Short Paper: Assessing Procedural Knowledge in Open-ended Questions through Semantic Web Ontologies

Eric Snow, Chadia Moghrabi and Philippe Fournier-Viger

Département d'informatique, Université de Moncton, Moncton, Canada
{eric.snow,chadia.moghrabi,philippe.fournier-viger}@umoncton.ca

Abstract. This paper presents a novel approach for automatically grading students' answers to open-ended questions. It is inspired by the OeLE method, which uses ontologies and Semantic Web technologies to represent course material. The main difference in our approach is that we add a new category of concepts, named *functional concepts*, which allow specifying an ordering relation between concepts. This modification allows assessing procedural knowledge in students' answers by grading the ordering of these concepts. We present an example for grading answers in a course about computer algorithms, and report the corresponding results.

Keywords: E-Learning, Computer-Assisted Assessment (CAA), Ontology, Semantic Web, Procedural Knowledge.

1 Introduction

Assessing the students' learning in an e-learning environment often relies on multiple choice or fill-in-the-blank questions, which only trigger the lowest level (*Knowledge*) of Bloom's taxonomy [1] of knowledge acquisition. As we shall see in Section 2, several attempts have been made to incorporate open-ended questions in online assessment, which would possibly trigger the higher levels of Bloom's taxonomy (*Synthesis* and *Evaluation*) in the students' learning.

However, grading open-ended questions by hand can be time-consuming. To build an e-learning environment that can automatically grade free-text answers, a variety of techniques have been used, such as Information Extraction (IE) [4-5], Natural Language Processing (NLP) [6-11], or statistical techniques [13-15].

Our approach resembles that of the OeLE system [2]. This system also uses NLP to assess the level of understanding of the students. Course material is represented in an ontology and encoded in the Web Ontology Language (OWL). The use of Semantic Web technologies allows the sharing and reusing of course ontologies, thus potentially reducing the time spent designing the ontologies. This allows for a deeper understanding of the text than more superficial statistical techniques. Automatic assessment is much faster, and hopefully done more objectively, than manual scoring. The OeLE system has been used in two online courses, *Design and Evaluation of Didactic Media, and Multimedia Systems and Graphical Interaction.*

OeLE successfully assesses the semantic content of the students' answers if the answers contain static expressions of facts about didactic media or multimedia systems. However, when applying it to the assessment of a computer algorithms course, we observed that the ordering of the elements in students' answers is not taken into account. It is crucial that this ordering be considered because to describe how an algorithm works, certain concepts should be stated in a specific order. In this paper, we address this challenge by proposing a new approach in which we introduce the idea of *functional concepts*. The course ontology then incorporates ordering information about a subset of these functional concepts. The assessment process is modified to take into account the ordering of these concepts in the students' answers and adjust their grade accordingly. The novelty of our work is in applying a hybrid approach combining the OeLE system with functional concepts to assess students' answers in domains using highly procedural knowledge.

Section 2 of this paper is a review of other automatic free-text assessment systems. We only focus here on short-answer assessment systems where reference texts are tailored to the course material, although some other systems have also been developed for essay scoring, where more general texts about a topic are used for training. Section 3 presents the general methodology, followed by our preliminary results in Section 4. We conclude the paper in Section 5 with some future work which we are investigating.

2 Related Work

This section presents previous and ongoing research in automatic short-answer assessment. A good review of many of these systems can be found in [3]. Although these systems do not take advantage of Semantic Web ontologies, they contain nonetheless functionalities and techniques useful to our system.

Some systems compare students' answers to the ideal answer supplied by the teacher. For instance, Automated Text Marker [4] uses a pattern-matching technique. It has been tested in courses on Prolog programming, psychology and biology. Automark [5] uses IE techniques to grade students' answers by comparing them to a mark scheme template provided by the teacher. It achieved 94.7% agreement with human grading for six of the seven science-related questions asked on a test exam.

