal{L}$ that are entailed by an ontology $\mathcal{O}$.
- **Subsumption**: Is a concept C subsumed by a concept D with respect to an ontology $\mathcal{O}$, denoted $\mathcal{O} \models C \sqsubseteq D$? Note that this is a special

case of entailment checking. It should be noted also that the number of subsumers D of a concept C where $\mathcal{O} \models C \sqsubseteq D$ is infinite, as is the deductive closure (i.e., infinite). In some cases it is desired to get a finite set of subsumers for a concept C. In such cases we set a language $\mathcal{L}$ such that the set $\mathcal{L}$-subsumers of C is finite. Testing for **equivalence** (e.g., $\mathcal{O} \models C \equiv D$?) is similar to performing two subsumption tests $\mathcal{O} \models C \sqsubseteq D$ and $\mathcal{O} \models D \sqsubseteq C$. **Classification** refers to the process of computing all subsumption relationships between all atomic concepts in the signature of $\mathcal{O}$.

- **Disjointness**: Is a concept C disjoint with a concept D with respect to an ontology $\mathcal{O}$, denoted $\mathcal{O} \models C \sqcap D \sqsubseteq \bot$? Again, this is a special case of entailment checking.

- **Satisfiability**: Is a concept C satisfiable, denoted $\mathcal{O} \not\models C \sqsubseteq \bot$? For example, a concept C can be unsatisfiable if it was entailed to be subsumed by two disjoint concepts D and $\neg D$, denoted $\mathcal{O} \models C \sqsubseteq D \sqcap \neg D$. Note that subsumption checking can be reduced to satisfiability checking in DLs that allow concept intersections and negations (e.g., $C \sqsubseteq D$ iff $C \sqcap \neg D$ is unsatisfiable). Similarly, disjointness checking can be reduced to satisfiability checking by checking if $C \sqcap D$ is unsatisfiable.

- **Instantiation**: Is an individual a an instance of a concept C in an ontology $\mathcal{O}$, denoted $\mathcal{O} \models C(a)$? **Realisation** refers to the process of computing all instantiation relations for all individuals and atomic concepts in the signature of $\mathcal{O}$.

- **Coherence**: Is an ontology $\mathcal{O}$ coherent? The answer is yes if there are no unsatisfiable concept names in $\mathcal{O}$ and no (i.e., **incoherent**) if there is some unsatisfiable concept name C in $\mathcal{O}$.

- **Consistency**: Is an ontology $\mathcal{O}$ consistent? The answer is yes if there exists a model $\mathcal{I}$ of $\mathcal{O}$. The answer is no (i.e., **inconsistent**) if there is no model of $\mathcal{O}$, denoted $\mathcal{O} \models \top \sqsubseteq \bot$. It should be noted that an inconsistent ontology entails everything. Note also that satisfiability, subsumption and instantiation can be reduced to consistency checking for DLs which allow concept intersection and negation. A concept C is unsatisfiable with respect to $\mathcal{O}$ iff $\mathcal{O} \cup \{C(i)\}$ is inconsistent for $i \notin \widetilde{\mathcal{O}}$. A concept C is subsumed by a concept D with respect to $\mathcal{O}$ iff $\mathcal{O} \cup \{C \sqcap \neg D(i)\}$ is inconsistent for $i \notin \widetilde{\mathcal{O}}$. Finally, an individual $a \in \widetilde{\mathcal{O}}$ is an instance of a concept C iff $\mathcal{O} \cup \{\neg C(a)\}$ is inconsistent.

A reasoning problem is said to be decidable for a DL if there exists a decision procedure that returns a yes or no answer after a finite amount of time.

Historically, early DL reasoners, implementing the so-called structural subsumption algorithms [BS01], were efficient (i.e., polynomial) and sound but incomplete. That is, they cannot detect all existing subsumption relations in expressive DLs. The basic idea of such algorithms is to normalise the concept descriptions and then structurally compare the normalised concept descriptions [Neb90, BBL05, Sun11]. Later, Tableaux algorithms [BS01] (along with practically efficient optimisation techniques) for key reasoning services were developed. The first tableaux algorithm was proposed for the DL $\mathcal{ALC}$ [HNS90, SS91] and later generalised for other DLs [DGL96, Hor97, HST00a]. Such algorithms try to construct a model of a concept C for which satisfiability checking is required. The concept is first transformed into its negation normal form (NNF). The constructed model is represented as a tree in which nodes represent individuals and edges represent role successorships. A node $Node_x$ for individual x is labeled with the concepts that must be satisfied by the individual x. The algorithm recursively applies the so-called expansion rules and terminates when it finds either a model of C (i.e., no more rules can be applied) or a clash (i.e., C cannot be satisfiable). The expansion rules try to decompose the concepts in the node labels and expand the tree accordingly. The number of rules depends on the number of constructors allowed by a particular DL (i.e., each rule is usually associated with one constructor). A clash is found when the algorithm adds two contradictory concepts to the labels of a node (e.g., both D and $\neg D$). Other Tableaux algorithms also exist and the ones that have been developed are sound, complete and terminating [HST00a].

2.1.3.2 Non-standard reasoning services

In addition to the above reasoning services which are usually referred to as “standard”, the related literature also introduces some non-standard reasoning services [BKM99, BTK03, SC03]. We explore some of these services which have been referred to in this thesis.

- The **Least Common Subsumer** (LCS) of some concepts $C_1, \ldots, C_n$ w.r.t. an ontology $\mathcal{O}$ is the "least" concept description D expressible in a considered DL that subsumes the concepts $C_1, \ldots, C_n$. D must be a subsumer of those concepts (i.e., $\mathcal{O} \models C_i \sqsubseteq D$ for $i = 1, ..., n$). And to ensure that D is the "least" subsumer, another condition is added as follows: if there exists a concept description E that is expressible in the considered DL s.t. $\mathcal{O} \models C_i \sqsubseteq E$ for $i = 1, ..., n$ then E must be a subsumer of D (i.e., $D \sqsubseteq E$). Finding the least common subsumer is considered a well understood problem for which various algorithms have been proposed; see for example [BK98, BKM99, Baa03, BST04, TZ13, ZT13].

- The **Most Specific Concept** (MSC) of some individual a w.r.t. an ontology $\mathcal{O}$ is the "least" concept description C expressible in a considered DL that has this individual as an instance (i.e., $\mathcal{O} \models C(a)$ and if there exists a concept description D that is expressible in the considered DL s.t. $\mathcal{O} \models D(a)$ then $C \sqsubseteq D$) [BK98, Baa03]. Finding the most specific concept along with the least common subsumer is said to be useful in bottom-up development of ontologies. For example, before adding a new concept, an ontology developer can start by suggesting example individuals for that concept. To help the developer to find a suitable position for the new concept, the most specific concepts for each individual are first computed for the developer. Then, the least common subsumer of all the most specific concepts is also computed [Baa03].

- **Justifications** are associated with entailments. In particular, a set of axioms $\mathcal{J}$ is said to be a justification for an entailment $\mathcal{O} \models \alpha$ if $\mathcal{J} \subseteq \mathcal{O}$, $\mathcal{J} \models \alpha$ and $\mathcal{J}$ is minimal (i.e., there is no $\mathcal{J}' \subset \mathcal{J}$ s.t. $\mathcal{J}' \models \alpha$) [SC03, BPS07, Hor11a, Bai13]. Justifications can help us in understanding why a particular entailment holds and are considered when an unwanted entailment needs to be repaired. However, it must be noted that more than one justification can be extracted for each entailment.

2.1.4 Ontology modularisation

A module $\mathcal{M}$ for an ontology $\mathcal{O}$ is a subset $\mathcal{M} \subseteq \mathcal{O}$ which contains a set of axioms "relevant" to a particular signature $\Sigma \subseteq \widetilde{\mathcal{O}}$. For example, given an ontology $\mathcal{O}$ and a seed signature $\Sigma = \{A, B\}$, one might be interested in extracting a module, say $\mathcal{O}_\Sigma \subseteq \mathcal{O}$, which preserves *all* (and *only*) entailments relevant to concepts A and B (i.e., all axioms whose signatures contain A, B). In this case, $\mathcal{O}_\Sigma$ is expected to be much smaller, hence easier to process, than $\mathcal{O}$. There are many promising applications for ontology modularisation. For instance, let us assume that we have a rather large and general ontology $\mathcal{O}_1$, from which we would like to extract a relatively small part to be imported in another ontology $\mathcal{O}_2$. The goal in this case is to import only the part of $\mathcal{O}_1$, say $\mathcal{O}'_1$, which is relevant to $\mathcal{O}_2$ aiming at getting a self-contained, yet compact, ontology $\mathcal{O}'_1 \cup \mathcal{O}_2$. Another interesting application is to use modularisation techniques to decompose large ontologies (e.g., for collaborative development, ease of human comprehension, efficient reasoning ... etc.). The rationale behind ontology decomposition is that consuming an ontology in parts is much easier than consuming it at once; both for humans and reasoners. The growing concern about the complexity of some ontologies is a typical motivation for such applications.

Ontology modularisation is an area that is well-understood and many existing modularisation techniques have already been developed [CGPSK06, CGHKS07, CGHKS08, SSZ09, KLWW13, GKW13].

2.1.4.1 Notions and properties of modules

In general, modularisation techniques are either syntactic or semantic. Syntax-based modularisation techniques are considered cheap approximations of semantics-based techniques.

A family of (syntax and semantics)-based approaches are the so-called locality-based modules [CGHKS08]. The core properties of locality-based modules are as follows:

- They preserve all entailments relevant to a particular signature.
- They can be efficiently extracted (especially syntax-locality-based modules for acyclic terminologies).
- They are not necessarily minimal (i.e., may contain some irrelevant axioms as well, e.g., tautologies).
- For a given signature, it is guaranteed that a unique and minimal module exists.

2.1.4.2 Modularisation for QG and similarity methods

With all the above being said about ontology modularisation, one can think of many advantages that can be gained from using modularisation techniques for QG purposes. For instance, rather than generating questions arbitrarily from an ontology $\mathcal{O}$, one can attempt to decompose it into different "topics" and then generate questions that are nicely categorised according to existing topics. This can be done by extracting questions from the decomposed ontology rather than the whole ontology. This can allow, for example, for an easier navigation through the generated questions. In a different scenario, we might be interested in generating questions from a specific part of $\mathcal{O}$, especially if $\mathcal{O}$ is large, and probably not all of its parts are relevant to a particular course. This can be done, safely, by generating questions from the relevant module [DVPS12]. Or let us assume that we are interested in generating questions about the core or most important concepts of an ontology. One possible option is to utilise the Atomic Decomposition (AD) [DVPSS11] which is a dependency graph representing the modular structure of an ontology. To illustrate the idea of utilising the AD to extract the core concepts of an ontology, we re-use an example presented in [DVGK+11]. The example ontology is shown in Figure 2.1 and its atomic decomposition is shown in Figure 2.2. From Figure 2.2, we can infer that the concept *Animal* is a core concept in this ontology as all the other concepts in this ontology depend on it. In the same sense, *Person* is the second important concept in this ontology.

$$
\begin{aligned}
&(a_1)\ \mathsf{Animal} \sqsubseteq (= 1\mathsf{hasGender}.\top),\\
&(a_2)\ \mathsf{Animal} \sqsubseteq (\geq 1\mathsf{hasHabitat}.\top),\\
&(a_3)\ \mathsf{Person} \sqsubseteq \mathsf{Animal},\\
&(a_4)\ \mathsf{Vegan} \equiv \mathsf{Person} \sqcap \forall \mathsf{eats}.(\mathsf{Vegetable} \sqcup \mathsf{Mushroom}),\\
&(a_5)\ \mathsf{TeaTotaller} \equiv \mathsf{Person} \sqcap \forall \mathsf{drinks}.\mathsf{NonAlcoholicThing},\\
&(a_6)\ \mathsf{Student} \sqsubseteq \mathsf{Person} \sqcap \exists \mathsf{hasHabitat}.\mathsf{University},\\
&(a_7)\ \mathsf{GraduateStudent} \equiv \mathsf{Student} \sqcap \exists \mathsf{hasDegree}.\{\mathsf{BA},\ \mathsf{BS}\}
\end{aligned}
$$

Figure 2.1: An example to show how to compute the AD of an ontology [DVGK+11]

In terms of similarity measurement, modularisation techniques can also play an interesting role. For instance, to measure the similarity of two concepts C,

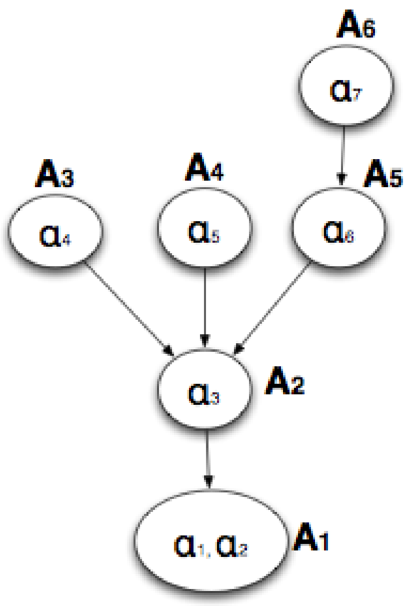


Figure 2.2: The AD of the ontology presented in Figure 2.1 [DVGK+11]

D in an ontology $\mathcal{O}$, it is sufficient to look only at the module $\mathcal{O}_{C,D} \subseteq \mathcal{O}$ which is relevant to those two concepts instead of taking the whole ontology $\mathcal{O}$ into account. The goal in this case is to improve performance, assuming that the extracted module $\mathcal{O}_{C,D}$ is much smaller than $\mathcal{O}$. As we will see in Chapter 5, in this thesis we present a new family of similarity measures in which, to measure the similarity of two concepts C, D, the new measures compute two sets $S(C)$, $S(D)$ which contain the subsumers of C, D, respectively. We know that locality-based $\bot$-modules preserve the subsumers of the concepts in the seed signature [CGHKS08]. So if $\mathcal{O} \models C \sqsubseteq E$, where E is a concept expressible in a considered DL, then extracting a module $\mathcal{O}_C$ with a seed signature $\Sigma = \{C\}$ also guarantees that $\mathcal{O}_C \models C \sqsubseteq E$. Therefore, computing the subsumers sets $S(C)$, $S(D)$ using a $\bot$-module $\mathcal{O}_{C,D}$ with a seed signature $\Sigma = \{C, D\}$ rather than using $\mathcal{O}$ is, first, sufficient (i.e., results in equivalent subsumers sets) and, second, expected to be much more computationally efficient.

2.2 Questions and students' assessment

In the following sections we introduce the core educational terms and techniques used throughout this thesis.

2.2.1 Basic concepts

In this thesis, the term **assessment** refers to educational assessment; in particular, assessing students' learning achievements. Generally speaking, educational assessment is about measuring knowledge, skills and attitudes [VH10]. It can be conducted on different levels of granularity, e.g., for individual learners, for a group of learners or for an educational institution/system. We focus on measuring learners' *knowledge*. With the increasing desire to achieve high standards in education, greater focus is put on educational assessment which is seen as a way to achieve better education [Bro94, WCG01, VH10].

Given the different purposes of assessment, we categorise assessment activities as follows:

- **Diagnostic assessment** or initial or preformative assessment aims at measuring learners' knowledge prior to teaching in order to tailor teaching activities according to learners' needs [Scr91].

- **Formative assessment** is ideally conducted throughout the teaching to provide feedback on how students are progressing in learning but it is not necessarily used to rank students. This can be done through the so-called **self-assessment**, **peer-assessment** or by teachers [Scr91]. Formative assessment is also referred to as assessment *for* learning [Ear03].

- **Summative assessment** is carried out to quantify students' achievements and translate that to pass or fail marks [Scr91]. Summative assessment is also referred to as assessment *of* learning [Ear03].

In this thesis, we use the terms assessments, **tests** and **exams** interchangeably. We consider written, rather than oral, forms of tests. This distinction is important because similarity, or the ability to distinguish between different concepts, is one of the main pillars upon which this thesis is built. This ability to distinguish between concepts can be affected by how these concepts are communicated to the examinee, whether in written or spoken form. Consider for example words of similar sounds such as *mat* and *cat* which might have low semantic similarities. In contrast, *cat* and *dinosaur* have more common features although they sound differently.

Standardised tests are usually used to provide consistent and fair results among students in large cohorts. Examples of standardised tests include, in the

Table 2.2: A sample multiple-choice analogy question [GRE]

Stem:	Cat: Mouse
Options:	(A) Lion: Tiger (B) Food: Warmth (C) Bird: Worm (D) Dog: Tail
Key:	(C) Bird: Worm

UK, the General Certificate of Secondary Education (GCSE), and in the US, the Graduate Record Examination (GRE) and the Graduate Management Admission Test (GMAT), to name only a few.

A test is made up of one or more **test items** or **questions**. These items can be categorised as objective or subjective. **Objective questions** can take different forms, for example, True/False or Multiple Choice Questions. An example of a **subjective question** is an essay. Objective questions, by their nature, are less biased than subjective questions because there is less variation in their marking. There is a pre-defined correct answer in objective questions while the correct answer in subjective questions can be expressed in different ways. In other terms, there is usually only one way to mark objective questions which means that no matter who is the marker, students should get the same result, excluding errors. However, it is also important to note that the quality of objective questions is influenced by the quality of their pre-defined answers.

A **Multiple Choice Question (MCQ)** is a tuple $\mathcal{MCQ} = \langle \mathcal{S}, \mathcal{K}, \mathcal{D} \rangle$ where $\mathcal{S}$ is a stem, $\mathcal{K}$ is a key and $\mathcal{D}$ is a set of distractors.

A **stem** is the part of a Multiple Choice Question which refers to the statement that introduces a problem to a student. A **key** refers to the correct answer(s). **Distractors** refer to a set of incorrect, yet plausible, answers.

An **analogy question** is a specific form of MCQs in which the stem and answers take the form X is to Y where X and Y are two concepts. The student is asked to locate the answer in which the underlying relation between the pair of concepts is similar to the underlying relation between the concepts in the stem. The difficulty of answering such questions can be affected by the degree of similarity between the correct answer and the stem as well as the degree of similarity between the correct answer and the distractors. An example of an analogy question is presented in Table 2.2.

2.2.2 Classification of assessment questions

To build a QG tool, we need some sort of a classification system to describe the desired questions. We summarise the classification of questions presented as part of the Question Generation campaign (2008) [GRC08]. This classification integrates some of the proposed schemes in artificial intelligence, computational linguistics, education, and cognitive science.

2.2.2.1 Purpose

Graesser et al. [GRC08] list the purposes of questioning as follows: (i) the correction of knowledge deficits (e.g., to fill a gap in knowledge such as asking "Where is the library?"), (ii) the social coordination of action (e.g., to request a permission by asking "Can I borrow your pen?"), (iii) the control of conversation and attention (e.g., rhetorical questions such as asking "Are you OK today?") and (iv) the monitoring of common ground (e.g., to assess a student by asking "Were dinosaurs carnivores?"). In this thesis, we focus on the last, i.e., the generation of questions to assess students' knowledge.

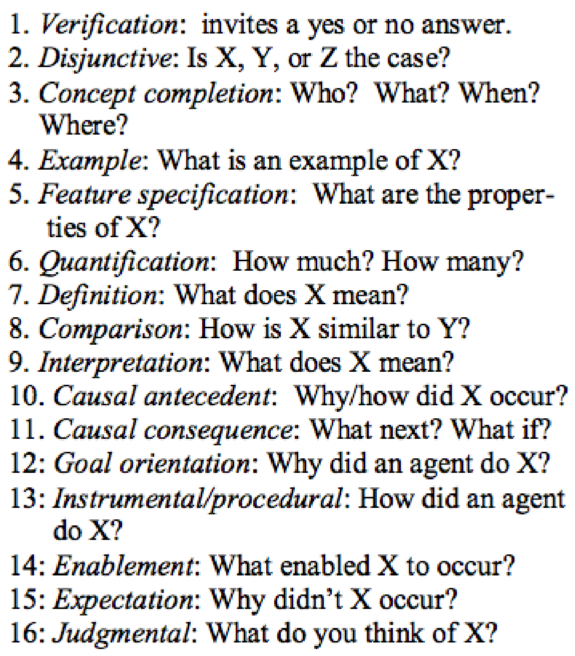
2.2.2.2 Type of information

Following on what was proposed by Lehnert [Leh77] and by Graesser & Person [GP94], Graesser et al. [GRC08] categorise questions to 16 categories that range from simple to complex questions. These categories are grounded in empirical studies carried out by Graesser & Person [GP94]. The categories are presented in Figure 2.3

In this thesis, we do not necessarily cover all the proposed categories but try to focus on some basic categories that can be easily extended to cover more categories. Details of the questions that will be focused on in this thesis will be presented in Chapter 4.

2.2.2.3 Source of answer

Does the student need to refer to a source of knowledge such as a text or a teacher to know the answer or does it come from world knowledge or common sense? In this thesis, we focus on generating questions for which the answer is in the ontology used for question generation.



1. *Verification*: invites a yes or no answer.
2. *Disjunctive*: Is X, Y, or Z the case?
3. *Concept completion*: Who? What? When? Where?
4. *Example*: What is an example of X?
5. *Feature specification*: What are the properties of X?
6. *Quantification*: How much? How many?
7. *Definition*: What does X mean?
8. *Comparison*: How is X similar to Y?
9. *Interpretation*: What does X mean?
10. *Causal antecedent*: Why/how did X occur?
11. *Causal consequence*: What next? What if?
12: *Goal orientation*: Why did an agent do X?
13: *Instrumental/procedural*: How did an agent do X?
14: *Enablement*: What enabled X to occur?
15: *Expectation*: Why didn't X occur?
16: *Judgmental*: What do you think of X?

Figure 2.3: Classification of questions [GRC08]

2.2.2.4 Length of answer

Questions differ in terms of the expected length of answer. For example, the answer can be lengthy as in essays or it can be short as in "fill in the gap" questions. In this thesis, we focus on multiple choice questions in which the answer is provided as part of the question. Mostly, the answer is limited to concept names or short concept expressions.

2.2.2.5 Cognitive process

Bloom's taxonomy of educational objectives [BK56] is commonly used to classify questions according to the cognitive process involved in answering the question. Aiken [Aik82] argues that, although it is certainly easier to write MCQs for knowledge recall, it is also possible to build MCQs that target higher order educational objectives. In this thesis, we mainly focus on knowledge recall questions but also show that it is possible to use the developed methods to generate more complex questions.

2.2.3 Evaluation of assessment questions

To evaluate assessment questions, we first need to know "what makes an exam or a question good?". In other terms, we need to know the quality metrics for evaluating assessment questions. Then we need to know the statistical methods which can be used to measure these quality metrics.

2.2.3.1 Quality metrics

Turnbull et al. [TGM98] summarise the desirable attributes for an ideal assessment tool as the following: (i) Validity, (ii) Reliability, (iii) Accountability, (iv) Flexibility, (v) Comprehensiveness, (vi) Feasibility, (vii) Timeliness and (viii) Relevance. Consistent with that, the Joint Committee on Standards for Educational Evaluation [oSfEE03] describes a good assessment as: proper, useful, feasible, and accurate. The last-mentioned includes validity and reliability.

Reliability refers to the reproducibility of test results. A common cause of low reliability of tests is the subjectivity of questions which in turn makes it possible for a given test to yield different results based on who marks the test. Reliability is usually measured in a scale from 0 to 1. A test of low reliability means that the test is totally independent of examinees' performance.

Validity refers to the ability of a test to measure what it is intended to measure. A common cause of low validity of tests is the unbalanced coverage of the topics to be assessed. Reliability is a necessary but not sufficient condition for validity. For example, a test of language proficiency is said to be reliable if it yields consistent results if administered multiple times over a short period to the same cohort of examinees; assuming that language proficiency does not change massively in short periods of time. If this test does not yield consistent results each time, then we cannot say that it is valid; simply because it does not measure what it is supposed to measure, i.e., language proficiency. An example of a reliable but not valid test for language proficiency is using examinees' heights to estimate their language proficiency. It yields consistent results over short periods of time, yet it does not estimate their language proficiency.

Accountability refers to the ability of a test to provide logical explanations for its results to all the stakeholders involved such as students, parents and educators [VH10].

Flexibility refers to how versatile a test is. A flexible test is suitable to be administered multiple times and in multiple settings.

Feasibility of a test is a fundamental feature of an assessment to facilitate its implementation. It describes how practical and cost-effective a test is.

Comprehensiveness refers to the scope of assessment. Consistent with what we have described as a main cause of invalidity of tests, this feature highlights the importance of properly covering what is meant to be assessed.

Timeliness relates to the time of delivering or administering the assessment. Ideally, there should be no delays in assessments and students should be assessed close to the time when they are expected to have achieved the learning goals [VH10]. Similarly, feedback should be given to students promptly so that they can make the most of it.

Relevance refers to the significance of a test and how important it is in achieving the objectives of the different stakeholders.

In addition to the above attributes for evaluating the quality of tests, there are some attributes for evaluating individual items in tests. These attributes include: **item difficulty**, **item discrimination**, **guessability** and **effectiveness**. Details of these attributes will be presented in the next section because their definitions depend on the specific statistical method used to derive them.

2.2.3.2 Statistical methods for the evaluation of assessments

Psychometric theory provides two alternative approaches to statistically analyse test data, namely: **Classical test theory (CTT)** [LN68, Gro82] and modern test theory or **Item response theory (IRT)** [Lor80, Bak01]. In this thesis, we are concerned only in the use of such approaches to perform item analysis in order to evaluate the quality of a group of multiple-choice items. Multiple-choice items are considered dichotomous items, which means that a student's answer to them is marked either as correct or incorrect. In contrast, polytomous items can have multiple responses with each response accounting for a different score (e.g., Likert-type items). Both theories have different models that can be used to analyse test data. A theory is a general specification of the approach. A model is a detailed specification of how to actually perform the analysis and compute values of the parameters under consideration [RR93]. A specific model is chosen based on how well it can fit the considered data. For example, some models are more suitable for dichotomous items while others are more suitable for polytomous items.

Classical test theory (CTT), sometimes referred to as True Score Theory, is a test-oriented theory which states that observed test scores (X) are equal to true scores (T) plus some measurement error (e). A person's true score in a test cannot be observed and is rarely equivalent to the observed score due to measurement errors. This is to say that measurement instruments are usually considered imperfect and not always reliable, i.e., e is seen as a random variable [Mag09].

One of the main applications of CTT is the evaluation of tests and test items. The main properties to be evaluated are [Gro82, Mag09]:

(i) Reliability of tests (denoted α),

(ii) Item difficulty (denoted p),

(iii) Item discrimination (denoted r), and

(iv) Effectiveness of distractors.

In CTT, the reliability of a test is defined as the ratio of true score variance to the observed score variance. This is equivalent to the ratio of true score variance to the sum of true score variance and error variance. The intuition here is that reliability becomes higher as error variance in test scores becomes lower. This definition of reliability cannot be used to estimate the reliability of a test in practice because true scores are unknown. Instead, Cronbach's α [Cro51] is usually used to estimate a lower bound of reliability. It is a measure of the internal consistency of a test and is calculated from the pairwise correlations between items of the test. For dichotomous items such as MCQs, Cronbach's α is mathematically equivalent to KR-20 [KR37] which can be calculated using the following formula:

$$\alpha = \frac{K}{K-1}\Big(1 - \frac{\sum_{i=1}^{K} p_i q_i}{\sigma_X^2}\Big) \tag{2.1}$$

where K is the number of items in the test, p_i is the proportion of correct responses to test item i, q_i is the proportion of incorrect responses to test item i and σ_X^2 is the variance of the observed total test scores (X).

Item difficulty (p) is indicated by frequency of correct responses, i.e., ratio of students answering the question correctly to the total number of students. By this definition, the more difficult the question i is, the lower the value of p_i. Some psychometricians prefer to use the ratio of students who got the question *wrong* to the total number of students, sometimes denoted q, i.e., $q = 1 - p$. It is also useful to identify the number of students who did not provide any answer to a certain question, sometimes referred to as error count.

Item discrimination (r) can be calculated using different methods, for example, by calculating item-total correlation. In CTT, usually the point biserial correlation coefficient is used for dichotomous items. The point biserial correlation coefficient is mathematically equivalent to Pearson's correlation coefficient. In this thesis, we use Pearson's coefficient for calculating item discrimination which is given by the following formula:

$$r = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{X_i - \bar{X}}{s_X}\right)\left(\frac{Y_i - \bar{Y}}{s_Y}\right) \tag{2.2}$$

which gives the correlation coefficient for a sample of paired data (X_i, Y_i) for $i = 1, ..., n$ where n is the sample size, $\bar{X}$ is the sample mean for X values, $\bar{Y}$ is the sample mean for Y values, s_X is the sample standard deviation for X values and s_Y is the sample standard deviation for Y values.

Another commonly used method to calculate item discrimination is to first order students according to their test scores starting from the highest score. Then, the third of the students with the highest test scores is selected and referred to as the upper group. Similarly, the third with the lowest scores is selected and referred to as the lower group. Then, item discrimination is calculated as the difference between the proportion of the upper group who answered an item correctly and the proportion of the lower group who answered the item correctly.

The effectiveness of distractors is indicated by the frequency of responses for each answer. More students from the upper group are expected to pick the correct answer compared to students from the lower group. Otherwise, the item should be reviewed. Similarly, distractors that are picked by more students in the upper group than students from the lower group are confusing distractors and should be reviewed. A distractor that is picked by only a few or none of the students is not effective as well and should be removed.

One of the main shortcomings of CTT, as with other statistical approaches, is that its results are sample-dependent. This means that item difficulty and item discrimination can differ from sample to sample for the same test. To overcome this limitation, larger sample sizes are recommended. For example, Hambleton

& Jones [RR93] suggests to use samples of sizes between 200 and 500. They also point out that the results obtained from CTT are more useful when the sample is similar to the population for which the test is developed [RR93]. The sample-dependence of CTT can be resolved with the careful design of experiments which must be backed up with high awareness to draw plausible conclusions. The use of two or more samples and the use of redundant *subsets* of questions, sometimes called anchor items, over the different samples can help to resolve the problem. This has been suggested by Hambleton & Jones [RR93] and Magno [Mag09]; and empirically undertaken by some researchers such as Mitkov et al. [MAHK06].

Item response theory (IRT) [Lor80], also previously referred to as latent trait theory, strong true score theory, or modern mental test theory, is an item-oriented theory which specifies the relation between examinees' performance on test items and the ability which is measured by those items. IRT is claimed to be an improvement over CTT, but IRT models are technically more complex than CTT models and they are not always available in common statistical softwares. There are many models under the general IRT framework. Each model is suitable for certain data. We consider only those models that are unidimensional (i.e., require a single trait or ability) and that are suitable for dichotomous items. The most well-known models of IRT are: one-, two- and three-parameter logistic models. The three parameters considered in three-parameter logistic (3PL) model are listed below. This model takes into account the possibility of getting an item right without actually knowing the answer (i.e., guessing). For example, for 4-option MCQ items, the probability of guessing is 0.25, given that all distractors are effective. The guessability parameter is omitted in the two-parameter model. This model is suitable when there is no chance of guessing, e.g., when the item requires a free response rather than picking an option out of pre-defined options. Alternatively, guessing is assumed to add randomly distributed noise to the data. The one-parameter model further omits the item discrimination parameter and assumes that all items are equivalent in terms of discrimination.

Three-parameter IRT models consists of the following parameters:

(i) Item difficulty (denoted b)

(ii) Item discrimination (denoted a)

(iii) Guessability or pseudo-guessing (denoted c)

The central concept in IRT is the Item Response Function (IRF) which specifies the probability of getting the item right by a student of certain ability θ, as specified in the following formula [RR93].

$$P_i(\theta) = c_i + \frac{1 - c_i}{1 + e^{-a_i(\theta - b_i)}} \tag{2.3}$$

The three parameters a, b, and c are estimated by first assuming that abilities are modelled as a sample from a normal distribution. IRFs are graphically represented by Item Characteristic Curves (ICC) as shown in Figure 2.4, for example. As the figure shows, c is the height of the lower asymptotic for the curve which corresponds to the probability of a correct response by the lowest ability students. Based on observed test scores of the sample, IRT models estimate the probability of guessing c_i for an item i. If i is a 4-option multiple-choice question and the estimated c_i is higher than 0.25 then some of the options are most probably not functioning well. Parameter b corresponds to the point on the ability axis where the probability of a correct answer equals $(1 + c)/2$ and also where the slope is maximised. Parameter a is proportional to the slope of the curve at the point b on the ability axis.

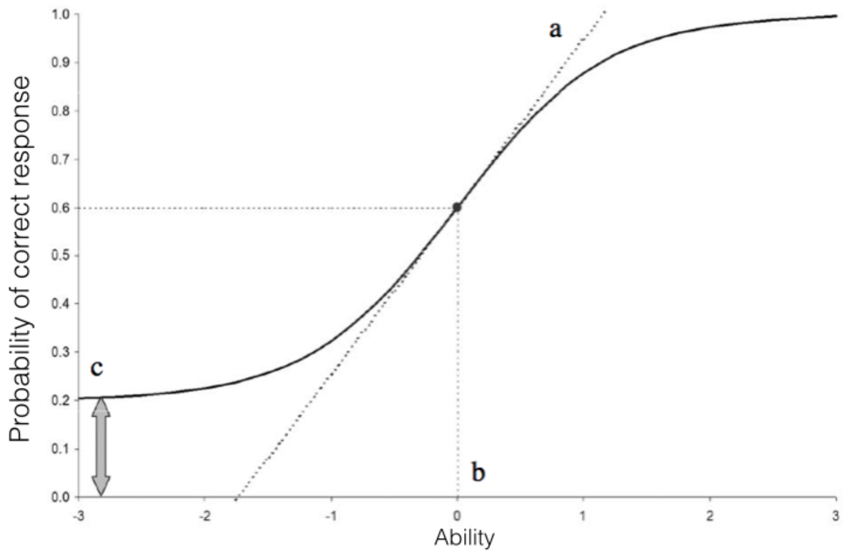


Figure 2.4: 3PL Item Characteristic Curve

Item information functions (IIF) and Test information functions (TIF) are alternative ways to assess the reliability of a test, i.e., the frequency of errors in measurement. Plots of IIFs are bell-shaped and are centered around their b value on the ability axis. IIFs shows how much information an item contributes to the assessment of ability [RR93]. Items with high discrimination power tend to have tall and narrow IIFs which indicates that their best contribution is over a narrow range of abilities. In contrast, items with low discrimination power contribute less information but over a wider range. For example, Figure 2.5 shows that item 1 is less difficult than item 2 and that items 3 and 4 are less discriminating than items 1 and 2.

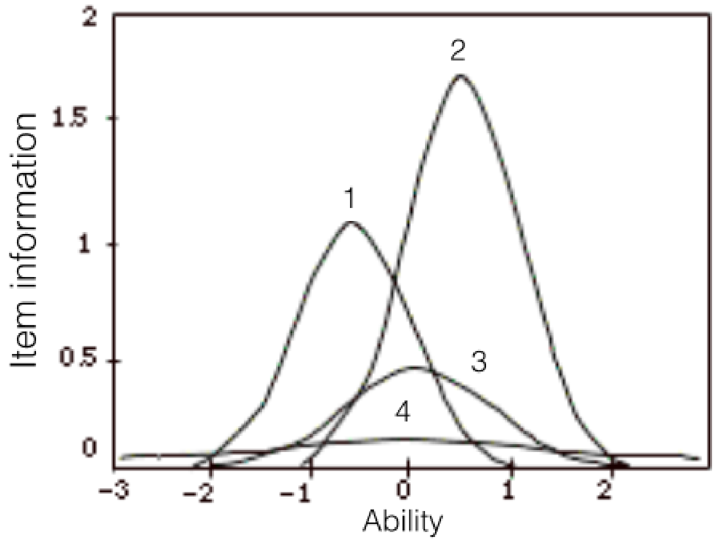


Figure 2.5: Item information functions for four test items

For a particular item i, the IIF of an item i at a specific ability θ, denoted $I_i(\theta)$, can be given by the following equation [dA09]:

$$I_i(\theta) = a_i^2 \frac{(p_i(\theta) - c_i)^2 q_i(\theta)}{(1 - c_i)^2 p_i(\theta)} \tag{2.4}$$

where a_i is the estimated item discrimination for item i, c_i is the estimated guessability for item i, $p_i(\theta)$ is the probability of getting item i right at ability θ and $q_i(\theta)$ is the probability of getting item i wrong at ability θ.

TIF is the sum of IIFs of all items in a test and it indicates the frequency of errors associated with the assessment of ability. The more information provided by TIF, the lower the frequency of errors at a particular ability level. This can be shown by the following formula which calculates the standard error of estimation, denoted $SE(\theta)$, at a particular ability θ from a test information function, denoted $I(\theta)$:

$$SE(\theta) = \frac{1}{[I(\theta)]^{\frac{1}{2}}} \tag{2.5}$$

Theoretically, IRT models are sample-independent as opposed to the sample-dependent CTT models. However, Hambleton & Jones [RR93] point out that IRT parameters are considered sample-independent only if the model fits the test data and point out that large samples are usually required, e.g., over 500. The fit of the data for the model can be assessed using Chi-square statistic.

To sum up, both classical and modern item analysis methods give some indications for the quality of an item. They should be used as an exploratory approach to identify possible defects in measurements. But they should be carefully used to provide generalisable information about the characteristics of test items, i.e., as a confirmatory approach.

In this thesis, due to experimental design limitations, we adopt CTT methods to calculate item difficulty (i.e., frequency of correct responses), item discrimination (i.e., Pearson's correlation coefficient) and effectiveness of distractors (i.e., frequency of responses for each answer).

Chapter 3

Foundations of MCQ generation

This chapter sets the theoretical foundations upon which question generation methods developed in this thesis are built. Previous attempts to develop methods for generating questions from ontologies [CNB03, HMMP05, HMMP06, ZSRG08, PKK08, CT09, CT10, ZPK11, AY11, AY14] or other sources [Ste91, Fai99, MH03, BFE05, HN05a, LWGH05, SSY05, MAHK06, Hei11, CCSO14] have not focussed on addressing the problem of controlling the difficulty of the generated questions.[1] As we have seen in the previous chapter, difficulty is indeed a core property of assessment questions. It is central for the *validity* of an assessment, which refers to how successful an assessment tool is in measuring what it intends to measure. Typically, in normal class settings, test developers aim at measuring mastery levels of students in a certain domain. A valid assessment tool must be able to measure the mastery level of all students, i.e., both high and low mastery students. If the assessment tool is too difficult, then it is not suitable for measuring mastery levels of the lower group. Similarly, if it is too easy, both high and low mastery students will get high grades and therefore we cannot distinguish between them. In other assessment settings, test developers are interested in those students who can pass the test at a certain level, e.g., 60%. In such a case, the test must be designed at a difficulty level suitable for the required level. Again, to construct a test that is *valid* for this purpose, a mechanism to control the difficulty of test items, and hence the overall test, is required (though, of course, is not sufficient).

Our main criticism for existing QG methods is that they are mainly technical and lack theoretical backing. This has resulted in developing ad-hoc generation methods with limited control over the quality and difficulty of the generated

[1] although some have tackled the problem as we will see in detail in Chapter 4.

questions. This chapter aims at presenting theoretical foundations of QG. In particular, the chapter examines the plausibility of the conjecture that controlling the difficulty of multiple choice questions can be done using a similarity-based approach. We explain the impact of controlling the difficulty of a question on its overall quality and describe the role of similarity in the difficulty control process. We explore psychological theories and empirical evaluations that support this conjecture.

The main purpose of this thesis is to prove the possibility of generating *useful* questions from ontologies. Generating a reasonable number of questions (rather than all possible questions) is sufficient for this purpose. Usefulness is defined in this thesis within two contexts: (i) useful for assessing students' performance and (ii) useful for ontology development/comprehension. The first context is the main focus of the thesis and the experiments involved whereas the second context serves the purpose of showing that the methods can be applied within other contexts.

Some existing works [MH03, MAHK06] on question generation describe a successful generation method as a method in which (i) the overall time spent on generation is reasonable (e.g., less than or equal to the time spent on generating the questions manually) and (ii) questions have functional distractors and good item discrimination. In this thesis, we extend (ii) and consider a broader description for a successful generation method in which the method is able to control the difficulty of the question as well. This will indeed affect the generation methods we develop and the experiments we carry out to evaluate them.

3.1 Desired properties of assessment questions

Students' assessment is concerned with measuring students' mastery, usually in terms of amount of *knowledge* and *skills* they have. We focus on *knowledge* assessment. Merrill [Mer94] classifies knowledge in four categories: facts, concepts, principles and procedures. We focus on assessing facts and concepts as they are more suitable to be modelled in an OWL ontology. Assessment of knowledge can be carried out along different cognitive complexity levels. Bloom's taxonomy [BK56] is a categorisation of the cognitive domain which is widely accepted and used by educational researchers and practitioners [Sed78]. The main categories of this taxonomy are (bottom-up): (1) Knowledge, (2) Comprehension, (3) Application, (4) Analysis, (5) Synthesis and (6) Evaluation. The lowest category refers to demonstrating a student's ability to *recall* knowledge. Higher categories require demonstrating higher abilities such as applying knowledge to new problems. A revised version of the taxonomy which was proposed in 2000 by Anderson et al. [AK00] is presented in Figure 3.1. Depending on the purpose of the assessment, it might be necessary to target both the lower and higher levels of the taxonomy.

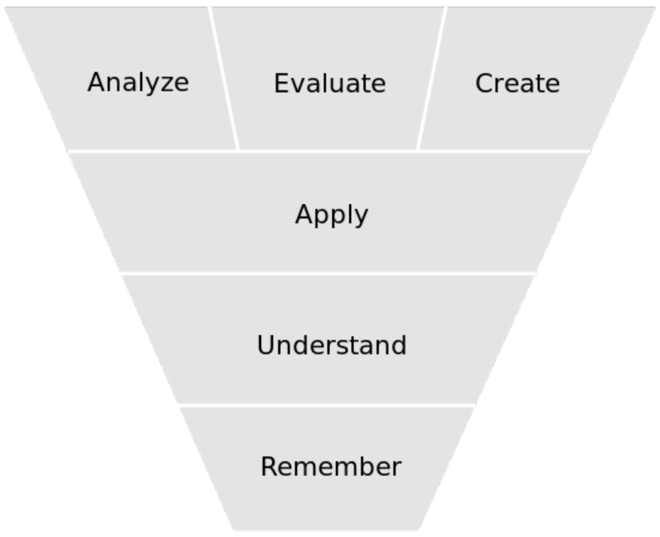


Figure 3.1: Categories of the revised Bloom's taxonomy, taken from [AK00]

Measuring students' knowledge is not straightforward as measuring their height or weight; knowledge models constructed in a student's mind are complex and not visible. In addition to the amount of knowledge, a distinction is usually made (in terms of quality of knowledge) between knowledge of high and low mastery students [WCG01]. Knowledge of high mastery students is well-organised which enables them to notice patterns of information that might be neglected by low mastery students [CGR82, CK83].

A good assessment question (backed with good interpretation of its results) is supposed to make explicit those implicit knowledge schemas in a student's mind. A desired property of an assessment (i.e., a collection of questions) is its ability to distinguish between high and low mastery students. In particular, there must be a strong correlation between students' mastery and the amount and quality of their knowledge. This property cannot be achieved using a single multiple-choice question. Instead, we need a collection of questions that work well individually and that (together) discriminate well among different levels of understanding [WCG01]. As we explain in the previous chapters, a key aspect of constructing a *good* multiple-choice question is the use of good distractors that 1) appear as plausible answers to a student who does not know the correct answer and 2) are clearly recognisable as wrong by a student who knows the correct answer.

To illustrate the desired properties of distractors (which make their generation challenging), we present the following example.

3.1.1 Distractor selection example

Consider the ontology in Figure 3.2 which describes some diseases and associated body parts.

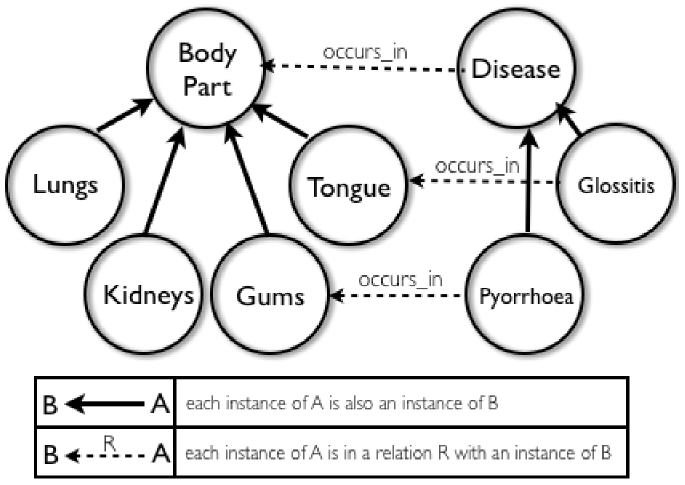

Body Part
occurs_in
Disease
Lungs
Tongue
occurs_in
Glossitis
Kidneys
Gums
occurs_in
Pyorrhoea
B A each instance of A is also an instance of B
B R A each instance of A is in a relation R with an instance of B

Figure 3.2: Example ontology

Given the example ontology, one can construct the multiple-choice question in Table 3.1. Note that, in this example, some distractors could have been replaced by better ones. For example, Disease, which can be easily eliminated even by a low mastery student by knowing that disease is not a body part and therefore is not an appropriate answer. This naive elimination process increases the chance of guessing the correct answer without actually having the required knowledge that is being assessed. Glossitis can be considered a better distractor compared to Disease because it requires deeper knowledge to recognise that Glossitis is a disease which is again a reason for elimination.

Table 3.1: Example question

Stem:	Pyorrhoea occurs in ...:
Options:	(A) the tongue (B) glossitis (C) the gums (D) a disease
Key:	(C)
Distractors:	(A), (B), (D)

As explained earlier, difficulty is a core property that must be carefully controlled during assessment design. It is suggested that only a few questions on any exam should be answered correctly by more than 90% or less than 60% of students [Low95]. This notion of difficulty is referred to as statistical difficulty. However, beyond knowing that 80% of students were unable to solve a question, we would like to know why this is the case. Several studies have explored sources of difficulty in questions [FHH96, FHHB94]. A distinction must be made between valid and invalid sources of difficulty. Examples of invalid sources of *high* difficulty include providing extra information. Similarly, examples of invalid sources of *low* difficulty include triviality, providing clues. Providing clues might lead to *guessing* the correct answer while providing extra information not needed for solving the question might mislead a high mastery student and hence lead to poor *discrimination*. Thus, avoiding invalid sources of difficulty allows us to eliminate guessability and get better discrimination. This is to say, controlling difficulty (in a valid way) leads to controlling the other two properties: guessability and discrimination. In addition, an assessment designer must be able to set the difficulty of each individual question. Therefore, we focus on the difficulty property in particular when we explore plausible question generation methods. One statement that can summarise the importance of the difficulty property is: "If you do not know why this question is harder than that one, then you do not know what you are measuring" [FHH96]. This statement refers to the impact of valid difficulty on the validity of the test.

We present a similarity-based theory that can be used to inform the generation of questions and control their difficulty. We validate it by showing its consistency with and accountability for both the psychological and educational theories.

3.2 A similarity-based theory of controlling difficulty of MCQs

Bearing in mind that our goal is to be able to control the difficulty of the generated questions, a key assessment design requirement is to understand what makes a question difficult for a group of students. We conjecture that similarity between parts of a question influences difficulty. In particular, the distinction between high and low mastery students can be mapped to their ability to distinguish between similar answers and identify any commonalities and differences between them. This conjecture is supported by existing studies on expertise and theories about knowledge retrieval from long-term memory as we will see in the next section. The conjecture seems to be plausible because of its consistency with the fact that the more knowledge students have about a topic, the more ability they have to answer questions about that topic. This relation between the amount and depth of knowledge and question answering ability can be explained in terms of the ability to account for similarity when answering certain kinds of questions.

As an example, refer back to the question in Table 3.1: "Pyorrhoea occurs in ...?" where Gums is the correct answer. Now let us consider the different possible distractors that can be used with this question. If the student only remembers that Pyorrhoea occurs in the mouth, then we can notice that Tongue would be more difficult for the student to eliminate compared to Lungs; both Gums (the key) and Tongue (the distractor) are in the mouth and hence share more commonalities (i.e., they are more similar to each other than to Lungs). To be able to answer the question correctly, the student must have knowledge about the particular parts of the mouth and the name of the disease that occurs in each of these parts; that is the particular difference between the key and distractor. Similarly, Lungs is more difficult to eliminate compared to Glossitis; both Gums and Lungs are body parts and hence are more similar to each other than to Glossitis.

As illustrated by the example, to compute the similarity between two concepts, we account for both the common and distinguishing features. Based on this notion of similarity, we will later define similarity functions such as $sim : C \times C \rightarrow [0, 1]$ where C is the set of concepts to be compared and $sim(X, Y) > sim(Y, Z)$ if the two concepts X, Y are more similar than the concepts Y, Z.

As explained above, one would expect that students would succeed in answering a question if they know the similarities and differences between the answers provided with the question. Similarly, they would fail if they do not know the similarities or differences. Based on this observation, we present the following hypothesis for controlling the difficulty of MCQs. The presented hypothesis is not suitable for controlling the difficulty of all classes of MCQs. It is suitable for controlling the difficulty of MCQs with parts (e.g., key, distractors) of similar kinds which require particular knowledge to distinguish them. Examples of questions belonging to this class of MCQs include: "What is X?", "Which of the following is X?", "Which of the following is the odd one out?". The hypothesis might need to be slightly altered to suit other types of MCQs (e.g.,"A is to B as .. is to ..?") where a condition can be added to control the similarity between the stem and the key. Variations of questions and similarity patterns that can be suitable for them are presented in Table 3.2 below. Examples of MCQs that are not suitable for our similarity theory are presented in Section 3.4.

Hypothesis 1

The difficulty of (some classes of) MCQs consisting of a stem S, a key K and distractors $D = \{D_1, \ldots, D_n\}$ is (with other things being equal) proportional to the degree of similarity between K and D_i where $i = 1, \ldots, n$.

Given that an MCQ Q can have more than one distractor (i.e., $n \geq 1$), each distractor $D_i \in D$ can have a different similarity $sim(K, D_i)$ to the key K. Let us define the following two parameters (Δ_1, Δ_2):

1. $\Delta_1 \coloneqq min\{sim(D_i, K) \mid 1 \leq i \leq n\}, 0 < \Delta_1 < 1$

2. $\Delta_2 \coloneqq max\{sim(D_i, K) \mid 1 \leq i \leq n\}, 0 < \Delta_2 < 1$

To construct a good MCQ, Δ_1 must be sufficiently greater than 0 (and less than 1). Decreasing Δ_1 increases the likelihood of having non-functional distractors or distractors that can negatively affect discrimination between good and poor students. Increasing Δ_1 increases the difficulty of Q, in general, and increasing it to reach a value close to Δ_2 decreases the likelihood of having non-functional distractors. Similarly, increasing Δ_2 increases the difficulty of Q.

3.3 Supporting the similarity theory

The plausibility of the above similarity-based theory for controlling difficulty must be assessed according to both psychological and educational considerations. In general, sources of difficulty are explained in terms of two aspects:[2] (i) the underlying knowledge and (ii) the cognitive ability required to solve the question [WCG01].

3.3.1 Psychological theories

In the following subsections, we examine two categories of psychological theories: (a) theories of semantic memory and (b) theories of expertise. These theories explain how knowledge is learned, i.e., *stored* in memory, and how knowledge is used, i.e., *retrieved* from memory. In general, knowledge is retrieved from, rather than stored in, memory during an assessment. In contrast, knowledge is expected to be learned before (and after) an assessment.

To fully understand the influence of these theories on controlling the difficulty of assessments, we structure our discussion of each theory around the following questions: (i) what are the main properties of the theory?, (ii) what are the main controversies around the theory? and (iii) how does the theory support our similarity conjecture and/or (iv) does the theory hint at factors, other than similarity, that can affect the difficulty of assessments?

