Improving Large-Scale Assessment Tests by Ontology Based Approach

Vinu E. V and **P Sreenivasa Kumar**
Artificial Intelligence and Databases Lab
Department of Computer Science and Engineering
Indian Institute of Technology Madras, Chennai, India

Abstract

Knowledge formalized in ontologies can assist Intelligent Tutoring Systems (ITS) in generating question items like multiple choice questions (MCQs), to assess the level of knowledge of a learner. Existing ontology based MCQ generation techniques generate unmanageably large number of questions, but not necessarily all are relevant to the domain. These question items need to be vetted by human experts to choose a suitable subset of questions (as question-set), to conduct an assessment test. Currently, there are no automated methods to achieve this task. We propose three heuristic techniques to choose a desired number of significant MCQs that cover the required knowledge boundaries, from a given ontology. Through experimental results, we show that the question-sets generated based on our approach compare satisfactorily to the ones prepared by domain experts, in terms of precision and recall.

Introduction

Ontologies are knowledge representation structures which can be used as a platform for building many intelligent applications. Recently, due to the advancement in Semantic Web technologies and ease in publishing knowledge in the form of ontologies, many researchers have focused their research on utilizing these knowledge structures in (e-learning) educational applications. One major research area, under the broad areas of e-learning, is the ontology based assessment systems, where the ontologies are used to generate multiple choice questions (MCQs) to conduct assessment tests, for assessing the knowledge and skill of learners.

Objective questions like MCQs are widely adopted in large-scale assessment tests than their counterpart, subjective questions (e.g., essay or short answer). Most of the country-wide and world-wide tests, and tests conducted as part of online courses like MOOCS (Massive Open Online Courses) typically consist mainly of MCQs (Simon, Ercikan, and Rousseau 2013). They have advantages, such as enabling questioner to cover a large content area, and are easier to administer and score using computer. However, research (Barbara Gross 1993; Sidick, Barrett, and Doverspike 1994) shows that designing valid questions and responses is a demanding skill that can be time consuming. Hence, an automated method to achieve this task becomes necessary.

Stem:	Choose the movie which won the Oscar award of 2008, based on the novel Q&A.
Options:	a) Slumdog Millionare (*Key*)
	b) Million Dollar Baby (*Distractor 1*)
	c) Oliver (*Distractor 2*)
	d) Argo (*Distractor 3*)

There are several papers for arguing the use of ontologies in generating MCQs (M.Tosic and M.Cubric 2009; Cubric and Tosic 2010; Alsubait, Parsia, and Sattler 2012; Al-Yahya 2011; Zoumpatianos, Papasalouros, and Kotis 2011). Studies by Al-Yahya (2014) have shown that ontologies are good for generating *factual* (or *knowledge-level*) MCQs. These knowledge-level questions help in testing the first level of Bloom's taxonomy (Bloom et al. 1956), a classification of cognitive skills required for learning. An example of a factual MCQ (in short, F-MCQ) from movie domain is shown below.

F-MCQ items of the above mentioned type can be generated from the assertional facts associated with the ontology (ABox axioms). The Abox axioms corresponding to the above example are:

- `Movie(slumdog_millionare)`
- `wonAward(slumdog_millionare, oscar_acdmy_2008)`
- `basedOn(slumdog_millionare, q_&_a)`

Approaches which use ABox axioms for question generation can be categorized into two types: (1) Generic (pattern-based) factual-question generation and (2) Ontology specific factual-question generation.

Generic factual-questions are those questions which can be generated from an ontology using simple SPARQL[1] templates. These questions, which contain a set of conditions, normally ask for a solution which is explicitly present in the ABox of the ontology. The existing approaches which generate factual-questions of the form: *Which is C?* or *Which of the following is an example of concept C?* (where C is a concept symbol) or questions based on ontology statements ($< subject, predicate, object >$) with missing information (subject or object), can be considered as generic pattern-based question generation techniques.

[1] http://www.w3.org/TR/rdf-sparql-query/

Ontology specific factual questions are questions which are domain specific and may not necessarily be generated from a generic pattern (or template). For example, *Choose the movie which won more than one Oscar award.* and *Choose the shortest river in Alaska.* are two ontology specific factual-questions (unless there are predicates which explicitly specify the answer). These questions generally require additional inferencing or logic for key and distractor generation.

Even though there are various methods in the literature for generating both kinds of factual-questions, there exist two major drawbacks which hinder the practicality of these approaches: (1) human intervention is needed to screen the irrelevant or out-of-domain questions, (2) most of the MCQ generation approaches generate thousands of MCQs, making it difficult even for a human expert to make the selection of a small question-set.

Research by Abacha, Silveira, and Pruski (2013); Alsubait, Parsia, and Sattler (2014) support our argument that a strategy is needed to choose a significant set of questions from the massive number of generated question items. Herein, we address this issue by proposing three screening techniques based on a few observed heuristics.

In this paper, we focus on significant question selection of generic factual-question types alone. We propose a systematic method to generate these generic factual-questions. In addition to our novel screening method and the approach that we follow for generating generic factual-questions, the other contribution of this paper is a simple SPARQL based distracting answer generation technique.