Some systems require teachers to provide training sets of marked student answers. For example, Auto-marking [6] uses NLP and pattern-matching techniques to compare students' answers to a training set of marked student answers. This system obtained 88% of exact agreement with human grading in a biology course. Bayesian Essay Test Scoring System (BETSY) [7] uses naive Bayes classifiers to search for specific features in students' answers. In a biology course, it achieved up to 80% accuracy. CarmelTC [8] uses syntactic analysis and naive Bayes classifiers to analyze essay answers. On an experiment with 126 physics essays, it obtained 90% precision and 80% recall. The Paperless-School Marking Engine (PS-ME) [9] is commercially available and requires a training set of marked answers. The system uses NLP to grade the students' answers in addition to implementing Bloom's taxonomy heuristics. However, the exact implementation is not disclosed. C-rater [10] uses a set of marked training essays to determine the students' answers grade using NLP. In a large-scale experiment of 170,000 answers to reading comprehension and algebra questions, it achieved 85% accuracy. In [11], a combination of NLP and Support Vector Machines is used to classify answers into two classes (above/below 6 points out of 10). It obtains an average of 65% precision rate (the only reported metric).

The MultiNet Working Bench system [12] uses a graphical tool to represent the students' knowledge visually. It compares the semantic network extracted from the student answer to that submitted by the teacher. Verified parts of the network are displayed in green, while wrong or unverified parts (not supported by logic inference) are displayed in red.

Other systems rely on Latent Semantic Analysis (LSA). For example, Research Methods Tutor [13] uses LSA to compare the students' answers to a set of expected answers. If the student answers incorrectly, the system guides the student into obtaining the right answer. The Willow system [14] requires several unmarked reference answers for each question. It also uses LSA to evaluate students' answers written in English or Spanish. In a computer science course, it achieved on average 0.54 correlation with the teacher's grading. A system currently in use at the University of Witwatersrand [15] uses LSA and clustering techniques. It achieves between 0.80 and 0.95 correlation with the teacher's grading.

3 Methodology

In this section, we briefly present the work on OeLE [2] and how we have adapted it and expanded on it in our system. Our focus has been on grading students' answers to questions in a computer algorithms course taught in French.

3.1 Natural Language Processing

For each of the online exam's questions in OeLE [2], the ideal answer provided by the teacher and the students' answers are processed similarly. The GATE software performs most of the NLP tasks, and the Protégé software is used to build the course ontology and encode it in OWL. While OeLE is written in Java and uses the Jena framework to process the encoded ontology, our system is done in PHP and we developed our own ontology-processing code. It is important to note that OeLE and our system use OWL for knowledge representation, but do not utilize its inference services. In this paper, we use the same terminology as [2]. We refer to OWL classes as *concepts*, to object properties as *relations*, and to data properties as *attributes*. Also, *entity* is used as a generic term for concept, relation, or attribute, while *property* is used for relation or attribute.

The NLP consists of three phases: *Preparation*, *Search*, and *Set in a context*. The *Preparation* phase consists of spell-checking, sentence detection, tokenization and POS tagging. In the *Search* phase, the linguistic expressions are detected and matched against the course ontology. Finally, the *Set in a context* phase associates the attributes and values to their respective concept, and also identifies which concepts participate in a relation.

In OeLE, the texts are annotated *semiautomatically*, meaning that the teacher only needs to manually annotate the fragments unknown to the system or incorrectly tagged. In our system, the natural language processing is done manually for the moment, as GATE does not sufficiently support French (out-of-the-box) for our purposes. Performing automatic French annotation is planned as a future work.

As an example, we use an actual question from a computer algorithms course given at our university: "Describe Depth-First Search (DFS)". Table 1 shows the annotation set (at the end of the NLP phase) of the partial student's answer: "Depth-First Search (DFS) is an exhaustive algorithm that explores a graph..." The ideal answer supplied by the teacher is similarly annotated; however, for every annotated entity, a numerical value ought to be supplied specifying the relative importance of that entity within the question.

Table 1. Example annotated answer of a student to describe DFS.