3.3.1.1 Theories of semantic memory

Semantic memory [SC02] refers to the part of the brain that is responsible for the acquisition, representation, and processing of *shared conceptual knowledge* (e.g., concepts, objects, states and events). It is often contrasted with episodic memory [Tul02] which enables human beings to store and recall *unique personal experiences* (e.g., losing someone you care about). This distinction between the two forms of memory was first established by Endel Tulving [Kle13]. Various models of semantic memory have been proposed. Tartter [Tar98] lists some of the most influential models of semantic memory such as network models, prototype models and features models. We briefly present these models below.

[2]In addition to knowledge and cognitive abilities, a skill might be required to answer a question. We assume that the students are equally equipped with the required skills for simplicity.

Network models are historically one of the early models of semantic memory which have had and still wield considerable influence [Tar98]. They describe semantic memory in terms of a rich organisation where all concepts are related through an associative net. The nodes of a semantic network can be either concepts (e.g., "birds") or properties (e.g., "can fly") and both kinds of nodes are treated equally and can be accessed in a similar way. The links between nodes represent different relations between them. For example, a link between the nodes "birds" and "animals" captures the fact that birds are kinds of animals. Similarly, a link between "birds" and "has wings" associates wings to the class birds. Moreover, those links can vary in their strength; the higher the weight of the link between two nodes, the stronger the relation between them. This enables the model to account for the familiarity effect (i.e., more familiar concepts can be accessed faster). Within network models, some theories [RM04, SF08] assume that knowledge upon which people make inferences is associative. Others [HR94, KT09] assume that knowledge is structured. In contrast, others [BF14] have reported that both associative and structured knowledge have an impact on the inferences we make and suggested that the different kinds of knowledge apply best under different processing conditions (e.g., required response time).

Spreading activation theory is an attempt to explain how knowledge is processed within a semantic network. It was originally developed by Collins & Quillian [CQ69] and later improved by Collins & Loftus [CL75]. The theory views memory search as activation spreading from concept nodes in a semantic network which stops when an intersection is found. Given a certain concept, the theory explains what will be the next concept that a person will automatically think of. This unconscious process is referred to as "associative priming" in the original theory [And95]. A core notion of this theory is that knowledge is organised in memory along the lines of semantic similarity. The more links there are between two concepts (i.e., the more commonalities they have), the more closely related the concepts are. This means that people are more likely to retrieve information from memory if related information has been presented to them first [And95]. Moreover, the stronger the relation between the two pieces of information, the more likely the chance to access one through the other. The theory can be used to explain a person's behaviour when confronted with a question like "Is bird an animal?". To answer with either "True" or "False", the person tries to access one node through the other. If a link is found, the returned answer will be "True".

There have been a number of experiments investigating spreading activation theory [And83b, Bal83, BL86, Lof74, SS97]. For example, Sharifian & Samani [SS97] carried out an experiment to compare subjects' response times when asked to verify the relation between different word pairs. The results show that there is a significant difference between the time required to verify direct and indirect relations. For instance, the response time required to verify the relation "rose is a plant" is significantly greater than the response time to verify either "flower is a plant" or "rose is a flower". A study conducted by Balota & Lorch [BL86] supported the fact that memory activation can spread between directly and indirectly related concepts. For example, the word "lion" can activate the word "stripes" although the two words are related only through a mediating concept "tiger". Other studies have shown empirical evidence that the spread of activation is done automatically as opposed to being under control [Bal83].

One of the controversies around semantic network models is the notion of cognitive economy. It refers to the assumption that information about a certain concept is stored only once in the appropriate level. For example, the fact that birds have skin is not stored at the bird level, rather, it is stored at the animal level. Some experiments carried out by Collins and Quillian [CQ69] support the cognitive economy property of semantic networks. For example, they report that subjects' response time to a question such as "does a bird have feathers?" is less than the response time to a question such as "does a bird have skin?"; where feathers is stored at the bird level and skin at the animal level. The controversy here is that cognitive economy can be confound with frequency of co-occurrence. For example, let us compare the property of having skin for both birds and snakes. Although it might not be frequent to talk about birds' skin, it is more frequent to talk about snakes' skin. This implies that there is no direct storage of skin for birds and, in contrast, there is a direct storage of skin for snakes (ignoring the cognitive economy property of the network).

Network models in general and spreading activation in particular can provide a logical explanation of why our similarity conjecture can be psychologically plausible. The theory explains that once a concept has been activated in memory, the most similar concepts to that concept would be activated immediately. This means that those similar concepts can act as distracting factors, i.e., functional distractors in MCQ terminology. This can explain why it is harder to verify the key answer when the distractors are very similar to the key. In addition to the distractors-key relation, the theory also explains why it is easier to verify the key answer when it is closer to the stem; a very plausible factor that we did not mention in Hypothesis 1. We have chosen not to consider this as a primary factor of our difficulty-control theory to keep it easier to understand and verify, and more importantly, applicable to a wider range of questions. However, we examine the impact of varying the similarity between the stem and the key in our evaluation experiments presented in Chapter 7 (see the automated evaluation experiment).

Prototype models introduced the notion of a "basic level" which is the most prototypical category among a hierarchically ordered set of categories. The basic level is said to be more accessible and more likely to be learned first. Prototype models were first introduced by Eleanor Rosch [Ros73, Ros75]. Rosch and others have studied different hierarchies in an attempt to define the prototypical category in each hierarchy. As an example, let us consider the categories *animal*, *dog* and *poodle*. Notice first that these categories are hierarchically arranged, i.e., poodles are kinds of dogs which are kinds of animals. Notice also that each of the three categories can have different subcategories, e.g., animals can be dogs, cats, reptiles and others. In her empirical studies, Rosch [Ros75] reports that most subjects have found it easier to imagine, e.g., recall the features of, a prototypical dog than to imagine a prototypical poodle or animal. Poodles are too specific while animals are too general. Also, when the subjects were asked to give an example of an animal, it was more likely to respond "dogs" rather than responding "poodles". Similarly, it was more likely that a subject who was asked to provide an example of a furniture will respond "chair" rather than responding "stool".

Prototype models provide a minimal representation of semantic memory; compare it for example to the more rich representation provided by network models. This does not necessarily reflect a disagreement between the two models. Loftus [Lof75] provides a spreading activation-based explanation of the plausibility of Rosch's results. However, it must be noted that prototype models have serious shortcomings. The main criticism is that the model does not provide a precise definition of the prototypical category. For example, if we assume that we are interested in the hierarchy (*animal*, *mammal* and *dog*), would *dog* be still the prototypical category? Another shortcoming of the theory is that it fails to explain why the same concept can be more prototypical for a given category and less prototypical for another category (e.g., *car* is more prototypical as a *vehicle* than as a *toy*) [Lof75].

Prototype models explain why it is easier to identify the key answer when it is more commonly associated with the stem; i.e., when the key is prototypical. So it is easier to verify that a chair is a furniture compared to verifying that a stool is a furniture. The theory can be used to control the difficulty of MCQs by varying the key-stem relation, provided a good (i.e., shared and explicit) understanding of prototypicality is available. Due to the unavailability of this understanding at the moment, we do not account for prototypicality in our difficulty control theory.

Feature models have not been as influential in cognitive psychology as network and prototype models. Nevertheless, we will briefly present the model to give greater insight in cognitive process models of semantic memory. In feature models, concepts are described in terms of their features, with a distinction between characteristic and defining features [SSR74]. Characteristic features are found in most instances of a category, i.e., they are typical features (e.g., a typical bird can fly). Defining features are those features that are necessary for membership and shared by all instances of a category (e.g., all birds lay eggs and have feathers). Feature models have two main properties: (a) feature lists are not structured and (b) features are stored locally (i.e., cognitive economy is not considered). To verify whether a *canary* is a *bird*, the feature overlap is considered. If the overlap is large (the model does not specify how large), then the sentence is verified. Otherwise, the answer is “no”. Smith, Shoben and Ribs [SSR74] explain that sentence verification can take one or two steps, with the second step being much slower than the first. When a subject is asked to verify a sentence such as A is a subcategory of B, the number of steps is determined based on the amount of feature overlap between A and B. If a large number of features is recognised, a fast “yes” answer is provided. If only a few number of features overlap, then a fast “no” is provided. However, if the amount of overlap is intermediate then a second step is required. In step 2, the defining features of A and B are compared to verify the sentence. For example, verifying that *canary* is a *bird* can be faster than verifying that *canary* is an *animal* because feature overlap between canaries and birds is greater than feature overlap between canaries and animals.

As with the other models, there is a debate around feature models, especially around defining features. For example consider a plucked chicken and a feather pillow. The former does not have feathers but it still is a bird and the latter does have feathers while it is not a bird. This questions the fact that “feathers” are considered a defining feature of birds. Moreover, feature models do not explain how the semantic memory is organised, i.e., how concepts are related [Tar98].

As with semantic models, feature models can explain why it is easier to verify the key answer when it is similar to the stem. Distractors that are similar to the key share a large number of features with the key. Hence, the comparison stage, i.e., step 2 above, would be harder or at least slower.

In addition to the above three models of semantic memory, we mention a very related theory of cognition which also models semantic memory, or more generally, the mind. The theory is referred to as Adaptive Control of Thought (ACT) [And83a] and was later extended to Adaptive Control of Thought-Rational (**ACT-R**) [And93, And07]. The theory does not only provide a model of the mind, but it also models and predicts human cognitive behaviour. It represents the memory as units, referred to as "chunks", interconnected by links representing the relations between those chunks (as in network models). Each chunk has a label and a set of properties specific to that chunk. The current state of the memory is held in what is referred to as a "buffer" which contains the currently activated chunks. As in network models, activation spreads to similar chunks. In addition, activation of a chunk speeds up with frequent retrievals of that chunk.

One of the most important notions studied under the ACT-R framework is what is referred to as the "fan effect". It points out to the observation that the time taken to recognise an item becomes longer as its fan increases. The fan is defined as the number of associations that an item has with other items in memory [And74, SA12]. ACT-R in general and the fan effect in particular can be used to support our similarity conjecture. It can explain why it gets harder to recognise the correct answer when the distractors share a lot of associations with the key. Of course one can argue that the time spent by a student to answer a question does not sufficiently indicate whether the student will get the question right or wrong. Some students will quickly recognise the correct answer, and yet, some students will recognise the correct answer after given sufficient time. This of course excludes time-limited tests in which students are encouraged to answer as fast as they can and their speed will affect their total grades.

3.3.1.2 Theories of expertise

Another challenging area of cognitive science aims at explaining the superior performance of experts in various domains. Stability is an important characteristic of their superior performance which eliminates the influence of luck or unique environmental circumstances [ES91]. Here, experts refers to those individuals who can be labeled as outstanding in a certain field [ES91]. In the context of learning and assessment, we choose to refer to those individuals as high mastery students. Existing studies of expertise have investigated the way knowledge is structured and used by experts and compared that to novices' knowledge. Various studies have explored the relation between expertise and performance in different domains such as playing chess [CS73], computer programming [MRRC81], music [Slo76], basketball [AB85], to name a few. Notice that these studies explore experts' behaviour in situations that require the application of *skills*. Instead, we focus below on studies that explore experts' behaviour when recalling *knowledge*.

Experts' knowledge is usually described as more cohesive and integrated compared to novice knowledge [GC86, CGR82, CK83]. This cohesiveness of knowledge was operationally defined by Chi and Koeske [CK83] in terms of the pattern of interrelations between concepts either through direct or indirect links. For example, a semantic network derived from a child who is learning about dinosaurs would have more links between dinosaurs which are more familiar to the child compared to links between less familiar dinosaurs. Moreover, dinosaurs belonging to the same family would share more common links compared to dinosaurs belonging to a different family. This pattern of interrelations was found only in parts of the semantic network corresponding to familiar knowledge.

Chi and Koeske [CK83] also investigated the impact of the degree of cohesiveness on a child's ability to perform subsequent memory tasks. The results show that a child has more information about the more familiar dinosaurs. This supports the fact that the degree of cohesiveness of knowledge can predict performance on memory tasks [CK83]. Gobbo and Chi [GC86] further investigated the impact of expertise on the success of performing other complicated tasks such as making semantic comparisons, inferring new knowledge and reasoning about new information as it is related to existing knowledge. In their study, Gobbo and Chi support the view that success in reasoning is based largely on knowledge as opposed to the other view that reasoning is a skill that children acquire as they mature. The results of their investigations show that expert children can infer more implicit facts about dinosaurs compared to novice children. Another interesting result is that expert children could infer implicit knowledge about both known and unknown dinosaurs. This means that the inferred knowledge is not always retrieved from memory but can also be generated by an expert child based on their sophisticated knowledge.

Theories of expertise can explain why difficult MCQs, as defined in Hypothesis 1, can only be answered correctly by high mastery students (experts), a necessary property to govern the reliability and validity of an assessment. High mastery students have the ability to account for larger, and better structured, amounts of knowledge which gives them a better ability to distinguish the correct answer from other distracting answers in a fast and reliable mechanism.

To sum up, the psychological theories presented above, together with a general theory of cognitive load, can provide a psychological support for our similarity-based theory of controlling MCQs difficulty. The increased level of similarity between the answers of an MCQ increases the likelihood that the students will be distracted by the wrong answers which increases the cognitive load. This in turn can have a negative impact on their academic success. Among the presented theories, network models together with theories of expertise are more applicable to the context of this thesis.

3.4 Applicability of the similarity conjecture to different classes of MCQs

It is very important to examine whether the similarity conjecture is applicable to different classes of questions. Although assessment questions may be specified in an almost unlimited number of ways, the student behaviours involved in these assessments can be described by a relatively small number of categories, see for example [BK56]. We are interested in investigating whether the similarity conjecture is suitable for different classes of assessment questions as classified by Bloom's taxonomy [BK56].

The different categories are arranged in a hierarchical order according to the complexity of the involved cognitive process. In addition, a student performing at a specific level is assumed to make use of and built on the behaviours required to perform at the preceding level [BK56, Sed78, Smi70]. It is important to distinguish between the notion of item complexity and item difficulty. Item complexity is based upon the quantity and quality of *effort* required to answer the item whereas item difficulty is based upon the quantity and quality of *knowledge*. For example, one can say that subtraction is more complex than addition, but a question involving subtraction is not necessarily more difficult than a question involving addition. Guttman [Gut53] and others [Cra68] demonstrated that it is still possible for more complex tasks to be either more or less difficult than less complex tasks. It is also important to note that, depending on the nature of the prior learning experiences, different students can solve the same question in different ways and that one test item can actually be placed in different categories [BK56, GH13]. This suggests that, in general, item difficulty can be controlled along the different levels of Bloom's taxonomy.

It remains now to examine whether our similarity conjecture is suitable for controlling the difficulty of questions along the different levels of Bloom's taxonomy. There is, in fact, a debate around the suitability of MCQs to assess higher levels of Bloom's taxonomy [Aik82]. Although we acknowledge that MCQs can be used to construct questions on both lower and higher levels of Bloom's taxonomy (see for example [Aik82]), but we also acknowledge that not all MCQs are suitable for our similarity theory. For example, consider questions that require the student to do some calculations and pick the correct numerical value (e.g., long division questions). By picking the correct answer, the student demonstrates the correct usage of a specific mathematical method or procedure. Similarly, by picking a distractor, the student demonstrates the wrong usage of the required procedure. Each distractor should correspond to making a mistake in a specific step of the procedure but similarity might not be directly applicable for selecting the resulting numerical distractors. Other counter examples include evaluation questions in which the answers should correspond to all possible opinions (e.g., excellent, good, bad); again, it makes no sense to vary the similarity between such answers. However, there are plenty of examples for MCQs on different Bloom's levels that are suitable for our similarity theory, i.e., the similarity between the different parts of the MCQ (stem, key and distractors) can be varied in order to vary the difficulty. We provide an example for each Bloom level in Table 3.2. Rather than constructing new examples, the presented questions are mostly adopted from existing educational sources [CDM96, GRE].

3.5 Summary and directions

In this chapter, we have discussed numerous psychological models to demonstrate the psychological plausibility of the similarity-based theory for controlling the

Table 3.2: Example questions on each Bloom's levels

Knowledge:	Who is the author of "Das Kapital"?
Key: (B)	(A) Mannheim (B) Marx (C) Engels
Similarity pattern:	key-distractors (German authors)
Comprehension:	Which one of the following describes the PREPARATION stage of the creative process?
Key: (A)	(A) The problem is explored and defined (B) An attempt is made to see if the proposed solution to the problem is acceptable (C) The person goes through some experience leading to a general idea of how the problem can be solved
Similarity pattern:	key-distractors (stage descriptions)
Application:	Which one of the following memory systems does a piano-tuner use?
Key: (A)	(A) Echoic memory (B) Long-term memory (C) Mono-auditory memory
Similarity pattern:	key-distractors (memory systems)
Analysis:	Cat: Mouse AS
Key: (B)	(A) Lion: Tiger (B) Bird: Worm (C) Dog: Tail
Similarity pattern:	stem-key, key-distractors (relation between two concepts)
Synthesis:	Predict a new conclusion: IF flying is a necessary and sufficient condition to be a Bird THEN
Key: (C)	(A) Bats can fly (B) Bats are Birds (C) Bats are Birds and Mammals
Similarity pattern:	stem-key, key-distractors (relation between two concepts)
Evaluation:	Judge the following statement: "Bats can fly BECAUSE Birds can fly"
Key: (B)	(A) The assertion and the reason are both correct, and the reason is valid. (B) The assertion and the reason are both correct, but the reason is invalid. (C) The assertion is correct but the reason is incorrect.
Similarity pattern:	stem parts (the so-called assertion & reason)

difficulty of some classes of MCQs. Other theories need to be explored to develop principled methods for controlling the difficulty of other classes of MCQs (e.g., calculation and evaluation questions).

One of the possible steps that needs to be taken in order to expand the difficulty-control theory to other classes of MCQs is to build and analyse a large corpus of real-world MCQs; trying to find different sources of difficulty. In addition, a general model of students can be incorporated to address possible differences between different cohorts.

Chapter 4

Generating MCQs from Ontologies

This chapter presents an overview of the landscape of ontology-based question generation. The landscape is wide and has multiple dimensions and its understanding is relevant to understand the decisions regarding which questions and generation methods will be considered in later chapters. This will also help to identify the specific contributions of this thesis in comparison with existing work on automatic question generation.

Every assessment has three foundational elements [WCG01]: (i) a representation of students' knowledge of a particular subject, (ii) a task that shows how do students perform on this subject and (iii) an interpretation method to reason about students' knowledge mastery from the evidence obtained (i.e., their performance). An example of element (ii) is a question in an exam. Students' performance on an exam is taken as an evidence of what they know or are able to do. Indeed, both the quality and the quantity of tasks (e.g., exam questions) has an impact on the validity of the assessment process. In general, MCQs can be of 1) good or bad quality, 2) high or low difficulty, and 3) high or low cost (for setting and marking). Similarly, a set of questions (i.e., an exam) can be of good or bad quality which depends on the quality of the individual questions and quantity and choice of questions (of certain levels of difficulty, covering the different areas of the domain). Ideally, we want a large number of good MCQs of different difficulty levels for the lowest possible cost (i.e., with as little human intervention as possible).

To lower the cost, various attempts have been made to automate the generation of assessments. We study how these attempts have evolved over time by researchers in different communities. We do not only present attempts made by the ontology community but rather give a wider picture by including some attempts made outside the ontology community. The reason is that studying the growing body of literature on ontology-based MCQ generation cannot be carried out in isolation of related literature in other communities, e.g., Natural Language Processing (NLP).

4.1 Systematic review of existing QG approaches

A large body of research exists on automatic QG approaches from different types of knowledge sources. To gain a deeper understanding of the field and to avoid overlooking specific subfields, we carried out a systematic review of the field. Systematic reviews are a standard research methodology that aims to minimise bias in selecting what to be reviewed and it can help to conduct reviews that can be replicable by following a well-defined procedure. The procedure we followed to conduct our systematic review is outlined in Figure 4.1. Five academic databases were used to find relevant peer-reviewed articles on automatic QG approaches published between 1969 and 2015. The search was restricted to the first 50 results sorted by relevance to the search term “question generation”. Although we were interested in QG approaches that are (i) automatic and (ii) based on some knowledge source, it was tricky to capture these two criteria in the search term. On the one hand, it is tricky to capture the first criterion because various terms have been used in the literature to describe automatic approaches, e.g., automatic, computer-aided, technology-aided, to name a few. On the other hand, it was tricky to capture the second criterion because various types of knowledge sources have been used for QG purposes. However, we filter out irrelevant papers in a later step as we will see below. The search was conducted on the following databases: INSPEC, ACM Digital Library, IEEE Xplore, ScienceDirect and ERIC, in a decreasing order of their contribution to the initial phase of gathering related articles (using the specified search/inclusion criteria). In this initial phase, the exclusion criteria were based on reading the title and abstract of the paper to judge its relevance. Irrelevant topics include self-generation of questions as a learning strategy to improve reading comprehension. In addition, 16 papers were excluded after reading their full texts. The exclusion criteria at this phase include: (1) the paper presents a work on progress only (i.e., position paper) and the description of the QG approach is not sufficient to understand how the questions are generated, (2) the paper presents a computer-aided mechanism to *deliver* assessments, rather than generating assessments, (3) the paper focuses on question *answering* rather than question generation, (4) the paper is presented in a language other than English, e.g. Japanese or (5) the presented QG generation approach is mainly based on a template and different questions are generated by substituting some place holders by random numerical numbers. We decided to include the last criterion because it violates our definition of QG, as defined in Chapter 1, which states that a QG system should take as input a knowledge source (e.g., text or ontology). We include in our review template-based QG approaches that populate some templates using domain-specific concepts extracted from a domain knowledge source.

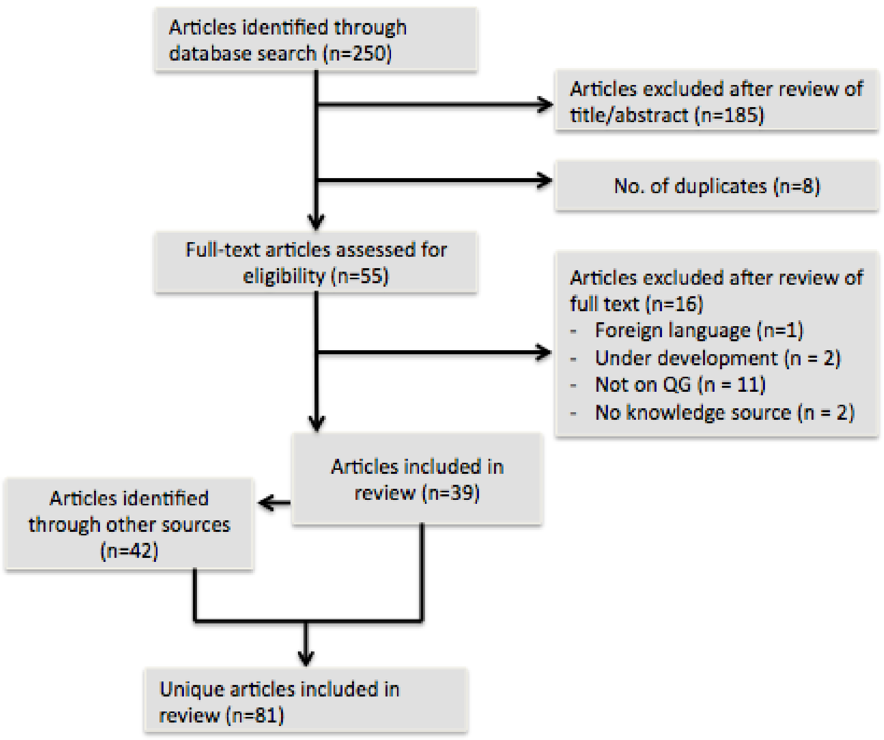

Articles identified through database search (n=250)
Articles excluded after review of title/abstract (n=185)
No. of duplicates (n=8)
Full-text articles assessed for eligibility (n=55)
Articles excluded after review of full text (n=16)
- Foreign language (n=1)
- Under development (n = 2)
- Not on QG (n = 11)
- No knowledge source (n = 2)
Articles included in review (n=39)
Articles identified through other sources (n=42)
Unique articles included in review (n=81)

Figure 4.1: Procedure followed to systematically review existing QG approaches

As shown in Figure 4.1, the total number of reviewed papers up until this phase was 39. We have also manually selected additional papers appearing in bibliographies of the papers gathered in the previous phase. Moreover, we gathered additional papers from different sources such as: (i) forwarded by peers, (ii) identified through reading related textbooks or attending related conferences and/or (iii) identified through manual search in Google or other search engines at different times using different search terms.[1] This has resulted in a total number of 81 papers included in our review. We have focused on selecting papers which can contribute better to our understanding of the QG field in general. For instance, we have selected earlier or later papers by authors of papers in our corpus, papers that have been cited frequently in our corpus and/or papers published before 2000 or after 2013 (i.e., very old or very recent publications).

After a quick investigation of these papers, we structured our observations based on the following coding scheme: who (i.e., contributors), when (i.e., the specific year the contribution started), why (i.e., what was the purpose of the contribution and which discipline/domain was it applied to), how (i.e., required input, QG method and distractor generation method if applicable), what (i.e., question format, answer format, feedback format if available and whether or not the method controls difficulty of questions) and finally how the method was evaluated. A detailed table showing all the reviewed approaches can be found in Appendix A.

Yao et al. [YBZ12] have classified QG approaches as follows: (i) syntax-based, (ii) template-based and (iii) semantics-based. We have extended this classification in order to accommodate some QG approaches which do not fit in these categories. In particular, we have added two additional categories, namely (iv) rule-based and (v) schema-based. Syntax-based QG methods mainly (syntacticly) manipulate unstructured knowledge sources (i.e., text) while semantics-based methods make use of structured knowledge sources. Structured sources of knowledge can take different formats. As defined in Chapter 2, we use the term knowledge base to refer to different formats of structured knowledge sources (e.g., lightweight ontologies/taxonomies, rich ontologies, or other knowledge bases such as semantic networks, concept maps, linked data). Template-based methods use structured or unstructured knowledge sources in addition to *domain-dependent* templates. Note that the QG space is multidimensional and hence a QG approach can fit in various categories.

[1]Some of the search terms used included: “ontology-based question generation”, “automatic question generation”, “computer-aided question generation”, in addition to changing the term “question generation” for the term “item generation”.

4.2 Past, present and future of automatic MCQs generation

We illustrate the relatively short history of MCQ generation in Figure 4.2. The automatic generation of assessment questions started with generating free-response (i.e., open-ended) questions and, later, approaches for generating MCQs were introduced. Although we focus on generating MCQs, we review approaches to generate both kinds of questions. Exploring the literature of free-response QG approaches is relevant to understand why and when MCQ generation approaches have emerged. In addition, free-response QG approaches can be extended to generate MCQs by adding a mechanism to generate suitable distractors. Syntax-based QG methods were the first to be developed. The shift from syntax-based QG methods to semantics-based methods was triggered by the need to minimise the generation of questions which do not make sense (e.g., "Who was the investment?" [HS09]) and/or to generate good distractors. We elaborate on these issues below.

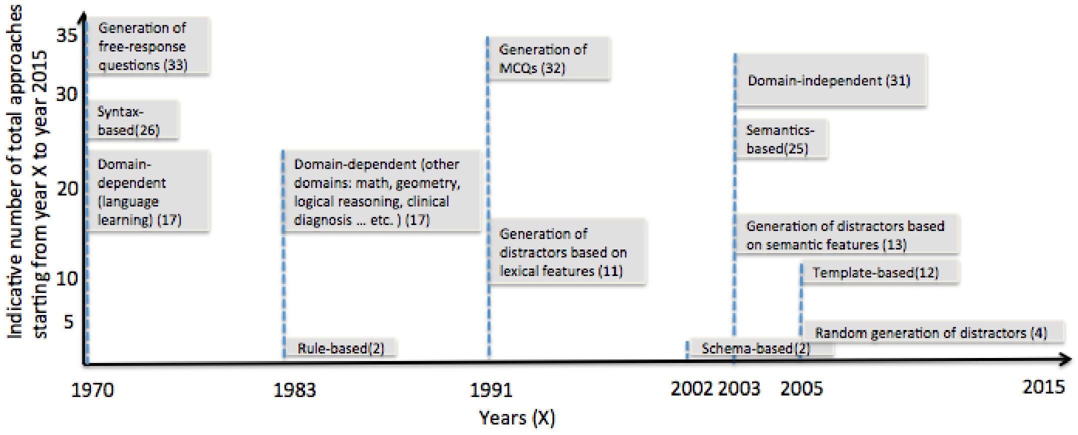


Figure 4.2: Evolution of automatic QG methods

An important deal of research effort has been devoted to generate questions for language learning and testing. We elaborate on various examples of QG systems that are mainly targeted at language learning/testing. Historically speaking, research on automatic QG techniques can be traced back to the 70's when Bormuth (1970) [Bor70] introduced the *prose-based item generation* theory for the automatic generation of questions from prose passages. Bormuth proposed to use existing learning materials (e.g., reading passages) to generate questions with the least possible human interference. The basic motivation behind Bormuth's theory is that manual assessment generation is subjective and inefficient. Early research on automatic QG, including Bormuth's work, involved syntactic transformations of sentences to construct WH-questions, i.e., who, what, where and when. For example, a sentence such as "The boy rode the horse", would be transformed to "Who rode the horse?". In a study carried out by Roid and Haladyna [RH76], the authors acknowledge that one of the problems of subjective-item writing is that it generates questions of varying quality (e.g., different writers generate questions of different difficulties), but the authors also report that Bormuth's item-generation rules did not eliminate "subjectivity" of item writing. They suggested that further investigation of automatic QG techniques is needed. However, Bormuth's theory has been abandoned later and no further investigations or improvements of the theory have been presented. Wolfe's AUTOQUEST system (1975) [Wol75, Wol76] is another well-known early work which also generates WH-questions by syntactic transformations of individual sentences using a pattern-matching approach.

In another early project on QG, Stevens (1991) [Ste91] suggested to utilise concordance tools to derive language learning exercises from general corpora. He described how to use concordance tools to generate a set of truncated sentences; each of these is displayed to the student on a separate line, with a centred blank in each sentence. The student is given a set of words to choose from, and only one choice (i.e., the key) will make the lines meaningful and grammatically correct. The distractors are morphological variants of the key. The motivation behind introducing this kind of concordance-based exercises, as reported by Stevens, is that student performance on traditional gap-filling exercises, which were open-ended, can be unexpectedly poor, especially for beginner language learners. In particular, Stevens suggested that adding distractors to gap-filling exercises can help in making these exercises more accessible, and therefore more useful, to such students. This shows that the interest in generating MCQs has emerged to solve previously observed problems with free-response questions.

It should be noted that MCQs were already popular in the assessment domain before the introduction of automatic methods to generate them (and before any form of technology-aided assessment). However, with the rapid rise of technology-aided education, assessment has been forced to pick up the pace. Large, and sometimes massive, number of students can enrol in online courses.

Making the easy-to-mark MCQs more popular than the hard-to-mark open ended questions. This is equally true for traditional large-scale assessments such as standardised college admission tests (e.g., SAT) which have also become popular in the US and in other countries around the world. All of these reasons have contributed to the interest in automatic MCQ generation methods. However, none of them were the main motivation behind early MCQ generation approaches such as Stevens' concordance-based system [Ste91]. Consistent with this, Moscow et al. (2004) [MBB+04] generated multiple-choice cloze questions[2] and reported that free-response cloze questions were too difficult to answer and mark because many correct answers were possible. The shift to generate MCQs in Moscow's case was to reduce the costs associated with free-response questions (cost of answering and marking).

In addition to the above systems, another early QG generation system that was targeted at the language learning domain was introduced by Conaim (1997) [Con97]. He suggested three different ways to generate multiple-choice cloze questions. The blank position is determined either mechanically by deleting every n^{th}-word or selectively by either selecting words with certain frequencies or a certain word class (e.g., verb, noun or adjective). Conaim initially points out that a target of 50% acceptable test items is reasonable (compared to acceptable-item rate of 66% for an experienced human setter), but he later reports that acceptable-item rate for questions generated by removing the nth word was much less than his initial target. He also reports that questions generated using the selective strategies were of better quality in terms of item difficulty and item discrimination. Later, Fairon (1999) [Fai99] developed a dynamic item banking system (EVALING) that utilises linguistic tools for facilitating the creation and management of French language tests. Manual creation of exercises is avoided in the EVALING system by designing search tools that apply manually compiled linguistic patterns to large corpora to retrieve sentences that can be used in the exercises. Although the system generates free-response exercises, Fairon reports that these exercises can be transformed to multiple choice format (by adding distractors), but does not explain how can this be accomplished. A number of other NLP-based QG techniques have also targeted the language learning domain [HN05a, SSY05, MC09, HS09, Hei11, GK10, MN14]. This might be due to the

[2]Cloze questions are usually used to assess vocabulary and reading comprehension. A portion of text (with certain words removed from the text) is presented to the student to fill in the blanks with suitable words that "close" the text.

fact that generating distractors for language testing is sometimes easier than generating distractors for other domains. For example, in grammar tests, different tenses of the same word can be good (and easily generated) distractors. Another explanation of why this line of QG approaches was, and is still, attractable is that you can simply take any book, story, newspaper article and use it to generate questions.

A number of QG approaches have been devoted to improve the distractor generation mechanism. Early MCQ generation approaches generate distractors based on syntactic or lexical features (e.g., same part of speech, same frequency, derivative words of the same prefix or suffix). Later, an interest in semantics-based distractor generation mechanisms has developed. The approach introduced by Mitkov et al. (2003) [MH03, MAHK06] marks the beginning of interest in semantics-based QG methods in which the distractors are generated according to their similarity, in terms of their meaning, to the key. The distractors in the system of Mitkov et al. are extracted from a lexicon (e.g., WordNet). The study shows that computer-aided MCQ generation performs better than manual construction of test items in terms of time without compromising quality. In two in-class experiments, the generated questions discriminate well between students and high percentage of distractors were functional. Although Mitkov et al. report that their approach is applicable to various domains, they have only evaluated their system in the linguistics domain. Later, Karamanis et al. (2006) [KLM06] have conducted a pilot study to use the system of Mitkov et al. in a medical domain and have reported that some questions were simply too vague or too basic, indicating that using a lexicon might not be suitable for all domains.

Ontology-based QG techniques have only evolved around 2006, see for example [HMMP06, ZSRG08, PKK08]. In general, ontology-based methods are domain-independent which means that they do not target a specific domain. Existing attempts involve generating questions for various domains such as relational databases [HMMP06], computer hardware [ZSRG08], science [AP11], maths [Wil11] and history [AY14], to name a few. Ontology-based QG methods, along with other methods that utilise structured knowledge sources, are usually classified as semantics-based methods. The main reasons for using ontologies as a knowledge source are: (i) their ability to generate good distractors and (ii) their ability to generate deep questions about the domain of interest. An example of a deep question is a questions that asks about relations between the different notions of the domain. A classical motivation behind ontology-based QG methods, and semantics-based methods in general, is the availability of high quality knowledge sources that can be (cheaply) reused (e.g., Guidon [Cla83]).

In short, research on automatic QG has been growing since the 70's of the last century with contributions from the education community and the NLP and KR communities. This has resulted in many books (e.g., Item Generation for Test Development [IK02], Automatic Item Generation [GH13]) and workshops (e.g., the series of international workshops on QG from 2008 to 2012 [RG09, BP10]). Looking at the (short) history of automatic QG, we notice the diversity of contributions and the lack of cohesion among contributors [GH13]. In addition, there is a lack of a unifying theme. So on the one hand, some contributions have targeted the theoretical foundations of QG. And on the other hand, other (non-theoretical) contributions have targeted the practical needs of QG by providing a technology that can alleviate the burden of manual QG. The future of automatic QG would benefit from a marriage between theorists and technologists which can be achieved by promoting multidisciplinary collaborations.

4.3 Dimensions of MCQs automatic generation

Taking into account the diversity of existing contributions on QG, we structure our discussion on available design options around the following topics: (i) why do people generate questions automatically?, (ii) what is required as input?, (iii) how to generate?, (iv) what will be generated (as output)? and (v) how to evaluate the generated questions? These design options are illustrated in Figure 4.3.

4.3.1 Purpose of questions

As we have already discussed in previous chapters, our main purpose of generating questions is to use them as assessment items. Consistent with our intentions, educational assessment has been the main purpose driving many other QG approaches, see for example [MH03, HN05a, HMMP06, PKK08, AY14]. These approaches generate questions that are suitable for both formative and summative assessment purposes. Other approaches have focused on generating formative assessment questions, e.g., to support self-studying [AP11, CCSO14].

Within the assessment domain, some of the existing QG approaches are suitable for generating questions for specific domains/disciplines. Examples of such

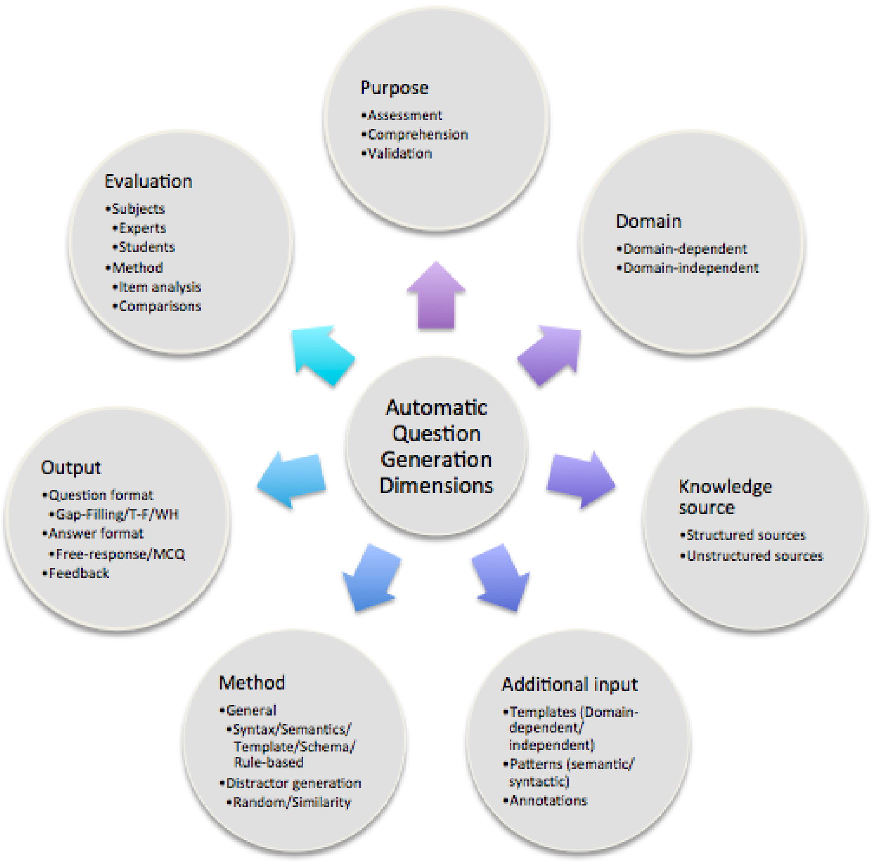


Figure 4.3: Design space of MCQs automatic generation

domain-dependent approaches include approaches for the generation of math word problems[3] [SB02, DS03, NB11, Wil11] or approaches for the generation of language assessments (including reading comprehension, vocabulary and grammar assessments) [LWGH05, CLC06, LSC07, GK10]. In contrast, other approaches [MAHK06, ZSRG08, PKK08, CCSO14] are domain-independent and are suitable for generating questions for various domains.

It is important to mention that educational assessment is not the only useful application for automatic QG approaches. Such approaches have been shown to be useful for other purposes such as validation and comprehension. For example, Bertolino et al. [BDDS11] have suggested to automate the validation of domain models by generating questions from these models. The generated questionnaires

[3]A math word problem is a mathematical exercise in which the background information on the problem is presented as text, i.e., in words rather than in mathematical notation. For example, a mathematical problem written in mathematical notation as “What is the result of 3 + 2?” might be presented in a word problem as “John has three apples and two oranges. How many fruits does John have in total?”.

can help in identifying the parts in the model which require further consideration and can guide the dialogues between domain experts and modelling experts. The results of evaluating their approach, although not statistically significant, are promising in that they have successfully identified some faults in domain models. Their results also show that the generated questionnaires have helped domain experts in gaining better understanding of the model which was built according to the views of multiple domain experts from multiple domains. Along similar lines, Liu et al. [LCR12, LCAP12, LC12, LCR14] have introduced various methods to assist students in validating their self-generated academic content (e.g., literature reviews). Their approach is capable of generating questions along the different levels of Bloom's taxonomy [BK56] such as: verification, comparison, procedural, casual and judgemental questions. Their evaluation shows that their approach generates questions that can be as useful as questions generated by human supervisors (after filtering out questions with grammatical and semantic errors).

4.3.2 Domain knowledge source

After defining the purpose of generation, a suitable knowledge source must be selected. We focus on generating questions from structured sources, in particular, OWL ontologies, although other alternative knowledge sources can be utilised such as unstructured text [Hei11, BBV12] or other structured sources (e.g., linked data [dM11, LL14, JS14], databases [SH14], knowledge bases [CCSO14]). In addition, some existing QG systems use multiple sources (e.g., text and ontologies [MAHK06]). We briefly review the most influential approaches below, starting with unstructured sources for historical reasons.

4.3.2.1 Unstructured sources

In one of the influential[4] attempts to generate MCQs, Mitkov et al. (2003) [MAHK06, MH03] introduced an NLP methodology for generating multiple-choice test items from electronic texts and utilised a lexicon (e.g., WordNet) for generating appropriate distractors. The procedure involves three main tasks: (i) term extraction, (ii) stem generation and (iii) distractor selection. Following similar lines, Heilman and Smith (2009) [HS09, HS10a, HS10b, Hei11] described a system

[4]in terms of number of citations, compared to other QG related publications.

that can automatically generate factual questions. They also discussed some of the computational and linguistic challenges related to extracting questions from text such as the limited ability of NLP-techniques to generate deep questions and difficulty in extracting relations between concepts (and hence generating good distractors). Following on, Liu et al. [LCR12] have overcome these challenges and reported that they have generated deep questions from texts using NLP-based approaches. In addition, Huang et al. (2014) [HTSC14] extended Hielman and Smith's approach to generate plausible distractors by replacing the head word of the answer phrase with *similar* words from those that appear in the given content (similarity here refers to same part of speech). Moreover, some recent NLP-based approaches [AMF11, AM14, MN14] have successfully extracted relations from unstructured texts to support the automatic generation of MCQs.

Grammaticality of questions is an issue that has been considered in many NLP-based QG approaches [BBV12], especially in those approaches that are based on syntactical transformations of individual sentences [YZ10]. For example, Heilman and Smith [HS09] report that 36.7% of their generated questions were rated as ungrammatical by reviewers. In addition, 39.5% of the questions were rated as not making sense which is a category that was suggested to be merged with the grammaticality category after analysing reviewers' responses and their agreements/disagreements. Later, Mazidi and Nielsen [MN14] report a 61% reduction in grammatical errors compared to Heilman and Smith's results.

As pointed out by Vanderwende [Van08], one of the important characteristics of QG systems is to be able to generate the *important* questions about the domain of interest. To achieve this, NLP-based QG methods first (automatically) identify the central concepts and sentences in the source text [OGP12]. Then, questions around these central notions are constructed. This step is part of many existing NLP-based QG approaches, including Mitkov et al.[MAHK06] and Becker et al. [BBV12]. In addition, general purpose NLP-based methods for text summarisation are also available [Lin04]. In contrast, Olney et al. [OGP12] point out that this step is less relevant in a pedagogical context and suggested that the key terms are usually already identified in textbook indices or glossaries.

4.3.2.2 Structured sources

A number of ontology-based QG approaches have been proposed [HMMP05, HMMP06, PKK08, CT09, CT10, ZPK11, AY14]. For example, Zitko et al. (2008)

[ZSRG08] proposed templates and algorithms for the automatic generation of objective questions from ontologies. The focus in their work was to extend the functionality of a previously implemented tutoring system (Tex-Sys) by concentrating on the assessment component. The main difference between this approach and our approach is in the distractor selection mechanism. The mechanism adopted by Zitko et al. is to generate a set of *random* distractors for each MCQ without an attempt to filter them according to their pedagogical appropriateness.

The distractor selection mechanism was enhanced by Papasalouros et al. (2008) [PKK08] who presented various ontology-based strategies for the automatic generation of MCQs. These strategies are used for selecting keys and distractors. The evaluation of the produced questions by domain experts shows that the questions are satisfactory for assessment but not all of them are syntactically correct. The major problem related to this approach is the use of highly constrained rules with no theory backing that would motivate the selection of these rules. For example, the distractors in each MCQ are mainly picked from the set of siblings of the correct answer while there might be other plausible distractors. Later, Cubric and Tosic (2009) [CT09] reported on their experience in implementing a *Protégé* plugin for question generation based on the strategies proposed by Papasalouros et al. [PKK08]. More recently, Cubric and Tosic (2010) [CT10] extended their previous work by considering new ontology elements, e.g., annotations. In addition, they suggested employing question templates to avoid syntactical problems in the generated questions. They have also illustrated, by some examples, that their method is suitable for generating questions of both lower and higher levels of Bloom's taxonomy [BK56].

In addition to the distractor selection mechanism, it is important to consider some presentation issues that might affect the quality of the generated questions, e.g., naturalness and fluency of the language. Consistent with this, Williams [Wil11] extends the use of SWAT[5] 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". Presentation issues, which are generally out of the scope of this thesis, can be left as a post-generation step. A human editor can easily modify presentation errors later during the reviewing process which must be carried out anyway, and the cost of editing is expected to be marginal. However, the thesis addresses presentation issues that

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

can have an impact on the difficulty of the generated questions. For example, we aim to remove clues that can reveal the correct answer even for those students who do not have sufficient knowledge about the topic since this can destroy the validity of assessment items. Examples of clues caused by presentation issues are grammatical inconsistencies between the stem and distractors or word repetitions between the stem and key.

In order to make ontology-based QG accessible to test developers with no prior experience in building ontologies, we need to provide them with strategies that can help them to build or extend ontologies in systematic ways. For example, Gavrilova et al. [GFB05] 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. [SG06] present a case study for utilising the above 5-step strategy to develop an ontology for the domain of C programming. Another related topic that we like to mention here is the availability of (semi)-automatic approaches for building ontologies from textual materials (e.g., wikipedia pages). This is still an open research topic attracting considerable attention. An interested reader is referred to [BCM05] for a general overview and to [ZN08] for a discussion about the automatic building of ontologies for educational purposes. In case an existing ontology is used as a source for QG, it might be necessary to select parts of the ontology for QG, rather than using the ontology as a whole. As we discussed in Chapter 2, ontology modularisation [SPS09, SSZ09] techniques can play an important role in extracting parts of an ontology in a logically-safe manner.

4.3.3 Additional input

Looking at existing QG approaches, we notice that various types of supporting inputs are utilised by the different approaches. For example, some approaches, mainly template-based approaches, use domain-dependent templates [MC09, Wil11] to support the generation of questions for a particular domain. The drawback of such approaches is that they require a huge manual effort to develop these templates, and yet, they are not applicable for other domains. Other approaches [ZSRG08, CT10, CCSO14] benefit from domain-independent templates to enhance the readability and naturalness of the generated questions. Although these templates require manual effort as well, they are suitable for generating questions for various domains.

Some QG systems take as input a user query [ZSCZ11, CCSO14] and generate a natural language question that can be answered by the knowledge base available in the system. Such systems can support learners in their knowledge acquisition process. Other systems require, as input, a set of syntactic-patterns [Wol76, Fai99, CLC06] or semantic-patterns [AMF11, AM14, MN14] to extract suitable parts from the knowledge source which can be used to construct a question. In some systems [CLC06], the search for phrases satisfying the required patterns is performed over an external corpus which can be different from the main source of domain knowledge. Other systems use corpora for other reasons such as performing statistical analysis [Con97, LWGH05, AMF11, AM14] or machine learning techniques [HN05a, HN05b, CBEM12].

Many existing QG approaches utilise lexicons or thesauri to generate plausible distractors, for example [SSY05, MAHK06, LSC07, GK10, YBZ12, MGL12, MN14, HTSC14]. Other approaches require as input a set of annotations either to train classifiers to find sentences that are optimal for QG [BK12a, BK12b] or to support the generation of questions with images [PKK11].

4.3.4 Generation method

4.3.4.1 General method

As discussed earlier, QG systems can be generally classified as: (i) syntax-based, (ii) semantics-based, (iii) template-based, (iv) schema-based or (v) rule-based. Syntax-based systems, for example [Bor70, MBB$^+$04, HS09], apply syntactic transformations on suitable sentences in order to generate factual questions. Such systems use readily available (textual) learning materials or informational sources. The main limitation of syntax-based systems is that they cannot infer the relations between the different parts of the (unstructured) knowledge source. This can result in the generation of vague questions, questions that do not make sense or questions whose answer is not available in the source text [HS09]. Semantics-based systems, for example [ZSRG08, PKK08, AY14], try to overcome these limitations by utilising structured sources that are ideally rich in terms of defined relations between the main concepts of the domain. Some QG approaches transform the source text into some sort of structured representation and then apply semantics-based QG method in order to generate deep questions about the main concepts in the text and their relations [YZ10, OGP12]. Template-based systems, for example [TG05, MC09], rely on domain-dependant templates and fill these templates by extracting relevant parts from structured or unstructured knowledge sources. Schemas are similar to templates in that they are domain-dependant but schemas are more abstract than templates. In other words, different wordings of templates have to be enumerated exhaustively. However, schemas can provide a higher level of abstraction for a group of templates that represent variants of the same problem. The so-called schema theory has been applied to the automatic generation and variation of mathematical problems [SB02, DS03]. Each schema defines the underlying problem structure by identifying the equations that relate the entities of the problem to one another. Finally, rule-based systems, for example [SH14], utilise rule-based knowledge sources to generate questions that can assess students in terms of their understanding of the important rules of the domain.

4.3.4.2 Distractors generation method

Generating distractors was identified by Haladyna [Hal94] as the most difficult part of MCQs generation. While some QG approaches select distractors at random [HN05a, HN05b, ZSRG08], others try to filter distractors according to their appropriateness. Filtering can be achieved in various ways. For example, Lin et al. [LSC07] consult Google's search engine to filter out obviously wrong distractors. Some of the disadvantages of utilising the web to get distractors is that it goes outside textbook knowledge and may produce unsuitable distractors (e.g., correct but synonymous). Alternatively, others [Con97, MBB$^+$04, BFE05] choose distractors based on lexical features (e.g., same part of speech (POS) or similar frequency to the correct answer). In some template-based approaches, each template has a different mechanism for selecting distractors, e.g., changing part of speech or changing the verb into different form [CLC06]. Others [TG05] generate distractors manually based on student misconceptions as identified by experienced domain instructors or related literature.

We choose to filter distractors according to their semantic similarity to the key. Consistent with our approach, existing approaches have already attempted to utilise the notion of semantic similarity to generate MCQs, for example [MH03, MAHK06]. Mitkove et al. [MHVR09], have carried out an investigation to find a similarity measure that is suitable for question generation. The reported results show that, among the evaluated measures, Lin's measure [Lin98] was the most effective measure in generating questions with quality distractors.[6] The quality of distractors was evaluated using classical item analysis methods [Keh95], see Chapter 2 for more details. Although this investigation was not carried out in a statistically significant fashion, the preliminary results are promising and show that computable similarity measures exist that can be used to generate questions of good quality. An important observation in this study is that the average item difficulty was high (i.e., greater than 0.5). These results can be explained by the similarity-based theory presented in this thesis, see Hypothesis 1. Given that the distractors were always chosen to be highly similar to the key, it is not surprising to get questions with high average difficulty. Following Mitkov's findings, a number of automatic QG methods utilised similarity measures to generate distractors that are semantically similar to the key, see for example [MGL12, AY14]. Rather than automatically measuring similarity, Al-Yahya [AY14] gathers similarity judgments from human experts during the ontological engineering process, limiting the applicability of the method to other existing ontologies.