Category	Description
Concept	DepthFirstSearch
Concept	Algorithm
Concept	Graph
Attribute	Exhaustive
Relation	IsA
Relation	Explores

3.2 Conceptual Grading

The grading stage consists of calculating the semantic distance between the annotation sets (obtained in Section 3.1) of each student's answer and that of the teacher's ideal answer, with respect to the course ontology. Because of space limitations, we cannot give detailed calculations for the example. The reader is advised to see the full explanation in the original publication [2], or an easy-to-follow example in [16].

The formulas used in [2] for calculating the semantic distances are given below. In every function, teacher-provided constants allow for certain elements to be weighted more or less heavily according to their importance. The best combination of these constants is problem-dependent and should be discovered empirically. The "linguistic distance" between the textual representation of the entities in the student and teacher's answer is also taken into account. All functions return values in the [0,1] interval.

Concept similarity. To calculate the concept similarity (CS) between concepts c_i and c_j, the following function is used:

$$CS(c_i,\, c_j) = cp_1 \times CP(c_i,\, c_j) + cp_2 \times PS(c_i,\, c_j) + cp_3 \times EQ(c_i,\, c_j) \tag{1}$$

The constants cp_1, cp_2, cp_3 indicate the relative importance of the corresponding elements. Also, $cp_1 + cp_2 + cp_3 = 1$ and $0 \leq cp_k \leq 1$.

The concept proximity (*CP*) is calculated using the taxonomy formed in the ontology by the class hierarchy defined in OWL. Note that the <*is-a*> relation is explicitly added to the course ontology (with the class as domain and the subclass as image) where `rdfs:subClassOf` is used:

```
<owl:Class rdf:about="DepthFirstSearch">
  <rdfs:subClassOf rdf:resource="Algorithm"/>
</owl:Class>
```

If the concepts c_i and c_j have no taxonomic parent (other than the root), this value is zero, otherwise it is defined as such:

$$CP\left(c_i,\ c_j\right) = 1 - \frac{\left|nodes\left(c_i, c_j\right)\right|}{|concepts|} \tag{2}$$

where $\left|nodes\left(c_i,\ c_j\right)\right|$ is the number of concepts separating c_i and c_j through the shortest common path through the taxonomic tree, and $|concepts|$ is the total number of concepts in the ontology. A shorter path thus indicates a stronger similarity between the two concepts.

The properties similarity (*PS*) calculates the similarity between the set of properties associated with c_i and c_j. The *properties* of a concept *c* are the union of the set of attributes that have *c* as domain, and the set of relations that have *c* as domain or image.

Lastly, $EQ\left(c_i,\ c_j\right)$ uses the Levenshtein distance between the string representation of concepts c_i and c_j, written $L(c_i,\ c_j)$ below, and is defined as follows:

$$EQ\left(c_i,\ c_j\right) = \frac{1}{1+L(c_i,\ c_j)} \tag{3}$$

Attribute similarity. The attribute similarity between two attributes a_i and a_j of two concepts is calculated by a similar function:

$$\begin{gathered} AS\left(a_i,\ a_j\right) = at_1 \times EQ\left(a_i,\ a_j\right) \\ +at_2 \times VS\left(a_i,\ a_j\right) + at_3 \times CS\left(conc(a_i),\ conc(a_j)\right) \end{gathered} \tag{4}$$

Here also, the non-negative constants at_1, at_2, at_3 must add up to 1. The function $conc(a)$ returns the (most specific) concept which is in the domain of *a*. The function $VS\left(a_i,\ a_j\right)$ is defined as such:

$$VS\left(a_i,\ a_j\right) = \frac{\left|vals(a_i) \cap vals(a_j)\right|}{\left|min_{k=i,j}\{|vals(a_k)|\}\right|} \tag{5}$$

that is, the similarity of their value sets. The function $vals(a)$ returns the image of the attribute *a*.

Relation similarity. The relation similarity between two relations r_i and r_j is calculated in a similar manner:

$$RS(r_i, r_j) = rl_1 \times EQ(r_i, r_j)$$
$$+rl_2 \times CS(dconc(r_i),\ dconc(r_j)) \times CS(dconc(r_i),\ dconc(r_j)) \tag{6}$$

It is required that the sum of the non-negative constants rl_1, rl_2 be 1. The function $dconc(r)$ returns the most specific concept in the domain of r, while $iconc(r)$ returns the most specific concept in the image of r. The concept similarity is calculated twice, to compare the domains of the relations r_i and r_j (obtained by $dconc(r)$) and the images of the relations (obtained by $iconc(r)$), respectively.

Global evaluation. In order to accomplish the evaluation of a question, each of the concepts of the student's answer is associated with the closest concept of the ideal answer, given that each concept can only be used once. The similarity between each pair of concepts is then calculated and is multiplied by the relative numerical value of the concept in the ideal answer. The similarity is then added to the final grade. The same process is repeated for relations and attributes.

3.3 Procedural Knowledge Grading

Our system uses the same grading algorithm as OeLE [2]. The students' answers are compared to the teacher's ideal answer. The grades are calculated based on the most similar entity in the expected answer. In OeLE, the order of the entities is not factored in the grade and any permutation of the linguistic expressions of the student's answer yields the same grade.

However, this is not appropriate for assessing procedural knowledge in our system. If the above method is applied to evaluate text describing procedural knowledge such as algorithms-related answers, the grade calculation ought to take into account the relative order of a subset of concepts expressing procedural knowledge.

Functional concepts. In order to address this issue, we propose to add *functional concepts* to the course ontology. A functional concept represents a global procedure, a sequence of sub-procedures or individual steps to accomplish a given task.

Let us consider the following example algorithm, *DepthFirstSearch*, given in pseudocode:

```
procedure DepthFirstSearch
  VisitRoot
  VisitFirstChildNode
  VisitOtherSiblings
end
procedure VisitRoot [...]
procedure VisitFirstChildNode [...]
procedure VisitOtherSiblings [...]
```

For every procedure or sub-procedure, we create a corresponding functional concept: *DepthFirstSearch*, *VisitRoot*, *VisitFirstChildNode*, and *VisitOtherSiblings*. The last three sub-procedures could in turn be further decomposed.

The functional concepts allow for a high-level description of the algorithm and mask implementation details, which would be difficult to express in the ontology using relations or attributes. Further decomposition of *VisitRoot* into individual steps could be stated in any of the following ways:

`DepthFirstSearch <visits> Root` [using relation <visits>]

`VisitRoot <visits> Root` [same relation with a more specific concept]

`Root.visited=true` [the value of the attribute <visited> *becomes* true]

Representing functional concepts in OWL. Relationships between functions are defined as meta-functions in [17]. These meta-functions are implemented in our system as relations between two functional concepts. In this example, two instances of the <*is-preceded-by*> relation are needed. One instance is needed between *VisitFirstChildNode* and *VisitRoot*, because the root has to be visited first, and another between *VisitOtherSiblings* and *VisitFirstChildNode*, because the first child node should be visited first. Similarly, three instances of the <*is-achieved-by*> relation are used between *VisitRoot* and each of the remaining functional concepts.

The same idea is found in [18], where the relation *preceded_by* is defined similarly to <*is-preceded-by*> and can be used to order any pair of classes P and P_1. In other words, P *preceded_by* P_1 is defined as "Every P is such that there is some earlier P_1". This relation is defined as transitive, and is neither symmetric, reflexive nor antisymmetric.

In [19], an irreflexive and transitive relation *precedes* is used when "the sequence of the related events is of utmost importance for the correct interpretation". This paper also defines the inverse relation *follows*.

Similarly, the working draft: "Time Ontology in OWL" [20] of the World Wide Web Consortium (W3C) states that: "There is a *before* relation on temporal entities, which gives directionality to time. If a temporal entity T_1 is before another temporal entity T_2, then the end of T_1 is before the beginning of T_2." This relation is part of the *time* namespace.