4.3.5 Output

4.3.5.1 Question and answer format

Questions can be classified according to the format of the question and the format of the answer. Question formats include WH-questions, fill in the blank, T/F questions and many others. The answers to these questions can take different formats such as: (i) free-response, (ii) multiple-response MCQs or (iii) single-response MCQs. Of course, some question formats are single-response MCQs in nature. For example, T/F questions have two fixed choices to pick from, i.e., True or False. Other questions can be presented to the students either as free-response or fixed-response questions (e.g., WH-questions, fill in the blank).

Some of the approaches presented earlier in this chapter have focused on generating MCQs [MAHK06, CT10, AY14]. Other approaches have chosen to generate free-response questions [DS03, WHL08, SSIS08, Kim08]. We choose to generate MCQs to address the challenge in generating good distractors which

[6]However, as we will see in detail in Chapter 5, Lin's similarity measure requires an annotated corpus (in addition to an ontology) to measure the similarity between the concepts in the ontology. This limits the applicability of this similarity measure to ontologies that do not have such an accompanying corpus. We explore alternative similarity measures in Chapter 5.

can be varied, in terms of similarity to the key, to generate questions of different difficulties. In terms of question formats, some systems have developed approaches to generate fill-in-the-blank [BFE05, HN05a, BBV12] while others generate True/False statements [PKK08, ZSRG08, BDDS11] or WH-questions [SSIS08, Kim08, MSK09, MC09, HS09].

4.3.5.2 Feedback

The ability to provide instant feedback to many students is one of the advantages that comes with MCQs. Some existing QG systems have highlighted the importance of providing feedback to students after providing their answers. Ideally, feedback should be tailored according to students' needs based on their provided answer. Ahmed and Parsons [AP13] automatically generate hints and suggestions to guide students in learning science through a series of MCQs. Liu and Lin [LL14] provide, as feedback, links for extra learning materials related to the current question. The feedback in this case is independent of the particular answer selected by the student. Mostow and Chen [MC09] provide students with hints to assist them through some reading comprehension exercises. Most existing QG systems ignore the importance of generating informative feedback or assume that revealing the correct answer is sufficient as a feedback.

4.3.6 Evaluation

When it comes to evaluating the generated questions, the purpose for which the questions have been generated plays an important role. For example, if the questions were generated for educational assessments purposes, then evaluating these questions has to involve administrating them to students. Some of the existing QG systems that have reported the results of evaluating their generated questions in studies involving students include [Con97, MBB$^+$04, BMB04, BFE05, KWDH14]. Such student-centred studies involve analysing the results of students answering these questions in order to evaluate questions' difficulty, item discrimination and usefulness of distractors (if they are available). Alternatively, or additionally, evaluations may include assessing the effectiveness of using the generated questions to enhance students' learning. For example, Chaudhri et al. [CCSO14] compared a group of students who have access to the QG system with a control group and reported that students who have used their system scored significantly better on a subsequent quiz. Similarly, Kuyten et al. [KBS$^+$12] assessed learners' enhanced comprehension of domain knowledge after reading question/answer pairs generated by the system. Moreover, students' performance on the generated questions can be compared to their performance on standardised tests in the domain of interest, see for example [MBB$^+$04, SSY05, BFE05].

Student-centred studies are not always possible due to various reasons (e.g., the system is in early stages). A more common method to evaluate automatically generated questions is through expert-based studies, see for example [CLC06, LSC07, PKK08, MC09, HS09, YZ10, PKK11, dM11, BDDS11, AY14]. In addition, expert-based evaluations have been combined with student-based evaluations in some studies, see for example [MAHK06]. Expert-based evaluations can be as simple as asking the expert whether a generated question is acceptable or unacceptable [MEAF12, BBV12] or they can be more systematic in identifying the aspects to be evaluated. Various aspects have been the focus of expert-based evaluations. For example, Yao et al. [YZ10] have focused on the following aspects: (i) relevance, (ii) syntactic correctness and fluency, (iii) ambiguity and (iv) variety. In contrast, Agarwal et al. [AM11] have focused on the quality of distractors in terms of readability and semantic meaning. The main drawback of expert-based evaluations is that they are time-consuming and labor intensive which can limit the number of experts willing to participate in them (including those experts who are friends of the investigator). Alternatively, Heilman and Smith [HS10b] have used Amazon Mechanical Turk to recruit raters of their generated questions. They report that each question has costed 27.5 cents for 5 raters. Another possible method to evaluate the automatically generated questions is through comparing them to questions generated by experts [LWGH05, LCR12, HTSC14].

The above evaluation methods are usually not part of the QG workflow, but rather a post-generation step to assess the efficiency of the QG approach. In contrast to this, Heilman [Hei11] proposed an overgenerate and rank method in which evaluating the generated questions is part of the QG workflow. The purpose of the ranking phase is to select higher quality questions out of a collection of automatically generated questions possibly containing a large number of low quality questions. In his PhD thesis, Heilman [Hei11] reports that the ranker roughly doubles the acceptability rate of top-ranked questions.

One observation that is specifically related to ontology-based QG approaches is that most existing approaches have used handcrafted ontologies for evaluating their QG approach, see for example [CNB03, HMMP06, ZSRG08, PKK08, AY14]. It is very important to evaluate these approaches by evaluating their utility over existing real ontologies. Obviously, using existing ontologies, rather than building new ones, for QG purposes lowers the cost of generation. Evaluating the use of existing ontologies for QG is important for understating the issues that may arise when using existing (probably big) ontologies and how to deal with these issues to enhance the quality (e.g., relevance, coverage) of the generated questions.

4.4 Proposed method to generate MCQs

Before discussing what methods will be adopted in this thesis to generate MCQs, we provide a general overview of the methods presented in the previous section. The goal is to emphasise the gaps in existing methods that will be addressed in this thesis. As can be observed from our discussion in the previous section, attempts to automate the generation of assessment questions have started in the 70's with the emphasise in the beginning being on free-response questions. This might be explained by the difficulty of generating proper distractors. Methods to generate MCQs were focused in the beginning on the generation of assessments for the language learning domain for which it seemed easy to generate distractors. Mechanisms to generate distractors for this domain included the generation of distractors based on their lexical features, e.g., mutating a verb to different forms such as: play, playing, played, to play. Clearly, this mechanism cannot be applied in other domains. Some existing ontology-based MCQ generation methods basically select distractors randomly. Obviously, this can affect the quality of the generated questions by generating distractors that are not functional. We aim to propose more principled mechanisms to generate distractors.

A general observation regarding existing QG approaches presented in this chapter is the lack of control over the difficulty of the generated questions. In contrast, controlling difficulty is one of the main contributions of this thesis. Current attempts to control the difficulty of automatically generated questions include [Wil11, KS13]. However, the proposed models to control difficulty in these systems have not been validated by any empirical studies. Williams [Wil11] discussed a few factors to control the difficulty of (free-response) mathematical word problems such as order of presentation and providing extraneous information. Kovacs et al. [KS13] have presented an approach to control the difficulty of MCQs that is similar to our approach in that it is based on the notion of similarity. However, their approach is different from our approach in that they vary the similarity between the stem and distractors, rather than between the key and distractors. Moreover, the single example presented in their paper shows that the utilised distance measure is not precise (not sensible to all similarities/differences between the compared concepts). Again, their approach has not been validated by presenting the generated questions to students in order to assess their difficulty. Newstead et al. [NBH$^+$02, NBH$^+$06] have presented different difficulty models to predict the difficulty of analytical reasoning questions. The main factors which can affect the difficulty of such questions, according to Newstead et al., are complexity of the rules needed to solve the questions, the number of mental models required to represent the problem, and question type. The limitation of Newstead's approach is that different difficulty models are needed for the different question types and that the models are applicable to specific question types (i.e., analytical reasoning questions). We present a more general approach to control the difficulty of MCQs and evaluate our approach in a series of empirical studies (showing promising results).

Our difficulty-control mechanism is based on the use of similarity measures. In Chapter 5, we present a new family of similarity measures for general OWL ontologies. Existing off-the-shelf similarity measures could not be utilised due to their limitations. For example, some of them are suitable for taxonomies rather than rich ontologies whereas some of the measures which were designed for DL ontologies are suitable for a limited range of ontologies (e.g., inexpressive ontologies, ontologies with acyclic TBoxes or ontologies with ABoxes). We have addressed these limitations by presenting similarity measures that are applicable to a wider range of ontologies.

So far, throughout this chapter and previous chapters, we have hinted at (in different places) the design options which have been adopted by our QG approach. Educational assessment is the main purpose for which we generate questions but also show, in Chapter 8, that the generated questions can be useful for ontology validation purposes. We focus on developing QG methods that can be suitable for various domains (i.e., domain-independent methods). We also focus on generating questions to assess knowledge rather than skills or attitudes. All of this has been discussed in both the theoretical foundations established in Chapter 3 and the empirical evaluations presented later in Chapter 7. We have chosen to evaluate our QG approach in a series of student-based and expert-based studies in order to gain a better understanding of the benefits and limitations of the proposed approach.

We have also chosen to utilise structured knowledge bases, namely OWL ontologies, as the main input to our QG system. Domain-independent templates are also required as an additional input. The motivation of using such templates is twofold: first it allows the generation of different kinds of questions and second it allows end-users to extend the system by adding additional templates to suit their needs. In Table 4.1, we present a few possible templates which have been generally proposed by existing QG approaches, for example [ZSRG08, CT10]. These templates naturally fit with the source, i.e., ontologies. The last template, i.e., analogy template, has not been proposed by other QG systems. It addresses higher levels of Bloom's taxonomy. Other templates can be easily added to our system. The templates shown in Table 4.1 are the ones used in the empirical evaluation studies presented in later chapters. We have focused on templates that can generate basic, but important, questions about the domain of interest. The questions, including distractors, are generated in a semantics-based manner by accessing the ontology via a reasoner. This means that the (correct and wrong) answers are selected according to their implicit and explicit relation to the stem and to each other.

In Chapter 7, we present details on implementing a prototype ontology-based QG system. The prototype system also includes a module to measure the pairwise similarity of some of the (possibly complex) concepts in the ontology.

Table 4.1: Basic questions templates

Definition:	What is the following definition describing? [Annotation N]
Key: concept name annotated with N	distractors: concept names not annotated with N
Recognition:	Which is the odd one out?
Key: concept name not subsumed by S1	distractors: concept names subsumed by S1, S1 is a concept name
Generalisation:	What is [Concept name A]?
Key: concept name that is a subsumer of A	distractors: concept names that are non subsumers of A, excluding subsumers of the key
Generalisation 2:	What is [Concept name A]?
Key: concept expression that is a subsumer of A	distractors: concept expressions that are non subsumers of A, excluding subsumers of the key
Specification:	Which of these is [Concept name A]?
Key: concept name that is a subsumee of A	distractors: concept names that are non subsumees of A, excluding subsumers and siblings of the stem
Specification 2:	Which of these is [Concept expression X]?
Key: concept name that is a subsumee of A	distractors: concept names that are non subsumees of A, excluding subsumers of the stem
Analogy:	[Concept name A] is to [Concept name B] as:
Key: a pair of concept names that have the same relation as A, B	distractors: pairs of concept names that have a relation other than the relation between A, B

Chapter 5

Similarity measures

In earlier chapters, we have shown that similarity measures can be used for generating educationally useful assessment questions and conjectured a similarity-based theory of controlling MCQs difficulty. In this chapter we elaborate on the topic of measuring similarity in general and focus on similarity measures for concepts.

We define similarity measurement as the process of assigning a numerical value reflecting the degree of resemblance between two objects (e.g., concepts) w.r.t. a specific ontology $\mathcal{O}$. There are many forms of similarity (e.g., semantic, syntactic or lexical). Although different forms of similarity might be valuable for different purposes, we focus on semantic similarity between ontology concepts and refer to this as conceptual similarity.

Similarity is at the core of numerous ontology-related applications such as ontology alignment [ES07], ontology learning [CW10], ontology clustering and comprehension [MISR11], to name a few. Consider for example the Gene Ontology [Con00]: measuring genes' similarity [BSTP$^+$10, BAB05, SDRL06, WDP$^+$07] would allow scientists to infer novel potential (undiscovered) gene functions. The diversity of similarity-based applications and the centrality of similarity for controlling the difficulty of a range of assessment questions have motivated us to explore similarity measures in detail.

Several attempts have been made to develop methods for measuring conceptual similarity in knowledge bases (e.g., taxonomies, rich ontologies) [RMBB89, Res95, Lin98, JC97, WP94, ODI07, Jan06, dSF08, LT12]. In addition, the problem of measuring similarity is well-founded in psychology and a number of similarity models have already been developed [Est55, Wag08, Blo01, She87, Nos92, GS04, Tve77, Gen83, Lev66, HCR03, Jam50]. Rather than adopting a psychological model for similarity as a foundation, we will see that existing similarity measures for ontologies are either ad-hoc and unprincipled or not computationally possible. Accordingly, no off-the-shelf similarity measure was suitable for our question generation purposes.

This chapter reviews some fundamental psychological theories which can be considered as a foundational stone for similarity measures. We also analyse some existing similarity measures before presenting a new family of similarity measures that addresses some of the problems that are present in existing measures. In addition, we discuss the desired properties of similarity measures and examine whether they hold for the new proposed measures. An empirical evaluation of the new family of similarity measures is presented in Chapter 6 which shows (among other things) that the new measures are best correlated with human similarity judgement which is very important when using these measures for QG purposes.

5.1 Background on similarity

A number of theories of similarity have been proposed by psychologists during the last few decades. From the point of view of a psychologist, the ability to assess similarity is the backbone of humans' thinking [Jam50]. Hence, theories of similarity are closely related to theories of cognition. For example, our success in solving a new problem depends on finding a similar problem that we previously solved [GS04]. Other cognitive processes are also founded on this sameness notion, such as categorisation, generalisation and recognition, to name a few. The need to understand these cognitive processes has motivated psychologists to build general theoretical models for similarity. In what follows, we shed a light on some of these models.

5.1.1 Psychological models of similarity

Note that this section is not aimed at providing a comprehensive survey of the whole (broad) research area, but rather provides some answers for the following questions: (i) Which representation systems can be used to represent the concepts to be compared? and (ii) How can we compare these representations in order to compute the similarity of two concepts?

5.1.1.1 Common elements model

In this model, the compared objects are represented as collections or sets of undifferentiated elements. The similarity between two objects is calculated by counting the number of common elements and/or summing up their values. For example, Figure 5.1 shows two objects within which individual elements are represented as Xs. The proportion of shared elements (red Xs) represents the similarity between the two objects. This model is useful in comparing simple objects that can be defined using what can be called the *elemental approach* [Est55, Wag08]. It might be certainly useful in revealing certain aspects of the compared objects, e.g., their discrimination [Blo01]. However, it is not applicable to measuring similarity between more complex objects that have more properties other than their membership in a collection/set of objects.

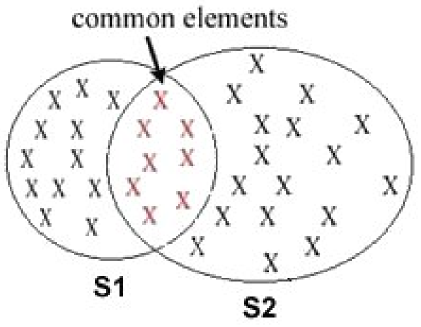


Figure 5.1: Common elements model, taken from [Blo01]

5.1.1.2 Geometric model

A geometric map represents a set of objects in an N-dimensional space where each object is represented as a point in the space [She87, Nos92]. Figure 5.2 shows an example of a geometric model with two dimensions (size and colour).

The similarity of two objects (a, b) is inversely related to their distance d_{ab} which can be calculated by the following formula:

$$d_{ab} = (\sum_{k=1}^{N} |x_{ak} - x_{bk}|^r)^{1/r}$$

where x_{ak} is the value of object a in dimension k, x_{bk} is the value of object b in

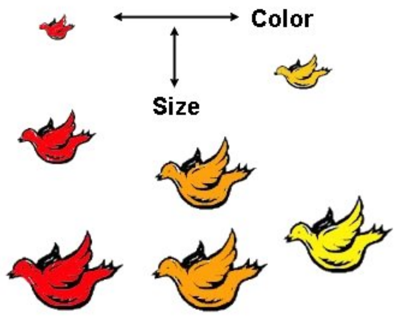


Figure 5.2: Geometric model, taken from [Blo01]

dimension k and r is a spatial metric which can be set to different values (e.g., Euclidean metric (r=2) or City-block metric (r=1)) [GS04]. While Euclidean metric yields constant distance values regardless of how coordinates are rotated w.r.t. objects in the space, city-block distances are sensitive to rotations. The value of each object in a certain dimension can be measured in different ways, e.g., using subjective human judgements.

Geometric models might be useful in comparing objects that can be represented in a few continuous dimensions. However, in cases where objects are characterised by non-continuous (e.g., qualitative) features, things get tricky as geometric models cannot be used to represent such features. Moreover, geometric models are founded on the use of metric distances. Hence, it was assumed that they fulfil the three metric properties: (i) minimality, (ii) symmetry and (iii) the triangle inequality. However, Tversky [Tve77] provided some counter examples in which geometric models fail to fulfil the aforementioned properties. Later, Tversky [Tve77] introduced the features model which is explained in Section 5.1.1.5.

5.1.1.3 Alignment-based model

In alignment-based models, objects are represented propositionally and/or hierarchically (Directed Acyclic Graph (DAG) representing partonomic relations). The advantage of representing objects in such relational structures is that in many applications (e.g., analogy detection) the interest is in comparing objects based on their underlying relations. Similarity is based on the degree of "correspondence" between the two structures. As an example, Figure 5.3 illustrates the similarity between an atom and the solar system as presented in the structure mapping theory developed by Gentner [Gen83].

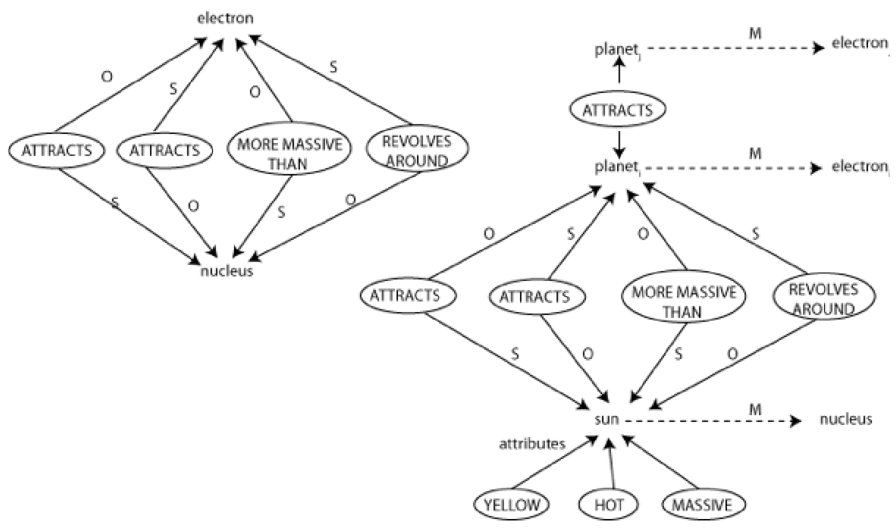


Figure 5.3: Alignment-based model, taken from [Gen83]

5.1.1.4 Transformational model

Transformational models are founded on the premise that similarity between two objects is relative to the number of *operations* required to transform one object into the other. For example, Levenshtein's distance corresponds to counting the number of necessary changes (e.g., insert, remove, replace) to transform a string into another string [Lev66]. Along similar lines, Hahn et al. [HCR03] define the similarity of two objects as a function of the complexity of transforming one into the other. Similarly, Distel et al. [DAB14] proposed a dissimilarity measure that is based on description trees for the lightweight description logic $\mathcal{EL}$. The difficulty in applying such models is in specifying the possible transformational operations. This difficulty limits the applicability of this model to complex representations.

5.1.1.5 Features model

To overcome the limitations of geometric models, Tversky [Tve77] proposed to use contrast models. In Tversky's contrast model, an object is represented as a set of features. The degree of similarity S_{ab} between objects (a, b) corresponds to features common to a and b, unique features of a and unique features of b. Figure 5.4 better illustrates this model. Similarity can be computed by the following formula:

$$S_{ab} = xf(a \cap b) - yf(a - b) - zf(b - a)$$

where $f(a \cap b)$ is the number of shared features between a and b, $f(a-b)$ is the number of features in a but not in b, $f(b-a)$ is the number of features in b but not in a, and (x, y, z) are weights used to change the focus of comparison. According to this model, similarity of object a to object b is different from similarity of b to a (i.e., symmetry is not assumed).

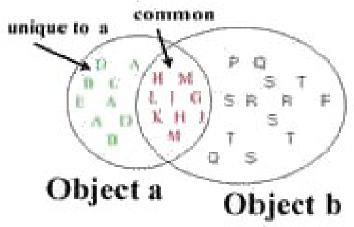


Figure 5.4: Features model, taken from [Blo01]

5.2 Measuring similarity in ontologies: existing approaches and challenges

All the previous models assume things that we do not necessarily have in ontologies, at least directly. For instance, it is tricky to define what we mean by features of a concept in an ontology. Due to richness of ontologies, many things may be associated with a given concept (e.g., atomic subsumers/subsumees, complex subsumers/subsumees, instances, referencing axioms, role successors). Looking at existing approaches for measuring similarity in DL ontologies, one can notice, on the one hand, that approaches which aim at providing a numerical value as a result of the similarity measurement process are mainly founded on feature-based models [Tve77], although they might disagree on which features to consider. On the other hand, approaches aiming at providing a descriptive result are mainly founded on transformational models [Lev66, HCR03] and are usually characterised as distances or dissimilarity measures rather than similarity measures. Instance-based measures of similarity are similar to the common elements model where instances represent the elements of the model. But they can also be seen as simple feature-based models where only a single feature is considered (i.e., membership feature). Other measures compute similarity by measuring distances in some graph of the ontology. We will see in Section 5.2.3.1 that these measures can also be seen as feature-based models where only distinguishing features of the compared objects, rather than common features, are considered.

In what follows, we concentrate on feature-based notions of similarity where the degree of similarity S_{CD} between concepts (C, D) depends on features common to C and D, unique features of C and unique features of D. Considering both common and distinguishing features is a vital property of the feature-based model. Tversky [Tve77] exemplified this notion of similarity by exploring the similarity between English Letters. For instance, if we try to compare the letters E, F and I, we can say that E is more similar to F than to I because they share more common features. And I is more similar to F than to E because they have fewer distinguishing features.

Looking at existing approaches for measuring similarity in ontologies, we find that some of these approaches consider either common or unique features (rather than both) and that some approaches consider features that some instances (rather than all) of the compared concepts have. Indeed, this has an impact on the similarity measurement result. Of course, a good similarity measure should be sensitive to both the explicit and implicit parts of the underlying ontology (i.e., considering entailments). Thus, to account for all the features of a concept, we could consider all (possibly complex) entailed subsumers of that concept. Again, this is not always the case in some existing similarity measures. Detailed inspection of the problems of some existing measures is presented in Section 5.2.2.

5.2.1 Desired properties of similarity measures

To understand some of the important properties which need to be considered when measuring similarity in ontologies, we present the following example:

Example 5.1 Consider the ontology $\mathcal{O}_{\text{Organisms}}$:

$$\begin{aligned} &\mathit{Animal} \sqsubseteq \mathit{Organism} \sqcap \exists \mathit{eats}.\top, && \mathit{Plant} \sqsubseteq \mathit{Organism}, \\ &\mathit{Carnivore} \sqsubseteq \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Animal}, && \mathit{Herbivore} \sqsubseteq \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Plant}, \\ &\mathit{Omnivore} \sqsubseteq \mathit{Animal} \sqcap \exists \mathit{eats}.\mathit{Animal} \sqcap \exists \mathit{eats}.\mathit{Plant} \end{aligned}$$

Please note that our "Carnivore" is also known as *obligate* carnivore. A good similarity function $Sim(\cdot)$ is expected to derive that Sim(Carnivore, Omnivore) $>$ Sim(Carnivore, Herbivore) because the first pair share more **common** subsumers and have fewer **distinguishing** subsumers than the second one. On the one hand *Carnivore*, *Herbivore* and *Omnivore* are all subsumed by the following **common** subsumers (abbreviated for readability): $\{\top, Org, A, \exists e.\top\}$. In addition to those subsumers, *Carnivore* and *Omnivore* have the following **common** subsumer: $\{\exists e.A\}$. On the other hand, they have the following **distinguishing** subsumers: $\{\exists e.P\}$ while *Carnivore* and *Herbivore* have the following **distinguishing** subsumers: $\{\exists e.P, \forall e.P, \exists e.A, \forall e.A\}$. Here, we have made a choice to ignore (infinitely) many subsumers and only consider a select few (by limiting the considered language). Clearly, this choice has an impact on $Sim(\cdot)$. Details on such design choices are discussed later.

We refer to the property of accounting for both common and distinguishing features as **rationality** which is defined in Definition 1.

Definition 1. *A* rational similarity function $Sim(\cdot)$ *satisfies the following conditions:*

- *If the number of common subsumers increases and the number of distinguishing subsumers remains constant then the similarity should also increase.*
- *If the number of distinguishing subsumers decreases and the number of common subsumers remains constant then the similarity should decrease.*

In addition, the related literature refers to some other properties for evaluating similarity measures. In the following definition, we briefly present some of these properties. For a detailed overview, the reader is referred to [dSF08, LT12].

Definition 2. *Let $\mathcal{O}$ be an ontology. Let $\mathcal{L}$ be a concept language and C, D, E, L, U be concepts in $\mathcal{L}$. A similarity measure $Sim : \mathcal{L} \times \mathcal{L} \to [0,1]$ is*

1. Equivalence closed *if* $Sim(C, D) = 1 \Leftrightarrow \mathcal{O} \models C \equiv D$.

2. Equivalence invariance *if* $\mathcal{O} \models C \equiv D \Rightarrow Sim(C, E) = Sim(D, E)$.

3. Symmetric *if* $Sim(C, D) = Sim(D, C)$.

4. Fulfilling the triangle inequality property *if* $1 + Sim(D, E) \geq Sim(D, C) + Sim(C, E)$.

5. Subsumption preserving *if* $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E \implies Sim(C, D) \geq Sim(C, E)$.

6. Reverse subsumption preserving *if* $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E \implies Sim(D, E) \geq Sim(C, E)$.

7. Monotonic *if* $\mathcal{O} \models C \sqsubseteq L \sqcap U, D \sqsubseteq L \sqcap U, E \sqsubseteq U, E \not\sqsubseteq L, \nexists H \in \mathcal{L}$ *s.t.* $C \sqsubseteq H \wedge E \sqsubseteq H \wedge D \not\sqsubseteq H \implies Sim(C, D) \geq Sim(C, E)$.

The quality of the similarity function/measure depends on the above measures. Typically, a similarity coefficient (e.g., Jaccard [Jac01], Tversky [Tve77], Dice [Dic45]) is used to compute similarity. The definitions of these coefficients are provided in Table 5.1.

Similarity coefficient	Definition
Tversky	$T(A, B) = \frac{\lvert(A \cap B)\rvert}{\lvert(A \cap B)\rvert + \alpha\lvert A - B\rvert + \beta\lvert B - A\rvert}$
Dice	$D(A, B) = \frac{2 \cdot \lvert(A \cap B)\rvert}{\lvert A\rvert + \lvert B\rvert}$
Jaccard	$J(A, B) = \frac{\lvert(A \cap B)\rvert}{\lvert(A \cup B)\rvert}$

Table 5.1: Some standard similarity coefficients for sets of "features" A,B

Note that Dice's coefficient is more sensitive to shared features than to distinguishing features. Tversky's coefficient may be asymmetric. However, it has been shown [Jac01] that Jaccard's distance (obtained by subtracting the Jaccard's coefficient from 1) is a proper metric, i.e., it satisfies Properties 1-3 in Definition 2.

5.2.2 Overview of existing approaches

For clarity, we classify existing similarity measures according to two dimensions. In the first dimension, we classify similarity measures to (i) taxonomy-based measures and (ii) ontology-based measures. In the second dimension, similarity measures are classified into (i) intentional measures and (ii) extensional measures.

5.2.2.1 Taxonomy vs. ontology based measures

Taxonomy-based measures [RMBB89, WP94, Res95, Lin98, JC97] only consider a taxonomy, i.e., a (possibly acyclic) directed graph or (possibly) a tree. For DLs, we *could use* the inferred class hierarchy (i.e., the Hasse diagram of the partial order on concept names in $\mathcal{O}$ induced by the entailment relation $\mathcal{O} \models A \sqsubseteq B$) as this graph and would thus only consider atomic subsumptions (e.g., $Carnivore \sqsubseteq Animal$). In fact, this can be considered an approximated solution to the problem which might be sufficient in some cases. However, the user must be aware of the limitations of such approaches. For example, direct siblings are always considered equi-similar although some siblings might share more features or subsumers than others.

Ontology-based measures [dSF08, Jan06, LT12] take into account more of the knowledge in the underlying ontology (e.g., $Carnivore \sqsubseteq \forall eats.Animal$). These measures can be further classified into (a) structural measures (b) interpretation-based measures or (c) hybrid measures. Structural measures [Jan06, LT12] first transform the compared concepts into a normal form (e.g., $\mathcal{EL}$ normal form [LT12] or $\mathcal{ALCN}$ disjunctive normal form [Jan06]) and then compare the syntax of their descriptions. To avoid being purely syntactic, they first unfold the concepts w.r.t. the *TBox* which limits the applicability of such measures [Jan06] to cyclic terminologies. Moreover, some structural measures [LT12] are applicable only to inexpressive DLs (e.g., $\mathcal{EL}$) and it is unclear how they can be extended to more expressive DLs. Interpretation-based measures mainly depend on the notion of canonical models (e.g., in [dSF08, dFE05] the canonical model based on the *ABox* is utilised) which do not always exist (e.g., consider DLs with disjunctions). Hybrid measures are those structural measures that use canonical interpretations (e.g., [dFE06, Fd06, EPT15]).

5.2.2.2 Intensional vs. extensional measures

Intensional measures [RMBB89, WP94, Jan06, LT12] exploit the terminological part of the ontology while extensional measures [Res95, Lin98, JC97, dSF08, dFE05, dFE06] utilise the set of individual names in an *ABox* or instances in an external corpus. Extensional measures are very sensitive to the content under consideration; thus, adding/removing an individual name would change similarity measurements. These measures might be suitable for specific content-based applications but might lead to unintuitive results in other applications because they do not take concept definitions into account. Moreover, extensional measures cannot be used with pure terminological ontologies and always require representative data.

5.2.3 Detailed inspection of some existing approaches

After presenting a general overview of existing measures, we examine in detail some measures that can be considered “cheap” options for measuring similarity and explore their properties. For what follows, we use $S_{\text{Atomic}}(C, \mathcal{O})$ to denote the set of atomic subsumers of C (i.e., $S_{\text{Atomic}}(C, \mathcal{O}) = \{D \in N_C \mid \mathcal{O} \models C \sqsubseteq D\}$).[1] We also use $\text{Com}_{\text{Atomic}}(C, D, \mathcal{O}), \text{Diff}_{\text{Atomic}}(C, D, \mathcal{O})$ to denote the sets of common and distinguishing atomic subsumers, respectively, i.e.:

$$\begin{aligned} \text{Com}_{\text{Atomic}}(C, D, \mathcal{O}) &= S_{\text{Atomic}}(C, \mathcal{O}) \cap S_{\text{Atomic}}(D, \mathcal{O}) \\ \text{Diff}_{\text{Atomic}}(C, D, \mathcal{O}) &= S_{\text{Atomic}}(C, \mathcal{O}) \triangle S_{\text{Atomic}}(D, \mathcal{O}) \end{aligned}$$

5.2.3.1 Rada et al.

This measure utilises the length of the shortest path [RMBB89] between the compared concepts in a taxonomy (e.g., one could use the inferred class hierarchy). To measure the distance between Carnivores and Herbivores of Example 5.1, we count the number of links in their shortest path. The corresponding class hierarchy is shown in Figure 5.5. The length in this case is 2. Similarly, if we count the number of links between Carnivores and Omnivores we get the same numerical value. However, we explained earlier that Carnivores and Omnivores are more similar than Carnivores and Herbivores. Therefore, this (dis)similarity measure is (too) coarse-grained since it does not always differentiate between specific cases (where it could do so if it had considered the information in the ontology). Moreover, if we count the number of links between Animals and Plants we also get the same numerical value, although Carnivores and Omnivores are more specific concepts than Animals and Plants and therefore could be more similar. Also, for all C, the similarity of C and its direct subsumers is always the same as its similarity to any of its subsumees since the number of edges is always 1.

Since this measure is based on counting the number of edges in the shortest path connecting two concepts, C and D, we can formulate the measure as:

[1] N_C is the set of atomic concepts

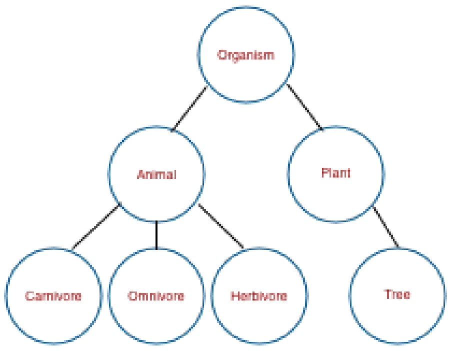


Figure 5.5: Inferred class hierarchy of the ontology in Example 5.1

$$d_{\text{Rada}}(C, D, \mathcal{O}) = |\text{Diff}_{\text{Atomic}}(C, D, \mathcal{O})|$$

To understand the sources of the above problems, we examine the subsumers that are considered by the measure. As can be clearly seen in the above formula, this measure only considers (atomic) distinguishing subsumers of C and D. The set of atomic subsumers of Carnivore is $S_{\text{Atomic}}(Carnivore) = \{\top, C, A, Org\}$. $S_{\text{Atomic}}(Herbivore) = \{\top, H, A, Org\}$ and $S_{\text{Atomic}}(Omnivore) = \{\top, O, A, Org\}$. To measure the dis-similarity $d_{\text{Rada}}(Carnivore, Omnivore)$,[2] we take the cardinality of the set of distinguishing subsumers $\text{Diff}_{\text{Atomic}}(Carnivore, Omnivore)$ which equals to 2 as provided by the original measure (i.e., length of the shortest path). Obviously, the essential problem here is that the measure takes only distinguishing features into account and ignores any possible common features.

5.2.3.2 Wu and Palmer

To account for both common and distinguishing features, Wu & Palmer [WP94] presented a different formula for measuring similarity in taxonomies (e.g., tree-shaped inferred class hierarchies). Their similarity is originally formulated as:

$$S_{\text{Wu \& Palmer}}(C, D) = \frac{2 \cdot N3}{N1 + N2 + 2 \cdot N3}$$

where E is the least common atomic subsumer of C and D (i.e., the most specific concept name that subsumes both C and D), $N1$ is the number of nodes on the path from C to E, $N2$ is the number of nodes on the path from D to E and $N3$

[2]To compute similarity, we can either take the multiplicative inverse of dissimilarity or subtract the dissimilarity value from 1 and then normalise it by dividing over the maximum similarity value.

is the number of nodes on the path from E to the root. The above formula is equivalent to:

$$S_{\text{Wu \& Palmer}}(C, D, \mathcal{O}) = \frac{2 \cdot |\text{Com}_{\text{Atomic}}(C, D, \mathcal{O})|}{2 \cdot |\text{Com}_{\text{Atomic}}(C, D, \mathcal{O})| + |\text{Diff}_{\text{Atomic}}(C, D, \mathcal{O})|}$$

Although this measure accounts for both common and distinguishing features, it only considers atomic concepts and it is clearly more sensitive to common features than to distinguishing features.

5.2.3.3 Resnik and other IC measures

In information theoretic notions [Res95, Lin98, JC97] of similarity, the information content IC_C of a concept C is computed according to the following formula:

$$IC_C = -logP_C$$

where the probability (P_C) of a concept C is the probability of encountering an instance of it. For example, $P_\top = 1$ and $IC_\top = 0$. We say that $\top$ is not informative. Accordingly, Resnik [Res95] defines the similarity $S_{\text{Resnik}}(C, D)$ as follows:

$$S_{\text{Resnik}}(C, D) = IC_{LCS}$$

where LCS is the least common atomic subsumer of C and D. As discussed earlier, this measure, along with any extensional measure, assumes that the instances set is present and that it is of good quality which may be difficult to achieve. Another problem is that those measures take into account features that some instances of C and/or D have, which are not necessarily neither common nor distinguishing features of all instances of C and D. In addition, Resnik's measure in particular does not take into account how far the compared concepts are from their least common subsumer. To overcome the last problem, two [Lin98, JC97] other IC-measures have been proposed:

$$S_{\text{Lin}}(C, D) = \frac{2 \cdot IC_{LCS}}{IC_C + IC_D}$$

$$S_{\text{Jiang\&Conrath}}(C, D) = 1 - IC_C + IC_D - 2 \cdot IC_{LCS}$$

5.3 A new family of similarity measures for ontologies

Following our exploration of existing measures and their associated problems, we present a new family of similarity measures that addresses these problems. To understand the limitations of existing measures in fulfiling the needs of real world ontologies, we briefly present some statistics gathered from the NCBO BioPortal corpus of ontologies [MP15]. Firstly, let us recall that extensional measures require the availability of a set of instances or individuals with the ontology. We want to know the percentage of ontologies that have such a set in the corpus. In fact, only 32% of ontologies in the corpus have at least one individual. The number of individuals is not normally distributed (i.e., massive deviation) across the corpus. This means that extensional measures cannot be used with at least 68% of ontologies in the corpus. Secondly, let us recall that structural measures require acyclic terminologies. However, cycles have been detected in at least 45.8% of the ontologies. Finally, 22.2% of ontologies have cycles and no individuals, which means neither extensional measures nor structural measures can be used for these ontologies. This clearly shows the need for new measures that can cope with all ontologies.

The new measures use Jaccard's coefficient and adopt the features model as the psychological foundation. The features under consideration are the subsumers of the concepts being compared. Note that we aim at similarity measures for general OWL ontologies and thus a naive implementation of this approach would be trivialised because a concept has infinitely many subsumers. To overcome this issue, we present two possible refinements for the similarity function.

5.3.1 A first refinement to the similarity function

As a first refinement to the similarity function, we do not simply count all subsumers but consider subsumers from a set of (possibly complex) concepts of a concept language $\mathcal{L}$. Let C and D be concepts, let $\mathcal{O}$ be an ontology and let

$\mathcal{L}(\widetilde{\mathcal{O}})$ be a concept language defined over the signature of $\mathcal{O}$. We set:

$$\begin{aligned}
S(C,\mathcal{O},\mathcal{L}) &= \{D \in \mathcal{L}(\widetilde{\mathcal{O}}) \mid \mathcal{O} \models C \sqsubseteq D\}\\
\mathrm{Com}(C,D,\mathcal{O},\mathcal{L}) &= S(C,\mathcal{O},\mathcal{L}) \cap S(D,\mathcal{O},\mathcal{L})\\
\mathrm{Union}(C,D,\mathcal{O},\mathcal{L}) &= S(C,\mathcal{O},\mathcal{L}) \cup S(D,\mathcal{O},\mathcal{L})\\
\mathrm{Diff}(C,D,\mathcal{O},\mathcal{L}) &= S(C,\mathcal{O},\mathcal{L}) \bigtriangleup S(D,\mathcal{O},\mathcal{L})\\
\mathrm{Sim}(C,D,\mathcal{O},\mathcal{L}) &= \frac{|Com(C,D,\mathcal{O},\mathcal{L})|}{|Union(C,D,\mathcal{O},\mathcal{L})|}
\end{aligned}$$

where, as before, $|M|$ denotes the cardinality of a set M, $\mathrm{Com}(C,D,\mathcal{O},\mathcal{L})$ is the set of common subsumers of concepts C and D, $\mathrm{Union}(C,D,\mathcal{O},\mathcal{L})$ is the set of all subsumers of C or D and $\mathrm{Diff}(C,D,\mathcal{O},\mathcal{L})$ is the set of distinguishing subsumers of C and D. Note that $\mathrm{Com}(C,D,\mathcal{O},\mathcal{L})$ and $\mathrm{Diff}(C,D,\mathcal{O},\mathcal{L})$ are disjoint, but there can be concepts in $\mathcal{L}(\widetilde{\mathcal{O}})$ that are in neither of them. In what follows, we omit $\mathcal{O}$ and/or $\mathcal{L}$ from $\mathrm{S}(\cdot), \mathrm{Com}(\cdot), \mathrm{Union}(\cdot), \mathrm{Diff}(\cdot)$ and $\mathrm{Sim}(\cdot)$ whenever it is clear from the context.

To design a new similarity measure, it remains to specify the set $\mathcal{L}(\widetilde{\mathcal{O}})$. As a first example for a simple similarity measure which captures taxonomy-based measures (i.e., it considers atomic concepts only), we present:

$$AtomicSim(C,D) = Sim(C,D,\mathcal{O},\mathcal{L}_{\mathrm{Atomic}}(\widetilde{\mathcal{O}})),\ \text{and}\ \mathcal{L}_{\mathrm{Atomic}}(\widetilde{\mathcal{O}}) = \widetilde{\mathcal{O}} \cap N_C.$$

Another measure to be presented here is:

$$SubSim(C,D) = Sim(C,D,\mathcal{O},\mathcal{L}_{\mathrm{Sub}}(\widetilde{\mathcal{O}})),\ \text{and}\ \mathcal{L}_{\mathrm{Sub}}(\widetilde{\mathcal{O}}) = Sub(\mathcal{O}).$$

where $Sub(\mathcal{O})$ is the set of (possibly complex) concept expressions in $\mathcal{O}$.
The rationale of $SubSim(\cdot)$ is that it provides similarity measurements that are sensitive to the modeller's focus which is captured in the subconcepts of the ontology. In addition, it provides a cheap (yet principled) way for measuring similarity in expressive DLs since the number of candidates is only linear in the size of the ontology. To capture more possible subsumers, we present:

$$\begin{aligned}
&GrammarSim(C,D) = Sim(C,D,\mathcal{O},\mathcal{L}_{\mathrm{G}}(\widetilde{\mathcal{O}})),\ \text{and}\\
&\mathcal{L}_{\mathrm{G}}(\widetilde{\mathcal{O}}) = \{E \mid E \in Sub(\mathcal{O})\ \text{or}\ E = \exists r.F, \text{for some}\ r \in \widetilde{\mathcal{O}} \cap N_R\ \text{and}\ F \in Sub(\mathcal{O})\}.
\end{aligned}$$

where N_R is the set of role names. To understand some of the cases in which $SubSim(\cdot)$ fails to capture some relevant subsumers, we take a look at the following example:

Example 5.2 Consider the following ontology $\mathcal{O}$:
$TBox = \{A \sqsubseteq \exists r.B\}$
$RBox = \{r \sqsubseteq s\}$
$Sub(\mathcal{O}) = \{\top, A, B, \exists r.B\}$

In the above example, $\exists s.B \notin Sub(\mathcal{O})$ but it is considered as a candidate subsumer for $GrammarSim(\cdot)$. Of course, other examples can be easily constructed. In the empirical evaluations in Chapter 6, we have chosen to include only grammar concepts which are subconcepts or which take the form $\exists r.F$ to make the experiments more manageable. However, the grammar can be extended easily.

5.3.2 Properties of the new measures

The new similarity measures presented above were designed to address certain problems in existing similarity measures. In particular, there was a need for a similarity measure that can be suitable for use with any arbitrary OWL ontology. For instance, the new measures are suitable to be used with ontologies of high expressivity. Of course, the "semantic sensitivity" ranges from "low" in $AtomicSim(\cdot)$ to a "higher" sensitivity in $GrammarSim(\cdot)$. The new measures are also safe to be used with ontologies with cyclic definitions and GCIs. In addition, since the new measures are intensional (i.e., rely on the $TBox$ rather than the *Abox*), they do not require a representative set of instances which can be difficult to establish.

In addition to the ontology-related properties discussed above, in what follows we examine whether the new measures satisfy the properties presented in Section 5.2.1. Theorem 1 states the properties of the new measures. See Definition 1 for a description of the rationality property and Definition 2 for all the other properties.

Theorem 1. *Let $\mathcal{O}$ be an ontology, $\mathcal{L}$ be a concept language, $Sim(C, D, \mathcal{O}, \mathcal{L}) = \frac{|Com(C,D,\mathcal{O},\mathcal{L})|}{|Union(C,D,\mathcal{O},\mathcal{L})|}$ be the similarity of concepts C, D such that the set of subsumers $S(C, \mathcal{O}, \mathcal{L})$ is finite for any concept $C \in \mathcal{L}$, then $Sim(\cdot)$ is:*

i. Rational,

ii. Equivalence closed,

iii. Equivalence invariance,

iv. Symmetric,

v. Satisfying triangle inequality,

vi. Subsumption preserving,

vii. Reverse subsumption preserving,

viii. Not monotonic.

Proof.

i. To prove the rationality property, we need to show that for all $C, D, E \in \mathcal{L}$, the following properties hold:

   (a) $|\text{Com}(C, D, \mathcal{O}, \mathcal{L})| > |\text{Com}(C, E, \mathcal{O}, \mathcal{L})| \wedge |\text{Diff}(C, D, \mathcal{O}, \mathcal{L})| = |\text{Diff}(C, E, \mathcal{O}, \mathcal{L})| \implies Sim(C, D, \mathcal{O}, \mathcal{L}) > Sim(C, E, \mathcal{O}, \mathcal{L})$

   (b) $|\text{Diff}(C, D, \mathcal{O}, \mathcal{L})| < |\text{Diff}(C, E, \mathcal{O}, \mathcal{L})| \wedge |\text{Com}(C, D, \mathcal{O}, \mathcal{L})| = |\text{Com}(C, E, \mathcal{O}, \mathcal{L})| \implies Sim(C, D, \mathcal{O}, \mathcal{L}) < Sim(C, E, \mathcal{O}, \mathcal{L})$

   By its definition, $Sim(\cdot)$ is rational.

ii. By definition of $Sim(\cdot)$ and Definition 2, it is obvious that $Sim(\cdot)$ is equivalence closed because the sets of (entailed) subsumers for any two equivalent concepts are always the same.

iii. By definition of $Sim(\cdot)$ and Definition 2, it is obvious that $Sim(\cdot)$ is equivalence invariance because, again, the sets of (entailed) subsumers for any two equivalent concepts are always the same.

iv. By definition of $Sim(\cdot)$ and Definition 2, it is obvious that $Sim(\cdot)$ is symmetric.

v. By definition of $Sim(\cdot)$ which is based on the Jaccard's coefficient, $Sim(\cdot)$ satisfies the triangle inequality property just as the Jaccard's coefficient. For a proof of triangle inequality property for Jaccard's coefficient, the reader is referred to [Lip99].

vi. To prove the subsumption preservation property, we need to show that for all $C, D, E \in \mathcal{L}$, the following property holds:

if $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E$ then $Sim(C, D, \mathcal{O}, \mathcal{L}) \geq Sim(C, E, \mathcal{O}, \mathcal{L})$.

Since $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E$ then:

(a) $|Com(C, D, \mathcal{O}, \mathcal{L})| = |S(D, \mathcal{O}, \mathcal{L})|$

(b) $|Com(C, E, \mathcal{O}, \mathcal{L})| = |S(E, \mathcal{O}, \mathcal{L})|$

(c) $|Union(C, D, \mathcal{O}, \mathcal{L})| = |S(C, \mathcal{O}, \mathcal{L})|$

(d) $|Union(C, E, \mathcal{O}, \mathcal{L})| = |S(C, \mathcal{O}, \mathcal{L})|$

(e) $|S(D, \mathcal{O}, \mathcal{L})| \geq |S(E, \mathcal{O}, \mathcal{L})|$

By definition of $Sim(\cdot)$, we have: $Sim(C, D, \mathcal{O}, \mathcal{L}) \geq Sim(C, E, \mathcal{O}, \mathcal{L})$

Therefore, $Sim(\cdot)$ is subsumption preserving.

vii. To prove the reverse subsumption preservation property, we need to show that for all $C, D, E \in \mathcal{L}$, the following property holds:

if $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E$ then: $Sim(D, E, \mathcal{O}, \mathcal{L}) \geq Sim(C, E, \mathcal{O}, \mathcal{L})$.

Since $\mathcal{O} \models C \sqsubseteq D \sqsubseteq E$ then:

(a) $|Com(D, E, \mathcal{O}, \mathcal{L})| = |S(E, \mathcal{O}, \mathcal{L})|$

(b) $|Com(C, E, \mathcal{O}, \mathcal{L})| = |S(E, \mathcal{O}, \mathcal{L})|$

(c) $|Union(D, E, \mathcal{O}, \mathcal{L})| = |S(D, \mathcal{O}, \mathcal{L})|$

(d) $|Union(C, E, \mathcal{O}, \mathcal{L})| = |S(C, \mathcal{O}, \mathcal{L})|$

(e) $|S(D, \mathcal{O}, \mathcal{L})| \leq |S(C, \mathcal{O}, \mathcal{L})|$

By definition of $Sim(\cdot)$, we have: $Sim(D, E, \mathcal{O}, \mathcal{L}) \geq Sim(C, E, \mathcal{O}, \mathcal{L})$

Therefore, $Sim(\cdot)$ is reverse subsumption preserving.

viii. To prove the monotonicity property, we need to show that for all C, D, E, L, U, $H \in \mathcal{L}$, the following property holds:

if $\mathcal{O} \models C \sqsubseteq L \sqcap U, D \sqsubseteq L \sqcap U, E \sqsubseteq U, E \not\sqsubseteq L$, $\nexists H \in \mathcal{L}$ s.t.
$C \sqsubseteq H \land E \sqsubseteq H \land D \not\sqsubseteq H$ then:

$\mathrm{Sim}(C, D, \mathcal{O}, \mathcal{L}) \geq \mathrm{Sim}(C, E, \mathcal{O}, \mathcal{L})$

But there are many cases that demonstrate that this is not always the case. Consider the counter example for $AtomicSim(\cdot)$:

Let $H_c, H_{d1}, H_{d2}, H_{d3}, H_{d4}, H_{d5}, H_{d6} \in \mathcal{L}$ such that:

$$\mathcal{O} \models C \sqsubseteq H_c, D \sqsubseteq H_{d1} \sqcap H_{d2} \sqcap H_{d3} \sqcap H_{d4} \sqcap H_{d5} \sqcap H_{d6}$$

$$\mathcal{O} \models D \not\sqsubseteq H_c, C \not\sqsubseteq H_{d1}, C \not\sqsubseteq H_{d2}, C \not\sqsubseteq H_{d3}, C \not\sqsubseteq H_{d4}, C \not\sqsubseteq H_{d5}, C \not\sqsubseteq H_{d6}$$

and C, D, E have no other subsumers, then:

$$\text{AtomicSim}(C, D, \mathcal{O}, \mathcal{L}) := \frac{2}{11}$$

and

$$\text{AtomicSim}(C, E, \mathcal{O}, \mathcal{L}) := \frac{1}{5}$$

then

$$\text{AtomicSim}(C, D, \mathcal{O}, \mathcal{L}) \ngtr \text{AtomicSim}(C, E, \mathcal{O}, \mathcal{L})$$

Therefore, the counter example shows that AtomicSim$(\cdot)$ is not monotonic.

It makes sense that the new measures do not satisfy the monotonicity property because its definition (see Definition 2) depends on the notion of common subsumers only and ignores any possible distinguishing subsumers which is a core property of the new measures. Some existing similarity measures have already been evaluated according to the properties presented in Definition 2. The interested reader is referred to [LT12, TS14, dSF08] for more details.