In our implementation, the functional concepts and the <*is-preceded-by*> relation are defined as such in OWL:

```
<owl:Class rdf:about="FunctionalConcept"/>
<owl:Class rdf:about="DepthFirstSearch">
  <rdfs:subClassOf rdf:resource="FunctionalConcept"/>
</owl:Class>
<owl:Class rdf:about="VisitRoot">
  <rdfs:subClassOf rdf:resource="DepthFirstSearch"/>
</owl:Class>
<owl:Class rdf:about="VisitFirstChildNode">
  <rdfs:subClassOf rdf:resource="DepthFirstSearch"/>
</owl:Class>
```

```
<owl:Class rdf:about="VisitOtherSiblings">
  <rdfs:subClassOf rdf:resource="DepthFirstSearch"/>
</owl:Class>
<owl:ObjectProperty rdf:about="IsPrecededBy"/>
```

Note that the <*is-achieved-by*> relation is implied by the class hierarchy rooted at the concept *FunctionalConcept*, just as the <*is-a*> relation is implied by the class hierarchy in OeLE.

For every algorithm, a separate (meta) ontology lists the required orderings specific to that algorithm. Although there exists many algorithms for graph exploration, we only need to define the functional concepts once in the course ontology, and their ordering can then be declared in a separate ontology. For instance, the *BreadthFirstSearch* algorithm can be defined with the same functional concepts as above, only ordered differently.

For *DepthFirstSearch*, the meta-ontology is as follows:

```
VisitFirstChildNode <is-preceded-by> VisitRoot
VisitOtherSiblings <is-preceded-by> VisitFirstChildNode
```

Note that the following relation is also inferred by the transitive property:

```
VisitOtherSiblings <is-preceded-by> VisitRoot
```

Grading with functional concepts. In our approach, the question evaluation process remains mostly unchanged. No special treatment is given to the functional concept class hierarchy rooted at the concept *FunctionalConcept*, even though its implied relation is <*is-achieved-by*>, rather than the <*is-a*> relation implied for the other concepts. This takes into account function nesting and composition, while allowing calculating the proximity of the functional concepts.

However, the global evaluation of a student answer *R* takes into account the algorithm-specific orderings of the meta-ontology. The new evaluation function is given below:

$$FG(R) = GE(R) \times (1 - od(1 - DF(R))) \tag{7}$$

The final grade (*FG*) for the student answer *R* is proportional to the global evaluation of the answer, $GE(R)$, obtained from Section 3.2. Here, od is a constant in the interval [0,1] allowing the teacher to adjust the relative importance of the correct ordering of concepts in the global evaluation. The ordering factor of the answer, $DF(R)$, is defined as follows:

$$DF(R) = \frac{DD(R)}{|orderings|} \tag{8}$$

where $DD(R)$ represents the number of functional concepts having the right ordering in the student answer *R*, and $|orderings|$ the number of functional concepts orderings in the meta-ontology.

It should be noted that if functional concepts in the student's answer are ordered with the opposite relation (that is, <*is-followed-by*>), the evaluation algorithm inverts the relation between the functional concepts.

Also, the individual student grades are affected by the number of defined orderings. If there are only a few orderings, as demonstrated below, students are strongly penalized for every mistake. This is also the case with the concept proximity defined in Formula 2, where the number of concepts in the ontology affects students' grades. However, we can assume that the course ontology is fixed during evaluation, and that the students' grades are therefore affected similarly (in a linear fashion).

4 Working Example and Results

Using Depth-First Search as an example, we can quantify the effect of the new evaluation function on a student's answer. To simplify, we omit the conceptual grading of the answer and concentrate on the functional grading. Since the same entities are present in both the student and teacher's answers, the conceptual grade is 100%. The ideal functional answer could be as follows: "Depth-First Search **first** visits the root [of a graph], then [recursively] visits its first child node **before** visiting its other siblings." Table 2 shows the produced functional concepts.

Table 2. Example annotation of ideal answer to describe DFS (using only functional concepts).

Category	Description
(Functional) Concept	DepthFirstSearch
(Functional) Concept	VisitRoot
(Functional) Concept	VisitFirstChildNode
(Functional) Concept	VisitOtherSiblings

Any permutation of this ideal answer taken as input by the original approach would yield a grade of 100%. Now, consider the following student's answer: "Depth-First Search visits the root [of a graph], then [recursively] visits its first child node **after** visiting its other siblings." Here, "after" inverts the ordering of the two last concepts (highlighted in bold below), yielding the following answer:

Table 3. Example annotation of student's answer for DFS (using only functional concepts).

Category	Description
(Functional) Concept	DepthFirstSearch
(Functional) Concept	VisitRoot
(Functional) Concept	**VisitOtherSiblings**
(Functional) Concept	**VisitFirstChildNode**

The student gave here the incorrect ordering:

```
VisitFirstChildNode <is-preceded-by> VisitOtherSiblings
```

However, these two student orderings are correct:

```
VisitFirstChildNode <is-preceded-by> VisitRoot [inferred]
VisitOtherSiblings <is-preceded-by> VisitRoot
```

As stated above, the conceptual grading of this answer, as performed by OeLE, is 100%. By using the new evaluation function (Formula 7), the final grade (*FG*) becomes:

$$FG(R) = 100\% \times \big(1 - 1.0(1 - 66.67\%)\big) = 66.67\% \tag{9}$$

where the global evaluation (*GE*) is 100%, the ordering factor (*DF*) is 66.67%, and the constant od is given a value of 1.0. Considering that the ideal answer to this algorithm contains only three orderings for pairs of functional concepts (one is inferred) and that a third is out of order, this low grade seems acceptable, or at least a reasonable improvement over the former grade of 100% that would have been attributed had we only used the conceptual grading system.

5 Conclusion and Future Work

The work presented in this paper adapts the OeLE system to include procedural knowledge. The example was taken from an algorithms course given at Université de Moncton. This approach could be used in other domains where procedural knowledge is central to processing the text. For example, [18] and [19] apply similar methods to biomedical ontologies.

The approach put forth in this paper introduces functional concepts to represent procedural knowledge in ontologies. The class hierarchy of functional concepts is considered as a series of instances of the relation <*is-achieved-by*> instead of <*is-a*>. For every computer algorithm (or procedure, for other domains), a series of instances of the relation <*is-preceded-by*> specify an ordering for pairs of functional concepts.

In this paper, the texts were annotated manually. We are considering annotating the French texts semiautomatically as future work. The detection of the orderings (detecting keywords such as “first”, “before”, “after” in the example of Section 4) could also be performed automatically.

In the case where the student answer uses the opposite ordering relation (<*is-followed-by*>), the relation between the functional concepts is inverted prior to evaluation. Some more complex answers could require more inversions, for example if the student wrote “*X* and *Y* should be done after *Z*”.

Future work could also consider flow control structures, such as loops or branches, although the textual representation of those structures without proper indentation or braces could be ambiguous. For example, the *VisitOtherSiblings* functional concept can be decomposed into the following loop: (for every other sibling, *VisitNode*).

Another idea that could be explored would be to add the notion of recursive procedures, such as Depth-First Search. *VisitFirstChildNode* and (every *VisitNode* of) *VisitOtherSiblings* should include recursive calls. As an ideal answer, the teacher could give either: *DFS.isRecursive=true*, or *VisitFirstChildNode.isRecursive=true* and *VisitOtherSiblings.isRecursive=true*. Depending on the ideal answer given and their own answer, students could be unjustly penalized.