It remains to specify the computational complexity of the new measures presented above. The lower bound of complexity is the complexity of reasoning in the DL under consideration. The upper bound depends on the number of candidate subsumers considered by each measure. For $AtomicSim(\cdot)$, the number of candidate subsumers is the number of atomic concepts. For $SubSim(\cdot)$, the number of candidate subsumers is the number of subconcepts in $\mathcal{O}$ or 2^x where x is the size of the longest axiom in $\mathcal{O}$. Finally, for $GrammarSim(\cdot)$, this number is further multiplied by the number of subconcepts times the number of role names in $\mathcal{O}$.

5.3.3 A second refinement to the similarity function

As another possible refinement to the similarity function we define similarity as a weighted function $WSim(\cdot)$. The definition of the function is flexible in that it has two changeable components: a concept language $\mathcal{L}$ and a weight function $W(\cdot)$. Changing one of the components or both allows us to design different similarity measures as we will see in the examples thereafter.

Definition 3. *Given an ontology $\mathcal{O}$ and two concepts C, D, the* weighted similarity $WSim(C, D, \mathcal{O})$ *for C and D is defined as follows:*

$$WSim(C, D, \mathcal{O}) = \frac{W(Com(C, D, \mathcal{O}))}{W(Union(C, D, \mathcal{O}))}$$

where $W : \wp(\mathcal{L}(\widetilde{\mathcal{O}})) \to \mathbb{R}^+$ is a weight function on sets of concepts such that for any set of concepts M, W satisfies the following properties:

1. $$W(M) = \begin{cases} 0 & \textit{if } M = \emptyset; \\ > 0 & \textit{otherwise.} \end{cases}$$

2. $M \subseteq M' \implies W(M) \leq W(M')$

3. *$W(M)$ is defined and is a real number for all $M \subseteq \mathcal{L}(\widetilde{\mathcal{O}})$*

In what follows, we omit $\mathcal{O}$ from $WSim(\cdot)$ whenever it is clear from the context.

A first candidate for a suitable weight function that can be used with different sets $\mathcal{L}(\widetilde{\mathcal{O}})$ is the following:

$$W(M) = \Sigma_{C \in M} \frac{1}{Size(C)}$$

where the motivation behind using it is that names and concepts close to names weigh more. We can think we could consider $\mathcal{L}$ such that $S(C, \mathcal{O}, \mathcal{L})$ is infinite and still base our similarity measurement on it. That is, we can think that the above candidate $W(M)$ is defined and is a real number for all $M \subseteq \mathcal{L}(\widetilde{\mathcal{O}})$. However, this function is a harmonic divergent infinite series for infinite sets M. Another example to be presented here simply overcomes this by using a bound K on the length of concepts considered:

$$W(M) = \Sigma_{C \in M} w_K(C)$$

$$w_K(C) = \begin{cases} \frac{1}{Size(C)} & \text{if } Size(C) \leq K; \\ 0 & \text{otherwise.} \end{cases}$$

This last weight seems attractive since it is sensitive to the vocabulary introduced in the ontology in that it prefers concept names and, depending on the DL $\mathcal{L}$, polynomially reducible to standard reasoning problems in $\mathcal{L}$.

As a refinement for the measure *SubSim*(.) introduced earlier, we introduce *SubWSim*(.) which uses a weight function $w_{Sub}(.)$:

$$w_{Sub}(C) = \begin{cases} \frac{1}{Size(C)} & \text{if } C \in \mathcal{L}_{\text{Sub}}(\widetilde{\mathcal{O}}); \\ 0 & \text{otherwise.} \end{cases}$$

A weighted grammar-based measure *GrammarWSim*$(\cdot)$ can be defined using the following weight function:

$$w_G(C) = \begin{cases} \frac{1}{Size(C)} & \text{if } C \in \mathcal{L}_{\text{G}}(\widetilde{\mathcal{O}}); \\ 0 & \text{otherwise.} \end{cases}$$

Other interesting weight functions can be designed. For example, a user might be interested in measuring the similarity with respect to a specific context or domain. To design such a weight function, it is possible to extract a module [CGHKS08] which captures the desired context and assign higher weight values to concepts in the signature of that module.

5.4 Approximations of similarity measures

Some of the presented examples for similarity measures might be practically infeasible due to the large number of candidate subsumers. For this reason, it would be nice if we can explore and understand whether a “cheap” measure (e.g., AtomicSim(.)) can be a good approximation for a more expensive one (e.g., GrammarSim(.)). We start by characterising the properties of an approximation in the following definition.

Definition 4. *Given two similarity functions* $Sim(\cdot)$, $Sim'(\cdot)$, *and an ontology* $\mathcal{O}$, *we say that:*

- $Sim'(\cdot)$ preserves the order of $Sim(\cdot)$ *if* $\forall C, D, E, F \in \widetilde{\mathcal{O}}$: $Sim(C, D) \leq Sim(E, F) \implies Sim'(C, D) \leq Sim'(E, F)$.
- $Sim'(\cdot)$ approximates $Sim(\cdot)$ from above *if* $\forall C, D \in \widetilde{\mathcal{O}}$: $Sim(C, D) \leq Sim'(C, D)$.
- $Sim'(\cdot)$ approximates $Sim(\cdot)$ from below *if* $\forall C, D \in \widetilde{\mathcal{O}}$: $Sim(C, D) \geq Sim'(C, D)$.
- *if* $Sim'(\cdot)$ *preserves the order of* $Sim(\cdot)$ *or approximates it from above or below, we say that* $Sim'(\cdot)$ *is an* approximation *of* $Sim(\cdot)$.

We start by examining whether, or not, $AtomicSim(\cdot)$ is an approximation of $SubSim(\cdot)$. The first thing to notice is that the set of candidate subsumers for the first measure is actually a subset of the set of candidate subsumers for the second measure ($\widetilde{\mathcal{O}} \cap N_C \subseteq Sub(\mathcal{O})$). However, we need to notice also that the number of entailed subsumers in the two cases does not need to be proportionally related. For example, if the number of atomic candidate subsumers is n and two compared concepts share $\frac{n}{2}$ common subsumers. We cannot conclude that they will also share half of the subconcept subsumers. They can actually share all or none of the complex subsumers. Therefore, the order-preserving property is not always satisfied. As a concrete example, let the number of common and distinguishing atomic subsumers for C and D be 2 and 4, respectively (out of 8 atomic concepts) and let the number of their common and distinguishing subconcept subsumers be 4 and 6, respectively (out of 20 subconcepts). Let the number of common and distinguishing atomic subsumers for C and E be 4 and 4, respectively and let the number of their common and distinguishing subconcept subsumers be 4 and 8, respectively. In this case, $AtomicSim(C, D) = \frac{2}{6} = 0.33$, $SubSim(C, D) = \frac{4}{10} = 0.4$, $AtomicSim(C, E) = \frac{4}{8} = 0.5$, $SubSim(C, E) = \frac{4}{12} = 0.33$. Notice that $AtomicSim(C, D) < AtomicSim(C, E)$ while $SubSim(C, D) > SubSim(C, E)$. Here, $AtomicSim(\cdot)$ is not preserving the order of $SubSim(\cdot)$ and$AtomicSim(\cdot)$ underestimates the similarity of C,D and overestimates the similarity of C,E compared to $SubSim(\cdot)$.

A similar argument can be made to show that entailed subconcept subsumers are not necessarily proportionally related to the number of entailed grammar-based subsumers. In Chapter 6, we investigate the relation between these measures in practice.

Proposition 1. *Although the set of candidate subsumers for $AtomicSim(\cdot)$ is a subset of the set of candidate subsumers for $SubSim(\cdot)$ which is a subset of the set of candidate subsumers for $GrammarSim(\cdot)$, $AtomicSim(\cdot)$ is* not an approximation *of $SubSim(\cdot)$ and $SubSim(\cdot)$ is* not an approximation *of $GrammarSim(\cdot)$.*

5.5 Relatedness and relational similarity

After introducing some approaches for measuring conceptual similarity, here we briefly discuss other forms of similarity that can also be relevant for QG. We also explore the notion of relatedness which is more general than similarity. We present some methods for measuring the degree of relatedness between concepts. The main idea is to generalise the principles introduced above for measuring the conceptual similarity to other forms of similarity, e.g., relational similarity, and relatedness.

5.5.1 Overview of relatedness

Similarity is a specific form of the general notion of relatedness. For example, we say that **cars** and **fuel** are closely related compared to **cars** and **bicycles** which are closely similar.

In earlier chapters, we have shown that similarity can play an important role in distractor generation. It seems reasonable to generalise this idea to, e.g., relatedness. To do this, we need to develop methods for measuring the degree of relatedness between different ontology components (e.g., between concepts). Unfortunately, there are no many known relatedness measures for ontologies compared to the relatively high number of existing similarity measures [MS08]. In what follows we explore some of the existing relatedness measures before presenting two new measures that are intended to be useful for QG purposes.

5.5.2 Existing relatedness measures

One of the first attempts was made by Hirst & St-Onge [HSO95] who developed a relatedness measure based on WordNet [MIL95]. The relatedness measure developed by Hirst & St-Onge focuses on the notion of a *semantically correct* relation between two words and defines it as a path that corresponds to one of the allowable patterns shown in Figure 5.6. Consistent with this, Mazuel & Sabouret

[MS08] also adopt the *semantically correct relation* patterns introduced earlier by Hirst & St-Onge. Three kinds of links appear in these patterns: Upward, Downward and Horizontal. An upward link corresponds to a generalisation (e.g., hypernymy and meronymy relations). A downward link corresponds to a specialisation (e.g., hyponymy and holonymy relations). A horizontal link corresponds to other relations which can help in specifying the meaning of a word (e.g., synonymy and antonymy relations). The rationale behind these patterns is that, firstly, generalising the context is not allowed after specifying it by following a downward or horizontal link. Secondly, at most one change of direction is allowed to avoid large changes in meaning with the exception that using a horizontal link is permitted to make a transition from an upward to a downward link.

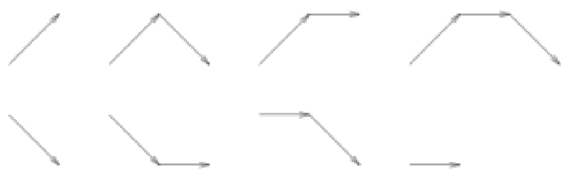


Figure 5.6: Relatedness patterns allowed by Hirst & St-Onge, taken from [HSO95]

Multiple paths can exist between any two words. To compute the weight of each path, Hirst & St-Onge suggested to use the following formula:

$$Relatedness_{H\&SO}(C, D) = J - LSP(C, D) - K \times CDP(C, D)$$

where J and K are constants defined empirically, $LSP(C, D)$ is the length of the shortest path between C, D and $CDP(C, D)$ is the number of changes of direction in this path.

While the measure developed by Hirst & St-Onge [HSO95] is based on the notion of the shortest path (e.g., similar to Rada et al. dissimilarity measure [RMBB89]), the relatedness measure developed by Mazuel & Sabouret [MS08] combines the shortest path notion with information theoretic notions (e.g., similar to Jiang & Conrath similarity measure [JC97]). So, Mazuel & Sabouret assume that the different links in the path connecting the two concepts may not have the same information content, and hence should have different weights. Also, Mazuel & Sabouret treat hierarchical relations (i.e., *is_a* and *includes*) differently from non-hierarchical ones. Hence, the weight of a hierarchical link is computed using a formula different from the formula used to compute the weight of a non-hierarchical link. In cases where different paths exist between two concepts, Mazuel & Sabouret suggest to compute the relatedness of those two concepts based on the path which has the minimum weight.

Mazuel & Sabouret [MS08] acknowledge the difficulty of classifying a relation as either an upward, downward or horizontal relation, thus limiting the generalisability of this approach to ontologies other than WordNet.

5.5.3 New relatedness measures

We choose to distinguish between hierarchical and non-hierarchical relations, just as the method proposed by Mazuel & Sabouret [MS08]. Thus, we present two separate relatedness measures for each kind of relations. Both measures are suitable for general OWL ontologies. The first measure was developed to support a QG tool that is designed to generate a specific kind of questions, namely analogy questions, which rely on relatedness notions. Details of designing and evaluating this QG tool are presented in Chapter 7. This measure is a path-based measure which is suitable only for hierarchal relations. The second relatedness measure presented below is suitable for non-hierarchical relations and it was designed to complement the first measure.

Both measures extend our new family of similarity measures to measure new kinds of similarities, namely relational similarity. For each relatedness measure, we present a relational similarity measure. The relational similarity [TL05, Tur05] corresponds to similarities between pairs of concepts in their underlying relations. For example, we say that the relation between **Lion** and **Mammal** is similar to the relation between **Frog** and **Amphibian**. Similarly, we say that the relation between **food** and **body** is similar to the relation between **fuel** and **car**. When two pairs of concepts have a strong relational similarity, we say that they are analogous. Measuring relational similarity of two pairs of concepts depends on how you measure relatedness of the two concepts of each pair. For instance, in the above two examples, the relation between Lion and Mammal is hierarchical whereas the relation between Food and body is non-hierarchical. Therefore, we use the path-based relatedness measure for generating questions about the first kind of relations and we use the set-based relatedness measure for generating questions about the second kind of relations. Both measures are presented below.

5.5.3.1 A new path-based relatedness measure

Consistent with the existing relatedness measures discussed above, our new path-based measure identifies a set of *semantically correct relation* patterns. We consider a pair of atomic concepts to be sufficiently related (i.e., have a semantically correct relation) if they have one of the hierarchal structures in Figure 5.7. Note that these structures are based on subsumption relationships only. These structures are sub-graphs of the inferred class hierarchy of an ontology $\mathcal{O}$. The adopted structures have at most one change in direction in the path connecting the two concepts and at most two steps in each direction. Other structures that have more steps and/or changes in direction were discarded to avoid generating questions with difficult-to-track relations, assuming that students' working memory has a limited capacity.

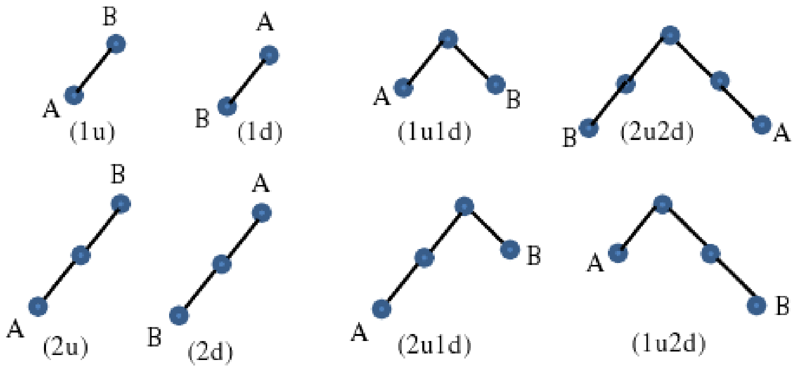


Figure 5.7: Valid relations structures (labels represent no. of steps and direction (up or down) in the path that connects concepts A & B in the inferred class hierarchy, starting from A)

Definition 5. *Let $\mathcal{O}$ be an ontology. Let A, B be concept names in $\widetilde{\mathcal{O}}$. Let $\pi(C, D)$ be the set of acyclic paths from A to B in the inferred class hierarchy of $\mathcal{O}$, Π be the set of valid paths according to Figure 5.7, then we define the path-based relatedness $Relatedness_{path-based}(A, B)$ between two concept names A and B as follows:*

$$Relatedness_{path-based}(A,B) = \begin{cases} 1 & \textit{if there is a path } p \in \pi(A,B) \cap \Pi; \\ 0 & \textit{otherwise.} \end{cases}$$

It remains to define a relational similarity measure for the path-based relatedness measure. Here, there are two sets of features that we are interested in comparing. On the one hand, given the paths (in the inferred class hierarchy) connecting two pairs of concepts, we want to know how many shared/unique *steps* the two pairs have. On the other hand, we want to know how many shared/unique *directions* the two pairs have in their paths. Similar to the way we defined conceptual similarity (i.e., based on the features-model), the path-based relational similarity can be defined as follows:

Definition 6. *Let C_1, C_2, D_1, D_2 be concepts in a concept language $\mathcal{L}$ such that $Relatedness_{path-based}(C_1, D_1) = 1$ and $Relatedness_{path-based}(C_2, D_2) = 1$, we define* the path-based relational similarity $RelSim_{\textit{path-based}} : \mathcal{L} \times \mathcal{L} \to [0, 1]$ *as:*

$$RelSim_{path-based}(C_1, D_1, C_2, D_2) = \frac{SS(P_1, P_2)}{TS(P_1, P_2)} \times \frac{SD(P_1, P_2)}{TD(P_1, P_2)}$$

where $SS(P_1, P_2)$ is the number of shared steps between the structures of the two pairs $P_1 = (C_1, D_1)$ and $P_2 = (C_2, D_2)$, $TS(P_1, P_2)$ is the number of total steps in the structures of the two pairs, $SD(P_1, P_2)$ is the number of shared directions between the structures of the two pairs and $TD(P_1, P_2)$ is the number of total directions in the structures of the two pairs.

5.5.3.2 A new set-based relatedness measure

The above path-based measure was a good start to generate relatedness-based questions that take into account hierarchal relations only. To generalise the approach, we need a relatedness measure that takes into account domain-specific relations (i.e., relations between some instances of a concept and some instances of the same or another concept). First, we represent, as a set, the relation $Rel(C, D, \mathcal{O})$ between two concepts C, D w.r.t. domain-specific relations, as follows:

$$Rel(C, D, \mathcal{O}) = \{r \in N_R \mid \mathcal{O} \models C \sqsubseteq \exists r.D\}$$

To quantify the relatedness of two concepts C and D, we take the cardinality of their set-based relation such that:

$$Relatedness_{set-based}(C, D, \mathcal{O}) = |Rel(C, D, \mathcal{O})|$$

To define a relational similarity measure based on the set-based relatedness measure, we want to know how many shared/unique *domain-specific relations* the two pairs of concepts have. Based on the features-model, the set-based relational similarity can be defined as follows:

Definition 7. *Let $\mathcal{O}$ be an ontology, C_1, C_2, D_1, D_2 be concepts in a concept language $\mathcal{L}$ such that $|Rel(C, D, \mathcal{O})|$ is defined and is a real number for all $C, D \in \mathcal{L}$, we set:*

1. the set of common relations between two pairs of concepts (C_1, D_1) and (C_2, D_2):
   $RelCom(C_1, D_1, C_2, D_2, \mathcal{O}) = Rel(C_1, D_1, \mathcal{O}) \cap Rel(C_2, D_2, \mathcal{O})$
2. the set of all relations between a pair of concepts (C_1, D_1) or another pair of concepts (C_2, D_2):
   $RelUnion(C_1, D_1, C_2, D_2, \mathcal{O}) = Rel(C_1, D_1, \mathcal{O}) \cup Rel(C_2, D_2, \mathcal{O})$
3. the set-based relational similarity between two pairs of concepts (C_1, D_1) and (C_2, D_2):
   $RelSim_{set-based}(C_1, D_1, C_2, D_2, \mathcal{O}) = \frac{|RelCom(C_1, D_1, C_2, D_2, \mathcal{O})|}{|RelUnion(C_1, D_1, C_2, D_2, \mathcal{O})|}$

5.6 Summary and directions

In this chapter, we have introduced a new family of similarity measures with a different computational cost for each individual measure. The intuition behind the new measures $SubSim(\cdot)$ and $GrammarSim(\cdot)$ is that considering more of the knowledge in the ontology (e.g., in our case more complex subsumers) enables us to calculate the similarity in a more precise way. However, it must be noted that this might lead to double counting some subsumers in some cases. For example, the measures count both an existential restriction and a number restriction that yields the same meaning (e.g., at least one). Other cases for double counted subsumers may exist. Thus, the newly introduced similarity measures must be refined in order to avoid such problems. An empirical investigation to test the new measures and examine whether such problems have a negative impact on similarity measurement is presented in the following chapter.

It is also interesting to develop and evaluate the weighted measures in order to examine their utility for applications that require measuring similarity w.r.t. contexts. There are many possible applications for context-based similarity measures, especially for large and heterogeneous ontologies. For example, given a heterogeneous ontology, a user might choose to give more weight to concepts belonging to a specific context and choose to (totally or partially) ignore concepts belonging to other contexts by giving them less weight.

Chapter 6

Evaluating similarity measures

Following on from our conceptual discussion on similarity measures in the previous chapter, this chapter studies the behaviour of some similarity measures in practice. Given a range of similarity measures with different costs, we want to know, on the one hand, how good an *expensive* measure is, its cost and the cases in which we are required to pay that cost to get a precise similarity measurement. On the other hand, we want to know how bad a *cheap* measure is and the specific problems associated with it. We also want to know how likely it is for a cheap measure to be a good approximation for more expensive measures. Although we have just seen (in Chapter 5) some cases where cheap measures are not (theoretically) approximations for expensive measures, we want to know how likely it is for such measures to be (close) approximations in practice. In addition, finding a cheap approximation for an expensive measure is interesting only if the expensive measure is shown to be good enough (and the approximation is close enough). We compare the new measures to human-based similarity judgements to confirm that the expensive measures can be more precise than the cheap ones.

In Chapter 5, we have reviewed some cheap existing measures that suffer from some problems. Such problems can negatively affect the application in which these measures are used. The degree of the negative impact depends on two aspects: (1) how rich the underlying ontology is and (2) the task/scenario in which the similarity measure is used. In some cases, depending on how simple the ontology/task is, using a computationally expensive "good" similarity measure is no better than using a cheap "bad" measure. Thus, we need to understand the computational cost of a given similarity measure and its strengths and weaknesses in different scenarios. Unfortunately, to date, there has been no thorough investigation of ontology-based similarity measures with respect to these issues.

For the purpose of studying the impact of the underlying ontology on the behaviour of the utilised similarity measures, we need an independently motivated corpus of ontologies which are actually used in practise. We make use of the NCBO BioPortal[1] corpus which contains over 300 ontologies that are used by the biomedical community which is a community that has a high interest in the similarity measurement problem [BSTP+10, BAB05, SDRL06, WDP+07]. However, note that it is difficult to classify ontologies according to how rich they are given that many factors contribute to the richness of an ontology.

To understand the major differences between similarity measures w.r.t. the task in which they are involved in, consider, for example, the following three tasks:

Given a concept C and some threshold Δ:

- Task 1: retrieve all atomic concepts D s.t. Similarity$(C, D) > 0$.
- Task 2: retrieve the N most similar atomic concepts to C.
- Task 3: retrieve all atomic concepts D s.t. Similarity$(C, D) > \Delta$.

We expect most similarity measures to behave similarly in Task 1 because we are not interested in the particular similarity values nor any particular ordering among the similar concepts. However, Task 2 gets harder as N gets smaller. In this case, a similarity measure that underestimates the similarity of some very similar concepts and overestimates the similarity of others can fail the task. In Task 3, the actual similarity values matter. Hence, using the most accurate similarity measure is essential.

The empirical evaluation of the new measures consists of two parts. In Experiment 1, we carry out a comparison between the new measures $GrammarSim(\cdot)$, $SubSim(\cdot)$ and $AtomicSim(\cdot)$ against (human) experts-based similarity judgments. In [PPPC07], IC-measures along with Rada et al. measure [RMBB89] have been compared against human judgements using the same data set which is used in the current study. Pedersen et al. [PPPC07] point out that the results of IC-measures highly depend on the external corpus they use and report that their results need to be confirmed through further studies involving the use of representative data. We excluded from the study any instance-based measure

[1]http://bioportal.bioontology.org/

since they require such a representative data. We also include another path-based measure which is Wu & Palmer measure [WP94]. In Experiment 2, we study in detail the behaviour of our new family of measures in practice. $GrammarSim(\cdot)$, with limited nesting, is considered as the expensive and most precise measure in this study. We use $AtomicSim(\cdot)$ as the cheap measure as it only considers atomic concepts as candidate subsumers. Studying this measure can allow us to understand the problems associated with taxonomy-based measures as they all consider atomic subsumers only. Recall that taxonomy-based measures suffer from other problems that were presented in Section 5.2.3. $AtomicSim(\cdot)$ can be considered the best candidate in its class since it does not suffer from these problems. We also consider $SubSim(\cdot)$ as a cheaper measure than $GrammarSim(\cdot)$ and more precise than $AtomicSim(\cdot)$ and we expect it to be a better "approximation" for $GrammarSim(\cdot)$, compared to $AtomicSim(\cdot)$. We also study the impact of adding a weight function to the similarity measure by examining $SubWSim(\cdot), GrammarWSim(\cdot)$ which were introduced in Chapter 5.

6.1 Experimental set-up

6.1.1 Infrastructure

With respect to hardware, the following machine has been used to carry out the experiments presented in this chapter:

Intel Quad-core i7 2.4GHz processor, 4 GB 1333 MHz DDR3 RAM, running Mac OS X 10.7.5 (Mac Pro late-2011 model).

As for the software, firstly, Oracle Java Runtime Environment (JRE) v1.6 is installed as the default JRE. Secondly, the OWL API v3.4.4 [HB09] is used. Thirdly, a range of freely available reasoners were utilised: FaCT++ [TH06], HermiT [SMH08], JFact [2], and Pellet [SPCG+07]. The need for more than one reasoner is justified by the fact that no available reasoner to date is capable of handling all existing ontologies. Therefore, to avoid runtime errors caused by reasoners, we used a stack of reasoners (used in the above order of appearance).

[2]http://jfact.sourceforge.net/

6.1.2 Test data

6.1.2.1 Experiment 1

In 1999, SNOMED-CT was jointly developed by the College of American Pathologists (CAP) and the National Health Service (NHS) in the UK. For the purposes of our comparison study, we use the 2010 version of SNOMED CT (Systematised Nomenclature of Medicine, Clinical Terms).[3] This ontology has been described as the most complete reference terminology in existence for the clinical environment [CCS+97]. It provides comprehensive coverage of diseases, clinical findings, therapies, body structures and procedures. In February 2014, the ontology has 397,924 classes. These are organised into 13 hierarchies. The ontology has the highest views amongst all BioPortal ontologies with over 13,600 views.

The reason for choosing this particular ontology is the availability of test data that shows the degree of similarity between some concepts from that ontology as rated by medical experts. Pedersen et al. [PPPC07] have introduced a publicly available test set for evaluating similarity measures in the biomedical domain. The test bed consists of 30 pairs of clinical terms. The similarity between each pair is rated by two groups of medical experts: physicians and coders. For details regarding the construction of this dataset, the reader is referred to [PPPC07]. We consider the average of physicians and coders similarity values in the comparison. We include in our study 19 pairs out of the 30 pairs after excluding pairs that have at least one concept that has been described as an ambiguous concept in the ontology (i.e., is assigned as a subclass of the concept ambiguous_concept) or not found in the ontology.

6.1.2.2 Experiment 2

The NCBO BioPortal library of biomedical ontologies has been used as a corpus for evaluating different ontology-related tools such as reasoners [KLK12], module extractors [DVKP+12], justification extractors [HPS12], to name a few. The corpus contains 438 user contributed OWL and OBO ontologies (as in April 2015) with varying characteristics such as axiom count, class count and expressivity. For example, the expressivity ranges from the inexpressive $\mathcal{AL}$ DL to the very expressive $\mathcal{SROIQ}$ DL. OBO ontologies were translated into OWL 2 and included in our study. The corpus is publicly available for download and targets

[3]http://www.ihtsdo.org/snomed-ct

the biomedical domain which has a noticeable interest in similarity measurement. The ontologies in the corpus are actively used in various applications. The corpus provides a history for each ontology showing all prior versions of the ontology. While some ontologies have many prior versions, some have only the original uploaded version. Some basic statistics about the ontologies included in our study are presented in Table 6.1.

	Class count	TBox size	RBox size
Mean	2,916	5,919	40
Median	507	924	4
Sum	565,661	1,148,241	7,717

Table 6.1: Some statistics about ontologies in our corpus

As for expressivity, Table 6.2 shows OWL 2 profiles across the corpus. Note that an ontology can belong to more than one category.

Profile	percentage of ontologies
DL	83%
EL	51%
QL	34%
RL	25%

Table 6.2: OWL 2 profiles across the corpus

A snapshot of the BioPortal corpus from November 2012 was used. It contains a total of 293 ontologies. For the purpose of our study, we excluded 86 ontologies which have only atomic subsumptions (including but not limited to RDFS files). The reason for this exclusion is that the behaviour of the considered similarity measures will be identical, i.e., we already know that $AtomicSim(\cdot)$ is good and cheap. We also excluded 5 more ontologies that have no classes and 33 ontologies that could not be processed by any reasoner due to run time errors. This has left us with a total of 169 ontologies.

Due to the large number of classes (565,661) and difficulty of spotting interesting patterns by eye, we calculated the pairwise similarity for a sample of classes from the corpus. The size of the sample is 1,843 classes with 99% confidence level. To ensure that the sample encompasses classes with different characteristics, we picked 14 classes from each ontology. The selection was not purely random. Instead, we picked 2 random classes and for each random class we picked some neighbour classes (i.e., 3 random siblings, a superclass, a subclass, a sibling of a direct superclass). This choice was made to allow us to examine the behaviour of the considered similarity measures even with special cases such as measuring similarity among direct siblings.

6.2 Experiment workflow

The general steps involved in the experiments presented in this chapter are summarised in Figure 6.1. In what follows, we describe the steps involved in more detail.

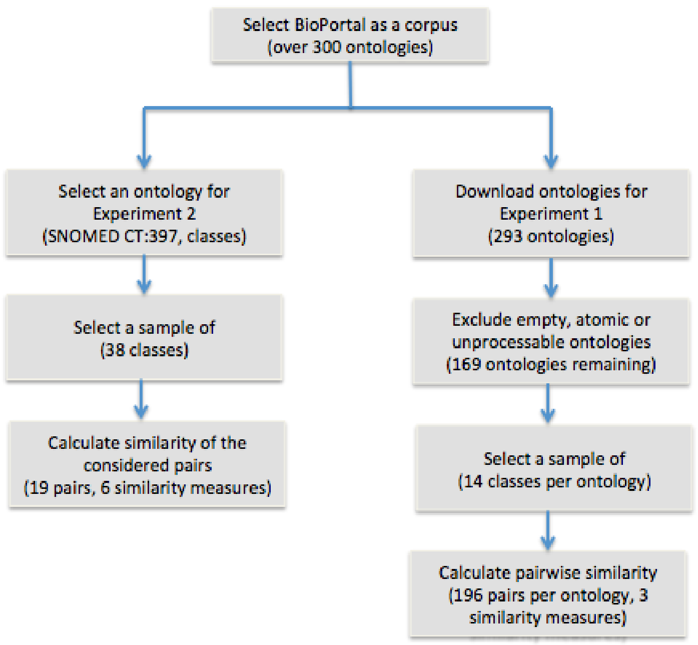

Select BioPortal as a corpus (over 300 ontologies)
Select an ontology for Experiment 2 (SNOMED CT:397, classes)
Select a sample of (38 classes)
Calculate similarity of the considered pairs (19 pairs, 6 similarity measures)
Download ontologies for Experiment 1 (293 ontologies)
Exclude empty, atomic or unprocessable ontologies (169 ontologies remaining)
Select a sample of (14 classes per ontology)
Calculate pairwise similarity (196 pairs per ontology, 3 similarity measures)

Figure 6.1: Steps involved in the evaluation experiments

6.2.1 Experiment 1

The similarity of 19 SNOMED CT concept pairs was calculated using the three measures along with Rada et al. [RMBB89] and Wu & Palmer [WP94] measures. We compare these similarities to human judgements taken from the Pedersen et al. [PPPC07] test set.

6.2.2 Experiment 2

Experiment 2 goes in steps as follows:

6.2.2.1 Consistency checking and classification

The first stage in the processing of each ontology is consistency checking and classification. We use the method precomputeInferences(InferenceType.CLASS HIERARCHY) provided by the OWL API to classify the ontology. This is a necessary step before selecting a sample of 14 classes with certain relations between them (e.g., superclasses, subclasses).

6.2.2.2 Sample selection

The second stage is to select two random classes from each ontology. For each random class, 6 "neighbour" classes are also selected as described above. If the total number of neighbour classes is less than 6, non-neighbour random classes are selected instead to keep the total number of selected classes to 14 classes for each ontology.

6.2.2.3 Module extraction

For optimisation, rather than working on the whole ontology, the next steps are performed on a $\bot$-module [CGHKS08] with the set of 14 classes as a seed signature. One of the important properties of $\bot$-modules is that they preserve almost all the seed signature's subsumers. There are 3 cases in which a $\bot$-module would miss some subsumers. The first case occurs when $\mathcal{O} \models C \sqsubseteq \forall s.X$ and $\mathcal{O} \models C \sqsubseteq \forall s.\bot$. The second case occurs when $\mathcal{O} \models C \sqsubseteq \forall s.X$ and $\mathcal{O} \models \forall s.X \equiv \top$. The third case occurs when $\mathcal{O} \models C \sqsubseteq \forall s.X$ and $\mathcal{O} \not\models C \sqsubseteq \exists s.X$. Since in all three cases $\forall s.X$ is a vacuous subsumer of C, we chose to ignore these, i.e., use $\bot$-modules without taking special measures to account for them.

6.2.2.4 Atomic concepts extraction

In this stage, all the atomic concepts in the $\bot$-module are extracted. These concepts will be considered as candidate subsumers for the three similarity measures $AtomicSim(\cdot)$, $SubSim(\cdot)$ and $GrammarSim(\cdot)$.

6.2.2.5 Subconcepts extraction

We recursively use the method getNestedClassExpressions() provided by the OWL API to extract all subconcepts from all axioms in the $\perp$-module. These concepts will be considered as candidate subsumers for the measures $SubSim(\cdot)$ and $GrammarSim(\cdot)$ only.

6.2.2.6 Grammar-concepts extraction

The subconcepts extracted in the previous step are used to generate grammar-based concepts. For practical reasons, we only generate concepts taking the form $\exists r.D$ s.t. $D \in Sub(\mathcal{O})$ and r a role name in the signature of the extracted $\perp$-module. Focusing on existential restrictions is justifiable by the fact that they are dominant in our corpus (77.89% of subconcepts) compared to other complex expression types (e.g., universal restrictions: 2.57%, complements: 0.14%, intersections: 13.89, unions: 2.05%). These concepts are used as candidate subsumers for $GrammarSim(\cdot)$ only.

6.2.2.7 Testing for subsumption entailments

For each class C_i in our sample and each candidate subsumer S_j, we test whether the ontology entails that $C_i \sqsubseteq S_j$. If the entailment holds, the candidate subsumer S_j is added to the set of C_i's subsumers. Note that S_j is already tagged as atomic, subconcept or grammar-based concept.

6.2.2.8 Calculating pairwise similarities

The similarity of each distinct pair in our sample (33,124 total pairs) is calculated using the three measures: $AtomicSim(\cdot), SubSim(\cdot)$ and $GrammarSim(\cdot)$.

6.2.2.9 Gathering stats along the way

For each ontology, the time required to process the ontology and calculate its pairwise similarities is recorded. In addition, we gather general properties of the ontology such as: OWL profile, DL expressivity, number of logical axioms, number of classes, number of object properties, number of individuals, length of the longest axiom, number of subconcepts.

We have shown in Chapter 5 that the above three measures are not approximations of each other. However, this might not be the case in practice as we will explore in the this experiment. To study the relation between the different measures in practice, we compute the following five properties: (1) order-preservation, (2) approximation from above (3) approximation from below, (4) correlation and (5) closeness. The time required to compute these properties is not included in the ontology processing time. Properties 1-3 are defined in Definition 4. For correlations, we calculate Pearson's coefficient for the relation between each pair of measures. Finally, two measures are considered close if the following property holds: $|Sim_1(C, D) - Sim_2(C, D)| \leq \Delta$ where $\Delta = 0.1$ in this experiment.

6.3 Results and discussion

6.3.1 Experiment 1

6.3.1.1 How good are the new measures?

Not surprisingly, *GrammarSim* and *SubSim* had the highest correlation values with experts' similarity judgements (Pearson's correlation coefficient for both $r = 0.87, p < 0.001$). Secondly comes *AtomicSim* with ($r = 0.86, p < 0.001$). Finally comes Wu & Palmer measure then Rada et al. measure with ($r = 0.81, p < 0.001$) and ($r = 0.64, p < 0.005$), respectively. Table 6.3 shows similarity values for the considered clinical terms using the different similarity measures. Note that all columns are normalised between, and including, 0 and 1 to allow for comparisons. The values in the expert's similarity column represent the average of physicians and coders similarity values taken from Pedersen et al. [PPPC07] data set and then normalised. As Table 6.3 shows, values in the columns representing *GrammarSim* and *SubSim* are very close (i.e., $|GrammarSim(x, y) - SubSim(x, y)| \leq 0.2$) and sometimes identical. This explains the fact that they have the same r values.

We also compare our calculated correlation values to the correlation values reported by Pedersen et al. [PPPC07]. They report that the best correlation value was 0.76 for a vector-based similarity measure that is ontology-independent (i.e., based on external data). Yet, this value is less than the correlation value calculated for the new measures. They also report that the ontology-dependent measure that was nearly as good as the vector-based measure was Lin's measure [Lin98] with correlation value of 0.69. We also compare our calculated correlation values for the path-based measure Rada et al. (0.64) with the correlation values reported for the same measure by Pedersen et al. (0.48). This difference is explained by the fact that we only consider 19 pairs out of the 30 pairs considered in the other study, for the reasons explained above. Note that Pedersen et al. [PPPC07] do not report the actual similarity values for each measure in their study; rather, they only report correlation values.

Term1	Term2	Exp.	Rad.	Wu.	Ato.	Sub	Gr.
Renal failure	Kidney failure	1.00	1.00	1.00	1.00	1.00	1.00
Heart	Myocardium	0.79	0.93	0.93	0.88	0.87	0.87
Abortion	Miscarriage	0.79	0.99	0.95	0.90	0.91	0.91
Delusion	Schizophrenia	0.65	0.80	0.30	0.18	0.14	0.13
Congestive heart failure	Pulmonary edema	0.55	0.71	0.58	0.41	0.39	0.40
Diarrhea	Stomach cramps	0.45	0.81	0.38	0.24	0.17	0.18
Mitral stenosis	Atrial fibrillation	0.45	0.72	0.68	0.51	0.51	0.51
Diabetes mellitus	Hypertension	0.38	0.64	0.36	0.22	0.16	0.17
Acne	Syringe	0.38	0.77	0.20	0.11	0.06	0.06
Antibiotic	Allergy	0.36	0.88	0.33	0.20	0.20	0.20
Cortisone	Total knee replacement	0.34	0.17	0.07	0.03	0.02	0.02
Cholangiocarcinoma	Colonoscopy	0.29	0.32	0.08	0.04	0.02	0.02
Lymphoid hyperplasia	Laryngeal cancer	0.29	0.48	0.28	0.16	0.12	0.13
Multiple sclerosis	Psychosis	0.25	0.86	0.44	0.29	0.14	0.13
Appendicitis	Osteoporosis	0.25	0.46	0.27	0.16	0.11	0.12
Rectal polyp	Aorta	0.25	0.00	0.05	0.03	0.01	0.01
Xerostomia	Alcoholic cirrhosis	0.25	0.52	0.40	0.25	0.22	0.22
Peptic ulcer disease	Myopia	0.25	0.77	0.47	0.30	0.25	0.27
Varicose vein	Entire knee meniscus	0.25	0.32	0.08	0.04	0.03	0.02

Table 6.3: Similarity between some clinical terms from SNOMED-CT

Clearly, the new expensive measures are more correlated with human judgements which is expected as they consider more of the information in the ontology. The differences in correlation values might seem to be small but this is expected as SNOMED is an $\mathcal{EL}$ ontology and we expect the differences to grow as the expressivity increases. Figure 6.2 shows the similarity curves for the 6 measures used in this comparison. Note that only circled points in these curves represent similarity values. But we have chosen to use continuous curves in order to make it easier to compare the curves w.r.t. over- and under-estimations. As we can see in the figure, the new measures along with Wu & Palmer measure mostly under-estimate human similarity whereas the Rada et al. measure mostly overestimates human similarity.

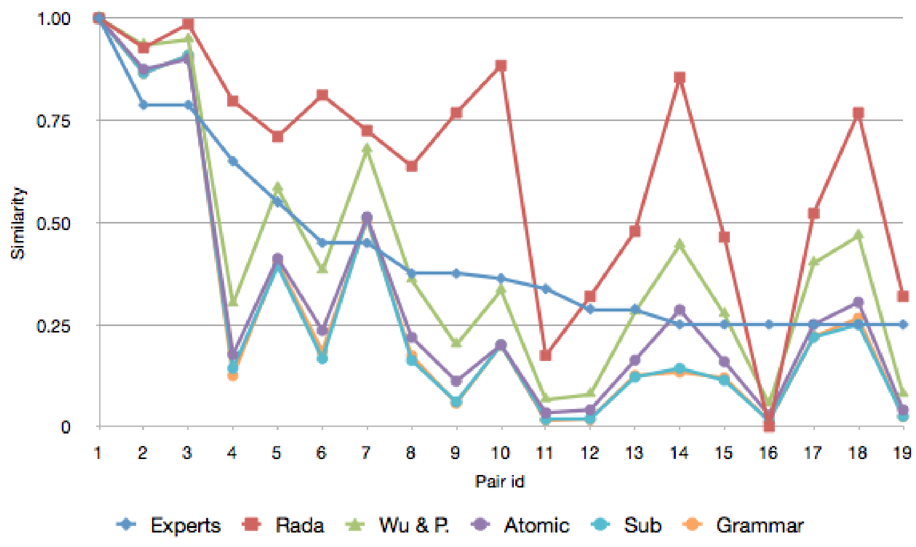


Figure 6.2: Curves of similarity between 19 clinical terms from SNOMED-CT using 6 similarity measures

6.3.2 Experiment 2

6.3.2.1 Cost of the expensive measure

One of the main issues we want to explore in this study is the cost (in terms of time) for similarity measurement in general and the cost of the most expensive similarity measure in particular. Note that the cost presented here is only for a small sample of 14 classes per ontology. Note also that the grammar used for the expensive measure was limited to existential restrictions only for practical reasons. Indeed, the cost of the expensive measure would be much higher without these limitations.

The average time per ontology taken to calculate grammar-based pairwise similarities was 2.3 minutes (standard deviation $\sigma = 10.6$ minutes, median $m = 0.9$ seconds) and the maximum time was 93 minutes for the Neglected Tropical Disease Ontology which is a $\mathcal{SRIQ}$ ontology with 1237 logical axioms, 252 classes and 99 object properties. For this ontology, the cost of $AtomicSim(\cdot)$ was only

15.545 seconds and 15.549 seconds for $SubSim(\cdot)$. 9 out of 196 ontologies took over 1 hour to compute $GrammarSim(\cdot)$. One thing to note about these ontologies is the high number of logical axioms and object properties. However, these are not necessary conditions for long processing times. For example, the Family Health History Ontology has 431 object properties and 1103 logical axioms and took less than 13 seconds to be processed. The average time per ontology taken to calculate $AtomicSim(\cdot)$ and $SubSim(\cdot)$ was 11.77 seconds and 11.79 seconds respectively. Although the average time required by $GrammarSim(\cdot)$ seems to be very high compared to the other two measures, in 89 ontologies out of 196 (45.41%) $GrammarSim(\cdot)$ was computed in less than 1 second.

Clearly, $GrammarSim(\cdot)$ can be far more costly than the other two measures. This is why we want to know how good/bad a cheaper measure can be.

6.3.2.2 Approximations and correlations

To study the five properties (order-preservation, approximation from above/below, correlation, closeness), we need to compare two measures at a time. We study the relation between $AtomicSim(\cdot)$ and $SubSim(\cdot)$ and refer to this as AS, the relation between $AtomicSim(\cdot)$ and $GrammarSim(\cdot)$ and refer to this as AG, the relation between $SubSim(\cdot)$ and $GrammarSim(\cdot)$ and refer to this as SG.

We want to find out how frequently can a cheap measure be a good approximation for (or have a strong correlation with) a more expensive measure. Recall that we have excluded all ontologies with only atomic subsumptions from the study. However, in 21 ontologies (12%), the three measures were perfectly correlated (Pearson's correlation coefficient $r = 1, p < 0.001$) mostly due to having only atomic subsumptions in the extracted module (except for three ontologies which have more than atomic subsumptions). In addition to these perfect correlations for all the three measures, in 11 more ontologies the relation SG was a perfect correlation (Pearson's correlation coefficient $r = 1, p < 0.001$) and the relations AS and AG were very high correlations (Pearson's correlation coefficient $r \geq 0.99, p < 0.001$). These perfect correlations indicate that, in some cases, the benefit of using an expensive measure is totally neglectable. Figure 6.3 shows the similarity curves for a perfect correlation case. Note that for presentation purposes, only part of the curve is shown. Note also that perfect correlations also means perfect order-preservations (100%), perfect approximations from above/-below (100%) and perfect closeness (100%).

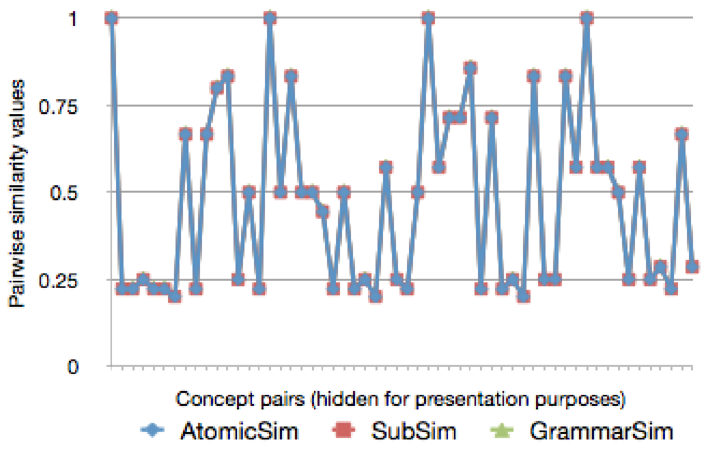


Figure 6.3: Three similarity curves coincide for perfect correlation in BioHealth ontological knowledge base

In about a fifth of the ontologies (21%), the relation SG shows a very high correlation ($1 > r \geq 0.99, p < 0.001$). Among these, 5 ontologies were 100% order-preserving and approximating from below. In this category, in 22 ontologies the relation SG was 100% close. As for the relation AG, in only 14 ontologies (8%) the correlation was very high.

In nearly half of the ontologies (49%), the correlation for SG was considered medium ($0.99 > r \geq 0.90, p < 0.001$). And in 19 ontologies (11%), the correlation for SG was considered low ($r < 0.90, p < 0.001$) with ($r = 0.63$) as the lowest correlation value. In comparison, the correlation for AG was considered medium in 64 ontologies (38%) and low in 55 ontologies (32.5%).

As for the properties (order-preservation, approximations from above/below and closeness) for the relations AG and SG, we summarise our findings in Table 6.4. Not surprisingly, $SubSim(\cdot)$ is more frequently a better approximation to $GrammarSim(\cdot)$, compared to $AtomicSim(\cdot)$.

Although one would expect that the properties of an ontology have an impact on the relation between the different measures used to compute the ontology's pairwise similarities, we found no indicators. With regard to this, we categorised the ontologies according to the degree of correlation (i.e., perfect, high, medium and low correlations) for the SG relation. For each category, we studied the following properties of the ontologies in that category: expressivity, number of logical axioms, number of classes, number of object properties, length of the

	AG	SG
Order-preservation	32	44
Approximations from below	32	49
Approximations from above	37	42
Closeness	28	56

Table 6.4: Ontologies satisfying properties of approximation

longest axiom, number of subconcepts. For ontologies in the perfect correlation category, the important factor was having a low number of subconcepts (≤ 1 except for 2 ontologies which had a count of 4 and 56 subconcepts). In this category, the length of the longest axiom was also low (≤ 11, compared to 53 which is the maximum length of the longest axiom in all the extracted modules from all ontologies). In addition, the expressivity of most ontologies in this category was $\mathcal{AL}$. Apart from this category, there were no obvious factors related to the other categories.

6.3.2.3 Are the weighted measures any better?

Given the weighted similarity measures $SubWSim(\cdot)$ and $GrammarWSim(\cdot)$, we want to know whether they are any better than the non-weighted measures. $SubWSim(\cdot)$ and $GrammarWSim(\cdot)$ preserved the order of $GrammarSim(\cdot)$ in 33 and 39 ontologies (out of 196), respectively, compared to 44 ontologies using $SubSim(\cdot)$. Also, $SubWSim(\cdot)$ and $GrammarWSim(\cdot)$ were close to $GrammarSim(\cdot)$ in 25% and 57% of the ontologies, respectively, compared to 33% of the ontologies using $SubSim(\cdot)$.

We also compared the weighted measures with human judgements of similarity and both weighted measures had a correlation value similar to their non-weighted version (i.e., Pearson's correlation coefficient $r = 0.87, p < 0.001$). These results indicate that the weighted measures are not necessarily better than the non-weighted measures.

6.3.2.4 How bad is a cheap measure?

To explore how likely it is for a cheap measure to encounter problems (e.g., fail Tasks 1-3 presented earlier in this chapter), we examine the cases in which a cheap measure was not an approximation for the expensive measure. *AG* and *SG* were not order-preserving in 80% and 73% of the ontologies respectively. Also, they were not approximations from above nor from below in 72% and 64% of the ontologies and were not close in 83% and 66% of the ontologies, respectively.

If we take a closer look at the African Traditional Medicine ontology for which the similarity curves are presented in Figure 6.4, we find that the SG is 100% order-preserving while AG is only 99% order-preserving. Both relations were 100% approximations from below. As for closeness, SG was 100% close while AG was only 12% close. In order to determine how bad are $AtomicSim(\cdot)$ and $SubSim(\cdot)$ as cheap approximations for $GrammarSim(\cdot)$, we study the behaviour of these measures w.r.t. Tasks 1-3.

Both cheap measures would succeed in performing Task 1 on the African Traditional Medicine ontology. However, only $SubSim(\cdot)$ can succeed in Task 2 (with 1% failure chance for $AtomicSim(\cdot)$). For Task 3, there is a higher failure chance for $AtomicSim(\cdot)$ since closeness is very low (12%).

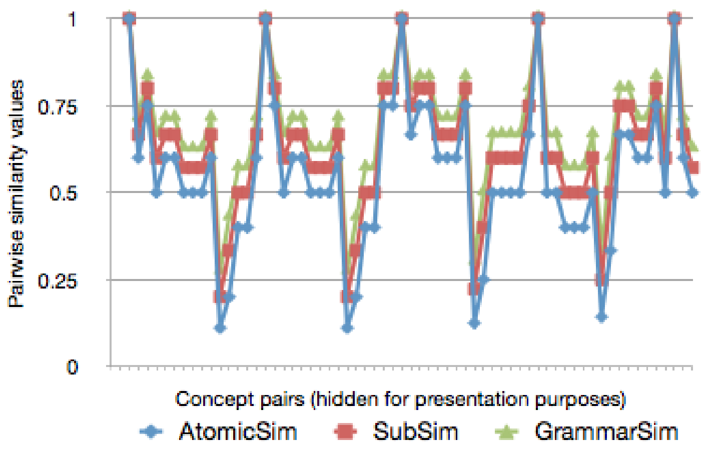


Figure 6.4: African Traditional Medicine ontology

As another example, we examine the Platynereis Stage Ontology for which the similarity curves are presented in Figure 6.5. In this ontology, both AG and SG are 75% order-preserving. However, AG was 100% approximating from above while SG was 85% approximating from below (note the highlighted red spots). In this case, both $AtomicSim(\cdot)$ and $SubSim(\cdot)$ can succeed in Task 1 but not always in Tasks 2 & 3 with $SubSim(\cdot)$ being worse as it can be overestimating in some cases and underestimating in other cases.