References

1. Bloom, B.S.: Taxonomy of Educational Objectives, Handbook 1: The Cognitive Domain. David McKay Co Inc., New York (1956)
2. Castellanos-Nieves, D., Fernández-Breis, J.T., Valencia-García, R., Martínez-Béjar, R., Iniesta-Moreno, M.: Semantic web technologies for supporting learning assessment. Information Sciences 181(9), 1517-1537 (2011)
3. Pérez-Marín, D., Pascual-Nieto, I., Rodríguez, P.: Computer-assisted assessment of free-text answers. The Knowledge Engineering Review 24(4), 353-374 (2009)
4. Callear, D., Jerrams-Smith, J., Soh, V.: CAA of short non-MCQ answers. In: Proceedings of the 5th International CAA Conference, Loughborough, UK (2001)
5. Jordan, S., Mitchell, T.: e-Assessment for learning? The potential of short-answer free-text questions with tailored feedback. British Journal of Educational Technology 40(2), 371-385 (2009)
6. Sukkarieh, J., Pulman, S., Raikes, N.: Auto-marking: using computational linguistics to score short, free text responses. In: Proceedings of the 29th IAEA Conference, Philadelphia, USA (2003)
7. Rudner, L. & Liang, T.: Automated essay scoring using Bayes' theorem. In: Proceedings of the Annual Meeting of the National Council on Measurement in Education, New Orleans, LA (2002)
8. Rosé, C., Roque, A., Bhembe, D., VanLehn, K.: A hybrid text classification approach for analysis of student essays. In: Proceedings of the HLT-NAACL Workshop on Educational Applications of NLP, Edmonton, Canada (2003)
9. Mason, O., Grove-Stephenson, I.: Automated free text marking with paperless school. In: Proceedings of the 6th International CAA Conference, Loughborough, UK (2002)
10. Burstein, J., Leacock, C., Swartz, R.: Automated evaluation of essays and short answers. In: Proceedings of the 5th International CAA Conference, Loughborough, UK (2001)
11. Hou, W.-J., Tsao, J.-H., Li, S.-Y., Chen, L.: Automatic Assessment of Students' Free-Text Answers with Support Vector Machines. LNCS 6096, 235-243 (2010)
12. Lutticke, R.: Graphic and NLP Based Assessment of Knowledge about Semantic Networks. In: Proceedings of the Artificial Intelligence in Education conference, IOS Press (2005)
13. Wiemer-Hastings, P., Allbritton, D., Arnott, E.: RMT: A dialog-based research methods tutor with or without a head. In: Proceedings of the 7th International Conference on Intelligent Tutoring Systems, Springer-Verlag, Berlin (2004)
14. Pérez-Marín, D., Alfonseca, E., Rodríguez, P., Pascual-Nieto, I.: Willow: Automatic and adaptive assessment of students free-text answers. In: Proceedings of the 22nd International Conference of the Spanish Society for the Natural Language Processing (SEPLN), Zaragoza, Spain (2006)
15. Klein, R., Kyrilov, A., Tokman, M.: Automated Assessment of Short Free-Text Responses in Computer Science using Latent Semantic Analysis. In: ITiCSE '11 Proceedings of the 16th annual joint conference on Innovation and technology in computer science education, New York, USA, pp. 158-162 (2011)

16. Fernández-Breis, J.T., Valencia-García, R., Cañavate- Cañavate, D., Vivancos-Vicente, P.J., Castellanos-Nieves, D. OeLE: Applying ontologies to support the evaluation of open questions-based tests. In: Proceedings of the KCAP'05 WORKSHOP. SW-EL'05: Aplications of Semantic Web Technologies for E-Learning (in conjunction with 3rd International Conference on Knowledge Capture (KCAP'05)), Banff, Canada (2005)
17. Aroyo, L., Dicheva, D.: Courseware authoring tasks ontology. In: Proceedings of the International Conference on Computers in Education, pp. 1319-1320. (2002)
18. Smith, B., Ceusters W., Klagges, B., Köhler, J., et al.: Relations in biomedical ontologies. Genome Biology 6(R46) (2005)
19. Schulz, S., Markó, K., Suntisrivaraporn, B. Formal representation of complex SNOMED CT expressions. BMC Medical Informatics and Decision Making 8(1) (2008)
20. World Wide Web Consortium (W3C), http://www.w3.org/TR/2006/WD-owl-time-20060927/, last accessed 2012-11-21.