In general, both $AtomicSim(\cdot)$ and $SubSim(\cdot)$ are good cheap alternatives w.r.t. Task 1. However, $AtomicSim(\cdot)$ would fail more often than $SubSim(\cdot)$

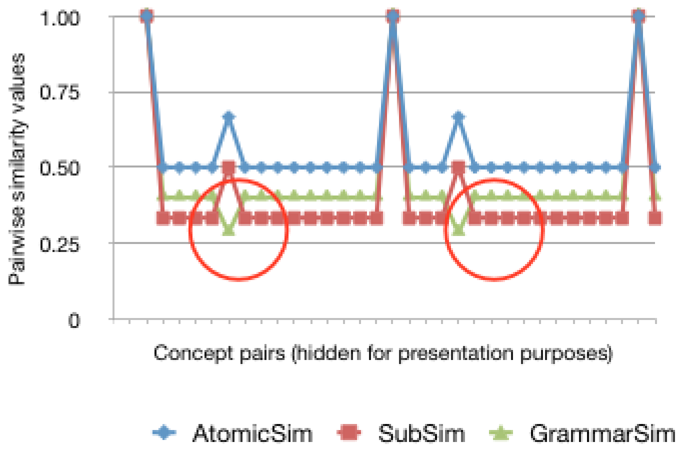


Figure 6.5: Platynereis Stage Ontology

when performing Tasks 2 or 3.

6.4 Threats to validity

6.4.1 Threats to internal validity

For practical reasons and due to the high runtime of the similarity measurement process, we had to restrict our analysis to a relatively small sample of classes per ontology in Experiment 2. Although the sample is statistically significant in terms of size, it could not be selected in a pure random mechanism. Rather than selecting 14 random classes, we selected 2 random classes and 6 neighbour classes for each random class. This design option was necessary for understanding the behaviour of similarity measures. Note that non-neighbour classes tend to have low similarity values. Including a lot of the non-neighbour classes in our sample could, for example, cause unwanted high percentages for order-preservation.

In addition, relying on only one ontology (i.e., SNOMED-CT) for comparing the new measures and some existing measures against human judgements might limit the generalisability of the results of Experiment 1. Rather than dealing with these results as confirmatory, they should be treated as preliminary indicators.

6.4.2 Threats to external validity

Although ontologies in the BioPortal corpus may not be representative for all available ontologies, it does contain a wide range of ontologies with different properties (e.g., size and expressivity). Moreover, it is built and maintained by a community that has a noticeable interest in the similarity measurement problem. Therefore, it is reasonable to adopt this corpus for testing services that would be provided for its community.

6.5 Summary and conclusions

To sum up, no obvious indicators were found to inform the decision of choosing between a cheap or expensive measure based on the properties of an ontology. However, the task under consideration and the error rate allowed in the intended application can help. In general, we find the similarity measure $SubSim(\cdot)$ to be a good candidate for QG purpose. It is a cheap measure, i.e., can be computed much faster than $GrammarSim(\cdot)$), and it performed equally well as we have seen in Experiment 1.

Chapter 7

Evaluating Ontology-based MCQ generation

So far, we have presented a similarity-based approach to generate MCQs of varied difficulties from ontologies. We have also presented a new family of similarity measures to support the generation of such questions. It remains to evaluate the presented QG approach in terms of: (1) how useful the generated MCQs are?, (2) how successful is the approach in controlling the difficulty of the generated MCQs? and (3) how easy/hard is it to adopt the approach by a test developer with no prior experience in ontologies? To answer these questions, we have conducted three studies.

The first study consists of two parts: (1) an expert-centred study to evaluate the appropriateness and usefulness of the generated questions and (2) a student-centred study to analyse the different properties of the questions, including their difficulty and quality of distractors. In the first study, we explore the case of generating MCQs from handcrafted ontologies. This can give us a greater insight into how the QG approach performs. As the utilised ontologies were developed in house, this allows us, for example, to tell, given two distractors, which one is more similar to the key. Hence, we would easily tell whether one distractor is more suitable than the other to generate an easy/difficult question. Using a handcrafted ontology also makes it easier to find teacher/student participants because you can tailor the handcrafted ontology to their needs. Otherwise, i.e., if an existing ontology is used, you would need to find a suitable group of students who are enrolled in a course for which there is an ontology that models the course content or at least covers part of it.

The second study is an expert-centred study which has been conducted with two goals in mind: (1) assessing an improved protocol to evaluate the appropriateness and usefulness of the generated questions and (2) assessing the challenges that might be faced by a test developer with no prior experience in ontologies when adopting an ontology-based QG tool. In this study, we report on the experience of an interested test developer to develop an ontology for QG purposes. After building the ontology and generating questions from it, we ask the interested test developer along with two more experts to evaluate the appropriateness, usefulness and difficulty of the generated questions.

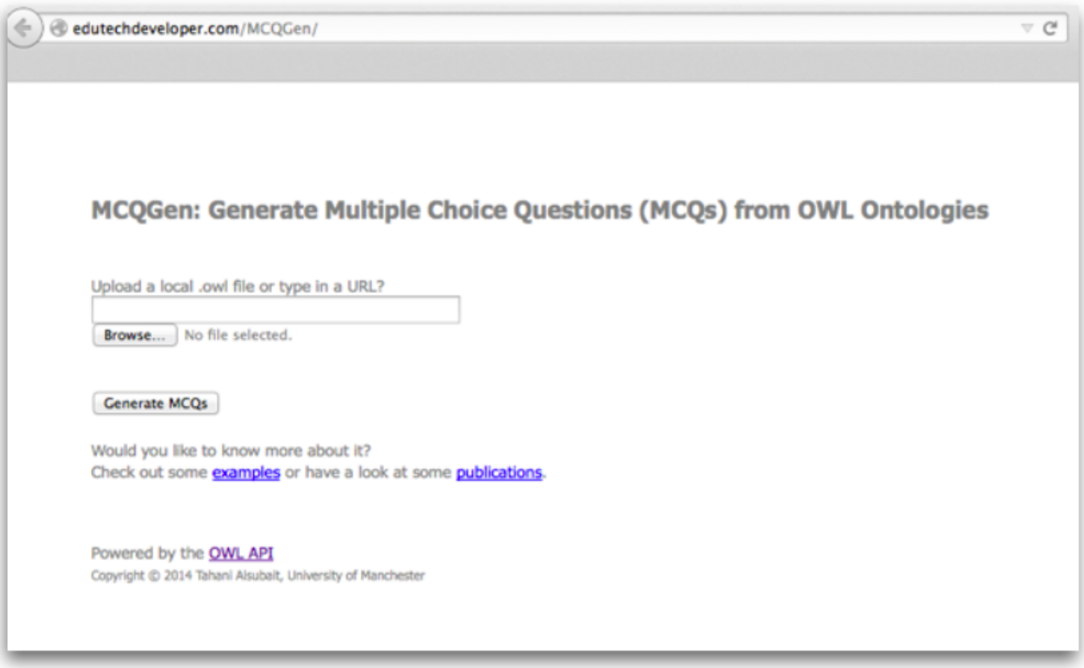
We structure the description of the first and second studies around the following phases:

1. The ontology development phase
2. The question generation phase.
3. The question evaluation phase.

The number of questions that can be evaluated by human participants (whether experts or students) is limited. To evaluate questions without human participants, we show in the third study that it is possible to utilise an automated solver which plays the role of *one* student attempting to answer a set of MCQs of varying difficulties. Given a large pool of questions, it is expected that a student would answer a higher number of easy questions compared to the number of difficult questions that this student can get right. Three existing ontologies have been used in this study. The results of this study show that, first, the general performance of the automated solver when answering questions about different subjects correlates with its (estimated) knowledge about those subjects. Second, the difficulty of the generated questions correlates with the degree of similarity between the key and distractors. The lesson to be learned from this study is that automatic mechanisms to evaluate a *large* number of questions can alleviate the burden of manual evaluation. Especially given that some existing ontologies can contain a large number of terms; hence a large number of questions can be generated from them, far more than what can be evaluated by a human expert.

7.1 Implementation

To evaluate the proposed QG approach, we have implemented a prototype QG application that allows to: (1) compute the pairwise similarity of (possibly complex) concepts from an ontology and (2) use these similarity values to generate MCQs of varied difficulties. The prototype which comes in two versions (a local version and a web-based version[1]) was developed using Java Runtime Environment JRE v1.6, the OWL API v3.4.4 [HB09] and FaCT++ reasoner [TH06]. The web-based application is shown in Figure 7.1 and it is hosted on a server running Linux v2.6.18, Apache Tomcat v6.0.36, JRE v1.6.0 and MySQL v5.5.42. The web-based application has limited memory resources (96MB JVM Heap Size) compared to the local version (3GB JVM Heap Size) of the application which has been used in the experiments presented in this chapter. Due to the very high memory requirements that are usually needed to process large ontologies by existing reasoners, only small and easy to process ontologies can be used with the web-based version.


edutechdeveloper.com/MCQGen/

MCQGen: Generate Multiple Choice Questions (MCQs) from OWL Ontologies

Upload a local .owl file or type in a URL?

Browse... No file selected.

Generate MCQs

Would you like to know more about it?
Check out some examples or have a look at some publications.

Powered by the OWL API
Copyright © 2014 Tahani Alsubait, University of Manchester

Figure 7.1: The web-based QG application

To generate MCQs from a given ontology, the following steps are followed:

[1]It can be accessed at http://edutechdeveloper.com/MCQGen/

1. Selecting the source ontology by either selecting a local file or providing a URL, see Figure 7.1.

2. Selecting the similarity measure(s) that will be used to calculate the pairwise similarity of all concept names and possibly sub-concept expressions in the selected ontology, see Figure 7.2. The more similarity measures selected, the more time will be required to process the ontology. After calculating the pairwise similarity, the average similarity between all siblings[2] in the selected ontology is calculated for all selected similarity measures.

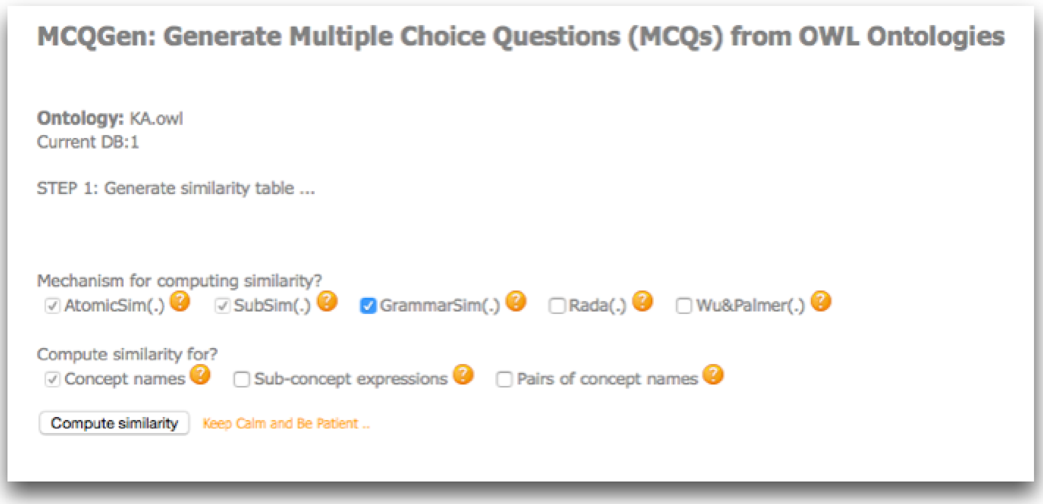
3. Selecting question template(s) that will be used to generate the questions. In Table 4.1, we have presented a few templates for generating basic questions about the domain of interest. We have chosen to focus on such questions due to their natural fit to the source (i.e., ontologies).

4. Generating all possible easy and difficult questions for each selected template. The generated questions have one stem, one key, a difficulty level and at least one distractor that is suitable for this difficulty level. Out of these questions, many sub questions of varied number of distractors can be constructed. In the following experiments, we randomly choose 3 distractors for each question. We report on the number of generated questions before and after selecting 3 distractors. These numbers are not necessarily the same due to the possibility of generating questions with less than 3 distractors. Questions with less (or more) than 3 distractors are used in the experiments presented in Chapter 8.

5. Specifying the difficulty of questions according to Hypothesis 1 presented in Chapter 3 by using the selected similarity measure(s). Any similarity measure can be used to generate questions of the required difficulty, however $GrammarSim(\cdot)$ (see Chapter 5) is used to generate questions which use concept expressions as answers, e.g., questions for the "Generalisation 2" category in Table 4.1. This is motivated by the fact that, when the answers are expressed in detail (e.g., concept expressions rather than simply concept names), the similarity measurement should be more precise. It remains to specify the upper (Δ_2) and lower (Δ_1) bounds of similarity values which are used to generate appropriate distractors for the required difficulty level.

[2]Siblings are concept names that have the same most specific atomic subsumer.

Rather than specifying a random number, we choose to use the average similarity value between all siblings in the ontology. This average similarity value is then used as a lower bound for generating a difficult distractor, where 1 is the upper bound. Distractors that are equivalent to the key are excluded. Hence, the upper bound is actually just less than 1. The lower bound to generate an easy distractor is set to be two thirds of the lower bound of difficult distractors. And, the upper bound to generate an easy distractor is, and excluding, the lower bound for difficult questions.



MCQGen: Generate Multiple Choice Questions (MCQs) from OWL Ontologies

Ontology: KA.owl
Current DB:1

STEP 1: Generate similarity table ...

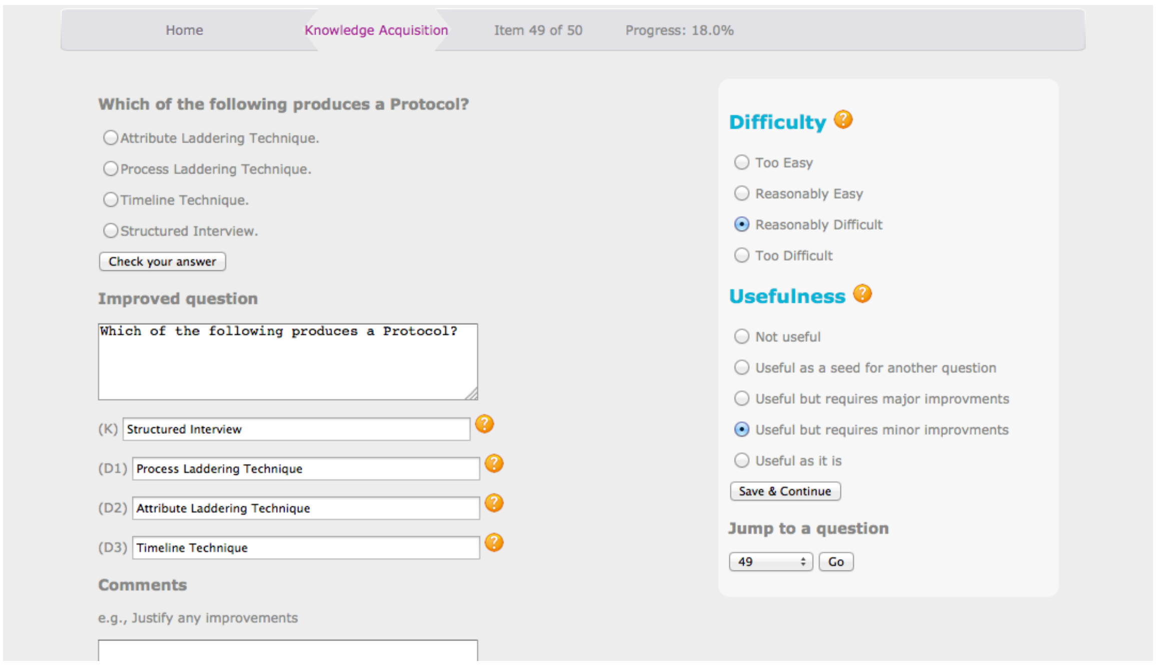
Mechanism for computing similarity?
☑ AtomicSim(.) ☑ SubSim(.) ☑ GrammarSim(.) ☐ Rada(.) ☐ Wu&Palmer(.)

Compute similarity for?
☑ Concept names ☐ Sub-concept expressions ☐ Pairs of concept names

Figure 7.2: Similarity measures available in the QG application

In addition to the prototype QG application, we have also implemented 2 versions of a web-based application that allows a group of domain experts to review a set of questions. The first version has been used in the first evaluation study presented below and the second version has been used in the second evaluation study. A screenshot of the first version is presented in Figure 7.3.

To accommodate the feedback gathered from the reviewers who participated in the first evaluation study, an improved version of the reviewing application has been built. In the first version of the application, the reviewers are asked to edit the question under review, in case they think it needs to, and then provide a free-response comment to justify their suggested modification. In the second version, for a deeper engagement, we also ask the reviewers to check/uncheck a list of 5 options. Each option presents a rule for constructing a good MCQ and asks the reviewer to indicate whether the question under review adheres to this rule. These rules were gathered from the qualitative analysis of reviewer comments



Which of the following produces a Protocol?

- ○ Attribute Laddering Technique.
- ○ Process Laddering Technique.
- ○ Timeline Technique.
- ○ Structured Interview.

Improved question

Which of the following produces a Protocol?

(K)	Structured Interview
(D1)	Process Laddering Technique
(D2)	Attribute Laddering Technique
(D3)	Timeline Technique

Comments

e.g., Justify any improvements

Difficulty

- ○ Too Easy
- ○ Reasonably Easy
- [✓ Reasonably Difficult]
- ○ Too Difficult

Usefulness

- ○ Not useful
- ○ Useful as a seed for another question
- ○ Useful but requires major improvments
- [✓ Useful but requires minor improvments]
- ○ Useful as it is

Jump to a question

49

Figure 7.3: The first version of the reviewing web-interface

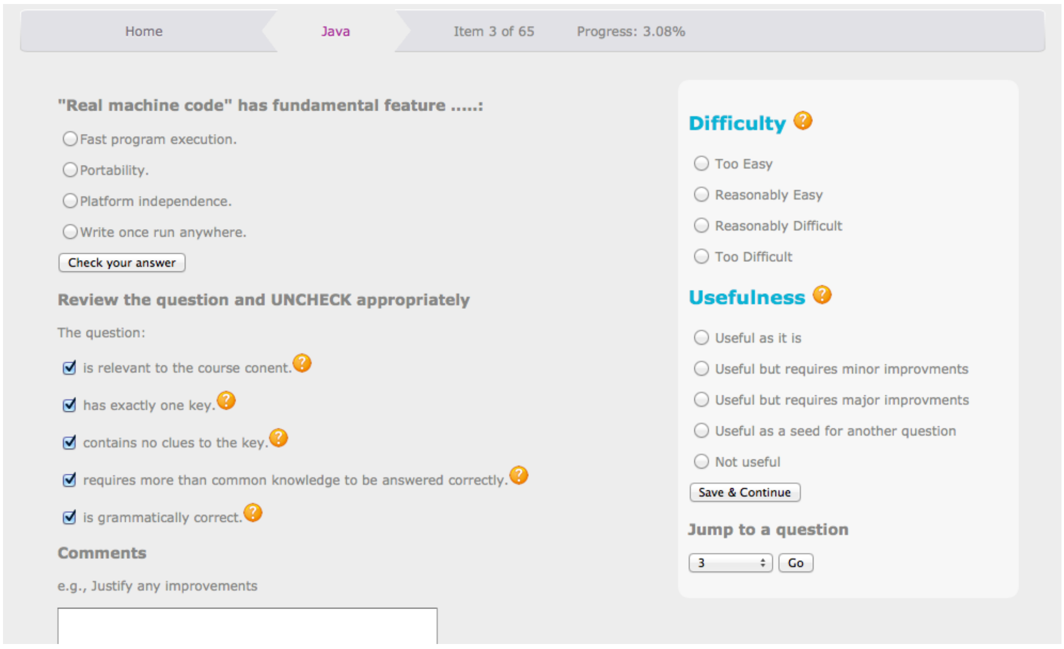
in the first evaluation study. A screenshot of the second version is presented in Figure 7.4. Details on the steps involved in using the reviewing application are presented in Section 7.2.2.4 , for the first version, and in Section 7.3.2.4, for the second version.

7.2 First evaluation study: generating questions from handcrafted ontologies

The importance of this study is that it helps to understand the overall effort required to build an ontology for a given course, if a relevant ontology is not available. To approach the problem of finding a suitable group of students willing to participate in the study, we have chosen to target a group of students and handcraft an ontology for the course they are enrolled in. The two versions are hosted in the same web server described above.

7.2.1 Goals

The main goal of this study is to answer the following questions:


Home | Java | Item 3 of 65 | Progress: 3.08%

"Real machine code" has fundamental feature .....:

○ Fast program execution.
○ Portability.
○ Platform independence.
○ Write once run anywhere.

Check your answer

Review the question and UNCHECK appropriately

The question:

☑ is relevant to the course conent.
☑ has exactly one key.
☑ contains no clues to the key.
☑ requires more than common knowledge to be answered correctly.
☑ is grammatically correct.

Comments

e.g., Justify any improvements

Difficulty

○ Too Easy
○ Reasonably Easy
○ Reasonably Difficult
○ Too Difficult

Usefulness

○ Useful as it is
○ Useful but requires minor improvments
○ Useful but requires major improvments
○ Useful as a seed for another question
○ Not useful

Save & Continue

Jump to a question

3 Go

Figure 7.4: The second version of the reviewing web-interface

1. Can we control the *difficulty* of MCQs by varying the similarity between the key and distractors?

2. Can we generate a reasonable number of educationally *useful* questions?

3. How *costly* is ontology-based question generation, including the cost of developing/enhancing an ontology, computation cost and post-editing cost?

7.2.2 Materials and methods

7.2.2.1 Equipment description

The reader is referred to Section 6.1.1 for a description of hardware and software used in this experiment.

7.2.2.2 Building the ontology

The Knowledge Representation and Reasoning course (COMP34512) is a third year course unit offered by The School of Computer Science at The University of Manchester. It covers various Knowledge Representation (KR) topics including Knowledge Acquisition (KA) techniques and KR formalisms. For the purposes of the study described in this section, a **Knowledge Acquisition (KA) ontology** (which models the KA part of the course) was developed from scratch.[3] This particular part of the course unit was chosen as it contains mainly declarative knowledge. Other parts of the course can be described as procedural knowledge and skills which are not suitable to be modelled in an OWL ontology. Assessing student's understanding of declarative knowledge is an essential part of various tests. Figure 7.5 shows a mind map for the overall course content and the area covered in the KA ontology (the shaded part).

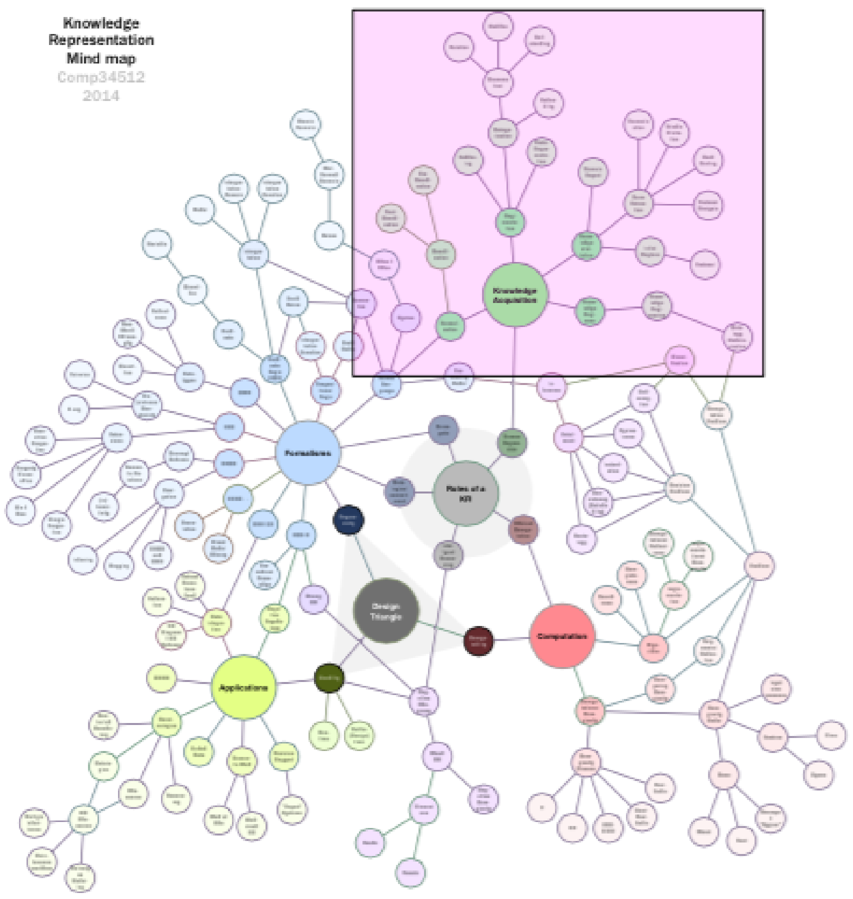


Figure 7.5: Mind map for the Knowledge Representation and Reasoning course

A total of 9 hours were spent by the ontology developer[4] to build the first version of the ontology, excluding the time required to skim through the contents of the course materials since the ontology was not developed by the instructor who is in charge of the course unit. The *Protégé* 4 ontology editor was used for building the ontology.

[3]It can be accessed at: http://edutechdeveloper.com/MCQGen/examples/KA.owl
[4]The author of this thesis

Several refinements were applied to the ontology after presenting the ontology to an experienced knowledge modeller[5] and getting useful feedback from her. The feedback session took around 2 hours and refinements took around 3 hours to be applied. The main refinements are described in the following points along with real examples from the KA ontology. These points are presented here as they can be *generally* useful when building ontologies.

1. The following points can help to enrich an ontology under development:

   (a) Model the *positives* as well as (the known) *negatives*. For example, if it is known that some properties hold for a certain KA technique and that some properties do not hold, then the ontology developer should add to the ontology both the properties and the non-properties of this technique and use negations when necessary.

   (b) Model both *tacit knowledge* and *explicit knowledge*. It is indeed harder to elicit tacit, i.e., implicit, knowledge from domain experts but once elicited, it should be fairly easy to model declarative tacit knowledge. For example, the concept of KnowledgeElicitation should be explicitly stated to involve the participation of a KnowledgeExpert and a KnowledgeModeller. This may be obvious to an expert in knowledge elicitation but may be less obvious to others.

   (c) Specify disjoint classes when possible (e.g., TacitKnowledge and ExplicitKnowledge).

   (d) If applicable, state that a property is transitive (e.g., involves) and declare any necessary property chains (e.g., involves o produces = produces).

2. The following points can help in making the ontology more *complete* and *specific*:

   (a) It is strongly advised to *close properties*. For example, if it is known that $Technique_X$ is good at, and only at, *Flexibility* (i.e., it is not good at other quality aspects), then the ontology should state that $Technique_X \sqsubseteq \exists isGoodAt.Flexibility \sqcap \forall isGoodAt.Flexibility$. This can help in getting more precise entailments from the ontology.

[5]One of the supervisors of this thesis

   (b) As OWL does not allow us to say that a property is the complement of another property, it is advised to declare properties in a positive form (e.g., $isGoodAt$ rather than $isNotGoodAt$) and use negation on class level (e.g., $Technique_Y \sqsubseteq \neg\exists isGoodAt.Flexibility$).

   (c) Be specific, e.g., introduce sub properties such as $produces$ of $isAssociatedWith$.

3. The following points can help in making the ontology more *organised*:

   (a) Stick to a *naming scheme*. For example, use either Camel Case or underscores to denote individual words in the name of a class/property. Also use either singular form (e.g., uses) or plural form (e.g., use) but avoid mixing the two forms in one ontology. This can help in rendering the generated questions.

   (b) Regardless of the similarity measure used, it is advised to use complex subsumptions involving conjunctions rather than separate atomic subsumptions (e.g., a definition like $Hammer \sqsubseteq Tool \sqcap \exists isUsedFor.Thing$ is better for QG than $Hammer \sqsubseteq Tool, Hammer \sqsubseteq \exists isUsedFor.Thing$ because in the first case the full definition of $Hammer$ is provided in one subconcept expression which can be used as a candidate stem/answer for QG).

4. The following point can help to debug an ontology and ensure that it is free of *obvious errors*:

   (a) Always use a reasoner to classify the ontology and look at the inferred class hierarchy to trace any logical errors. For example, if we add the following two axioms to an ontology: $\exists isGoodAt.TacitKnowledge \sqsubseteq Requirement$ and $Interviews \sqsubseteq \exists isGoodAt.TacitKnowledge$, then, the ontology will entail that: $Interviews \sqsubseteq Requirement$ which is clearly wrong but can be easily diagnosed by looking at the inferred class hierarchy.

The important advantage of considering the above points is that they can highly improve similarity measurement between the different concepts of the ontology and help to generate better questions. The resulting ontology, after applying these refinements, is an $\mathcal{SI}$ ontology consisting of 504 axioms. Among these are 254 logical axioms. Class and object property counts are 151 and 7, respectively, with one inverse and one transitive object property.

7.2.2.3 Generating questions

A variety of memory-recall questions (i.e., lowest Bloom level) have been generated which we describe below. Questions that require higher cognitive skills (e.g., reasoning) will be examined later in Section 7.4.

A total of 913 questions have been generated from the KA ontology described above. Among these questions are 633 easy questions and 280 difficult questions. These questions consist of one stem, one key and all possible distractors. A large number of questions can be generated out of the 913 questions by selecting different combinations of distractors. We choose to construct one question only for each generated stem. Only 535 questions out of the 913 questions have at least 3 distractors (with 474 easy questions and 82 difficult questions). Out of these, we randomly selected 50 questions for further evaluation by 3 reviewers. Those 50 questions are presented in Table B.1 in Appendix B. The 50 questions contain 5 easy and 5 difficult questions from the 6 different question categories which are described in Table 4.1. The number of optimal distractors for MCQs remains debatable [HD93]. We choose to randomly select 3 distractors for each question. The number of questions generated for each category is presented in Table 7.1. Refer to Table 4.1 for a detailed description of these question categories. $SubSim(\cdot)$ (see Chapter 5) has been used to generate all the questions described in this table with the exception of using $GrammarSim(\cdot)$ to generate questions in the "Generalisation 2" category (which is the only category that uses concept expressions as answers).

Category	Total	Total-easy	Total-difficult	3+dist	3+dist-easy	3+dist-difficult
Definition	27	17	10	21	16	5
Recognition	94	94	0	75	75	0
Generalisation	133	77	56	101	71	30
Generalisation 2	259	162	97	133	106	27
Specification	55	54	1	41	41	0
Specification 2	345	229	116	185	165	20

Table 7.1: The number of generated questions from KA ontology according to 6 different categories

As Table 7.1 shows, in all categories, the number of *easy* questions is higher than the number of difficult questions. This is explained by the fact that fewer distractors fit the criteria required to generate difficult questions compared to easy questions. It is generally harder to find distractors that are very similar to the key compared to finding distractors that are less similar.

7.2.2.4 Reviewing questions

Three reviewers involved in leading the course have been asked to evaluate the 50 randomly selected questions using the web interface presented in Figure 7.3. For each question, the reviewer first attempts to solve the questions and then specifies whether they think that the question is (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. Then, the reviewer predicts the difficulty of the question. To distinguish between acceptable and extreme levels of difficulty, we ask the reviewers to choose one of the following options for each question: (1) too easy, (2) reasonably easy, (3) reasonably difficult and (4) too difficult. In what follows, we refer to the reviewers by their job completion time. Hence, “first reviewer” refers to the reviewer who first finished the reviewing process.

7.2.2.5 Administering questions

Two samples of the questions which have been rated by the reviewers as useful (or useful with minor improvements) by at least 2 reviewers have been administered to third year students[6] who are enrolled in the course unit for the academic year 2013/14 and who were about to sit the final exam. The two sets of questions have been administered in two different rounds to increase participation rate and allow for comparisons between the two rounds which have been set up differently (e.g., the first is a closed-book test while the second is open-book). In the first round, a total of 6 questions (3 easy, 3 difficult) have been administered to 19 students using paper-and-pencils during a revision session at the end of the course. Refer to Table B.1 for a list of these 6 questions (Question IDs: 4, 7, 18, 23, 36 and 38). The students had 10 minutes to answer the 6 questions. In the second round, another set of 6 questions (3 easy, 3 difficult) have been administered to students via the university’s eLearning environment, BlackBoard, one week before the final exam and the students were allowed to answer the questions at any time during

[6]This study has been approved by the ethics committee in the School of Computer Science, The University of Manchester (approval number: CS125).

this week. These questions are presented in Table B.1 with IDs: 1 , 3, 16 , 20, 39 and 50. Only 7 students have participated in the second round.

7.2.3 Results and discussion

Overall cost. As mentioned earlier, the cost of QG might, in some cases, include any costs of developing a new ontology or reviewing/editing an existing one. For the current experiment, we experienced the extreme case of having to build an ontology from scratch for the purpose of QG. A total of 14 hours (spread over multiple days) were spent to develop the ontology described above. For computation time, we need to consider both the time required to compute pairwise similarity for the underlying ontology and the time required to compute the questions. Computing pairwise similarity for all sub-concept expressions (including concept names) in the KA ontology took 22 minutes. This includes the time required to compute similarities using both $SubSim(\cdot)$ and $GrammarSim(\cdot)$ for a total of 296 sub-concept expressions. Computing a total of 913 questions took around 21 minutes. Computing "which is the odd one out?" questions took 17 minutes out of the 21 minutes while computing all other questions took less than 4 minutes. So in total, around 1 hour was needed for computation.

Finally, we also have to consider any time required to review the questions (possibly including post-editing time). As the reviewers were allowed to review each item in an unrestricted manner, it is difficult to determine the exact time that each reviewer has spent on each item. For example, for a set of questions, a reviewer might start looking at a question on a given day and then submit the review on the next day after getting interrupted for any reason. We exclude questions for which the recorded time was more than 60 minutes as this clearly shows that the reviewer was interrupted in the middle of the reviewing process. The first reviewer spent between 13 and 837 seconds to review each of the 50 questions, including time for providing suggestions to improve the questions. The second reviewer spent between 12 and 367 seconds. And the third reviewer spent between 17 and 917 seconds. Note that these times include the time required to attempt to answer the question by the reviewer.

Usefulness of questions. A question is considered "useful" if it is rated as either "useful as it is" or "useful but requires minor improvements" by a reviewer. 46 out of 50 questions were considered useful by at least one reviewer. 17 out of the 46 questions were considered useful by at least 2 reviewers. This is illustrated in

Figure 7.6. The first reviewer rated 37 questions as being useful while the second and third reviewer rated 8 and 33 questions as useful, respectively. Note that the third reviewer is the main instructor of the course unit during the academic year in which the experiment has been conducted while the second reviewer taught the course unit in the previous year. The first reviewer has not taught this course unit before, but has general knowledge of the content.

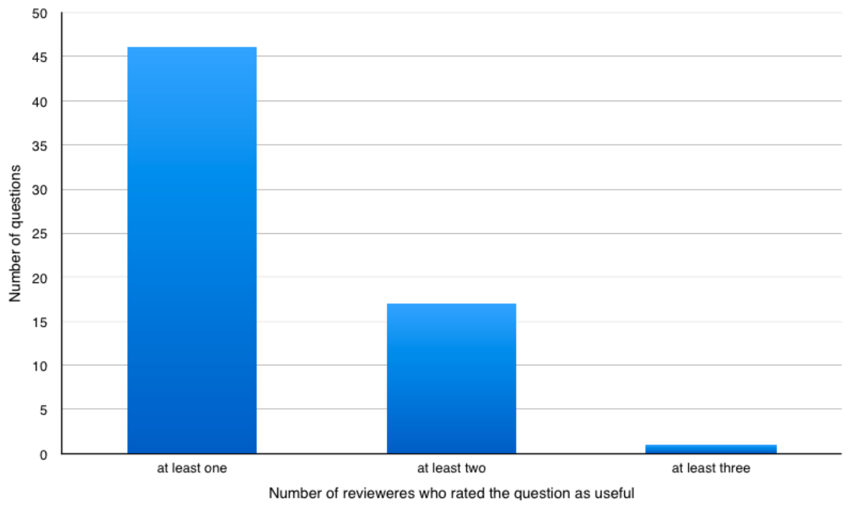


Figure 7.6: Usefulness of questions

Usefulness of distractors. A given distractor is considered “useful” if it has been functional (i.e., picked by at least one student). For the six questions which were administered on paper, at least two out of three distractors were useful. In five out of the six questions, the key answer was picked more frequently than the distractors. Exceptionally, in one question, a particular *linguistically unique*[7] distractor was picked more frequently than the key. The course instructor explained this by pointing out that this question was not covered explicitly in class. For the six questions which have been administered via BlackBoard, at least one distractor was useful except for one question which has been answered correctly by all the seven students.[8]

Item discrimination. We used Pearson’s coefficient to compute item discrimination to show the correlation between students’ performance on a given question and the overall performance of each student on all questions. The range of item discrimination is [-1,+1]. A good discrimination value is greater than

[7]This distractor was a verb phrase while the other answers, including the key, were nouns. See question number 36 in Table B.1.

[8]See question number 3 in Table B.1.

0.4 [Ebe54]. For the six questions administered on paper and four out of the six questions administered via BlackBoard, item discrimination was greater than 0.4. For one question administered via BlackBoard, item discrimination could not be calculated as 100% of students answered that question correctly. Finally, item discrimination was poor for only one question. The third reviewer pointed out that this question is highly guessable because of the conceptual similarities between the stem and the key.[9] These results are illustrated in Figure 7.7 and Figure 7.8.

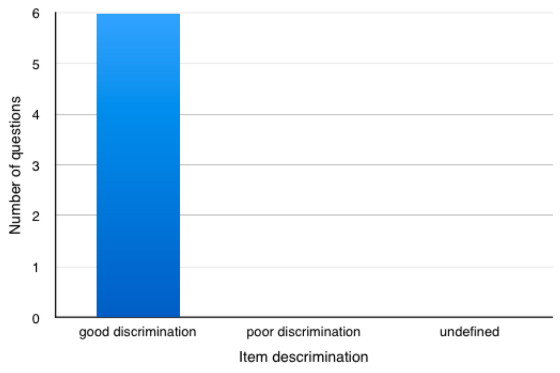


Figure 7.7: Item discrimination for questions administered in class

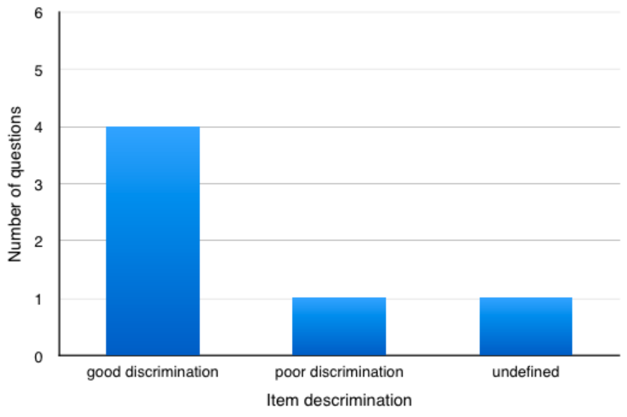


Figure 7.8: Item discrimination for questions administered online

Item difficulty. One of the core goals of the presented QG approach is to be able to control item difficulty. To evaluate this functionality, we examine tool-reviewers agreement and tool-students agreement. As described above, the tool generates questions and labels them as either easy or difficult. Each reviewer

[9]See question number 16 in Table B.1.

can estimate the difficulty of a question by choosing one of the following options: (1) too easy, (2) reasonably easy, (3) reasonably difficult and (4) too difficult. A question is too difficult for a particular group of students if it is answered correctly by less than 30% of the students and is too easy if answered by more than 90% of the students [Dav01]. In both cases, the question needs to be reviewed and improved. Accordingly, we consider a question to be difficult if answered correctly by 30-60% and easy if it is answered correctly by 60-90% of the students.

Before discussing tool-reviewers agreement, it is worth to note agreements among reviewers. We distinguish between loose agreements and strict agreements. A loose agreement occurs when two reviewers agree that a question is easy/difficult but disagree whether it is too easy/difficult or reasonably easy/difficult. Table 7.2 summarises agreements among reviewers. Each reviewer agrees with the tool on 31 (not necessarily the same) questions, see Figure 7.9.

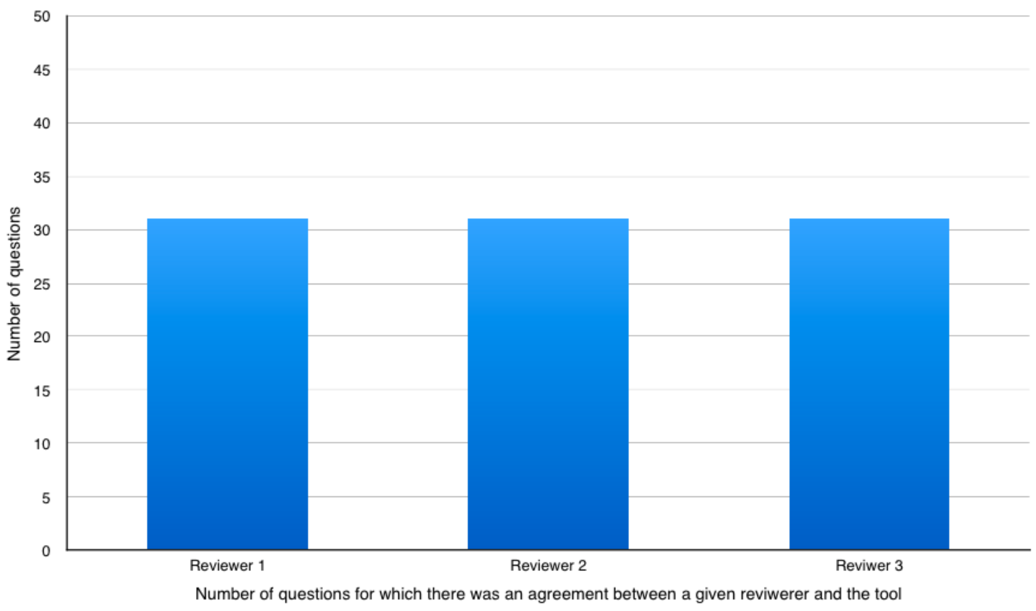


Figure 7.9: Tool-reviewers agreement on item difficulty

	1st & 2nd	1st & 3rd	2nd & 3rd
Loose agreements	31	26	33
Strict agreements	19	15	15

Table 7.2: Loose and strict agreements between the three reviewers

With regard to the six questions delivered on paper, two questions (no. 7 and 18 in Table B.1) were reasonably difficult and two (no. 7 and 38 in Table B.1) were reasonably easy for the students. These four questions were in line with difficulty estimations by the QG tool. One (no. 36 in Table B.1) out of the six questions was too difficult for the students. Most of the students picked a linguistically unique distractor rather than the key. Remarkably, the tool and the three reviewers have rated this item as easy. Finally, one question (no. 23 in Table B.1) was too easy for the students, however it was rated as difficult by the tool. This is due to having a clue in the stem. Similarly, for BlackBoard questions, one question (no. 50 in Table B.1) was reasonably difficult and one question (no. 1 in Table B.1) was reasonably easy for the students; just in line with tool estimations. One (no. 20 in Table B.1) out of the six questions was too easy for the students (100% correct answers). This question was rated as easy by the tool. Again, one question (no. 3 in Table B.1) was rated as difficult by the tool but was easy for the students due to having a clue in the stem. Two questions (no. 16 and 39 in Table B.1) were not in line with tool estimations but were in line with estimations of at least two reviewers. Results of item difficulty predictions are illustrated in Figure 7.10 for questions administered on paper and in Figure 7.11 for questions administered online.

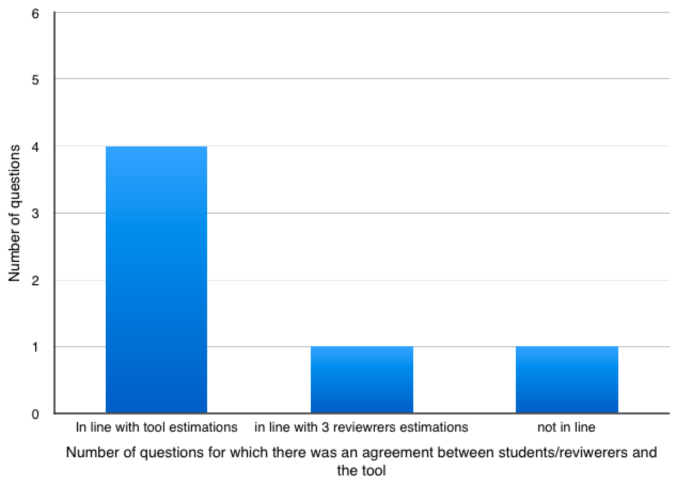


Figure 7.10: Item difficulty predictions-In class

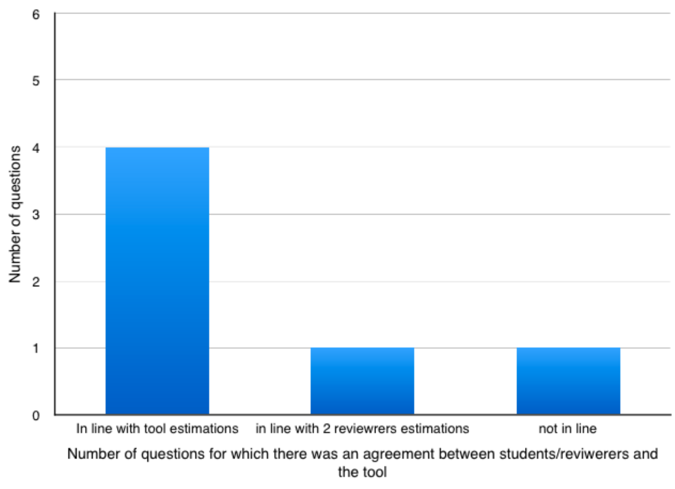


Figure 7.11: Item difficulty predictions-Online

7.2.4 Threats to validity

Due to time and resource limitations, the number of ontologies, generated questions, evaluated questions, student participants in this study were limited. Although the results are promising, they are far away from statistically significant. Moreover, the utilised ontology was handcrafted, limiting our ability to understand possible issues that may rise when generating questions from existing ontologies. We address both issues in the third evaluation study by looking at larger numbers of questions that are generated from existing ontologies.

7.3 Second evaluation study: using the QG approach by novice ontology developers

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 the 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.

7.3.1 Goals

This section presents another case study for generating MCQs from ontologies which have been handcrafted for the explicit purpose of generating questions (and evaluating them). Moreover, in the current case study, we examine the possibility of adapting our automatic approach to generate MCQs by instructors with no prior experience in building ontologies.

7.3.2 Materials and methods

7.3.2.1 Equipment description

See Section 6.1.1 for a description of hardware and software used in this experiment.

7.3.2.2 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. Also, 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 the terminological part was modelled in the ontology. This type of knowledge is typically a vital part of education in general and of assessment in particular. The development of the ontology has gone through the following steps:

- The instructor has been introduced to the basics of ontology development in an initial meeting which lasts for 2 hours.[10] This included a brief hands-on tutorial on using *Protégé* 4 ontology editor. Further online materials [Hor11b] 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.

[10]with the author of this thesis.

- 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.

- Before attempting to build a new version of the ontology, an attempt has been made to generate some questions from the first version of the ontology. A quick look at the questions has revealed some bugs in the ontology. The main feedback given to the ontology builder was to make sure that every "property" added to any class, via assigning a complex superclass, should also be applicable to its subclasses. For example, one of the classes in the first version was **API** and the ontology builder has asserted the following axiom **API** *SubClassOf*: **standsFor** *some* **ApplicationProgramInterface** to describe the abbreviation. As a consequence, for any subclass **X** of the class **API**, the following entailment holds: **X** *SubClassOf*: **standsFor** *some* **ApplicationProgramInterface**. One of the possible ways to overcome such a problem is to make use of annotation properties which do not have logical consequences.

- 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.

7.3.2.3 Generating questions

We have generated questions using the same templates used in the previous study and described in Table 4.1. Prior to generating questions, we computed the pairwise similarity for all the subconcept expressions in the ontology. 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 obvious. The previous evaluation study (QG from the KA ontology) have identified this as a possible problem (see Section 7.2.3 and question no. 23 in Table B.1). 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 65 questions are presented in Table B.2. Among these are 25 easy questions and 40 difficult questions.

7.3.2.4 Reviewing questions

Again, three reviewers have been asked to evaluate the 65 questions using the web interface shown in Figure 7.4. 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 each 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. The above three steps, i.e., attempting to answer the question, rate difficulty and rate usefulness, are exactly the same steps carried out in the reviewing phase in the first evaluation study. In addition to these steps, in this evaluation study, each reviewer is asked to check whether the question adheres to 5 rules for constructing good MCQs. 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.

7.3.3 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 pairwise similarities. Finally, Reviewers 1, 2 and 3 spent around 43 minutes, 141 minutes, and 56 minutes, respectively to review the selected 65 questions. 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 and the second scenario is where ontology-based question generation strategies are utilised. 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

Table 7.3: A sample question generated from the Java Ontology

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

shown to be possible in the second scenario (for example, see the study presented in Section 7.2). Third, no particular skills/creativity for MCQ construction are required when utilising ontology-based question generation strategies.

Usefulness of questions Figure 7.12 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 have been rated as either useful as it is or useful with minor improvements by all three reviewers. As a concrete example of a question that was rated useful by all 3 reviewers, we present the question in Table 7.3.

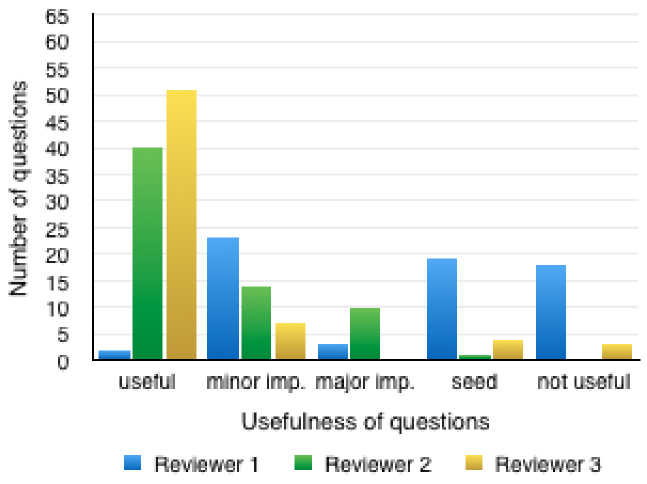


Figure 7.12: Usefulness of questions according to reviewers evaluations

Quality of questions The quality of questions is evaluated by adherence to 5 rules. Figure 7.13 shows the number 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.

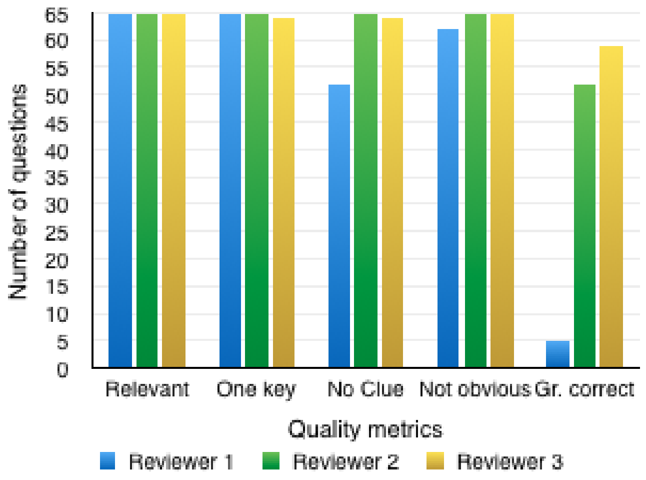


Figure 7.13: Quality of questions according to reviewers' evaluations

Difficulty of questions according to reviewers' ratings Part of the objectives of this study is to 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 asked 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 7.14 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 categorise the agreements according to the difficulty of questions. Table 7.4 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 (named) superclass while they could possibly be assigned to multiple (possibly complex) superclasses. Restructuring and enriching the ontology is expected to increase the ability of the tool to generate questions at certain levels of difficulty.

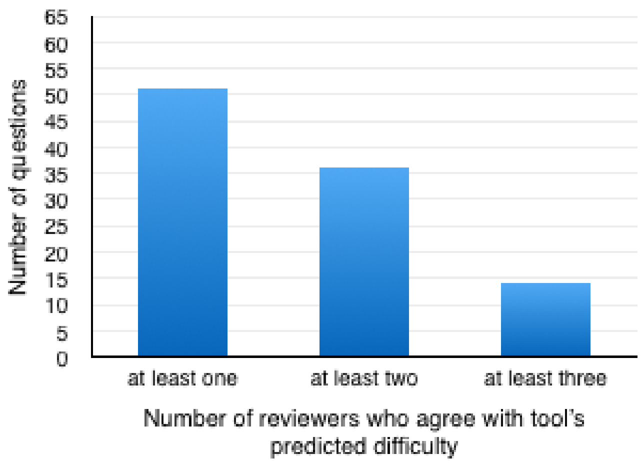


Figure 7.14: Difficulty of questions according to reviewers' evaluations

Table 7.4: Accuracy of difficulty predictions for easy and difficult questions

	$\geq$ 1 reviewer	$\geq$ 2 reviewers	$\geq$ 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, none 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 the reviewer has 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 7.13. This means that if a reviewer picks an answer other than the one identified by the tool as the correct answer then their answer will not be recognised as correct. Figure 7.15 shows the number of questions answered correctly by at least one, two and three reviewers.

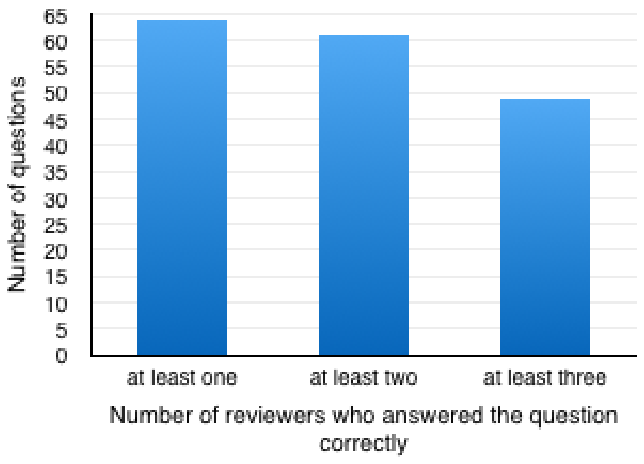


Figure 7.15: Difficulty of questions according to reviewers performance

In exactly one question (no. 28 in Table B.2), 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?”. The 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 particular question as “useful”, we believe that it is too difficult and, hence, not necessarily very useful as an assessment item.

7.3.4 Threats to validity

As discussed earlier in the evaluation of questions generated from the KA ontology, the Java ontology used in the second evaluation study is a handcrafted ontology which has been build for question generation purposes. This limits our understanding of any issues that may arise when generating questions from existing ontologies that were built for other (non-educational) purposes.

7.4 Third evaluation study: automated evaluation of questions

To examine, in a statistically significant manner, the practicality of using similarity to generate questions of certain levels of difficulty, we need a large number of questions. However, it is practically impossible to administer a large number of questions to real students, for different reasons such as: (i) ethical reasons and (ii) to avoid any possible bias caused by administering similar questions (with different distractors) to the same student. Rather than administering the questions to real students, we can alternatively utilise an automatic corpus-based solver. This also enables us to measure (although not precisely) the amount of knowledge that the solver has about a specific subject (e.g., number of resources related to this subject in the corpus).

Existing methods have already been developed to automatically solve MCQs using corpus-based solvers. In this experiment we follow the method described by Turney and Littman [TL05] who developed a method to automatically solve a specific kind of multiple-choice questions. In particular, the method was designed to solve multiple-choice analogy questions which were described in Chapter 2; see Table 2.2 for an example. The results show that their method can solve about 47% of multiple-choice analogy questions (compared to an average of 57% correct answers solved by high school students). The method takes a pair of words representing the stem (e.g., "$A : B$ ::") and 5 other pairs representing the answers presented to students (e.g., "$C_i : D_i$") and returns the answer with the highest analogy degree to the stem. The method is based on the Vector Space Model (VSM) of information retrieval. To calculate the analogy degree between the stem and a given answer, the solver creates two vectors, ($R1$) representing the stem and ($R2$) representing the given answer. Each vector consists of 128 elements that represent the frequency of encountering a certain phrase with a certain joining term in a large corpus. Each phrase is constructed by joining the two words (e.g., A & B) using one of 64 proposed joining terms in two ways (e.g., “A is B” and “B is A”). Each phrase is sent as a query to a search engine and the logarithm of the returned number of hits is saved in the corresponding element in the vector.

Due to the availability of methods to automatically solve analogy questions, we focus on generating this type of questions for the purposes of this experiment. One useful implication of generating one type of questions is that the cognitive ability required to solve the questions is guaranteed to be the same for all the questions, eliminating any external factors that could affect the difficulty of questions. In this case, the required cognitive ability is referred to as analogical reasoning.

7.4.1 Goals

The experiment is designed to answer two main questions: (i) Does the general performance of the solver correlate with the solver's amount of knowledge? and (ii) Does the performance of the solver on a particular question correlate with the estimated difficulty of that question?

7.4.2 Materials and methods

7.4.2.1 Equipment description

Due to the high computational demands of this experiment, two machines were used in parallel to solve the generated questions automatically. The first machine is the one described in Section 6.1.1. The second machine is an Intel Core i3 2.27GHz processor, 2 GB 1333 MHz SODIMM DDR3 RAM, running Windows 7 Home Premium (Dell Inspiron N3010 Mid 2010 model). Only the first machine was used to generate the questions.

7.4.2.2 Generating questions

To generate multiple-choice analogy questions, we extend the rules presented in Hypothesis 1 in Chapter 3 such that we can control difficulty of MCQs by varying: (1) similarity between the stem and the key, and additionally (2) similarity between the stem and distractors. The extended rules, which make use of the relational similarity measure $RelSim(\cdot)$ defined in Chapter 5, are presented below.

Proposition 2. *Let Q be a multiple-choice analogy question, let S be the stem part of Q consisting of two concepts S_x, S_y, let K be the key part of Q consisting of two concepts K_x, K_y, let D be a set of distractors in Q such that a distractor $D_i \in D$ consists of two concepts D_{xi}, D_{yi} and $1 < i \leq n$ where n is the number of distractors in Q. Let K be significantly more similar to S compared to any distractor $D_i \in D$, i.e., $RelSim(S_x, S_y, K_x, K_y) = RelSim(S_x, S_y, D_{xi}, D_{yi}) + \Delta_1$ and $\Delta_1 > 0$. Let K be sufficiently similar to S such that $RelSim(S_x, S_y, K_x, K_y) = \Delta_2$ and $\Delta_2 > 0$. For all distractors $D_i \in D$, let D_i be similar, to an extent, to S such that $RelSim(S_x, S_y, D_{xi}, D_{yi}) = \Delta_3$ where Δ_3 is assumed to be maximal. Then the following properties hold:*

1. *Increasing Δ_1 decreases the difficulty of Q.*
2. *Increasing Δ_2 decreases the difficulty of Q.*
3. *Decreasing Δ_3 for a distractor $D_i \in D$ decreases the difficulty of Q.*

Three ontologies were used to generate multiple-choice analogy questions using the above rules. One of the ontologies, which is the Gene Ontology, is considered to be a sophisticated ontology of specialised terms, though it is not very rich. The other two ontologies (Pizza Ontology and People & Pets Ontology) are ontologies that are usually used in ontology development tutorials; hence contain common knowledge. The decision to choose those ontologies was made to assess whether or not the difficulty of the domain influences the difficulty of the generated questions. More familiar domains to the solver are expected to be easier. Figure 7.16 presents a rough estimation of knowledge about each domain in the corpus as indicated by the number of related resources, i.e., when performing a search using the search engine provided by the corpus and using each subject as a search term. These simple statistics will be used to answer the first question of the experiment: "Does the general performance of the solver correlate with the solver's amount of knowledge?". Table 7.5 presents the number of classes in each ontology and the total number of generated questions from each ontology. These questions are of different levels of difficulty: easy, moderate and difficult. The parameters used to control difficulty of the generated questions are presented in Table 7.6 and were determined empirically (see Proposition 2 for details of these parameters). So, for Q to be easy, for example, Δ_1 has to be $= 0.5$, Δ_2 has to be $= 1$ and Δ_3 has to be $= 0$.

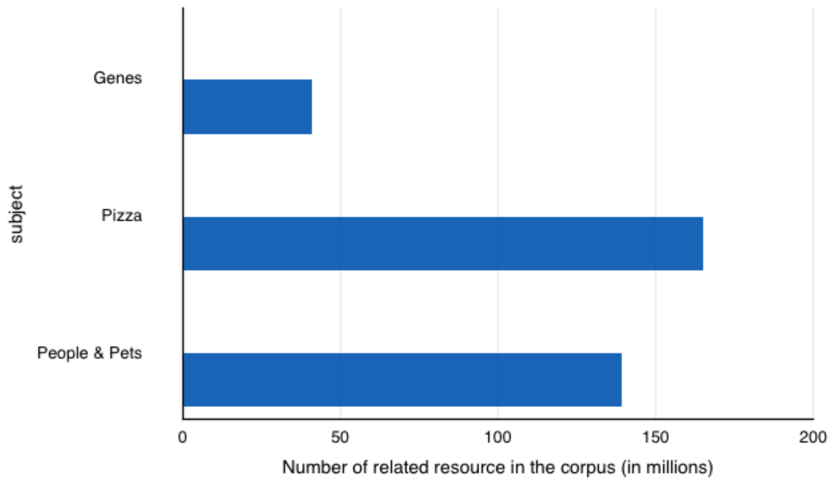


Figure 7.16: Rough estimation of knowledge in the corpus

Table 7.5: Number of classes and number of generated questions per each ontology

	No. of Classes	No. of questions
Gene Ontology	36146	187,924 Septillion
Pizza Ontology	97	18,933 Trillion
People & Pets Ontology	58	12,372 Billion

Table 7.6: Parameters used to generate analogy questions of different difficulties

Difficulty Level	Δ_1	Δ_2	Δ_3
Easy	0.5	1	0
Moderate	0.5	0.5625	0.0625
Difficult	0.5	0.5	0.1

To generate an analogy question from an ontology we need to, first, generate pairs of concepts that are sufficiently related in the ontology. Second, we need to compute the similarity degree between the generated pairs of concepts based on the underlying relation of each pair (i.e., their relational similarity). To generate a sufficiently related pair of concepts C, D, we use $Relatedness_{path-based}(\cdot)$ presented in Chapter 5 such that $Relatedness_{path-based}(C, D) = 1$. After generating pairs of related concepts, we need to compute the similarity degree between the

Table 7.7: A sample question generated from the People & Pets Ontology

Stem:	Haulage Truck Driver : Driver
Options:	(A) Quality Broadsheet : Newspaper (B) Giraffe : Sheep (C) Bus : Vehicle (D) Giraffe : Cat Liker
Key:	(C) Bus : Vehicle

generated pairs. To do so, we use $RelSim_{\text{path-based}}(\cdot)$ presented in Chapter 5. The final step in generation is iterating through all sufficiently related pairs of concepts in the present ontology, consider the current pair as a stem and follow rules of Proposition 2 to select keys and distractors targeting different levels of difficulty. We generate all possible analogy questions from a given ontology by exhausting all combinations of (sufficiently related) pairs which can appear either in the stem, key or distractors. We generate 3 distractors for each question.

To make the question solving phase more manageable and since we want to use the automated solver on a very high number of questions, we can only consider subsumption-based relations. To do this, we utilise $Relatedness_{pat-based}(\cdot)$. The more relations the automatic solver takes into account, the (far) more time it spends on each question. Note also that we have analysed a number of analogy questions (1,082 questions) available on the web and found out that a considerable proportion of them concentrate on the relations considered by $Relatedness_{pat-based}(\cdot)$. These analogy questions have been gathered using two search engines, Google and Yahoo!, using three search terms, "analogy examples", "analogy questions", "analogy test", considering only the two first pages of results. We have included in our corpus only textual questions written in English. In particular, 96 out of the 1,082 questions (8.9%) focus on is_a relations, 122 questions (11.3%) focus on sibling relations. Examples of other relations found in our corpus of analogy questions which are not considered in the current experiment include: object-characteristic (9.9%), part-whole relations (8.9%) and object-function (8.9%) relations.

Examples of the questions that were generated from the People & Pets ontology and Pizza ontology are presented in Table 7.7 and Table 7.8, respectively.

Table 7.8: A sample question generated from the Pizza Ontology

Stem:	Sloppy Giuseppe : Pizza De Carne
Options:	(A) Cogumelo : Pizza Vegetariana
	(B) Pizza : Food
	(C) Cogumelo : Napoletana
	(D) Cogumelo : Sorvete
Key:	(B) Pizza : Food

7.4.2.3 Building the automated solver

As explained earlier, we implemented a procedure similar to the one described by Turney and Littman [TL05] to automatically solve the generated analogy questions. First, we have constructed a table of joining terms relevant to the relations considered in the generation phase (e.g., "is a", "type", "and", "or"). Using these joining terms, we generate a set of phrases for each pair of concepts. Then we create vectors of 10 features for the stem, the key and each distractor. The constructed phrases are sent as a query to a search engine (Yahoo!) and the logarithm of the hits count is stored in the corresponding element in the vector. The hits count is always incremented by one to avoid getting undefined values.

7.4.2.4 Solving questions

Evaluating all the generated questions using the current automated solver was not possible due to the limited number of queries per user allowed by the utilised search engine (and the extremely high number of questions). Therefore, we only consider a representative sample of the questions (about 1800 questions per ontology). Three stratified samples of (easy, moderate and difficult) questions were selected randomly from the questions generated from the three ontologies (i.e., a sample for each ontology). The overall time required to solve all the questions in the considered sample (95% confidence interval with a margin of error of $\pm 3\%$) took over 120 hours using the two machines described in Section 7.4.2.1 working in parallel.

7.4.3 Results and discussion

Among the easy questions generated from the Gene ontology, the solver has correctly answered 16% of the questions. Similarly, for the People & Pets and Pizza

Table 7.9: Percentage of analogy-questions (per ontology) solved correctly by the automatic solver

	%Correct
Gene Ontology	16%
Pizza Ontology	49%
People & Pets Ontology	32%

ontology, the solver has correctly solved 32% and 49% of the easy questions, respectively. Those percentages are shown in Table 7.9. To answer the question of whether the performance of the solver correlate with the solver's amount of knowledge, we compare the solver's performance on the different sets of questions with its knowledge about each subject. As Table 7.9 and Figure 7.16 show, the more knowledge the automated solver has about a domain, the more ability it has to solve questions about that domain.

To evaluate the accuracy of the QG tool in terms of difficulty predictions, we compare the number of questions solved correctly by the solver in the different categories of questions (i.e., easy, moderate and difficult). Due to the solver's low performance on questions generated from the Gene ontology (see Table 7.9), we have only considered the two tutorial ontologies in this comparison. Figure 7.17 and Figure 7.18 show that, for each ontology, more easy questions were correctly solved by the automatic solver than moderate and difficult questions. This indicates that the similarity-based QG method was successful in controlling the difficulty of the generated MCQs.

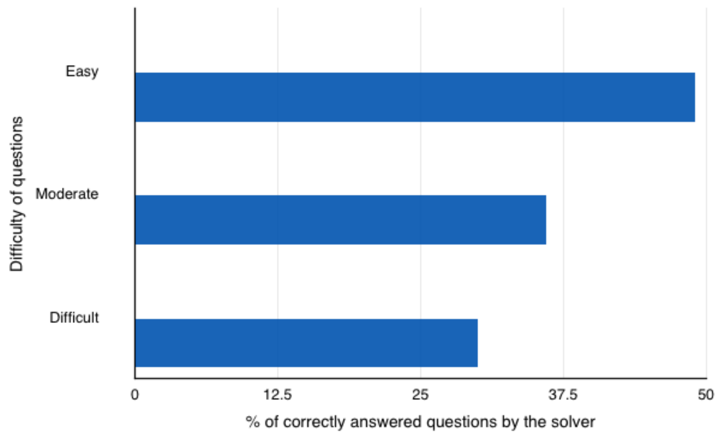


Figure 7.17: Percentages of questions of varied difficulties that were correctly answered by an automatic solver (Pizza ontology)

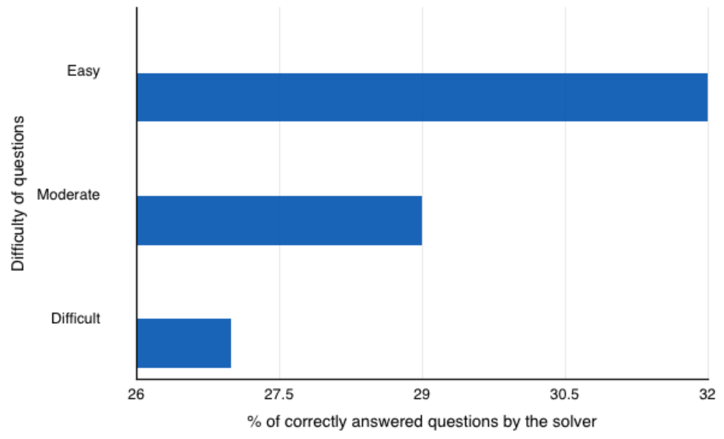


Figure 7.18: Percentages of questions of varied difficulties that were correctly answered by an automatic solver (People & Pets ontology)

7.4.4 Threats to validity

In the above experiment, an automated solver has been utilised to simulate a student attempting to answer a set of questions. A clear limitation of this approach is that the automated solver simulates a *single* student with a fixed ability (given that the questions solving phase took place in a relatively short period to time, hence corpus' resources are assumed to be relatively fixed). In real class situations, students have different abilities. A better simulation of such students would be to use different solvers with varied abilities (e.g., by varying the number of resources or number of joining terms). However, due to time limitations, we have focused on a single solver with fixed ability. Another threat to validity is the (relatively) low number of ontologies and the ad-hoc selection criteria of which ontologies to be considered.

7.5 Summary and directions

We have presented a series of studies to evaluate the usefulness and difficulty of MCQs generated using our similarity-based QG tool. We have focused on evaluating the usefulness and appropriateness of questions for educational purposes, in particular, assessments. It is also interesting to evaluate the usefulness of questions generated using the similarity-based QG tool for other purposes (e.g., validating an ontology). We briefly elaborate on this topic in the next chapter.

Chapter 8

Applications of QG methods

Although previous chapters have focused on generating questions for assessment purposes, this is not necessarily the only possible target application for QG methods. In this chapter, we explore the applicability of QG methods for ontology validation purposes. The chapter builds on the ideas presented in previous chapters by utilising the proposed similarity-based approach for generating questions from ontologies.

In Section 7.3, we have witnessed the usefulness of looking at MCQs generated from ontologies that are under development. Some important "errors" in the Java ontology were easily identified by looking at the MCQs generated from that ontology, in particular, MCQs with errors. Some errors were syntactic (e.g., typing mistakes) while others were logical (e.g., a wrong entailment identified by looking at an invalid key or a missing entailment identified by looking at an invalid distractor). Logical errors are generally harder to spot and considered more interesting when debugging an ontology. We will briefly present some specific examples from the Java ontology in Section 8.2.

In this chapter, we present a case study to further explore the applicability of QG methods for ontology validation purposes. Rather than validating an ontology under development (as we did in Section 7.3), we study the case of validating a previously built ontology in an attempt to suggest ways to improve it. We present some specific examples for possible errors in the SNOMED CT ontology as identified by some domain experts. In addition, QG methods can support ontology comprehension purposes which can be a goal in itself or it can be done prior to validating an ontology that has been built by a different ontology developer. We briefly tackle this in the study presented in this chapter.

8.1 Designing a QG-based protocol for ontology validation

Ontologies can grow large in terms of size and complexity, making them difficult to debug. The most hard-to-spot errors in ontologies are the ones that do not make the ontology inconsistent or incoherent, though cause either undesirable or missing entailments. This is similar to the so-called "logical errors" in programming languages which cause the program to produce undesired output but do not cause compilation errors or abnormal termination. In Table 8.1 and Table 8.2 we present two examples of MCQs that have been generated from the Java ontology presented in Chapter 7. Clearly, some logical errors in the Java ontology have resulted in producing the errors that appear in these MCQs. Identifying the errors in these MCQs by a Java expert has helped in finding and correcting some omissions in the Java ontology. These examples show that looking at questions generated from an ontology can be fruitful for identifying some omissions in the ontology.

Table 8.1: Missing entailment example

Stem:	A feature of Virtual Machine Code is:
Key:	(A) Portability
Distractors:	(B) Write once Run Anywhere
	(C) Platform Independence
	(D) Reusability
Explanation of error:	the distractors are correct answers (i.e., all the answers are features of Virtual Machine Code)
Reasons for the (missing) entailment:	Those features have been asserted (in the ontology) to be features of Java Programming but not features of Virtual Machine Code. However, due to the similarity between the features (answers A, B, C, and D) they have all appeared in the answer list of this MCQ.

Indeed, there are many possible ways to find errors in ontologies. Direct ontology inspection can be effective but has the obvious disadvantage of being time consuming. In addition, direct inspection might be more effective for finding soundness problems (i.e., invalid entailments) rather than completeness problems

Table 8.2: Undesired entailment example

Stem:	Swing stands for:
Key:	(A) Application Programming Interface
Distractors:	(B) Abstract Windowing Toolkit
	(C) Java Foundation Classes
Explanation of error:	the key is not a correct answer (i.e., Swing does not stand for Application Programming Interface)
Reasons for the (undesired) entailment:	Swing $\sqsubseteq$ API
	API $\sqsubseteq \exists$ standsFor.ApplicationProgrammingInterface
	Therefore, the ontology entails that:
	Swing $\sqsubseteq \exists$ standsFor.ApplicationProgrammingInterface

(i.e., missing entailments). Other approaches have been proposed to address completeness problems. For example, Formal Concept Analysis (FCA) has been used for such a purpose [BGSS07]. Another example is the approach presented by Dragisic et al. [DLWK14] that takes already found missing entailments as input and suggest logical solutions to repair the ontology by possibly adding missing axioms. We are not comparing our QG-based method to any other debugging method, rather, we are suggesting that using MCQs with high similarity between the key and distractors, can be useful in restricting the search space (in a principled way) when attempting to detect a specific class of omissions. These omissions include both missing atomic subsumptions and missing complex subsumptions. We expect that our method can detect more missing complex subsumptions compared to missing atomic subsumptions. Using similarity to elicit knowledge from domain experts has already been used in well known elicitation techniques. For example, the triadic elicitation technique involves presenting 3 concepts to domain experts who are asked to identify the two similar concepts and explain why the third is considered different.

We conjecture that asking a domain expert to look at a set of MCQs generated from an ontology can help in identifying some of the invalid and/or missing entailments based on the expert's knowledge. The questions should be presented to the expert in the form of a multiple-response question where the expert is asked to select all (and only) the correct answers. We use the QG application described in earlier chapters to generate questions that has exactly one answer entailed by the ontology to be correct. For the purpose of using these questions to validate the ontology, we select (for each question) a varied number, ranging for example from 1 to 10, of answers that are entailed to be wrong answers. The similarity between the key and distractors is set to be above a threshold. To examine whether using a threshold of a high value has an impact on the number and type of the identified errors, we experiment with two different thresholds as we will describe in detail in Section 8.3. In general, since the wrong answers are selected to be similar to the correct answer, we question whether the ontology should entail that they are correct answers as well (i.e., a missing entailment such as the one presented in Table 8.1).

8.2 Implementing a prototype QG-based application for ontology validation

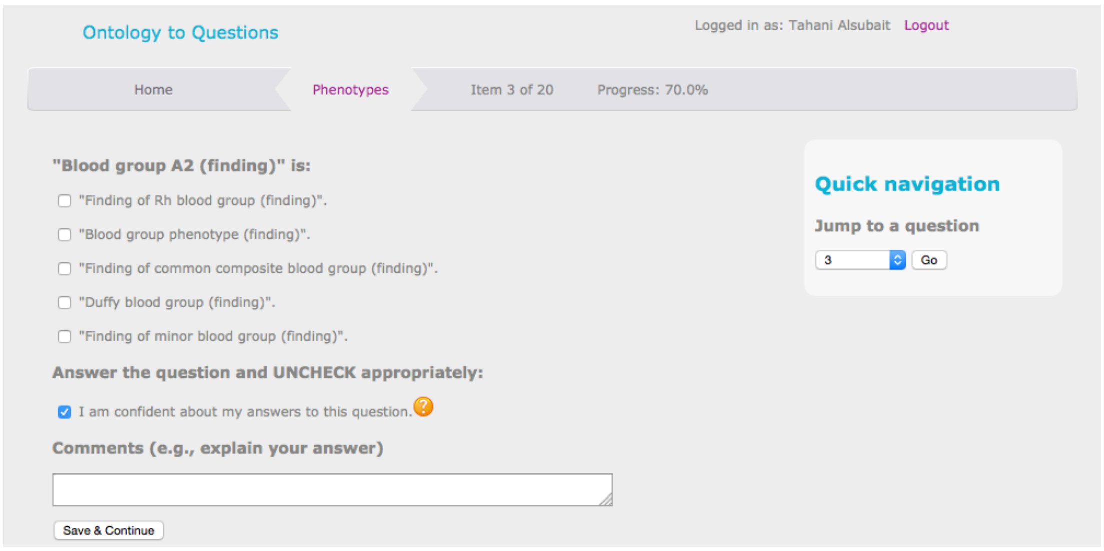
To evaluate the usefulness of the similarity-based QG approach presented in this thesis for ontology validation purposes, we have implemented a prototype web-based application that (1) presents a selected set of multiple-response questions generated from an ontology to a domain expert (see Figure 8.1) and (2) based on the expert's answers, the application suggests some possible wrong and/or missing entailments in the ontology (see Figure 8.2). As we describe in Section 8.1, the questions in fact are generated such that they have only one answer which is entailed by the ontology to be correct. However, experts answering these questions are asked to pick all the answers they believe to be correct. Experts are also asked to indicate whether they are confident about their answers, per question. They can also leave a comment for a detailed explanation.

When the answers provided by an expert are different from the ones entailed by the ontology, the expert is asked to confirm his/her answers, as shown in Figure 8.3. The aim of this extra verification step is to encourage deeper engagement.

8.3 A case study

8.3.1 Goals

The main goal of this case study is to evaluate the applicability of QG-methods for ontology validation purposes. To address this goal, we try to answer the following question: Can a domain expert identify some omissions in an ontology by looking at MCQs generated from that ontology? We focus on a specific class


Ontology to Questions

Logged in as: Tahani Alsubait Logout

Home | Phenotypes | Item 3 of 20 | Progress: 70.0%

"Blood group A2 (finding)" is:

- [ ] "Finding of Rh blood group (finding)".
- [ ] "Blood group phenotype (finding)".
- [ ] "Finding of common composite blood group (finding)".
- [ ] "Duffy blood group (finding)".
- [ ] "Finding of minor blood group (finding)".

Answer the question and UNCHECK appropriately:

- [x] I am confident about my answers to this question.

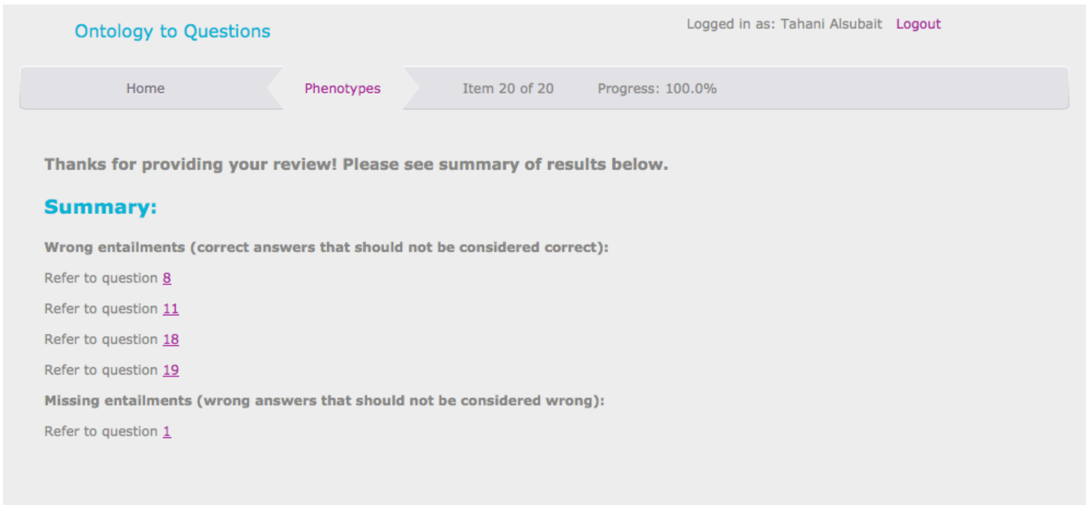
Comments (e.g., explain your answer)

Save & Continue

Quick navigation

Jump to a question: 3 Go

Figure 8.1: QG-based support for ontology validation


Ontology to Questions

Logged in as: Tahani Alsubait Logout

Home | Phenotypes | Item 20 of 20 | Progress: 100.0%

Thanks for providing your review! Please see summary of results below.

Summary:

Wrong entailments (correct answers that should not be considered correct):

Refer to question 8

Refer to question 11

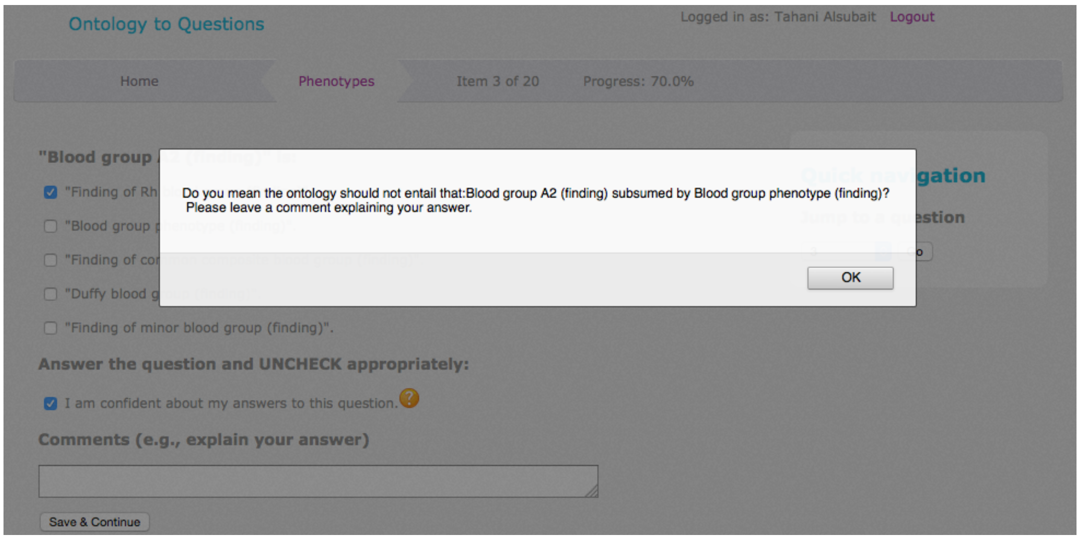
Refer to question 18

Refer to question 19

Missing entailments (wrong answers that should not be considered wrong):

Refer to question 1

Figure 8.2: Summary of suggestions to improve the ontology


Ontology to Questions

Logged in as: Tahani Alsubait Logout

Home | Phenotypes | Item 3 of 20 | Progress: 70.0%

"Blood group

- [✓] "Finding of Rh
- [ ] "Blood group
- [ ] "Finding of co
- [ ] "Duffy blood g
- [ ] "Finding of minor blood group (finding)".

Answer the question and UNCHECK appropriately:

- [✓] I am confident about my answers to this question.

Comments (e.g., explain your answer)

Do you mean the ontology should not entail that:Blood group A2 (finding) subsumed by Blood group phenotype (finding)?
Please leave a comment explaining your answer.

Figure 8.3: Extra verification step

of MCQs in which each wrong answer is similar to the correct answer (but entailed by the ontology to be a wrong answer). We expect that looking at such questions can reveal some omissions or missing statements (in the ontology) that might be difficult to spot without looking at the questions. This is because these wrong answers are similar to the correct answer and therefore raise the question of whether they have been considered as wrong answers due to having any missing statements in the ontology or due to actual constraints in the domain. The missing statements that are intended to be detected can be either *atomic* or *complex* subsumptions. Missing or invalid atomic subsumptions highlight problems in the inferred class hierarchy of the ontology. Since this hierarchy is frequently looked at by ontology developers, we expect, in general, that there are more missing/invalid complex subsumptions rather than atomic subsumptions in a given ontology. We examine this hypothesis in the current study by looking at two sets of questions, Set A1 and Set A2. The questions in the two sets are constructed:

1. in Set A1: based on atomic subsumptions.

2. in Set A2: based on complex subsumptions.

Another goal of this study is to explore the impact of varying the similarity degree between the key and distractors on the overall usefulness of the generated questions for validation purposes. To examine this factor, we generate and compare two sets of MCQs, Set B1 and Set B2 which are described below. We try to answer the following question: Is looking at MCQs from Set B1 more useful for ontology validation purposes than looking at MCQs from Set B2? The MCQs in the two sets are generated such that the similarity between the wrong answers and the correct answer is:

1. in Set B1: above a threshold Δ_{max}.
2. in Set B2: below a threshold Δ_{max} but above a second threshold Δ_{min}.

The two sets A1 and A2 are not disjoint from sets B1 and B2. To examine all possibilities, we generate four disjoint sets of questions such that the questions:

1. in Set 1: are selected from Set A1 and Set B1.
2. in Set 2: are selected from Set A1 and Set B2.
3. in Set 3: are selected from Set A2 and Set B1.
4. in Set 4: are selected from Set A2 and Set B2.

8.3.2 Materials and methods

8.3.2.1 Equipment description

The following machine has been used to carry out the experiment presented in this chapter:

Intel Core i5 1.4GHz processor, 4 GB 1600 MHz DDR3 RAM, running Mac OS X 10.10.2 (MacBook Air Early-2014 model). Details of the software used to carry out the experiment have been presented in Section 6.1.1.

8.3.2.2 Ontology selection

The current study requires the availability of a domain expert to answer a set of MCQs generated from a domain ontology. Due to the availability of an expert in BioInformatics, we have asked that expert to select some parts of an ontology which he thinks might be suitable for the purpose of this study. Due to the expert's interest in SNOMED CT in general and genetic findings in particular and his assumptions that the ontology is not detailed enough in this part, we have selected a (small) part of genetic findings that covers phenotypes (e.g., Blood groups). All the subclasses (197 classes) of the class *Phenotype* have been used as a seed signature to extract a $\perp$-module. In addition, the object property *RoleGroup* has been added to the seed signature. This property is used to group certain properties together [SDMW02] and is necessary for extracting the module. The resulting module has a total of 246 classes and 6 object properties.

8.3.2.3 Generating questions

Two sets of questions have been generated from the extracted module using the prototype QG application described in Section 7.1. This prototype generates two different sets of questions, namely difficult and easy questions. The difficult questions are generated such that the similarity between the key and distractors is above the average similarity between all siblings in the ontology (or in the current study, the extracted module). The easy questions are generated such that the similarity between the key and distractors is above two thirds of the average similarity between all siblings in the module (but less than the average similarity between all siblings). For the current study, we consider difficult questions to be questions of Set B1 and easy questions to be questions of Set B2. After computing the average similarity between all siblings in the module, the thresholds Δ_{max} and Δ_{min} have been set to 0.88 and 0.587, respectively. The generated questions take the form "What is X?" where X is a class name and the answers are either class names or class expressions. This kind of questions is suitable for finding missing/invalid entailments that we are interested in. Among the generated questions, 223 questions have class-name-based answers, referred to as Set A1 questions, and 24 questions have class-expression-based answers, referred to as Set A2 questions. Among the class-expression-based questions, only 5 questions are suitable for Set B1 (i.e., the similarity between the key and distractors is above the threshold Δ_{max}). These 5 questions are referred to as Set 3 as defined in Section 8.3.1. Each question has exactly one key but the number of distractors was variable. If the number of generated distractors for a given question is more than 10, we randomly select 10 distractors out of the available ones. We have not restricted the questions set to questions with exactly three distractors (as in the experiments presented in Chapter 7) because questions with lower or higher number of distractors might be equally interesting for validation purposes as the omissions can be in any part of the ontology. However, we restricted the number of distractors to be below or equal to 10 to make the question answering phase manageable.

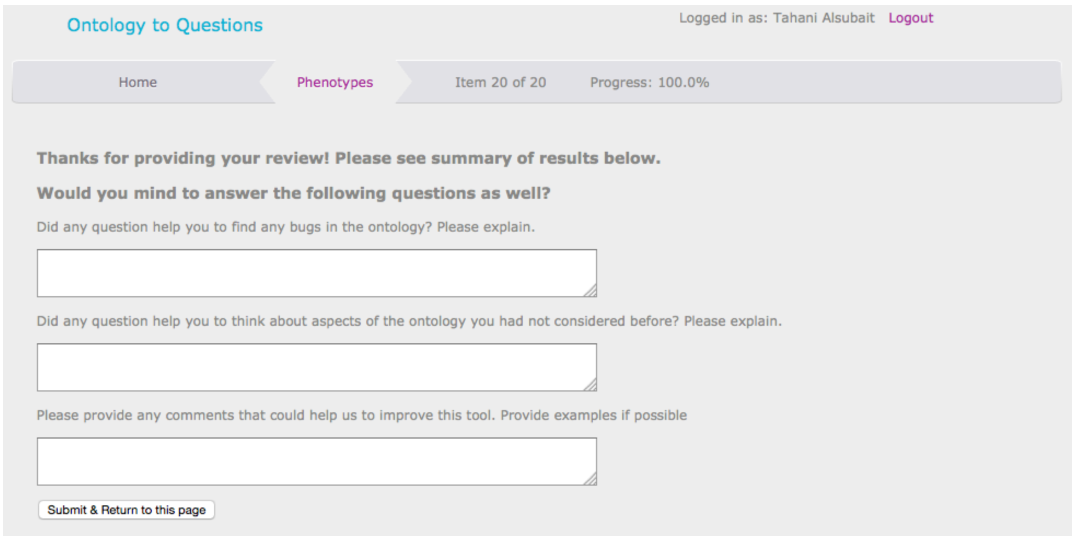
8.3.2.4 Answering questions

Two domain experts have been asked to answer a total of 20 questions (5 questions from each of the four sets Set 1, Set 2, Set 3 and Set 4). The first expert is a bioinformatician and the second expert is a physician. The 20 questions were selected randomly from the set of generated questions in the previous step and are presented in Table B.3. The questions were presented to the domain experts via the web-interface described in Section 8.1, see Figure 8.1. The first 10 questions are from Set A1 and the second 10 questions are from Set A2. We chose to present questions from Set A1 first, for deeper engagement, because they are expected to take less time to answer compared to questions from Set A2. Within Sets A1 and A2, questions from Sets B1 and B2 are randomly ordered. Also, a think-aloud technique was used to get a deeper insight into the advantages and limitations of the approach. The experts were allowed to use any external source to help them in answering the questions. After answering all the questions, the experts were asked to answer three last questions about their overall experience in answering the questions. These questions, which are shown in Figure 8.4, are:

1. Did any question help you to find any bugs in the ontology? Please explain.
2. Did any question help you to think about aspects of the ontology you had not considered before? Please explain.
3. Please provide any comments that could help us to improve this tool. Provide examples if possible.

8.3.3 Results and discussion

For 9 out of the 10 questions in Set B1, the first expert's answers were correct, i.e., equivalent to what is entailed by the ontology. The only question for which this expert's answers were different from the ones entailed by the ontology is question no. 15 in Table B.3. This question is the only question which contains an answer that contains an existential restriction; all the other answers contain either class names or conjunctions of class names. The expert has identified both a missing entailment (invalid wrong answer) and a wrong entailment (invalid correct answer). In particular, the expert indicated that the ontology should entail that a finding of common composite blood group is subsumed by a finding of blood


Ontology to Questions
Logged in as: Tahani Alsubait Logout
Home
Phenotypes
Item 20 of 20
Progress: 100.0%
Submit & Return to this page

Thanks for providing your review! Please see summary of results below.

Would you mind to answer the following questions as well?

Did any question help you to find any bugs in the ontology? Please explain.

Did any question help you to think about aspects of the ontology you had not considered before? Please explain.

Please provide any comments that could help us to improve this tool. Provide examples if possible

Figure 8.4: Using QG-methods to validate ontologies

group and phenotype finding. He also indicated that the ontology should not entail that a finding of common composite blood group is subsumed by a finding of blood group and interprets (attribute) ABO and Rho(D) typing (procedure). The expert indicated that he was not confident about his answers to this question and explained that by reporting that he was not familiar with the terminology used by the ontology to describe the concepts presented in this question, e.g., interprets (attribute). In consistent with the first expert's answers, the second expert answered all the questions in Set B1 correctly; hence she did not identify any possible omissions in this part of the ontology.

For 8 out of the 10 questions in Set B2, the first and second experts' answers were correct. The two questions for which the two experts' answers were different from the ones entailed by the ontology are questions no. 17 and 20 in Table B.3. In both questions, the answers are conjunctions of class names. Again, in both questions, the experts have identified a missing entailment (by selecting one of the distractors) and a wrong entailment (by not selecting the expected key). Both experts have agreed on the wrong answer that they chose to select as an answer. The two experts have indicated that they are not confident about their answers to these two questions. The first expert explained why he was not confident about his answers to question no.17 by pointing out that one of the terms used in the question, i.e., inherited, seems irrelevant since all blood groups are inherited. For question no. 17 the experts indicated that the ontology should entail that inherited weak D phenotype is subsumed by blood group phenotype and finding of minor blood group. Similarly, for question no. 20, the experts indicated that the ontology should entail that trans weak D phenotype is subsumed by blood group phenotype and finding of minor blood group.

In total, the first expert indicated that he was confident when answering only 7 questions out of the 20 questions. The second expert was confident when answering 13 questions out of the 20 questions. The first expert explained that by pointing out that although the terminology used in the ontology might seem to be natural to an ontology developer, it does not seem to be natural for a subject matter expert. Consistent with this, the second expert reported that the language of questions made it difficult to interpret what the question was asking. The first expert also reported that the questions seem to be of varying difficulty. For example, he pointed out that answering questions no. 1, 2, 3, 4, 5, 6, 8 and 10 (from Set A1) was straightforward. These questions use only class names as answers. In contrast, he reported that questions no. 7 and 9, which also use only class names as answers, were harder to answer. He explained that by pointing out that the answers were very similar and hence he found it difficult to decide which answer is the correct answer. The answers to these questions were: Blood laboratory and Blood bank which are indeed similar. The first expert further explains that he selected what he thought was the best answer, rather than the only correct answer. Consistent with this, the second expert reported that, for questions no. 7 and 9, she picked what she thought was the best answer. The experts did not identify any missing entailments in questions no. 7 and 9, i.e., they did not indicate that a wrong answer should be a correct answer. However, their explanation supports the hypothesis we are testing in this study, i.e., looking at MCQs with distractors that are similar to the key can be helpful in identifying missing entailments.

As described earlier, the similarity between the key and distractors in questions from Set B1 is higher than the similarity between the key and distractors in questions from Set B2. Although one would expect that questions in Set B1 would reveal more omissions in the ontology compared to questions in Set B2 (because the wrong answers are more similar to the correct answers), this was not the case. Questions in Set B1 have identified 2 (possible) omissions while questions in Set B2 have identified 4 (possible) omissions. This can be explained by the fact that errors can occur in *different* parts of the ontology. For example, questions in Set B1 would identify missing subsumees that are very close to their (potential) subsumer, e.g., in the inferred class hierarchy. In contrast, questions in Set B2 would identify missing subsumees that are not very close to their potential subsumer. In general, looking at this (rather small) set of questions was helpful in spotting some omissions in the ontology and suggesting improvements. Consistent with our expectations, the results also show that the method may be generally more helpful in identifying invalid/missing entailments involving complex subsumptions, i.e., Set A2, rather than atomic subsumptions, i.e., Set A1.

The aim of the second and third question presented to the experts after answering the questions was to evaluate the usefulness of the presented MCQs to support ontology comprehension purposes. According to the answers provided by the experts, the questions were not very helpful in identifying new aspects of the ontology they had not considered before. The first expert pointed out that this is due to having (1) questions that seem to be unnatural to a subject matter expert (due to describing concepts in an uncommon way) and (2) changes in the difficulty level of the questions (partly due to the first point). He further explains by pointing out that these two points might limit the usefulness of this form of MCQs for supporting students who want to learn about the subject. The second expert, who is a physician, did not respond to this question as she was not familiar with the ontology.

8.3.4 Analogus experiments

Bertolino et al. [BDDS11] have investigated the use of QG-based methods for validation purposes. Their method aims at validating models in general and can be applied to ontologies as well. A set of True/False questions generated from an (altered) model are presented to a group of domain experts. The responses gathered from domain experts are used to validate the model. The method proposed by Bertolino et al. is different from our method in that they suggest to alter the model by deliberately introducing some errors in it before the QG step. Their method is also suitable for finding invalid entailments but not missing entailments. Although they have reported that their method have helped the recruited experts to think about new aspects of the domain which they have not considered before, the method does not guarantee that this applies to the unaltered (error-free) parts of the domain only.

8.4 Summary and conclusions

We have presented a case study for evaluating the applicability of similarity-based QG methods for ontology validation purposes. Although the results seem to be promising, they are far from significant. Further efforts are needed to improve and evaluate the presented strategy. In particular, more user studies are needed. In addition, it might be useful to further consider the experts' comments gathered from the case study presented in this chapter regarding what can make a question difficult for a group of students. It would be useful to incorporate these comments into the QG approach and evaluate it by future studies.

Chapter 9

Conclusion and future work

In this chapter, we tie together the various issues covered in this thesis in order to discuss the overall contributions, the limitations of each contribution and the possible paths forward.

9.1 Thesis overview

The thesis presents a similarity-based approach to generate MCQs from ontologies. The main hypothesis of this thesis is that we can control difficulty of the generated MCQs by varying similarity between the correct and wrong answers. In Chapter 3, we have provided a psychological justification of why we think that this hypothesis is (1) important and (2) valid. The main evaluation studies presented in this thesis were set out to validate this hypothesis and explore the usefulness of the generated questions for assessing students' knowledge about a domain of interest. We have also investigated whether generating questions from domain ontologies can be useful, not only for assessing students' domain knowledge, but also for validating the ontology they are generated from. The thesis also presented a new family of similarity measures for ontologies. Similarity measures are an essential component of the presented MCQ generation approach and are essential to many other ontology-based applications.

9.2 Contributions, limitations and future directions

The major contributions of this thesis are summarised by topic in the following subsections. This thesis has covered a wide range of topics and there are still some open issues that need to be addressed. For each contribution, we discuss the limitations of the contribution and suggest some directions to extend the work presented in this thesis.

9.2.1 Establishing a theory of controlling MCQs difficulty

The key contribution of this thesis is establishing a theory for QG which takes into account the importance of controlling the difficulty of the generated questions. This theory is applicable both to automatic QG, e.g., ontology-based QG, and manual QG. Prior to this thesis, existing automatic QG methods have either ignored (or failed to recognise) the importance of controlling difficulty or proposed methods that have not been validated. Establishing theories/mechanisms to control the difficulty of assessment questions is clearly a big gap in existing QG literature. The thesis has advanced our knowledge on the psychological aspects of the problem and proposed strategies that exploit ontologies and similarity measures in order to provide better QG methods that can address this gap. Empirically, we have found that a solver's general performance (on solving questions of varied difficulty) correlates with the solver's amount of knowledge. This property is a very basic requirement to construct *valid* assessment questions. We have shown, although not in a statistically significant manner, that the proposed similarity-based QG approach generates questions that fulfil this requirement. Further studies are required to evaluate this aspect of the presented QG approach. We have, also, shown that students' performance correlates with the *estimated* difficulty level of the generated questions. This is a very important finding as existing attempts to control difficulty were, primarily, theoretical and unbacked with any empirical evidence (e.g., [Wil11, KS13]).

Broadly speaking, it is likely that the proposed QG method, including its ability to control difficulty, will prove to be useful for test developers. However, while we have provided a general model to control the difficulty of MCQs, the suggested similarity-based model is indeed not applicable to all classes of MCQs. Also, the empirical studies conducted to evaluate the MCQ generation method are limited in that they only consider a few classes of MCQs due to their natural fit with the source and due to time and resource limitations. Focusing on these classes, which are used in some existing QG methods, allows potential comparisons between the presented method and other already existing ones. However, there is still a need to extend these empirical studies and more importantly extend the presented MCQ generation method by studying other factors that can affect the difficulty of assessment questions and that may be applicable to other classes of MCQs. We are currently investigating such factors with some research partners in order to extend the model presented in this thesis.[1] For example, combining pedagogic content knowledge (PCK) [Shu86] with subject matter knowledge may help in controlling difficulty by taking into account teachers' knowledge of, e.g., what makes concepts difficult, regularly encountered students' misconceptions or misapplications of prior knowledge.

We have focused on developing methods to generate and control the difficulty of MCQs which seem to be more time consuming to generate (manually) compared to other forms of questions such as essay questions. The presented QG method is limited in that it is not, at least directly, applicable to other kinds of questions such as free response questions. Extending the method to include other kinds of questions is one of the interesting future paths.

Finally, the presented model of difficulty provides a relative, rather than absolute, notion of difficulty. Given two MCQs, the model can predict which one is more difficult than the other. Providing absolute difficulty values remains an open issue.

9.2.2 Reducing test developers' efforts

Consistent with the study presented by Mitkov et al. [MAHK06], this thesis shows that the manual effort to construct MCQs can be reduced by utilising automatic QG tools; especially if the ontology does not have to be built from scratch. We have not presented comparison studies to explicitly show the reduced effort; however, the thesis shows that a large number of (reasonably good) MCQs can be generated from a given (reasonably good) ontology. On the one hand, it is likely that the effort required to build a new ontology will be less than the effort required to manually construct the large number of MCQs (with suitable number of distractors) that can be automatically generated from that ontology. On the

[1] For an Elsevier research project.

other hand, we have shown that the automatically generated MCQs come with good predictions of their difficulty which can hardly be accomplished otherwise. It remains to evaluate the cost of adopting the proposed QG method in comparison with manual generation methods.

One of the issues that may affect such comparison and which has not been comprehensively covered in this thesis is the rendering of the generated questions. Indeed, the accuracy of language is an important aspect of questions that needs to be taken into account when generating questions automatically. In addition, the more accurate the generated questions, the less time required to post-edit them.

Another open issue that would help to reduce test developers' effort is to support the generation of complete *exams* rather than unrelated individual questions. Although the thesis has not covered this issue, we believe that by utilising both the presented model to control difficulty and the knowledge in the ontology, we can generate well balanced exams in terms of difficulty and topics covered. Moreover, different exams with similar properties (i.e., difficulty or related content) can be generated. In addition, extending the method to generate questions from a mix of sources might prove to be useful, especially considering that ontologies are not necessarily suitable for modelling all kinds of knowledge.

Also, reducing test developers' effort when generating questions is not interesting unless the generated questions are useful. Although we have shown, through the expert-centred evaluations, that the generated questions are educationally useful, the notion of usefulness was not precisely defined. Indeed, usefulness consists of multiple dimensions or perspectives and we have not (comprehensively) covered all of them. For example, we have not shown that they can improve students' understanding of the subject matter. In addition, there is a need to explore which questions templates are considered more useful by real test developers. One of the possible future paths is to conduct surveys to see which templates would be more interesting for a wide range of test developers.

9.2.3 Giving dimensions to automatic question generation

Several approaches have been proposed to automatically generate questions from electronic knowledge sources. In order to compare these approaches and understand the contributions and limitations of each approach, we need to understand the dimensions of the QG problem. In Chapter 4, we have conducted a systematic review of the QG literature in order to provide a better understanding of the problem and its dimensions. Our analysis of existing automatic QG methods has revealed that the main dimensions of QG are: (1) purpose (e.g., assessment, validation), (2) knowledge source (e.g., text or knowledge bases), (3) additional input (e.g., templates, patterns), (4) general generation methods (e.g., syntax-based, semantics-based), (5) distractor generation method (e.g., similarity, random), (6) output format (e.g., questions format, answer format), (7) feedback support (e.g., answer dependent/independent) and (8) evaluation method (e.g., student centred, expert-centred). We have explored the different design options for the automatic generation of questions and compared them whenever possible. We have also provided a historical analysis of the evolution of QG methods over the last five decades and provided an outlook on its future.

Although we have compared existing QG methods, on a conceptual level, with respect to each dimension, there is still a need for comparing QG methods in practice. The presented dimensions can be the basis for developing a shared evaluation challenge for QG. A similar evaluation challenge has existed in the past [RG09] but is no longer continued. Such an evaluation challenge can prove to be useful in knowing, e.g., which distractor generation methods are more effective or which knowledge source is better for generating questions for a specific purpose.

9.2.4 Developing a protocol to evaluate QG methods

This thesis contributes to the QG literature by presenting a series of evaluation studies that vary in their nature; while some require some sort of participants (e.g., students, domain/testing experts), others require no participants by utilising automated mechanisms to evaluate the questions. The evaluation studies vary in their goals as well; while some were set up to validate the proposed theory to control MCQs difficulty, others sought to explore the usefulness of the generated questions for different purposes. The use of automated evaluation methods have shown that it is possible to evaluate large numbers of questions, although they cannot evaluate all aspects of the generated questions. In summary, the thesis presents a protocol to evaluate automatically generated questions through user and/or automated studies to asses the difficulty and usefulness of questions.

There is plenty of room to advance the user studies presented in this thesis to cover different important aspects (e.g., different perspectives of usefulness). In addition, the studies can be extended/validated by recruiting larger number of participants and by utilising more ontologies, especially existing ones.

Developing automated question solving tools is an interesting research area in its own. We have seen that it can be used to facilitate the evaluation of a certain class of automatically generated MCQs. This part of the thesis can be extended by developing automated tools to solve other kinds of questions. This can allow to include such an automatic evaluation tool in the workflow of QG generation in order to, e.g., automatically rank the generated questions prior to presenting them to a human expert.

9.2.5 Developing a new family of similarity measures for ontologies

One of the key contributions of this thesis is a new family of similarity measures for ontologies. In Chapter 5, we have presented a review of the psychological foundations of similarity measures and highlighted the importance of taking these foundations into account when developing similarity measures for ontologies. We have discussed and provided examples for the desired properties of similarity measures and justified the need for a new similarity measure for general OWL ontologies. Prior to this thesis, existing similarity measures were applicable to only limited classes of ontologies for different reasons. For example, some measures are applicable only to inexpressive or acyclic ontologies while others require ontologies with ABoxes or additional corpora. Hence, those measures cannot be used for all ontologies.

The experiments presented in Chapter 6 shows that the new similarity measures, in particular $AtomicSim(\cdot)$, $SubSim(\cdot)$ and $GrammarSim(\cdot)$, correlate better with human similarity judgements, compared to some other existing measures. However, we have compared the proposed similarity measures to existing similarity measures in a rather small experiment (in terms of number of ontologies, similarity measures and size of the dataset). The comparison study can be conducted on a larger scale by considering more ontologies and more similarity measures. Indeed, suitable datasets are required to conduct such studies which, to the best of our knowledge, were not available at the time of writing this thesis. In addition, it remains to compare the performance of the different measures for QG purposes.

The members of the new family of similarity measures vary in terms of their computation cost and accuracy. This is why it seemed important to explore different notions of approximations in order to examine whether a cheap measure can be a good approximation for a more expensive one. We have developed a protocol to compare similarity measures of different computational costs based on comparing those measures in terms of the following notions: (1) order preservation, (2) approximation from above, (3) approximation from below, (4) closeness and (5) correlation. Given that the similarity measure $GrammarSim(\cdot)$ is the most expensive measure among the new measures, we have shown that the measure $SubSim(\cdot)$ is better than the measure $AtomicSim(\cdot)$ when considering them as (cheap) "approximations" to $GrammarSim(\cdot)$. Moreover, we have suggested three general scenarios that can be applicable to wide range of applications and have shown, empirically, that some measures can be more suitable than others for accomplishing certain tasks.

To design similarity measures of different computational costs, we had to make decisions to limit the *infinite* set of subsumers of a DL concept. We have discussed the theoretical and practical implications of restricting the (infinite) set of subsumers to (finite) sets of different sizes. We have presented three examples of (finite) sets of subsumers for the three new measures $AtomicSim(\cdot)$, $SubSim(\cdot)$ and $GrammarSim(\cdot)$, in increasing order of cost. We also presented some examples of *weighted* similarity measures as an alternative or additional method to limiting the infinite set of subsumers. A possible future path is to explore how to use these weighted measures to measure similarity with respect to a certain context.

We have shown, empirically, that some of the proposed similarity measures are computationally expensive, especially when dealing with large ontologies. However, they can perfectly be used in applications that do not require on-the-fly computation of similarity. In addition, one of the possible future paths is to further optimise the new similarity in order to reduce their cost. Indeed, it would be very important to optimise the proposed similarity measures in order to use them in applications that require on-the-fly similarity computation.

Our large-scale BioPortal experiment has also shown that it is likely (for over 12% of the ontologies) that using an expensive measure would be of no benefit over using a cheap measure as they will yield exactly the same results. These ontologies were too inexpressive. Hence, expressivity can be used to guide us in deciding when an expensive measure would make no difference. However, after analysing the results of using the different measures over the whole BioPortal corpus, we have reported that we have found no general indicators that can guide us in choosing which similarity measure to use for a given ontology. Further investigations are required to address this issue.

The proposed similarity measures can be also improved by extending their utility to measuring similarity over multiple ontologies. For example, it remains necessary to show how to extend the new measures to measure similarity of two ontologies or two concepts in two different ontologies. Such an extension of the new measures can be combined with lexical similarity measures to support ontology alignment applications.

9.2.6 Developing a protocol to validate ontologies using QG methods

In Chapter 8, we have evaluated the applicability and usefulness of QG methods, not only to assessment applications, but for other applications as well. We have shown, through a case study, that the generated questions can be useful for purposes other than students' assessment, e.g., ontology comprehension, validation and development.

The presented case study is rather small and can be extended in different ways. For example, more experts can be recruited which can allow to review a larger number of questions. Ideally, the extended studies should involve comparisons to a baseline to determine the percentage of errors that can be detected using the suggested QG-based method. In addition, we can measure the usefulness of looking at the generated questions for ontology comprehension purposes by using an independent testing procedure before and after looking at the questions.

Appendix A

Review of existing QG approaches

Table A.1: Some existing QG approaches

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[Bor70]	assessment	language learning	text	-	syntax-based	n/a	wh-questions	free response	-
[Wol75, Wol76]	assessment	language learning	text	patterns	syntax-based	n/a	wh-questions	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[Cla83, Cla87]	assessment	clinical diagnosis	rule-based kb	200 teaching rules	rule-based	n/a	identify the problem	free response	-
[Ste91]	assessment	language learning	text	corpora	syntax-based	morphological variants of the key	fill-in-the-blank	multiple-choice	students (comparison to traditional gap-filler questions)
[Con97]	assessment	language learning	text	corpora	syntax-based	similarity (word class and word frequency)	fill-in-the-blank	multiple-choice	students
[Fai99]	assessment	language learning	text	corpora, patterns	syntax-based	n/a	fill-in-the-blank	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[SB02]	assessment	math	domain schemas	-	schema-based	n/a	math word problems	free response	-
[NBH+02, NBH+06]	assessment	analytical reasoning	domain models	-	not clear	not clear	logical reasoning problems involving complex ordered array	multiple-choice	validate the difficulty prediction model
[DS03]	assessment	math	domain schemas	semantic frames	schema-based	n/a	math word problems	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[MH03, MAHK06]	assessment	general	text	WordNet, corpora, transformation rules	syntax-based + semantics-based	semantic similarity	wh-question	multiple-choice	experts + students
[MBB+04]	assessment	language learning	text	-	syntax-based	same part of speech and same frequency as the key	fill-in-the-blank	multiple-choice	students (predicting test scores)
[BMB04]	assessment	language learning	text	-	syntax-based	frequency or random from a set of hand-annotated words	fill-in-the-blank + which word means	multiple-choice	students (predicting test scores)

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[LWGH05]	assessment	language learning	text	corpora	syntax-based	same part of speech and collocation	fill-in-the-blank	multiple-choice	comparison to questions generated by experts
[BFE05]	assessment	language learning	WordNet	corpus	semantics-based	same part of speech and same frequency as the key	definition, synonym, antonym, hypernym, hyponym, cloze questions	multiple-choice	students (predicting test scores)
[HN05a, HN05b]	assessment	language learning	text	corpora	syntax-based	randomly from the source text	fill-in-the-blank	multiple-choice	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[SSY05]	assessment	language learning	text	thesaurus	syntax-based + semantics-based	similarity in meaning and grammatical form	fill-in-the-blank	multiple-choice	students (item analysis + predicting test scores)
[TG05]	assessment	programming	database of code snippets	domain-dependent templates	template-based	manual based on student misconceptions	What is the output of the program?	multiple-choice	students (evaluate usefulness)
[HMMP05, HMMP06]	assessment	relational databases	ontologies	database schema	schema-based	n/a	write an SQL query	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[CLC06]	assessment	language learning	text	domain-dependent templates, corpora, patterns	template-based	change part of speech or change the verb into different form	fill-in-the-blank	multiple-choice	experts
[LSC07]	assessment	language learning	text	WordNet	semantics-based	Google search to extract colloca-tions and filter out obviously wrong dis-tractors	which of the following adjectives can re-place the adjective in the sentence	multiple-choice	experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[WHL08]	assessment	medicine	text	domain-dependent templates, meta-thesaurus, semantic network, lexicon	template-based	n/a	wh-questions	free response	subjects
[SSIS08]	assessment	general	text	user defined templates, user defined tags	syntax-based	n/a	wh-questions	free response	-
[Kim08]	assessment	general	text	-	syntax-based	n/a	wh-questions	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[ZSRG08]	assessment	general	ontologies	domain-independent templates	semantics-based	random	What is X?, Who has R [Value]?, others	multiple-choice	-
[PKK08]	assessment	general	ontologies	-	semantics-based	custom strategies (mainly siblings of the key)	choose the correct sentence	multiple-choice	experts
[MSK09]	knowledge acquisition	general	ontologies	domain-independent templates	semantics-based	n/a	wh-questions	free response	-
[MC09]	assessment	language learning	text	domain-dependent templates	template-based	n/a	wh-questions	free response	experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[HS09, HS10a, HS10b, Hei11]	assessment	general	text	-	syntax-based	n/a	wh-questions	free response	experts
[CT09, CT10]	assessment	general	ontologies	domain-independent templates	semantics-based	custom strategies (adopted from [PKK08])	What is X?, What is an example of X?, others	multiple-choice	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[GK10]	assessment	language learning	text	WordNet	syntax-based	derivatives of the same prefix or suffix, synonym or antonym to key	fill-in-the-blank	multiple-choice	comparison with existing methods
[YZ10, YBZ12]	general	general	text	(optional) ontologies or semantic networks	semantics-based	n/a	wh-questions + T/F	free response	experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[CMC11]	general	general	text	corpora, patterns (learned from corpora)	syntax-based	n/a	wh-questions	free response	-
[NB11]	assessment	math	data store	domain-dependent templates	template-based	n/a	math word problem	free response	students
[PKK11]	assessment	general	ontologies	annotation for images	semantics-based	custom strategies	choose the correct sentence	multiple-choice	experts
[Wil11]	assessment	math	ontologies	domain-dependent templates	template-based	n/a	math word problem	free response	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[ZSCZ11]	knowledge acquisition	general	text	user queries, auto. induced question templates	template-based	n/a	wh-questions	free response	coverage
[dM11]	knowledge acquisition	general	linked data	auto. built lattice	semantics-based	n/a	wh-questions	free response	experts
[BDDS11]	validation	general	domain models	-	semantics-based	n/a	T/F	multiple-choice (T/F)	experts
[ASM11]	assessment	general	text	-	syntax-based	n/a	why, when, give an example	free response	check syntactic and semantic correctness

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[AM11]	assessment	general	text	-	syntax-based	contextual similarity, part of speech, frequency	fill-in-the-blank	multiple-choice	experts (evaluate quality of distractors)
[AP11, AP13]	knowledge acquisition	science	ontologies	a reference scenario	semantics-based	random	What do you think about ..?, Select a hypothesis	multiple-choice	students (evaluate effectiveness to enhance learning)
[AY11, AY14]	assessment	general	ontologies	-	semantics-based	similarity (manually elicited from experts)	What is the kind of X?, T/F, fill-in-the-blank	multiple-choice	experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[AMF11, AM14]	assessment	general	text	corpora, auto. extracted semantic patterns	semantics-based	contextual similarity	wh-questions	multiple-choice	experts
[KBS+12]	knowledge acquisition	general	text	-	syntax-based	n/a	wh-questions	free response	students (evaluate effectiveness to enhance comprehension)

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[MGL12]	assessment	general	text	WordNet	syntax-based + semantics-based	latent semantic analysis, part of speech, weight of concepts	fill-in-the-blank	multiple-choice	experts (evaluate quality of distractors)
[CBEM12]	assessment	language learning	text	corpora	syntax-based	not clear	fill-in-the-blank	multiple-choice	experts (evaluate quality of stems)
[Ata12]	assessment	general	ontologies	lecture notes	semantics-based	not clear	wh-questions	multiple-choice	students + experts
[BBV12]	assessment	language learning	text	-	syntax-based	n/a	fill-in-the-blank	free response	experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[GLT12]	assessment	clinical diagnosis	domain models	domain-dependent templates	template-based	manual	what is the best next step?	multiple-choice	-
[OGP12]	assessment	general	text	semi-auto. generated concept maps	semantics-based	n/a	T/F, what, how	free response	experts
[MEAF12]	general	general	text	-	syntax-based	n/a	wh-questions	free response	experts
[WLHL12]	assessment	logic	semantic networks	-	semantics-based	algorithm	logic questions	multiple-choice	-

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[LCR12, LCR14]	validation	writing support	text	domain-dependent templates	template-based	n/a	6 categories of questions to enhance literature review writing	free response	comparison to questions generated by experts
[LC12]	validation	writing support	text	domain-dependent templates, Wikipedia, patterns	template-based	n/a	3 categories of questions to enhance literature review writing	free response	comparison to questions generated by experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[LCAP12]	validation	writing support	text	domain-dependent templates, Wikipedia	semantics-based	n/a	9 categories of questions to enhance literature review writing	free response	comparison to questions generated by experts
[BK12a, BK12b]	assessment	general	text	annotations	syntax-based + semantics-based	similarity	fill-in-the-blank	multiple-choice	students
[DMMS13]	knowledge acquisition	general	text	auto. induced question templates	template-based	n/a	wh-questions	free response	automatic evaluation + manual evaluation

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[KS13]	assessment	general	ontologies	-	semantics-based	similarity	fill-in-the-blank	multiple-choice	-
[MN14]	assessment	general	text	WordNet, semantic patters	semantics-based	n/a	wh-questions	free response	comparison with existing methods
[CCSO14]	knowledge acquisition	general	knowledge base	user queries	semantics-based	n/a	What is X?, What is the R of X?, others	free response	students (evaluate effectiveness in enhancing students' learning)

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[PMSS14]	assessment	general	text	domain-independent templates	syntax-based	n/a	What is X?, Give an example of X?, others	free response	-
[HTSC14]	assessment	general	videos	WordNet, Corpus	syntax-based	replacing the head word of the answer phrase with words of same part of speech	what is the main idea? + wh-questions	multiple-choice	comparison to questions generated by experts

Continued on next page

Table A.1 – continued from previous page

Bib. ref.	Purpose	Domain	Knowledge source	Additional input	QG Method	Distractor generation method	Question format	Answer format	Evaluation
[SH14]	assessment	geometry	database of geometric figures	-	rule-based	n/a	geometry problems	free response	experts
[JS14]	assessment	history	Wikipedia, DBpedia, Freebase	domain-dependent templates	template-based	n/a	depends on used templates	free response	-
[KWDH14]	assessment	clinical diagnosis	probabilistic knowledge base	-	semantics-based	not clear	choose and rank from the following hypothesis	multiple-response	students

Appendix B

Questions used for evaluation

Table B.1: Questions generated from the KA ontology

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
1	What is the following definition describing? "KA technique used to capture the way in which an expert views the concepts in a domain. Involves presenting three random concepts and asking in what way two of them are similar but different from the other one."	Triadic elicitation	Diagram-based technique	Concept map technique	Hierarchy generation technique	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
2	What is the following definition describing? “The things (physical objects, information, people, etc.) that constitute a domain.”	Concept	Instance	Value	Attribute	Difficult
3	What is the following definition describing? “Shows the alternative courses of action for a particular decision. It also shows the pros and cons for each course of action.”	Decision ladder	Composition ladder	Attribute ladder	Process ladder	Difficult
4	What is the following definition describing? “An interview in which a pre-prepared set of questions is used to focus and scope what is covered, but which also involves unprepared supplementary questions to clarify and probe what the expert has said.”	Semi-structured interview	Laddering	Observation	Teach back	Easy
5	What is the following definition describing? “Knowledge acquisition technique in which the expert reports on their own or another?s task performance.”	Commentary	Unstructured interview	Structured interview	Interview	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
6	What is the following definition describing? “Shows the way a knowledge object is composed of its constituent parts”.	Composition ladder	Attribute ladder	Process ladder	Decision ladder	Easy
7	What is the following definition describing? “The specific property of an object or relationship such as its actual weight or age.”	Value	Instance	Concept	Attribute	Difficult
8	What is the following definition describing? “A quality or feature belonging to a concept, e.g. weight, maximum speed, age, duration.”	Attribute	Instance	Value	Relationship	Difficult
9	Which of these is the odd one out?	Weight	Fast	Accurate	Blue	Easy
10	Which of these is the odd one out?	Process laddering	Unstructured interview	Structured interview	Interview	Easy
11	Which of these is the odd one out?	HasPros	Book	Sky	Table	Easy
12	Which of these is the odd one out?	Velocity	Requires	HasValue	PartOf	Easy
13	A Process Map is ....:	a network diagram	a structured text	a symbolic character-based language	a hypertext	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
14	Logic is ....:	a symbolic character-based language	a network diagram	a hypertext	a matrix	Easy
15	PartOf is ....:	relationship	weight	speed	colour	Easy
16	A state transition technique is ....:	a diagram-based technique	a matrix-based technique	a protocol analysis technique	a hierarchy generation technique	Difficult
17	A ladder is ....:	a diagrammatic representation	a structured text	a symbolic character-based language	a tabular representation	Difficult
18	A process map technique is ....:	a diagram-based technique	a matrix-based technique	a protocol analysis technique	a hierarchy generation technique	Difficult
19	Table is ....:	a concept	an instance	a value	a relationship	Difficult
20	Colour is ....:	attribute	hasPros	is_a	requires	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
21	Commentary is ....:	protocol generation technique	process laddering technique	matrix-based technique	attribute laddering technique	Easy
22	Sky is ....:	a concept	an instance	a value	a relationship	Difficult
23	What is the following definition describing? "Type of knowledge object that represents the way other knowledge objects are related to one another. Usually represented as a labelled arrow on a network diagram."	Relationship	Instance	Value	Concept	Difficult
24	What is the following definition describing? "A way of representing knowledge in which each concept in a domain is described by a group of attributes and values using a matrix representation"	Frame	Hypothesis-diagnostic matrix	Problem-solution matrix	Concept-property matrix	Easy
25	Which one is the odd one out?	Laddering technique	State transition technique	Concept map technique	Process map technique	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
26	A structured interview is ....:	targeted at explicit knowledge	targeted at role knowledge or time-based knowledge	a tabular representation targeted at role knowledge or time-based knowledge	targeted at tacit knowledge	Easy
27	Limited-information and constrained-processing tasks is ....:	good at efficiency	good at flexibility	good at uncovering hidden correlations	good at recall	Difficult

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
28	Limited-information and constrained-processing tasks is ....:	targeted at procedural knowledge	targeted at declarative knowledge	targeted at role knowledge or time-based knowledge	a tabular representation targeted at role knowledge or time-based knowledge	Easy
29	A process ladder is ....:	targeted at procedural knowledge	targeted at declarative knowledge	targeted at role knowledge or time-based knowledge	a tabular representation targeted at role knowledge or time-based knowledge	Easy
30	A process ladder is ....:	composed of relationships	composed of values	composed of attributes	composed of concepts	Difficult

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
31	A state transition network is ....:	produced by state transition technique	diagrammatic representation produced by hierarchy generation technique	tabular representation produced by matrix-based technique or structured interview	produced by hierarchy generation technique	Easy
32	A concept map is ....:	a network diagram produced by concept map technique	a network diagram produced by state transition technique	a diagrammatic representation produced by hierarchy generation technique	a network diagram produced by process map technique	Difficult

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
33	Twenty-questions technique is ....:	good at time efficiency	good at flexibility	good at recall	good at uncovering hidden correlations	Difficult
34	A repertory grid technique ....:	involves selecting attributes	involves self-reporting or shadowing	involves identifying knowledge objects	involves observing	Difficult
35	A repertory grid technique ....:	involves repertory grid stage 1	involves self-reporting or shadowing	involves identifying knowledge objects	involves observing	Easy
36	Which of the following is a concept?	Table	HasPros	Weight	Speed	Easy
37	Which of the following is a hierarchy generation technique?	Process laddering technique	State transition technique	Unstructured interview	Process map technique	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
38	Which of the following is a diagram-based technique?	Concept map technique	Triadic elicitation	Card sorting	Interview	Easy
39	Which of the following is a protocol generation technique?	Unstructured interview	Process laddering technique	Attribute laddering technique	State transition technique	Easy
40	Which of the following is laddering technique?	Composition laddering technique	Matrix-based technique	State transition technique	Unstructured interview	Easy
41	Which of the following is a matrix that is produced by cluster analysis?	Focus matrix	Hypothesis-diagnostic matrix	Problem-solution matrix	Concept-property matrix	Difficult
42	Which of the following is a diagrammatic representation that is produced by a hierarchy generation technique?	Ladder	Structured text	Symbolic character-based language	Tabular representation	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
43	Which of the following is good at flexibility?	Semi-structured interview	Unstructured interview	Structured interview	Interview	Difficult
44	Which of the following is good at structured agenda?	Semi-structured interview	Laddering technique	Commentary	KA technique	Easy
45	Which of the following is good at efficiency?	Limited-information and constrained-processing tasks	Matrix-based technique	Triadic elicitation	State transition technique	Easy
46	Which of the following is targeted at initial interviews?	Unstructured interview	Structured interview	Semi-structured interview	Interview	Difficult
47	Which of the following is composed of concepts?	Concept ladder	Attribute ladder	Process ladder	Decision ladder	Easy

Continued on next page

Table B.1 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
48	Which of the following produces a matrix?	Structured interview	Unstructured interview	Interview	Semi-structured interview	Difficult
49	Which of the following produces a protocol?	Structured interview	Process laddering technique	Attribute laddering technique	Timeline technique	Easy
50	Which of the following is targeted at tacit knowledge?	Sorting technique	Diagram-based technique	Concept map technique	KA technique	Difficult

Table B.2: Questions generated from the Java ontology

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
1	“Char” is .....:	Primitive data type	Instance	Simple object	Actual parameter	Easy

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
2	“Swing” has fundamental feature .....:	Accessibility	Fast program execution	Re-usability	Platform independence	Easy
3	“Real machine code” has fundamental feature .....:	Fast program execution	Write once run anywhere	Portability	Platform independence	Easy
4	A container has .....:	Layout manager	Generic type	Method body	Default value	Easy
5	..... refers to “A non-static member variable of a class.”:	Instance variable	Loop variable	Reference variable	Primitive variable	Difficult
6	..... refers to “A Java keyword when applied to a variable is the Java equivalent of a ’constant’, that belongs to a class, not an object of a class.”:	Static Final	Implements	Throws	This	Easy
7	..... refers to “The interface to a class used by a programmer.”:	API	Concrete class	Mutator method	Map interface	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
8	…. refers to “A GUI complement that provides space in which an application can attach any other component.”:	Panel	Check box	Frame	Button	Difficult
9	…. refers to “A method used to set the value of an instance variable.”:	Mutator method	Interface method	Receiving method	Overloaded method	Difficult
10	…. refers to “The process of catching one type of exception and immediately throwing another, of a more appropriate class.”:	Converting exception	Runtime exception	Exception handling	Checked exception	Difficult
11	…. refers to “A type of binary operator that is used to handle one condition.”:	Relational operator	Additive operator	Unary operator	Equality operator	Difficult
12	…. refers to “A primitive data type that can hold a 32-bit signed two’s complement integer.”:	Int	Short	Char	Boolean	Easy

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
13	..... refers to “A stage in the software development process where customer needs are elicited.”:	Requirement analysis	Testing	Design	Implementation	Easy
14	..... refers to “A primitive data type that holds a 16-bit data type and has a minimum value of -32,768 and a maximum value of 32,767 (inclusive).”:	Short	Int	Boolean	Byte	Easy
15	Which of these is a primitive data type?	Float	Key event	Command line argument	Final	Easy
16	Which of these is a Java keyword?	Final	Immutable object	Key event	Complex object	Easy
17	..... refers to “A control flow statement used to decide whether to do something at a special point, or to decide between two courses of action.”:	If Else	While Loop	For Loop	Return statement	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
18	Which of these is also known as scalar?	Primitive data type	Instance	Object	Parameter	Easy
19	Which of these is also known as parameter?	Argument	Method signature	Scalar	Double	Easy
20	Which of these contains parameter?	Method signature	Immutable object	Simple object	Mouse event	Easy
21	Which of these declares Immutable object?	Final	Float	Short	Action event	Easy
22	Which of these has value True or has value False?	Boolean	Int	Short	Float	Easy
23	Which of these is the odd one out?	Input stream reader	Java compiler	Javac	JIT	Difficult
24	Which of these is the odd one out?	Iterator	Operator	Addition	Modulus	Difficult
25	Which of these is the odd one out?	Input stream reader	Array list	Hash map	Tree map	Difficult
26	Which of these is the odd one out?	Final	Int	Float	Short	Easy
27	Which of these is the odd one out?	Iterator	Loop condition	Initial statement	Continuation statement	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
28	Which of these is the odd one out?	Iterator	Buffered writer	Graphics class	Interface method	Difficult
29	..... refers to "A control flow statement that allows code to be executed repeatedly based on a given Boolean condition, where the code within the block is always executed first, and then the condition is evaluated.":	Do While Loop	While Loop	For Loop	For Each Loop	Difficult
30	..... refers to "A Java feature that supports the automatic conversion of object wrapper types to their corresponding primitive equivalents, if needed for assignments and method and constructor invocations.":	AutoUnBoxing	Converting Exception	AutoBoxing	Inheritance	Difficult
31	..... refers to "Data identifiers that are used to refer to specific values that are generated in a program.":	Variable	Assertion	Assignment	Abstract class	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
32	….. refers to "An access modifier that allows a method or data variable to be accessed by any other class anywhere.":	Public	Generic	Void	Abstract	Difficult
33	Which of these is composed of simple objects?	Complex object	Event object	Method signature	Argument	Easy
34	Which of these is also known as object?	Instance	Byte	Char	Method signature	Easy
35	….. refers to "A powerful Java programming language feature that allows types (classes and interfaces) and methods to operate on classes/objects of various types while providing compile-time safety.":	Generic	Polymorphism	Exception handling	Inheritance	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
36	..... refers to “A Java keyword used to declare an abstract type class which can be implemented by classes that declare the name of this class with the implements keyword.”:	Interface	Assert	Applet	Void	Difficult
37	Virtual machine code has fundamental feature .....:	Portability	Fast program execution	Write once run anywhere	Reusability	Easy
38	..... refers to “A reference type, similar to a class, that is used to define the mechanism by which an object is used and can contain only constants, method signatures, default methods, static methods, and nested types.”:	Interface	Mutator method	Reader	Writer buffer	Difficult
39	..... refers to “An individual object created from the blueprints specified by a class.”:	Instance	Method signature	Actual parameter	Byte	Easy
40	..... refers to “A statement that happens before the loop is executed.”:	Initial statement	Instantiation statement	Branching statement	Decision making statement	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
41	..... refers to “An OOP feature that allows a new class to be derived from a class that has already been defined. It is a mechanism for code reuse.”:	Inheritance	Autoboxing	Encapsulation	Polymorphism	Difficult
42	..... refers to “A special type of instance method that creates a new object, has the same name as the class and has no return value in the declaration.”:	Constructor	Concrete class	Mutator method	Map interface	Difficult
43	..... refers to “A Java keyword used in several different contexts to define an entity which cannot later be changed.”:	Final	Int	Parameter	Byte	Easy
44	..... refers to “A Java keyword that refers to a method or variable that is not attached to a particular object, but rather to the class as a whole.”:	Static	Throws	This	Implements	Easy

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
45	..... refers to “A collection of objects or primitives whose size is fixed at the point it is created (but not at compile time) and is accessed by indexing.”:	Array	Tree set	Hash set	Hash map	Difficult
46	..... refers to “An access modifier that allows a method or data variable to be accessible only within the class.”:	Private	Protected	Void	Abstract	Difficult
47	..... refers to “A class that cannot be instantiated, but can be subclassed.”:	Abstract class	Concrete class	Component class	Input stream class	Difficult
48	..... refers to “A mechanism for organising Java classes belonging to the same category or have similar functionality, providing access protection and name space management.”:	Package	Collection	Applet	Interface	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
49	..... refers to “A mechanism where a method being defined is called within its own definition.”:	Recursion	Object casting	Garbage collection	Dynamic binding	Difficult
50	..... refers to “The values passed to a method.”:	Parameter	Instance	Int	Byte	Easy
51	..... refers to “The process of converting a general reference to something more specific, a subclass of that general type.”:	Object casting	Converting exception	Map interface	Autoboxing	Difficult
52	..... refers to “A compiler that performs compilation during execution of a program (at run time) ? rather than prior to execution.”:	JIT	API	Buffered writer	Applet	Difficult
53	..... refers to “A Java keyword used to test if an object is of a specified type.”:	InstanceOf	Super	IsAKindOf	Extends	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
54	..... refers to “An ordered collection (sometimes called a sequence) of elements, where each element can occur more than once within the collection.”:	List	Set	Hash set	Map	Difficult
55	..... refers to “A fundamental Java, OOP construct that is the blueprint from which individual objects are created.”:	Class	Package	Catch block	Overloaded Method	Difficult
56	..... refers to “The ability to define more than one method with the same name in the same class, but that differ in the types of their arguments.”:	Method overloading	Receiving method	Mutator method	Interface method	Difficult
57	..... refers to “A statement used to give a value to a variable.”:	Assignment	Initial statement	Assertion	Object casting	Difficult
58	..... refers to “An instance of a class.”:	Object	Primitive data type	Scalar	Argument	Easy

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
59	..... refers to “The process of automatic memory management used to destroy objects that are no longer used by the program.”:	Garbage collection	Run time exception	CRC	Object factory	Difficult
60	..... refers to “A fundamental OOP feature used by objects to communicate with each other.”:	Message passing	Event listener	Interface method	Applet	Difficult
61	..... refers to “An object which allows a programmer to traverse through all the elements of a collection, regardless of its specific implementation.”:	Iterator	Buffered writer	Loop variable	Recursion	Difficult
62	..... refers to “A computer program that executes virtual machine code (Java byte codes), generated from compiling Java source code.”:	JVM	API	Java compiler	AWT	Difficult

Continued on next page

Table B.2 – continued from previous page

ID	Stem	Key	Distractor 1	Distractor 2	Distractor 3	Predicted difficulty
63	.... refers to "A Java feature that supports the automatic conversion of primitive types to their corresponding object wrapper classes, in assignments and method and constructor invocations.":	AutoBoxing	AutoUnBoxing	Exception handling	Inheritance	Difficult
64	.... refers to "An OOP property where the exact method implementation is determined by both the method name and the receiving object at the run-time, and not at compile-time.":	Dynamic binding	Object casting	Garbage collection	Autoboxing	Difficult
65	.... refers to "The process of generalisation by reducing the information content of a concept, in order to retain only information which is relevant for a particular purpose.":	Abstraction	Recursion	Event handling	Initialisation	Difficult

Table B.3: Questions generated from SNOMED CT

ID	Stem	Key	Distractors	Predicted difficulty
1	“k+ phenotype (finding)” is:	“K+ phenotype (finding)”	“K+k+ phenotype (finding)” “K-k- phenotype (finding)” “K-k+ phenotype (finding)” “Kell null phenotype (finding)” “Kell mod phenotype (finding)” “K- phenotype (finding)” “K+k- phenotype (finding)”	Difficult
2	“Blood group Bel (finding)” is:	“Blood group B (finding)”	“Blood group AB (finding)” “Blood group A variant (finding)” “Blood group A 1 (finding)” “Blood group A3B (finding)” “Blood group A1B (finding)” “Blood group A h (finding)” “Blood group O (finding)” “Blood group A2B (finding)” “Blood group A (finding)” “Blood group antigen A variant (finding)”	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
3	“Blood group A2 (finding)” is:	“Blood group phenotype (finding)”	“Finding of Rh blood group (finding)” “Finding of common composite blood group (finding)” “Duffy blood group (finding)” “Finding of minor blood group (finding)”	Easy
4	“ABO blood grouping (procedure)” is:	“Laboratory procedure (procedure)”	“Blood bank procedure (procedure)”	Easy
5	“Weak Fyb̂p̂henotype (finding)” is:	“Duffy blood group phenotype (finding)”	“Blood group Oh Bombay Indian type (finding)” “Yus type (finding)” “McLeod phenotype (finding)” “Rh negative Du positive (finding)” “Jk(a+b+) phenotype (finding)” “P2 phenotype (finding)” “Kx blood group phenotype (finding)” “Blood group O (finding)” “Weak E phenotype (finding)” “Weak S phenotype (finding)”	Easy

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
6	"Blood group B3 (finding)" is:	"Blood group B (finding)"	"Blood group AB (finding)" "Blood group A 2 (finding)" "Blood group A variant (finding)" "Blood group A 1 (finding)" "Blood group A3B (finding)" "Blood group A1B (finding)" "Blood group A h (finding)" "Blood group O (finding)" "Blood group A2B (finding)" "Blood group A (finding)" "Blood group antigen A variant (finding)"	Difficult
7	"Blood group typing (procedure)" is:	"Laboratory procedure (procedure)"	"Blood bank procedure (procedure)"	Easy

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
8	“Blood group Bh (finding)” is:	“Blood group B (finding)”	“Blood group AB (finding)” “Blood group A 2 (finding)” “Blood group A variant (finding)” “Blood group A 1 (finding)” “Blood group A3B (finding)” “Blood group A1B (finding)” “Blood group A h (finding)” “Blood group O (finding)” “Blood group A2B (finding)” “Blood group A (finding)” “Blood group antigen A variant (finding)”	Difficult
9	“Immunologic procedure (procedure)” is:	“Laboratory procedure (procedure)”	“Blood bank procedure (procedure)”	Easy

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
10	"Blood group Bm (finding)" is:	"Blood group B (finding)"	"Blood group AB (finding)" "Blood group A 2 (finding)" "Blood group A variant (finding)" "Blood group A 1 (finding)" "Blood group A3B (finding)" "Blood group A1B (finding)" "Blood group O (finding)" "Blood group A2B (finding)" "Blood group A (finding)" "Blood group antigen A variant (finding)"	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
11	“Blood group AB Rh(D) negative (finding)” is:	“Blood group AB (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)”	“Blood group B (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)” “RhD negative (finding)” and “Finding of common composite blood group (finding)” and “Blood group O (finding)” “Blood group AB (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)”) “Blood group A (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)”	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
12	“Blood group A Rh(D) positive (finding)” is:	“Blood group A (finding)” and ”RhD positive (finding)” and “Finding of common composite blood group (finding)”	“Blood group B (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)” “RhD positive (finding)” and “Finding of common composite blood group (finding)” and “Blood group O (finding)” “Blood group AB (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)” “Blood group A (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)”	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
13	“Blood group A Rh(D) negative (finding)” is:	“Blood group A (finding)” and ”RhD negative (finding)” and “Finding of common composite blood group (finding)”	“Blood group A (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)” “Blood group B (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)” “RhD negative (finding)” and “Finding of common composite blood group (finding)” and “Blood group O (finding)” “Blood group AB (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)”	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
14	“Blood group B Rh(D) positive (finding)” is:	“Blood group B (finding)” and ”RhD positive (finding)” and “Finding of common composite blood group (finding)”	“Blood group A (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)” “Blood group B (finding)” and “RhD negative (finding)” and “Finding of common composite blood group (finding)” “RhD positive (finding)” and “Finding of common composite blood group (finding)” and “Blood group O (finding)” “Blood group AB (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)”	Difficult
15	“Finding of common composite blood group (finding)” is:	“Finding of blood group (finding)” and “Interprets (attribute)” “ABO and Rho(D) typing (procedure)”	“Finding of blood group (finding)” and “Phenotype finding (finding)”	Difficult

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
16	“Blood group A Rh(D) positive (finding)” is:	“Blood group A (finding)” and “RhD positive (finding)” and “Finding of common composite blood group (finding)”	“Blood group phenotype (finding)” and “Finding of minor blood group (finding)” “Hh blood group phenotype (finding)” and “Blood group O (finding)” “Blood group B (finding)” and “Blood group Para-Bombay (finding)”	Easy
17	“Inherited weak D phenotype (finding)” is:	“Blood group phenotype (finding)” and “Finding of Rh blood group (finding)”	“Blood group phenotype (finding)” and “Finding of ABO blood group (finding)” “Blood group phenotype (finding)” and “Duffy blood group (finding)” “Blood group B (finding)” and “Blood group Para-Bombay (finding)” “Blood group phenotype (finding)” and “Finding of minor blood group (finding)”	Easy

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
18	“Rr-̂blood group phenotype (finding)” is:	“Blood group phenotype (finding)” and “Finding of Rh blood group (finding)”	“Blood group phenotype (finding)” and “Duffy blood group (finding)” “Blood group B (finding)” and “Blood group Para-Bombay (finding)” “Blood group phenotype (finding)” and “Finding of minor blood group (finding)” “Blood group phenotype (finding)” and “Finding of ABO blood group (finding)”	Easy
19	“Rh null phenotype (finding)” is:	“Blood group phenotype (finding)” and “Finding of Rh blood group (finding)”	“Hh blood group phenotype (finding)” and “Blood group O (finding)” “Blood group phenotype (finding)” and “Finding of minor blood group (finding)” “Blood group phenotype (finding)” and “Finding of ABO blood group (finding)”	Easy

Continued on next page

Table B.3 – continued from previous page

ID	Stem	Key	Distractors	Predicted difficulty
20	"Trans weak D phenotype (finding)" is:	"Blood group phenotype (finding)" and "Finding of Rh blood group (finding)"	"Blood group phenotype (finding)" and "Duffy blood group (finding)" "Blood group B (finding)" and "Blood group Para-Bombay (finding)" "Blood group phenotype (finding)" and "Finding of minor blood group (finding)" "Blood group phenotype (finding)" and "Finding of ABO blood group (finding)"	Easy

Bibliography

[AB85] F. Allard and N. Burnett. Skill in sport. *Canadian Journal of Psychology*, 39(2):294–312, 1985. (Cited on page 75.)

[ABB$^+$00] M. Ashburner, C. Ball, J. Blake, D. Botstein, H. Butler, J. M. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig, M. Harris, D. Hill, L. Issel-Tarver, A. Kasaerskis, S. Lewis, J. Matese, J. Richardson, M. Ringwald, G. Rubin, and G. Sherlock. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000. (Cited on page 40.)

[Ach00] Achieve. Testing: Setting the record straight. Technical report, Washington, DC: Author, 2000. (Cited on page 17.)

[ACKZ09] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *Journal of Artificial Intelligence Research*, 36:1–69, 2009. (Cited on page 34.)

[Aik82] L. Aiken. Writing multiple-choice items to measure higher-order educational objectives. *Educational and Psychological Measurement*, 42(3):803–806, 1982. (Cited on pages 53 and 77.)

[AK00] L. W. Anderson and D. R. Krathwohl, editors. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives.* Allyn and Bacon, 2000. (Cited on pages 10 and 64.)

[AM92] M.G. Aamodt and T. McShane. A meta-analytic investigation of the effect of various test item characteristics on test scores. *Public Personnel Management*, 21(2), 1992. (Cited on page 19.)

[AM11] M. Agarwal and P. Mannem. Automatic gap-fill question generation from text books. In *IUNLPBEA '11: Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, 2011. (Cited on pages 98 and 217.)

[AM14] N. Afzal and R. Mitkov. Automatic generation of multiple choice questions using dependency-based semantic relations. *Soft Computing*, 2014. (Cited on pages 21, 91, 94, and 218.)

[AMF11] N. Afzal, R. Mitkov, and A. Farzindar. Unsupervised relation extraction using dependency trees for automatic generation of multiple-choice questions. In *C. Butz and P. Lingras (Eds.): Canadian AI 2011, LNAI 6657*, page 3243, 2011. (Cited on pages 91, 94, and 218.)

[And74] J. R. Anderson. Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6(4):451474, 1974. (Cited on page 74.)

[And83a] J. R. Anderson. *The Architecture of Cognition.* Cambridge, MA: Harvard University Press, 1983. (Cited on page 74.)

[And83b] J. R. Anderson. A spreding activation theory of memory. *Journal of Verbal Learning and Verbal Behavior*, 22:261–295, 1983. (Cited on page 71.)

[And93] J. R. Anderson. *Rules of the mind.* Hillsdale, NJ: Erlbaum, 1993. (Cited on page 74.)

[And95] J. R. Anderson. *Cognitive psychology and its implications.* New York: W. H. Freeman, 4 edition, 1995. (Cited on page 70.)

[And07] J. R. Anderson. *How Can the Human Mind Occur in the Physical Universe?* Oxford Series on Cognitive Models and Architectures, 2007. (Cited on page 74.)

[AP11] S. Ahmed and D. Parsons. ThinknLearn: An ontology driven mobile web application for science enquiry based learning. In *proceedings of the 7th International Conference on Information Technology and Applications (ICITA 2011)*, 2011. (Cited on pages 87, 88, and 217.)

[AP13] S. Ahmed and D. Parsons. Abductive science inquiry using mobile devices in the classroom. *Computers & Education*, 63, 2013. (Cited on pages 97 and 217.)

[APS12a] T. Alsubait, B. Parsia, and U. Sattler. Automatic generation of analogy questions for student assessment: an ontology-based approach. *Research in Learning Technology*, 20:95–101, 2012. (Cited on page 31.)

[APS12b] 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. (Cited on page 31.)

[APS12c] T. Alsubait, B. Parsia, and U. Sattler. Mining ontologies for analogy questions: A similarity-based approach. In *OWLED*, 2012. (Cited on page 30.)

[APS12d] 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. (Cited on page 31.)

[APS13] 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. (Cited on page 31.)

[APS14a] T. Alsubait, B. Parsia, and U. Sattler. Generating multiple choice questions from ontologies: How far can we go? In *Proceedings of the First International Workshop on Educational Knowledge Management (EKM 2014)*, 2014. (Cited on page 31.)

[APS14b] 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. (Cited on page 31.)

[APS14c] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in ontologies: A new family of measures. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference (ISWC 2014)*, 2014. (Cited on page 31.)

[APS14d] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in ontologies: A new family of measures. In *Proceedings of The 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2014)*, 2014. (Cited on page 31.)

[APS14e] T. Alsubait, B. Parsia, and U. Sattler. Measuring similarity in ontologies: How bad is a cheap measure? In *27th International Workshop on Description Logics (DL-2014)*, 2014. (Cited on page 31.)

[ASM11] M. Agarwal, R. Shah, and P. Mannem. Automatic question generation using discourse cues. In *IUNLPBEA '11: Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, 2011. (Cited on page 216.)

[Ata12] T. Atapattu. Automated generation of practice questions from semi-structured lecture notes. In *ICER '12: Proceedings of the ninth annual international conference on International computing education research*, 2012. (Cited on page 219.)

[AY11] M. Al-Yahya. OntoQue: A question generation engine for educational assessment based on domain ontologies. In *11th IEEE International Conference on Advanced Learning Technologies*, 2011. (Cited on pages 21, 62, and 217.)

[AY14] M. Al-Yahya. Ontology-based multiple choice question generation. *The Scientific World Journal*, 2014. (Cited on pages 21, 62, 87, 88, 91, 94, 96, 98, 99, and 217.)

[Baa91] F. Baader. Augmenting concept languages by transitive closure of roles: an alternative to terminological cycles. In *Proceedings of the 12th international joint conference on Artificial intelligence (IJCAI-91), San Francisco, CA, USA*, pages 446–451. Morgan Kaufmann Publishers Inc., 1991. (Cited on page 43.)

[Baa03] F. Baader. Least common subsumers and most specific concepts in a description logic with existential restrictions and terminological cycles. In *Georg Gottlob and Toby Walsh, editors, Proceedings of the 18th International Joint Conference on Artificial Intelligence. Morgan Kaufmann*, pages 319–324, 2003. (Cited on page 46.)

[BAB05] O. Bodenreider, M. Aubry, and A. Burgun. Non-lexical approaches to identifying associative relations in the Gene Ontology. *Pac Symp Biocomput*, page 91102, 2005. (Cited on pages 103 and 133.)

[Bai13] S. Bail. *The Justificatory Structure of OWL ontologies.* PhD thesis, The University of Manchester, 2013. (Cited on page 46.)

[Bak01] Frank Baker. *The Basics of Item Response Theory.* ERIC Clearinghouse on Assessment and Evaluation, University of Maryland, College Park, MD, 2001. (Cited on page 55.)

[Bal83] D. A. Balota. Automatic semantic activation and episodic memory encoding. *Journal of Verbal Learning and Verbal Behavior*, 22:88–104, 1983. (Cited on page 71.)

[BBL05] F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh, Scotland, UK*, page 364369, 2005. (Cited on pages 34, 39, and 45.)

[BBV12] L. Becker, S. Basu, and L. Vanderwende. Mind the gap: Learning to choose gaps for question generation. *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 742–751, 2012. (Cited on pages 90, 91, 97, 98, and 219.)

[BCM05] P. Buitelaar, P. Cimiano, and B. Magnini. *Ontology Learning from Text: Methods, Evaluation and Applications, P. Buitelaar, P. Cimiano, and B.Magnini, eds.*, volume 123, chapter Ontology Learning from Text: An Overview, pages 3–12. Frontiers in Artificial Intelligence and Applications, IOS Press, 2005. (Cited on page 93.)

[BCM$^+$07] 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. (Cited on pages 23, 33, and 40.)

[BDDS11] A. Bertolino, G. DeAngelis, A. DiSandro, and A. Sabetta. Is my model right? let me ask the expert. *Journal of Systems and Software*, 84(7):1089–1099, 2011. (Cited on pages 89, 97, 98, 194, and 216.)

[BF14] A. K. Bright and A. Feeney. The engine of thought is a hybrid: Roles of associative and structured knowledge in reasoning. *Journal of Experimental Psychology: General*, 143(6):2082–2102, Dec 2014. (Cited on page 70.)

[BFE05] J. Brown, G. Firshkoff, and M. Eskenazi. Automatic question generation for vocabulary assessment. In *Proceedings of HLT/EMNLP*, pages 819–826, Vancuver, Canada, 2005. (Cited on pages 21, 62, 95, 97, 98, and 208.)

[BGSS07] F. Baader, B. Ganter, B. Sertkaya, and U. Sattler. Completing description logic knowledge bases using formal concept analysis. In *Proceedings of IJCAI 2007*, 2007. (Cited on page 185.)

[BK56] 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. (Cited on pages 53, 63, 76, 77, 90, and 92.)

[BK98] F. Baader and R. Küsters. Computing the least common subsumer and the most specific concept in the presence of cyclic ALN-concept descriptions. In *Proc. of the 22th German Annual Conf. on Artificial Intelligence (KI'98), volume 1504 of Lecture Notes in Computer Science, pages 129-140. Springer-Verlag*, 1998. (Cited on page 46.)

[BK12a] L. Bednarik and L. Kovacs. Automated ea-type question generation from annotated texts. In *Applied Computational Intelligence and*

Informatics (SACI), 2012 7th IEEE International Symposium on, pages 191–195, 2012. (Cited on pages 94 and 222.)

[BK12b] L. Bednarik and L. Kovacs. Implementation and assessment of the automatic question generation module. In *Cognitive Infocommunications (CogInfoCom), 2012 IEEE 3rd International Conference on*, pages 687–690, 2012. (Cited on pages 94 and 222.)

[BKM99] F. Baader, R. Küsters, and R. Molitor. Computing least common subsumers in description logics with existential restrictions. In *Dean, T. (ed.) Proc. of IJCAI, pp. 96101. Morgan Kaufmann, San Francisco*, 1999. (Cited on pages 45 and 46.)

[BL86] D. A. Balota and R. F. Lorch. Depth of automatic spreading activation: Mediated priming effects in pronunciation but not in lexical decision. *Journal of Experimental Psychology: Learning, Memory, Cognition*, 12:336–345, 1986. (Cited on page 71.)

[BL04] R. Brachman and H. Levesque. *Knowledge Representation and Reasoning.* Morgan Kaufmann/Elsevier, 2004. (Cited on page 40.)

[Blo01] D. S. Blough. The perception of similarity. In R. G. Cook (Ed.), Avian visual cognition, http://www.pigeon.psy.tufts.edu/avc/dblough/ [Accessed: 19-04-2014], 2001. (Cited on pages 10, 104, 105, 106, and 108.)

[BLS06] F. Baader, C. Lutz, and B. Suntisrivaraporn. Eficient reasoning in EL+. In *Proceedings of the 2006 International Workshop on Description Logics (DL2006)*, 2006. (Cited on page 41.)

[BMB04] J.E. Beck, J. Mostow, and J. Bey. Can automated questions scaffold childrens reading comprehension? In *Lester, J.C., Vicari, R.M., Paraguau, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 478490. Springer, Heidelberg*, 2004. (Cited on pages 97 and 207.)

[Bor70] J. R. Bormuth. *On a theory of achievement test items.* Chicago. University of Chicago Press, 1970. (Cited on pages 84, 94, and 204.)

[Bor96] A. Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence*, 82(1-2):353–367, April 1996. (Cited on page 33.)

[BP10] Kristy Elizabeth Boyer and Paul Piwek, editors. *Proceedings of QG2010: The third workshop on Question Generation.* Pittsburgh: questiongeneration.org, 2010. (Cited on page 88.)

[BPS07] F. Baader, R. Peñaloza, and B. Suntisrivaraporn. Pinpointing in the description logic el+. In *Proceedings of the 30th Annual German Conference on Artificial Intelligence (KI-07)*, pages 52–67, 2007. (Cited on page 46.)

[Bra04] S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and what else? In *R. Lopez de Mantaras and L. Saitta, editors, Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-2004)*, pages 298–302. IOS Press, 2004. (Cited on page 41.)

[Bro94] P. Broadfoot. Editorial. assessment in education; 1: 3-10, 1994. (Cited on page 50.)

[BS01] F. Baader and U. Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69:5–40, 2001. (Cited on page 45.)

[BST04] F. Baader, R. Sertkaya, and A. Turhan. Computing least common subsumers w.r.t. a background terminology. In *Proceedings of the International Workshop on DLs*, 2004. (Cited on page 46.)

[BSTP$^+$10] S. Benabderrahmane, M. Smail-Tabbone, O. Poch, A. Napoli, and M. Devignes. IntelliGO: a new vector-based semantic similarity measure including annotation origin. *BMC Bioinformatics*, 2010. (Cited on pages 103 and 133.)

[BTK03] S. Brandt, A.-Y. Turhan, and R. Küsters. Extensions of non-standard inferences for description logics with transitive roles. In *M. Vardi and A. Voronkov, editors, Proceedings of the tenth International Conference on Logic for Programming and Automated Reasoning (LPAR'03), LNCS. Springer*, 2003. (Cited on page 45.)

[BVL03] S. Bechhofer, R. Volz, and P. Lord. Cooking the semantic web with the OWL API. In *ISWC-03*, 2003. (Cited on page 35.)

[CBEM12] R. Correia, J. Baptista, M. Eskenazi, and N. Mamede. Automatic generation of cloze question stems. In *Proceedings of PROPOR 2012 the 10th International Conference on Computational Processing of the Portuguese Language*, pages 168–178, 2012. (Cited on pages 94 and 219.)

[CCS$^+$97] J.R. Campbell, P. Carpenter, C. Sneiderman, S. Cohn, CG. Chute, and J. Warren. Phase ii evaluation of clinical coding schemes: completeness, taxonomy, mapping, definitions, and clarity. *Journal of the American Medical Informatics Association*, 4:238–251, 1997. (Cited on page 135.)

[CCSO14] V. Chaudhri, P. Clark, A. Spaulding, and A. Overholtzer. Question generation from a knowledge base. In *Proceedings of The 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2014)*, 2014. (Cited on pages 62, 88, 89, 90, 93, 94, 97, and 223.)

[CDM96] J. Carneson, G. Delpierre, and K. Masters. *Designing and Managing Multiple Choice Questions*. University of Cape Town, South Africa, 1996. (Cited on page 77.)

[CGHKS07] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Just the right amount: Extracting modules from ontologies. In *Proceedings of the 14th International World Wide Web Conference (WWW-05)*, 2007. (Cited on page 47.)

[CGHKS08] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *J. of Artificial Intelligence Research*, 31:273318, 2008. (Cited on pages 47, 49, 123, and 138.)

[CGPSK06] B. Cuenca Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularity and web ontologies. In *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning (KR-06)*, 2006. (Cited on page 47.)

[CGR82] M.T.H. Chi, R. Glaser, and E. Rees. Expertise in problem-solving. *R.J. Sternberg (Ed.), Advances in the psychology of human intelligence (Volume 1). Hillsdale, NJ: Erlbaum*, 1982. (Cited on pages 64 and 75.)

[CK83] M.T.H. Chi and R.D. Koeske. Network representation of a childs dinosaur knowledge. *Developmental Psychology*, 19:2939, 1983. (Cited on pages 64 and 75.)

[CL75] A. Collins and E. Loftus. A spreading-activation theory of semantic processing. *Psychological review*, 82(6):407–428, 1975. (Cited on page 70.)

[Cla83] W. J. Clancey. GUIDON. *Journal of Computer-Based Instruction*, 10:8–15, 1983. (Cited on pages 24, 88, and 205.)

[Cla87] W. Clancey. *Knowledge-Based Tutoring: The GUIDON Program.* MIT Press, 1987. (Cited on pages 24 and 205.)

[CLC06] C. Chen, H. Liou, and J. Chang. FAST: an automatic generation system for grammar tests. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, COLING-ACL '06, pages 1–4, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. (Cited on pages 89, 94, 95, 98, and 210.)

[CMC11] S. Curto, A. C. Mendes, and L. Coheur. Exploring linguistically-rich patterns for question generation. In *UCNLG+EVAL '11: Proceedings of the UCNLG+Eval: Language Generation and Evaluation Workshop*, 2011. (Cited on page 215.)

[CNB03] G. Chung, D. Niemi, and W. L. Bewley. Assessment applications of ontologies. In *Paper presented at the Annual Meeting of the American Educational Research Association*, 2003. (Cited on pages 21, 62, and 99.)

[Con97] D. Conaim. A preliminary inquiry into using corpus word frequency data in the automatic generation of English language cloze tests. In *Computer Assisted Language Instruction Consortium*, volume 16, page 1533, 1997. (Cited on pages 86, 94, 95, 97, and 205.)

[Con00] T. G. O. Consortium. Gene ontology: tool for the unification of biology. *Nat. Genet. May 2000*, 25(1):25–9, 2000. (Cited on page 103.)

[CQ69] A. Collins and M. Quillian. Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, 8:240–248, 1969. (Cited on pages 70 and 71.)

[Cra68] W. Crawford. Item difficulty as related to the complexity of intellectual processes. *Journal of Educational Measurement*, 5(2):103–107, 1968. (Cited on page 77.)

[Cro51] LJ. Cronbach. Coefficient alpha and the internal structure of tests. *Psychometrika*, 16(3):297–334, 1951. (Cited on page 56.)

[CS73] W. Chase and H. Simon. Perception in chess. *Cognitive Psychology*, 1973. (Cited on page 75.)

[CT09] M. Cubric and M. Tosic. SeMCQ - Protégé plugin for automatic ontology-driven multiple choice question tests generation. In *11th Intl. Protg Conference, Poster and Demo Session*, 2009. (Cited on pages 21, 62, 91, 92, and 213.)

[CT10] M. Cubric and M. Tosic. Towards automatic generation of e-assessment using semantic web technologies. In *Proceedings of the 2010 International Computer Assisted Assessment Conference*, University of Southampton, July 2010. (Cited on pages 21, 62, 91, 92, 93, 96, 101, and 213.)

[CW10] T. Cohen and D. Widdows. Empirical distributional semantics: Methods and biomedical applications. *Journal of Biomedical Informatics*, 42(2):390405, 2010. (Cited on page 103.)

[dA09] R.J. de Ayala. *The Theory and Practice of Item Response Theory.* New York, NY: The Guilford Press, 2009. (Cited on page 60.)

[DAB14] F. Distel, J. Atif, and I. Bloch. Concept dissimilarity based on tree edit distance and morphological dilations. In *European Conference on Artificial Intelligence (ECAI), Prag, Czech Republic*, page 249254, 2014. (Cited on page 107.)

[Dav01] B. G. Davis. *Tools for Teaching.* San Francisco, CA: Jossey-Bass, 2001. (Cited on pages 19 and 164.)

[dFE05] C. d'Amato, N. Fanizzi, and F. Esposito. A semantic similarity measure for expressive description logics. *A. Pettorossi, editor, Proceedings of Convegno Italiano di Logica Computazionale (CILC05), Rome, Italy.*, 2005. (Cited on pages 29 and 112.)

[dFE06] C. d'Amato, N. Fanizzi, and F. Esposito. A dissimilarity measure for ALC concept descriptions. *Proceedings of the 21st Annual ACM Symposium of Applied Computing, SAC2006, Dijon, France. ACM.*, 2:16951699, 2006. (Cited on pages 29 and 112.)

[DGL96] G. De Giacomo and M. Lenzerini. Tbox and abox reasoning in expressive description logics. In *Proceedings of the 5th International Conference on the Principles of Knowledge Representation and Reasoning (KR-96)*, page 316327, 1996. (Cited on page 45.)

[Dic45] L. Dice. Measures of the amount of ecologic association between species. *Ecology 26*, 3:297–302, 1945. (Cited on page 111.)

[DLWK14] Z. Dragisic, P. Lambrix, and F. Wei-Kleiner. Completing the is-a structure of biomedical ontologies. In *10th International Conference on Data Integration in the Life Sciences*, 2014. (Cited on page 185.)

[dM11] M. d'Aquin and M. Motta. Extracting relevant questions to an rdf dataset using formal concept analysis. In *Proceedings of the sixth international conference on Knowledge capture*, page 121128. ACM, 2011. (Cited on pages 90, 98, and 216.)

[DMMS13] G. Dror, Y. Maarek, A. Mejer, and I. Szpektor. From query to question in one click: suggesting synthetic questions to searchers. In *WWW '13: Proceedings of the 22nd international conference on World Wide Web*, 2013. (Cited on page 222.)

[DS03] P. Deane and K. Sheehan. Automatic item generation via frame semantics: Natural language generation of math word problems. *Education Testing Service*, 2003. (Cited on pages 89, 95, 96, and 206.)

[dSF08] C. d'Amato, S. Staab, and N. Fanizzi. On the Influence of Description Logics Ontologies on Conceptual Similarity. In *EKAW '08 Proceedings of the 16th international conference on Knowledge Engineering: Practice and Patterns*, 2008. (Cited on pages 29, 103, 110, 112, and 121.)

[DVGK$^+$11] C. Del Vescovo, D. Gessler, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and A. Winget. Decomposition and modular structure of BioPortal ontologies. In *The Semantic Web ISWC 2011, volume 7031 of Lecture Notes in Computer Science, pages 130-145. Springer. Berlin / Heidelberg*, 2011. (Cited on pages 10, 48, and 49.)

[DVKP$^+$12] C. Del Vescovo, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and D. Tsarkov. Syntactic vs. semantic locality: How good is a cheap approximation? In *WoMO 2012*, 2012. (Cited on page 135.)

[DVPS12] C. Del Vescovo, B. Parsia, and U. Sattler. Logical relevance in ontologies. In *Description Logics (DL) workshop*, 2012. (Cited on page 48.)

[DVPSS11] C. Del Vescovo, B. Parsia, U. Sattler, and T. Schneider. The modular structure of an ontology: Atomic decomposition. In *Proc. of IJCAI*, pages 2232–2237, 2011. (Cited on page 48.)

[Ear03] Lorna Earl. *Assessment as Learning: Using Classroom Assessment to Maximise Student Learning.* Thousand Oaks, CA, Corwin Press, second edition edition, 2003. (Cited on page 50.)

[Ebe54] R.L. Ebel. Procedures for the analysis of classroom tests. *Educational and Psychological Measurement*, 14:352–364, 1954. (Cited on page 163.)

[End01] H. B. Enderton. *A Mathematical Introduction to Logic.* San Diego, CA: Academic Press, 2nd edition, 2001. (Cited on page 33.)

[EPT15] A. Ecke, M. Pensel, and A. Turhan. Elastiq: Answering similarity-threshold instance queries in EL. In *Description Logics 2015*, 2015. (Cited on page 112.)

[ES91] K. A. Ericsson and J. Smith, editors. *Toward a General Theory of Expertise: Prospects and Limits*. Cambridge University Press, 1991. (Cited on page 75.)

[ES07] J. Euzenat and P. Shvaiko. Ontology matching. *Springer-Verlag*, 2007. (Cited on page 103.)

[Est55] W. K. Estes. Statistical theory of distributional phenomena in learning. *Psychological Review*, 62:369–377, 1955. (Cited on pages 104 and 105.)

[Fai99] C. Fairon. A web-based system for automatic language skill assessment: Evaling. In *Proceedings of Computer Mediated Language Assessment and Evaluation in Natural Language Processing Workshop*, 1999. (Cited on pages 21, 62, 86, 94, and 205.)

[Fd06] N. Fanizzi and C. d'Amato. A similarity measure for the ALN description logic. In *Convegno Italiano di Logica Computazionale (CILC 2006)*, 2006. (Cited on page 112.)

[FHH96] H. Fisher-Hoch and S. Hughes. What makes mathematics exam questions difficult. *British Educational Research Association, University of Lancaster, England*, 1996. (Cited on page 66.)

[FHHB94] H. Fisher-Hoch, S. Hughes, and T. Bramley. What makes GCSE examination questions difficult? outcomes of manipulating difficulty of GCSE questions. *British Educational Research Association, University of York, England*, 1994. (Cited on page 66.)

[Gar04] L. M. Garshol. Metadata? thesauri? taxonomies? topic maps! making sense of it all. *Journal of Information Science*, 30:378–391, 2004. (Cited on pages 32 and 33.)

[GBJR+13] R. Gonçalves, S. Bail, E. Jiménez-Ruiz, N. Matentzoglu, B. Parsia, B. Glimm, and Y. Kazakov. OWL reasoner evaluation (ORE) workshop 2013 results: Short report, 2013. (Cited on page 34.)

[GC86] C. Gobbo and M. T. H. Chi. How knowledge is structured and used by expert and novice children. *Cognitive Development*, 1:221–237, 1986. (Cited on page 75.)

[Gen83] D. Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7:155–170, 1983. (Cited on pages 10, 104, 106, and 107.)

[GFB05] 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. (Cited on page 93.)

[GH13] M. J. Gierl and T. M. Haladyna, editors. *Automatic Item Generation: Theory and Practice.* Taylor and Francis Group. New York and London, 2013. (Cited on pages 77 and 88.)

[GHM+08] B. C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322, 2008. (Cited on pages 33 and 34.)

[GK10] T. Goto and T. Kojiri. Automatic generation system of multiple choice cloze questions and its evaluation. *Knowledge Management and E Learning: An International Journal*, 2(3), 2010. (Cited on pages 86, 89, 94, and 214.)

[GKW13] W. Gatens, B. Konev, and F. Wolter. Module extraction for acyclic ontologies. In *Proceedings of WoMo 2013*, 2013. (Cited on page 47.)

[GLT12] M. Gierl, H. Lai, and S. Turner. Using automatic item generation to create multiple- choice test items. *Medical Education*, 46:757765, 2012. (Cited on page 220.)

[GP94] AC. Graesser and NK. Person. Question asking during tutoring. *American Educational Research Journal*, 31:104–137, 1994. (Cited on page 52.)

[GPFLC04] A. Gómez-Pérez, M. Fernández-López, and O. Corcho. *Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web.* Springer-Verlag London Limited, 2004. (Cited on page 40.)

[GRC08] A. Graesser, V. Rus, and Z. Cai. Question classification schemes. In *Proc. of the Workshop on Question Generation*, 2008. (Cited on pages 10, 17, 52, and 53.)

[GRE] GRESampleQuestions. Best sample questions. retrieved march 10, 2012, from http://www.bestsamplequestions.com/gre-questions/analogies/. (Cited on pages 8, 51, and 77.)

[Gro82] N. Gronlund. *Constructing Achievement Tests*. New York, NY: Prentice-Hall, 1982. (Cited on pages 18, 55, and 56.)

[Gru93] T. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993. (Cited on page 33.)

[GS04] R. L. Goldstone and Ji Yun Son. Similarity. *Psychological Review*, 100:254–278, 2004. (Cited on pages 104 and 106.)

[Gut53] L. Guttman. Image theory for the structure of quantitative variates. *Psychometrica*, 18(4):277–296, 1953. (Cited on page 77.)

[Haa12] V. et al. Haarslev. The RacerPro knowledge representation and reasoning system. In *Semantic Web 3.3*, pages 267–277, 2012. (Cited on page 35.)

[Hal94] T. M. Haladyna. Developing and validating multiple-choice test items. *Hillsdale: Lawrence Erlbaum*, 1994. (Cited on page 95.)

[HB09] 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. (Cited on pages 134 and 151.)

[HCR03] U Hahn, N Chater, and LB Richardson. Similarity as transformation. *COGNITION*, 87 (1):1 – 32, 2003. (Cited on pages 104, 107, and 108.)

[HD93] 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. (Cited on pages 19 and 159.)

[Hei11] M. Heilman. *Automatic Factual Question Generation from Text.* PhD thesis, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, 2011. (Cited on pages 21, 62, 86, 90, 98, and 213.)

[HGR$^+$06] N. Horridge, M. Drummond, J. Goodwin, A. Rector, R. Stevens, and H. Wang. The Manchester OWL syntax. In *Proceedings of the 2nd International Workshop on OWL: Experiences and Directions (OWLED-06)*, 2006. (Cited on page 34.)

[HKS06] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible SROIQ. In *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), AAAI Press*, pages 57–67, 2006. (Cited on pages 42 and 43.)

[HMMP05] E. Holohan, M. Melia, D. McMullen, and C. Pahl. Adaptive e-learning content generation based on semantic web technology. In *Proceedings of Workshop on Applications of Semantic Web Technologies for e-Learning*, pages 29–36, Amsterdam, The Netherlands, 2005. (Cited on pages 21, 62, 91, and 209.)

[HMMP06] E. Holohan, M. Melia, D. McMullen, and C. Pahl. The generation of e-learning exercise problems from subject ontologies. In *Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies*, pages 967–969, 2006. (Cited on pages 21, 62, 87, 88, 91, 99, and 209.)

[HN05a] A. Hoshino and H. Nakagawa. Real-time multiple choice question generation for language testing: a preliminary study. In *Proceedings of the Second Workshop on Building Educational Applications using Natural Language Processing*, pages 17–20, Ann Arbor, US, 2005. (Cited on pages 21, 62, 86, 88, 94, 95, 97, and 208.)

[HN05b] A. Hoshino and H. Nakagawa. Webexperimenter for multiple-choice question generation. In *HLT-Demo '05: Proceedings of HLT/EMNLP on Interactive Demonstrations*, 2005. (Cited on pages 94, 95, and 208.)

[HN10] A. Hoshino and H. Nakagawa. Predicting the difculty of multiple-choice cloze questions for computer-adaptive testing. In *Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics*, Romania, March 21-27 2010. (Cited on pages 20 and 21.)

[HNS90] B. Hollunder, W. Nutt, and M. Schmidt-Schauß. Subsumption algorithms for concept description languages. In *Proc. of ECAI-90, Pitman Publishing, London*, 1990. (Cited on page 45.)

[Hor97] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, Department of Computer Science, The University of Manchester, 1997. (Cited on page 45.)

[Hor10] M. Horridge. Owl syntaxes. http://ontogenesis. knowledge-blog.org/88 [accessed: 17-04-2014], 2010. (Cited on page 34.)

[Hor11a] M. Horridge. *Justification Based Explanation in Ontologies*. PhD thesis, The University of Manchester, 2011. (Cited on page 46.)

[Hor11b] 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. (Cited on pages 42 and 167.)

[HPS03] I. Horrocks and P.F. Patel-Schneider. Reducing owl entailment to description logic satisfiability. In *Proceedings ISWC-2003, in: LNCS, vol. 2870, Springer*, page 1729, 2003. (Cited on page 43.)

[HPS11] M. Horridge, B. Parsia, and U. Sattler. The state of biomedical ontologies. In *BioOntologies 2011 15th-16th July, Vienna Austria*, 2011. (Cited on page 25.)

[HPS12] M. Horridge, B. Parsia, and U. Sattler. Extracting justifications from BioPortal Ontologies. *International Semantic Web Conference*, 2:287–299, 2012. (Cited on page 135.)

[HPSMW07] I. Horrocks, P.F. Patel-Schneider, D. L. McGuinness, and C. A. Welty. *The Description Logic Handbook: Theory, Implementation, and Applications (2nd Edition), Franz Baader, Diego Calvanese,* *Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (ed.)*, chapter OWL: a Description Logic Based Ontology Language for the Semantic Web. Cambridge University Press, 2007. (Cited on page 38.)

[HR94] E. Heit and J. Rubinstein. Similarity and property effects in inductive reasoning. *Journal of Experimental Psychology*, 20:411–422, 1994. (Cited on page 70.)

[HS04] I. Horrocks and U. Sattler. Decidability of SHIQ with complex role inclusion axioms. *Artificial Intelligence*, 160:79104, 2004. (Cited on page 43.)

[HS09] M. Heilman and N. Smith. Ranking automatically generated questions as a shared task. In *Scotty D. Craig and Darina Dicheva, editors, The 2nd Workshop on Question Generation, volume 1 of AIED 2009 Workshops Proceedings, Brighton, UK. International Artificial Intelligence in Education Society*, page 3037, 2009. (Cited on pages 84, 86, 90, 91, 94, 97, 98, and 213.)

[HS10a] M. Heilman and N. Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*, page 609617, Los Angeles, California, June 2010. Association for Computational Linguistics. (Cited on pages 90 and 213.)

[HS10b] M. Heilman and N. Smith. Rating computer-generated questions with mechanical turk. In *CSLDAMT '10: Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics, 2010. (Cited on pages 90, 98, and 213.)

[HSO95] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet, edited by Christiane Fellbaum, Cambridge, MA: The MIT Press*, 1995. (Cited on pages 10, 125, and 126.)

[HST98] I. Horrocks, U. Sattler, and S. Tobies. A PSpace-algorithm for deciding $\mathcal{ALCNI}_{R^+}$-satisfiability. LTCS-Report 98-08, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1998. (Cited on page 43.)

[HST99] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Ganzinger, H.; McAllester, D.; and Voronkov, A., eds., Proc. of LPAR-6, volume 1705 of LNAI , Springer*, page 161180, 1999. (Cited on page 43.)

[HST00a] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for very expressive description logics. *Journal of the Interest Group in Pure and Applied Logics 8*, pages 1–26, 2000. (Cited on pages 41, 43, and 45.)

[HST00b] I. Horrocks, U. Sattler, and S. Tobies. Reasoning with individuals for the description logic SHIQ. In *MacAllester, D., ed., Proc. of CADE-17, volume 1831 of LNCS . Germany: Springer.*, 2000. (Cited on page 43.)

[HTSC14] Y. Huang, Y. Tseng, Y.S. Sun, and M. Chen. Tedquiz: Automatic quiz generation for ted talks video clips to assess listening comprehension. In *Advanced Learning Technologies (ICALT), 2014 IEEE 14th International Conference on*, pages 350–354, 2014. (Cited on pages 91, 94, 98, and 224.)

[IK02] S. Irvine and S. Kyllonen, editors. *Item Generation for Test Development.* Routledge, 2002. (Cited on page 88.)

[Jac01] P. Jaccard. Etude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Societe Vaudoise des Sciences Naturelles*, 37:547–579, 1901. (Cited on page 111.)

[Jam50] W. James. The principles of psychology. dover: New york. (original work published 1890), 1890/1950. (Cited on page 104.)

[Jan06] K. Janowicz. Sim-dl: Towards a semantic similarity measurement theory for the description logic ALCNR in geographic information retrieval. In *SeBGIS 2006, OTM Workshops 2006, pages 1681-1692*, 2006. (Cited on pages 29, 103, and 112.)

[JC97] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th International Conference on Research on Computational Linguistics, Taiwan*, 1997. (Cited on pages 29, 103, 112, 115, and 126.)

[JS14] C. Jouault and K. Seta. Content-dependent question generation for history learning in semantic open learning space. In *Proceedings of the 12th International Conference on Intelligent Tutoring Systems, ITS 2014*, 2014. (Cited on pages 90 and 225.)

[KBS$^+$12] P. Kuyten, T. Bickmore, S. Stoyanchev, P. Piwek, H. Prendinger, and M. Ishizuka. Fully automated generation of question-answer pairs for scripted virtual instruction. In *IVA'12: Proceedings of the 12th international conference on Intelligent Virtual Agents.* Springer-Verlag, 2012. (Cited on pages 98 and 218.)

[Keh95] J. Kehoe. Basic item analysis for multiple-choice tests. *Practical Assessment, Research & Evaluation*, 4(10), 1995. (Cited on page 96.)

[Kim08] H. Kim. Design of question answering system with automated question generation. In *Proceedings of NCM 08. Fourth International Conference on Networked Computing and Advanced Information Management*, 2008. (Cited on pages 96, 97, and 211.)

[KKS11] Y. Kazakov, M. Krötzsch, and F. Simancik. Unchain my EL reasoner. In *Proceedings of the 24th International Workshop on Description Logics (DL-11)*, 2011. (Cited on page 34.)

[Kle13] Stanley B. Klein. Episodic memory and autonoetic awareness. *Frontiers in Behavioral Neuroscience*, 7(3), 2013. (Cited on page 69.)

[KLK12] Y.-B. Kang, Y.-F. Li, and S. Krishnaswamy. Predicting reasoning performance using ontology metrics. In *ISWC 2012 Lecture Notes in Computer Science. Volume 7649*, 2012. (Cited on page 135.)

[KLM06] N. Karamanis, H. LA, and R. Mitkov. Generating multiple-choice test items from medical text: A pilot study. In *Proceedings of the 4th international natural language generation conference*, page 111113, 2006. (Cited on page 87.)

[KLWW13] B. Konev, C. Lutz, D. Walther, and F. Wolter. Model-theoretic inseparability and modularity of description logic ontologies. *AIJ*, 2013. (Cited on page 47.)

[Kon14] R. et al. Kontchakov. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In *The Semantic WebISWC 2014*, pages 552–567. Springer International Publishing, 2014. (Cited on page 35.)

[KPS$^+$05] A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca-Grau, and J. Hendler. SWOOP: a web ontology editing browser. *Journal of Web Semantics*, 4(2):144153, 2005. (Cited on page 35.)

[KR37] G. F. Kuder and M. W. Richardson. The theory of the estimation of test reliability. *Psychometrika*, 2(3):151–160, 1937. (Cited on page 56.)

[KS13] L. Kovacs and G. Szeman. Complexity-based generation of multi-choice tests in aqg systems. In *Cognitive Infocommunications (CogInfoCom), 2013 IEEE 4th International Conference on*, pages 399–402, 2013. (Cited on pages 99, 197, and 223.)

[KT09] C. Kemp and J. B. Tenenbaum. Structured statistical models of inductive reasoning. *Psychological Review*, 116(1):20–58, 2009. (Cited on page 70.)

[KWDH14] N. Khodeir, N. Wanas, N. Darwish, and N. Hegazy. Bayesian based adaptive question generation technique. *Journal of Electrical Systems and Information Technology*, 1:10–16, 2014. (Cited on pages 97 and 225.)

[LB07] H. Levesque and R. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3(1):78–93, 2007. (Cited on pages 39 and 40.)

[LC12] M. Liu and R. Calvo. Using information extraction to generate trigger questions for academic writing support. In *ITS'12: Proceedings of the 11th international conference on Intelligent Tutoring Systems*, 2012. (Cited on pages 90 and 221.)

[LCAP12] M. Liu, R. A. Calvo, A. Aditomo, and L. A. Pizzato. Using Wikipedia and conceptual graph structures to generate questions for academic writing support. *IEEE Transactions on Learning Technologies*, 5(3):251–263, 2012. (Cited on pages 90 and 222.)

[LCR12] M. Liu, R. Calvo, and V. Rus. G-Asks: An intelligent automatic question generation system for academic writing support. *Dialogue and Discourse*, 3(2):101124, 2012. (Cited on pages 90, 91, 98, and 221.)

[LCR14] M. Liu, R. Calvo, and V. Rus. Automatic generation and ranking of questions for critical review. *Educational Technology & Society*, 17(2):333–346, 2014. (Cited on pages 90 and 221.)

[Leh77] W. Lehnert. *The process of question answering.* Phd thesis, Yale University New Haven, CT, USA, 1977. (Cited on page 52.)

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710, 1966. (Cited on pages 104, 107, and 108.)

[Lin98] D. Lin. An information-theoretic definition of similarity. In *Proc. of the 15th International Conference on Machine Learning*, San Francisco, CA, 1998. Morgan Kaufmann. (Cited on pages 29, 95, 103, 112, 115, and 140.)

[Lin04] C. Lin. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization*, 2004. (Cited on page 91.)

[Lip99] A. H. Lipkus. A proof of the triangle inequality for the tanimoto distance. *Journal of Mathematical Chemistry*, 26(1):263–265, 1999. (Cited on page 119.)

[LL14] D. Liu and C. Lin. Sherlock: a semi-automatic quiz generation system using linked data. In *Proceedings of ISWC 2014 Posters & Demonstrations Track*, 2014. (Cited on pages 90 and 97.)

[LN68] F. M. Lord and M. R. Novick. *Statistical theories of mental test scores*. Reading MA: Addison-Welsley Publishing Company, 1968. (Cited on page 55.)

[Lof74] E. F. Loftus. Activation of semantic memory. *American Journal of Psychology*, 86:331–337, 1974. (Cited on page 71.)

[Lof75] E. Loftus. Spreading activation within semantic categories: Comments on Rosch's cognitive representation of semantic categories. *Journal of Experimental Psychology: General*, 104(3):234–240, 1975. (Cited on page 72.)

[Lor80] F.M. Lord. *Applications of item response theory to practical testing problems*. Mahwah, NJ: Erlbaum, 1980. (Cited on pages 55 and 58.)

[Low95] J. Lowman. *Mastering the Techniques of Teaching (2nd ed.)*. San Francisco: Jossey-Bass, 1995. (Cited on pages 19, 21, and 66.)

[LSC07] Y. C. Lin, L. C. Sung, and M. C. Chen. An automatic multiple choice question generation scheme for english adjective understanding. In *ICCE 2007 Workshop Proceedings of Modeling, Management and Generation of Problems/Questions in eLearning*, pages 137–142, 2007. (Cited on pages 89, 94, 95, 98, and 210.)

[LT12] K. Lehmann and A. Turhan. A framework for semantic-based similarity measures for ELH-concepts. *JELIA 2012*, pages 307–319, 2012. (Cited on pages 29, 103, 110, 112, and 121.)

[LWGH05] C. Liu, C. Wang, Z. Gao, and S. Huang. Applications of lexical information for algorithmically composing multiple-choice cloze items. In *Proceedings of the Second Workshop on Building Educational Applications using Natural Language Processing*, pages 1–8, Ann Arbor, US, 2005. (Cited on pages 21, 62, 89, 94, 98, and 208.)

[Mag09] C. Magno. Demonstrating the difference between classical test theory and item response theory using derived test data. *The International Journal of Educational and Psychological Assessment*, 1(1), 2009. (Cited on pages 56 and 58.)

[MAHK06] R. Mitkov, L. An Ha, and N. Karamani. A computer-aided environment for generating multiple-choice test items. *Natural Language Engineering. Cambridge University Press.*, 12(2):177–194, 2006. (Cited on pages 21, 58, 62, 63, 87, 89, 90, 91, 94, 95, 96, 98, 198, and 207.)

[MBB+04] J. Mostow, J. Beck, J. Bey, A. Cuneo, J. Sison, B. Tobin, and J. Valeri. Using automated questions to assess reading comprehension, vocabulary, and effects of tutorial interventions. *Technology, Instruction, Cognition and Learning*, pages 97–134, 2004. (Cited on pages 86, 94, 95, 97, 98, and 207.)

[MC09] J. Mostow and W. Chen. Generating instruction automatically for the reading strategy of self-questioning. In *Proceeding of the 2009 conference on Artificial Intelligence in Education*, pages 465–472, Amsterdam, The Netherlands, 2009. IOS Press. (Cited on pages 86, 93, 95, 97, 98, and 212.)

[MCG+05] A. Morgan, J. Cafeo, K. Godden, R. Lesperance, A. Simon, D. L. McGuinness, and J. L. Benedict. The general motors variation-reduction adviser. *AI Magazine*, 26(2), 2005. (Cited on page 35.)

[MCGH+12] B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz. OWL 2 profiles. `http://www.w3.org/TR/owl2-profiles/` [Accessed 10-03-2014], 2012. (Cited on page 34.)

[MEAF12] C.S. Montenegro, V.G. Engle, M.G.J. Acuba, and A.M.A. Ferrenal. Automated question generator for tagalog informational texts using case markers. In *TENCON 2012 - 2012 IEEE Region 10 Conference*, pages 1–5, 2012. (Cited on pages 98 and 220.)

[Men12] J. Mendez. jcel: A modular rule-based reasoner. In *ORE*, 2012. (Cited on page 34.)

[Mer94] M. D. Merrill. *Instructional Design Theory.* Englewood Cliffs, NJ: Educational Technology Publications, 1994. (Cited on page 63.)

[MGL12] J.R. Moser, C. Gutl, and W. Liu. Refined distractor generation with lsa and stylometry for automated multiple choice question generation. In *Proceedings of the 25th Australasian Conference AI 2012:*

Advances in Artificial Intelligence, pages 95–106, 2012. (Cited on pages 94, 96, and 219.)

[MH03] R. Mitkov and L.A. Ha. Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2*, pages 17–22, Edmonton, Canada, 2003. (Cited on pages 21, 62, 63, 87, 88, 90, 95, and 207.)

[MHVR09] R. Mitkov, L.A. Ha, A. Varga, and L. Rello. Semantic similarity of distractors in multiple-choice tests: Extrinsic evaluation. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 49–56, Athens, Greece, March 2009. Association for Computational Linguistics. (Cited on page 95.)

[MIL95] G. MILLER. WordNet: A lexical database for English. *COMMUNICATIONS OF THE ACM*, 38(11), November 1995. (Cited on page 125.)

[MISR11] E. Mikroyannidi, L. Iannone, R. Stevens, and A. Rector. Inspecting regularities in ontology design using clustering. In *The Semantic Web, ISWC 2011, pages 438-453. Springer*, 2011. (Cited on page 103.)

[MN14] K. Mazidi and R. Nielsen. Linguistic considerations in automatic question generation. In *Proceedings of 52nd ACL*, 2014. (Cited on pages 86, 91, 94, and 223.)

[MP15] N. Matentzoglu and B. Parsia. Bioportal snapshot 27.01.2015. http://dx.doi.org/10.5281/zenodo.15667, Feb 2015. (Cited on page 116.)

[MRRC81] K. B. McKeithen, J. S. Reitman, H. H. Rueter, and Hirtle. S. C. Knowledge organization and skill differences in computer programmers. *Cognitive psychology*, 13:307–325, 1981. (Cited on page 75.)

[MS08] L. Mazuel and N. Sabouret. Semantic relatedness measure using object properties in an ontology. In *ISWC '08 Proceedings of the 7th International Conference on The Semantic Web*, pages 681–694, 2008. (Cited on pages 125, 126, and 127.)

[MSK09] B. Magnini, M. Speranza, and V. Kumar. Towards interactive question answering: An ontology-based approach. In *Semantic Computing, 2009. ICSC '09. IEEE International Conference on*, pages 612–617, 2009. (Cited on pages 97 and 212.)

[NB11] K. Nandhini and S. R. Balasundaram. Math word question generation for training the students with learning difficulties. In *ICWET '11: Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, 2011. (Cited on pages 89 and 215.)

[NBH$^+$02] S. Newstead, P. Bradon, S. Handley, J. Evans, and I. Dennis. *S. Irvine & P. Kyllonen (Eds.), Item generation for test development.*, chapter Using the psychology of reasoning to predict the difficulty of analytical reasoning items. Mahwah, NJ: Lawrence Earlbaum Associates, 2002. (Cited on pages 100 and 206.)

[NBH$^+$06] S. Newstead, P. Bradon, S. Handley, I. Dennis, and J. Evans. Predicting the difficulty of complex logical reasoning problems. *Thinking & Reasoning*, Volume 12(1):62–90, 2006. (Cited on pages 100 and 206.)

[Neb90] B. Nebel. Reasoning and revision in hybrid representation systems. *Lecture Notes In Artificial Intelligence. Springer-Verlag*, 422, 1990. (Cited on page 45.)

[Nos92] R. M. Nosofsky. Similarity scaling and cognitive process models. *Annual Review of Psychology*, 43:25–53, 1992. (Cited on pages 104 and 105.)

[NSW$^+$09] N. Noy, N. Shah, P. Whetzel, B. Dai, M. Dorf, C. Jonquet, D. Rubin, M. Storey, C. Chute, and M. Musen. BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Research*, 37(W170-W173), 2009. (Cited on page 35.)

[ODI07] R. Othman, S. Deris, and R. Illias. A genetic similarity algorithm for searching the gene ontology terms and annotating anonymous protein sequences. *Journal of BiomedInform*, 23, 2007. (Cited on pages 29 and 103.)

[OGP12] A. Olney, A. Graesser, and N. Person. Question generation from concept maps. *Dialogue & Discourse*, 3(2):75–99, 2012. (Cited on pages 91, 94, and 220.)

[oSfEE03] Joint Committee on Standards for Educational Evaluation. The student evaluation standards: How to improve evaluations of students. newbury park, ca: Corwin press, 2003. (Cited on page 54.)

[Pax00] M. Paxton. A linguistic perspective on multiple choice questioning. *Assessment & Evaluation in Higher Education*, 25(2):109–119, 2000. (Cited on page 19.)

[Pid14] W. Pidcock. What are the differences between a vocabulary, a taxonomy, a thesaurus, an ontology, and a meta-model? `http://www.metamodel.com/article.php?story=20030115211223271` [accessed: 19-04-2014], 2014. (Cited on pages 32 and 33.)

[PKK08] A. Papasalouros, K. Kotis, and K. Kanaris. Automatic generation of multiple-choice questions from domain ontologies. In *IADIS e-Learning 2008 conference*, Amsterdam, 2008. (Cited on pages 21, 26, 62, 87, 88, 89, 91, 92, 94, 97, 98, 99, 212, and 213.)

[PKK11] A. Papasalouros, K. Kotis, and K. Kanaris. Automatic generation of tests from domain and multimedia ontologies. *Interactive Learning Environments*, 19(1):523, 2011. (Cited on pages 94, 98, and 215.)

[PMSS14] P. Pabitha, M. Mohana, S. Suganthi, and B. Sivanandhini. Automatic question generation system. In *Proceedings of the International Conference on Recent Trends in Information Technology (ICRTIT)*, 2014. (Cited on page 224.)

[PPPC07] T. Pedersen, S. Pakhomov, S. Patwardhan, and C. Chute. Measures of semantic similarity and relatedness in the biomedical domain. *Journal of Biomedical Informatics*, 30(3):288–299, 2007. (Cited on pages 133, 135, 137, 140, and 141.)

[Res95] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *In Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI95)*, volume 1, pages 448–453, 1995. (Cited on pages 29, 103, 112, and 115.)

[RG09] V. Rus and A. Graesser. Workshop report: The question generation task and evaluation challenge. Institute for Intelligent Systems, Memphis, TN, ISBN: 978-0-615-27428-7, 2009. (Cited on pages 21, 88, and 200.)

[RGH12] A. A. Romero, B. C. Grau, and I. Horrocks. MORe: Modular combination of OWL reasoners for ontology classification. In *Proceedings of the 11th International Semantic Web Conference (ISWC 2012)*, 2012. (Cited on page 34.)

[RH76] G. H. Roid and T. M. Haladyna. A comparison of objective-based and modified-Bormuth item writing techniques. *Paper presented at the annual meetings of the American Educational research association (60th, San Francisco, California, April 19-23)*, 1976. (Cited on page 85.)

[RM04] T. T. Rogers and J. L. McClelland. *Semantic Cognition: A Parallel Distributed Processing Approach.* Cambridge, MA: MIT Press, 2004. (Cited on page 70.)

[RMBB89] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. In *IEEE Transaction on Systems, Man, and Cybernetics*, volume 19, page 1730, 1989. (Cited on pages 29, 103, 112, 113, 126, 133, and 137.)

[RNG93] A. Rector, A. Nowlan, and A. Glowinski. Goals for concept representation in the GALEN project. In *Proceedings of the 17th Annual Symposium on Computer Applications in Medical Care, October 30 November 3, 1993, Washington DC, American Medical Informatics Association*, pages 414–418, 1993. (Cited on page 40.)

[Ros73] E. Rosch. On the internal structure of perceptual and semantic categories. *T. Moore(Ed.), Cognitive development and the acquisition of languagee. New York: Academic Press*, 1973. (Cited on page 72.)

[Ros75] E. Rosch. Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, 1975. (Cited on page 72.)

[RR93] K. Hambleton Ronald and W. Jones Russell. An ncme instructional module on comparison of classical test theory and item response theory and their applications to test development. *Instructional Topics in Educational Measurement*, 1993. (Cited on pages 55, 58, 59, 60, and 61.)

[Rud10] L. Rudner. *Elements of adaptive testing*, chapter Implementing the Graduate Management Admission Test computerized adaptive test, pages 151–165. New York, NY: Springer, 2010. (Cited on page 17.)

[SA12] D. W. Schneider and J. R. Anderson. Modeling fan effects on the time course of associative recognition. *Cognitive Psychology*, 64:127–160, 2012. (Cited on page 74.)

[Sat96] U. Sattler. A concept language extended with different kinds of transitive roles. In *KI-96: Advances in Artificial Intelligence, 20th Annual German Conference on Artificial Intelligence, Dresden, Germany, volume 1137 of Lecture Notes in Computer Science, pages 333-345. Springer*, 1996. (Cited on page 41.)

[SB02] M. K. Singley and R. E. Bennett. *S. Irvine & P. Kyllonen (Eds.), Item generation for test development*, chapter Item generation and beyond: Applications of schema theory to mathematics assessment. Mahwah, NJ: Lawrence Earlbaum Associates, 2002. (Cited on pages 89, 95, and 206.)

[SBD94] J. T. Sidick, G. V. Barrett, and D. Doverspike. Three-alternative multiple-choice tests: An attractive option. *Personnel Psychology*, 47:829–835, 1994. (Cited on pages 19 and 20.)

[SC97] K. Spackman and K. Campbell. Snomed rt: A reference terminology for health care. In *Daniel R. Masys, editor, Proceedings of AMIA Annual Fall Symposium, Bethesda, Maryland, USA*, pages 640–644. Hanley and Belfus Inc., 1997. (Cited on page 40.)

[SC02] D. Saumier and H. Chertkow. Semantic memory. *Current Neurology and Neuroscience Reports*, 2:516–522, 2002. (Cited on page 69.)

[SC03] S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03), pages 355362*, 2003. (Cited on pages 45 and 46.)

[Scr91] M. Scriven. *Evaluation thesaurus*. Newbury Park, CA: Sage Publications, 4th ed edition, 1991. (Cited on page 50.)

[SDMW02] K. Spackman, R. Dionne, E. Mays, and J. Weis. Role grouping as an extension to the description logic of ontylog, motivated by concept modeling in snomed. In *Proceedings of the AMIA Symposium: American Medical Informatics Association*, pages 712–712, 2002. (Cited on page 190.)

[SDRL06] A. Schlicker, FS. Domingues, J. Rahnenführer, and T. Lengauer. A new measure for functional similarity of gene products based on gene ontology. *BMC Bioinformatics*, 7, 2006. (Cited on pages 103 and 133.)

[Sed78] G. M. Seddon. The properties of Bloom's taxonomy of educational objectives for the cognitive domain. *Review of Educational Research*, 48(2):pp. 303–323, 1978. (Cited on pages 63 and 76.)

[Ser13a] B. Sertkaya. The ELepHant reasoner system description. In *2nd OWL Reasoner Evaluation Workshop (ORE)*, 2013. (Cited on page 35.)

[SER13b] M. Simon, K. Ercikan, and M. Rousseau, editors. *Improving Large Scale Education Assessment: Theory, Issues, and Practice*. Routledge, New York, 2013. (Cited on page 20.)

[SF08] V. M. Sloutsky and A. V. Fisher. Attentional learning and flexible induction: How mundane mechanisms give rise to smart behaviors. *Child Development*, 79:639–651, 2008. (Cited on page 70.)

[SG06] 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. (Cited on page 93.)

[SH14] R. Singhal and M. Henz. Automated generation of region based geometric questions. In *Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on*, pages 838–845, 2014. (Cited on pages 90, 95, and 225.)

[She87] R.N. Shepard. Toward a universal law of generalization for psychological science. *Science*, 237:1317–1323, 1987. (Cited on pages 104 and 105.)

[Shu86] L. S. Shulman. Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15(2):4–31, 1986. (Cited on page 198.)

[SLL$^+$10] D. F. Savo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodrguez-Muro, V. Romagnoli, and Stella. Mastro at work: Experiences on ontology-based data access. In *Prooceedings of DL*, pages 20–31, 2010. (Cited on page 35.)

[Slo76] J. A. Sloboda. Visual perception of musical notation: Registering pitch symbols in memory. *Quarterly Journal of Experimental Psychology*, 28(1), 1976. (Cited on page 75.)

[SMH08] R. Shearer, B. Motik, and I. Horrocks. HermiT: A highly-efficient OWL reasoner. In *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED-08EU)*, 2008. (Cited on pages 34 and 134.)

[Smi70] R. Smith. An empirical investigation of complexity and process in multiple-choice items. *Journal of Educational Measurement*, 7(1):33–41, 1970. (Cited on page 76.)

[SPCG$^+$07] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2), 2007. (Cited on pages 34 and 134.)

[SPS09] H. Stuckenschmidt, C. Parent, and S. (Eds.) Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization, Series: Lecture Notes in Computer Science, Vol. 5445*. Springer, 2009. (Cited on page 93.)

[SS91] M. Schmidt-Schauß and G. Smolka. Attributive concept descreiptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991. (Cited on pages 39 and 45.)

[SS97] F. Sharifian and R. Samani. Hierarchical spreading of activation. In *In F. Sharifian (Ed.) Proc. of the Conference on Language, Cognition, and Interpretation*, pages 1–10, 1997. (Cited on page 71.)

[SSIS08] L. Stanescu, C.S. Spahiu, A. Ion, and A. Spahiu. Question generation for learning evaluation. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 509–513, 2008. (Cited on pages 96, 97, and 211.)

[SSR74] E. E. Smith, E. J. Shoben, and L. J. Rips. Structure and process in semantic memory: A featural model for semantic decisions. *Psychological Review*, 81:214–241, 1974. (Cited on page 73.)

[SSY05] E. Sumita, F. Sugaya, and S. Yamamoto. Measuring non-native speakers prociency of english using a test with automatically-generated fill-in-the-blank questions. In *Proceedings of the Second Workshop on Building Educational Applications using Natural Language Processing*, pages 61–68, Ann Arbor, US, 2005. (Cited on pages 21, 62, 86, 94, 98, and 209.)

[SSZ09] U. Sattler, T. Schneider, and M. Zakharyaschev. Which kind of module should I extract? In *Proceedings of the 22nd International Workshop on Description Logics (DL-09)*, 2009. (Cited on pages 47 and 93.)

[Ste91] V. Stevens. *In: Johns, Tim/King, Philip (ed.). Classroom Concordancing*, chapter Concordance-based Vocabulary Exercises: A Viable Alternative to Gap-fillers, pages 47–63. (ELR Journal 4), 1991. (Cited on pages 21, 62, 85, 86, and 205.)

[Sun11] B. Suntisrivaraporn. *Multi-disciplinary Trends in Artificial Intelligence, Lecture Notes in Computer Science*, volume 7080, chapter Structural Distance between EL+ Concepts, pages 100–111. Springer, 2011. (Cited on page 45.)

[Tar98] V. C. Tartter. *Language and its normal processing.* SAGE Publications, Inc, 1998. (Cited on pages 69, 70, and 74.)

[TG05] D. Traynor and J.P. Gibson. Synthesis and analysis of automatic assessment methods in cs1. In *The 36th SIGCSE Technical Symposium on Computer Science Education 2005, SIGCSE05*, page 495499, St. Louis Missouri, USA, February 2005. ACM Press. (Cited on pages 95 and 209.)

[TGM98] J. Turnbull, J. Gray, and J. MacFadyen. Improving in-training evaluation programs. *Journal General Internal Medcine*, 13(5):317–23, 1998. (Cited on page 54.)

[TH06] D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR)*, 2006. (Cited on pages 34, 134, and 151.)

[TL05] P. Turney and M. Littman. Corpus-based learning of analogies and semantic relations. *Machine Learning*, 60(1-3):251–278, 2005. (Cited on pages 127, 175, and 180.)

[Tob01] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation.* PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 2001. (Cited on page 43.)

[TP12] D. Tsarkov and I. Palmisano. Chainsaw: a metareasoner for large ontologies. In 1st OWL Reasoner Evaluation Workshop (ORE), 2012. (Cited on page 35.)

[TS14] S. Tongphu and B. Suntisrivaraporn. On desirable properties of the structural subsumption-based similarity measure. In *Proceedings of the 4th Joint International Semantic Technology Conference (JIST2014)*, Thailand, 2014. (Cited on page 121.)

[Tul02] E. Tulving. Episodic memory: From mind to brain. *Annual Review of Psychology*, 53:1–25, 2002. (Cited on page 69.)

[Tur05] P. Turney. Measuring semantic similarity by latent relational analysis. In *IJCAI is the International Joint Conference on Artificial Intelligence*, 2005. (Cited on page 127.)

[Tve77] A. Tversky. Features of similarity. *Psycological Review by the American Psycological Association, Inc.*, 84(4), July 1977. (Cited on pages 104, 106, 107, 108, 109, and 111.)

[TZ13] A. Turhan and B. Zarrieß. Computing the lcs w.r.t. general el+-tboxes. In *Description Logics*, pages 477–488, 2013. (Cited on page 46.)

[Urr77] V. Urry. Tailored testing: A successful application of latent trait theory. *Journal of Educational Measurement*, 14(2):181–196, 1977. (Cited on page 20.)

[Van08] L. Vanderwende. The importance of being important: Question generation. In *Vasile Rus and Art Graesser, editors, Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge, September 25-26*, 2008. (Cited on page 91.)

[VH10] A. Vergis and K. Hardy. Principles of assessment: A primer for medical educators in the clinical years. *The Internet Journal of Medical Education*, 1(1), 2010. (Cited on pages 50, 54, and 55.)

[Wag08] A.R. Wagner. Evolution of an elemental theory of pavlovian conditioning. *Learning and Behavior*, 36:253–265, 2008. (Cited on pages 104 and 105.)

[WCG01] J. W.Pellegrino, N. Chudowsky, and R. Glaser, editors. *Knowing what Students Know: The Science and Design of Educational Assessment.* NATIONAL ACADEMY PRESS, Washington, DC, 2001. (Cited on pages 50, 64, 65, 69, and 80.)

[WDP$^+$07] JZZ. Wang, Z. Du, R. Payattakool, PSS. Yu, and CFF. Chen. A new method to measure the semantic similarity of GO terms. *Bioinformatics*, 2007. (Cited on pages 103 and 133.)

[Wei13] R. Weingarten. Set the record straight on standardized tests, 2013. (Cited on page 18.)

[WHL08] W. Wang, T. Hao, and W. Liu. Automatic question generation for learning evaluation in medicine. In *Advances in Web Based Learning - ICWL 2007. 6th International Conference*, 2008. (Cited on pages 96 and 211.)

[Wil11] S. Williams. Generating mathematical word problems. In *2011 AAAI Fall Symposium Series*, 2011. (Cited on pages 87, 89, 92, 93, 99, 197, and 215.)

[WLHL12] K. Wang, T. Li, J. Han, and Y. Lei. Algorithms for automatic generation of logical questions on mobile devices. *IERI Procedia*, pages 258–263, 2012. (Cited on page 220.)

[Wol75] J. Wolfe. An aid to independent study through automatic question generation (autoquest). Technical report, NAVY PERSONNEL RESEARCH AND DEVELOPMENT CENTER SAN DIEGO CA, 1975. (Cited on pages 85 and 204.)

[Wol76] J. Wolfe. Automatic question generation from text-an aid to independent study. In *Proceedings of ACM SIGCSE-SIGCUE*, 1976. (Cited on pages 85, 94, and 204.)

[WP94] Z. Wu and M. Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd. Annual Meeting of the Association for Computational Linguistics (ACL 1994)*, page 133138, 1994. (Cited on pages 29, 103, 112, 114, 134, and 137.)

[WT08] D. Wiliam and M. Thompson. Integrating assessment with learning: What will it take to make it work? In C. A. Dwyer, editor, *The Future of Assessment: Shaping Teaching and Learning*, pages 53–82. New York: Lawrence Erlbaum Associates, 2008. (Cited on page 18.)

[YBZ12] X. Yao, G. Bouma, and Y. Zhang. Semantics-based question generation and implementation. *Dialogue and Discourse*, 3(2):1142, 2012. (Cited on pages 83, 94, and 214.)

[YZ10] X. Yao and Y. Zhang. Question generation with minimal recursion semantics. In *Proceedings of QG2010: The Third Workshop on Question Generation*, 2010. (Cited on pages 91, 94, 98, and 214.)

[ZN08] A. Zouaq and R. Nkambou. Building domain ontologies from text for educational purposes. *IEEE Transactions on Learning Technologies*, 1(1):49–62, January-March 2008. (Cited on page 93.)

[ZPK11] K. Zoumpatianos, A. Papasalouros, and K. Kotis. Automated transformation of SWRL rules into multiple-choice questions. In *FLAIRS Conference'11*, 2011. (Cited on pages 21, 62, and 91.)

[ZSCZ11] Z. Zheng, X. Si, E. Chang, and X. Zhu. K2q: Generating natural language questions from keywords with user refinements. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, page 947955, 2011. (Cited on pages 94 and 216.)

[ZSRG08] 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. (Cited on pages 21, 26, 62, 87, 89, 92, 93, 94, 95, 97, 99, 101, and 212.)

[ZT13] B. Zarrieß and A. Turhan. Most specific generalizations w.r.t. general EL-TBoxes. In *IJCAI*, 2013. (Cited on page 46.)