We study the efficacy of the proposed screening techniques in filtering a significant question-set by comparing it with question-sets created by domain experts. Our results show that the screening techniques are effective in generating question-sets, which can be compared satisfactorily to the ones prepared by domain experts, in terms of precision and recall.

Generic Factual-Question Generation

A generic F-MCQ stem can be considered as a set of conditions formed using different combinations of predicates (or properties) associated with an instance in an ontology. Due to space limitations, we limit our illustration to factual-questions which can be framed using at most two predicates.

Formation of predicate combinations

An instance in an ontology can have 2 types of binary predicates and 1 type of unary predicate associated with it.

- Binary predicates: Object property and Datatype property
- Unary predicates: Classes are modeled as unary predicates, using `rdf:type`

We can denote the possible property combinations of size one w.r.t. an instance x as: $x\ \overrightarrow{O_1}\ i_1,\ x\ \overleftarrow{O_1}\ i_1\ ,\ x\ \overrightarrow{D_1}\ v_1$ and $x\ \overrightarrow{a}\ C_1$, where i_1 is an instance, $\overrightarrow{a}$ is `rdf:type`, $\overrightarrow{O_1}$ and $\overleftarrow{O_1}$ represent object properties of different directions, $\overrightarrow{D_1}$ denotes datatype property, v_1 stands for the value of the datatype property and C_1 is a class name. We call the instance x as the *pivot-instance* of the question-pattern. The arrows ($\leftarrow$ and $\rightarrow$) represent the directions of the properties w.r.t. the pivot-instance. In this paper, we use the terms question-pattern and property combination interchangeably.

Property Combinations: ↓ Size: 1	2
1) $x\ \overrightarrow{O_1}\ i_1$	5) $i_2\ \overleftarrow{O_2}\ x\ \overrightarrow{O_1}\ i_1$ 6) $i_2\ \overrightarrow{O_2}\ x\ \overrightarrow{O_1}\ i_1$ 7) $v_1\ \overleftarrow{D_1}\ x\ \overrightarrow{O_1}\ i_1$ 8) $C_1\ \overleftarrow{a}\ x\ \overrightarrow{O_1}\ i_1$
2) $x\ \overleftarrow{O_1}\ i_1$	$i_2\ \overleftarrow{O_2}\ x\ \overleftarrow{O_1}\ i_1$ (6) 9) $i_2\ \overrightarrow{O_2}\ x\ \overleftarrow{O_1}\ i_1$ 10) $v_1\ \overleftarrow{D_1}\ x\ \overleftarrow{O_1}\ i_1$ 11) $C_1\ \overleftarrow{a}\ x\ \overleftarrow{O_1}\ i_1$

Property Combinations: ↓ Size: 1	2
3) $x\ \overrightarrow{D_1}\ v_1$	$i_1\ \overleftarrow{O_1}\ x\ \overrightarrow{D_1}\ v_1$ (7) $i_1\ \overrightarrow{O_1}\ x\ \overrightarrow{D_1}\ v_1$ (10) 12) $v_2\ \overleftarrow{D_2}\ x\ \overrightarrow{D_1}\ v_1$ 13) $C_1\ \overleftarrow{a}\ x\ \overrightarrow{D_1}\ v_1$
4) $x\ \overrightarrow{a}\ C_1$	$i_1\ \overleftarrow{O_1}\ x\ \overrightarrow{a}\ C_1$ (8) $i_1\ \overrightarrow{O_1}\ x\ \overrightarrow{a}\ C_1$ (11) $v_1\ \overleftarrow{D_1}\ x\ \overrightarrow{a}\ C_1$ (13) 14) $C_2\ \overleftarrow{a}\ x\ \overrightarrow{a}\ C_1$

Table 1: Property combinations of size 1 and 2.

Stem	Ptn.
Choose a `SoccerPlayer` *with* `has_team FC_Barcelona.`	8
`Hamlet is_written_by _____, died_on April 23,1616.`	7
Choose the `American_President, born_on Feb 12,1809.`	13

Table 2: Sample MCQ stems that can be generated using the patterns (Ptn. 8, 7 and 13).

In Table 1, we show the formation of possible property combinations of size two by adding predicates to the four combinations of size one. Repetitions in the combinations are marked with the pattern number of the matching pattern. It should be noted that, in the pattern representation, we consider only the directionality and the type of the properties, but not their order. Therefore the combinations like $i_2\ \overleftarrow{O_2}\ x\ \overleftarrow{O_1}\ i_1$ and $i_2\ \overrightarrow{O_2}\ x\ \overrightarrow{O_1}\ i_1$ are considered to be the same; we represent one as the duplicate of the other. After avoiding the duplicate combinations, we get four combinations of size one and ten combinations of size two.

To illustrate the use of the question patterns in Table 1 in question generation, a few examples of the possible stems of some of these patterns are given in Table 2. The conversion of these patterns into MCQ stems is based on the templates associated with each pattern (see the example in the next section).

Practicality Issue of Pattern-Based Question Generation

For the efficient retrieval of data from the knowledge base, we transform each of the patterns into SPARQL queries. For example, the stem-template and query corresponding to the pattern $C_1\ \overleftarrow{a}\ x\ \overrightarrow{O_1}\ i_1$ (Pattern-8) are:

– *Choose a* [`?C1`] *with* [`?O1`] [`?i1`].

```
select ?C1 ?x ?O1 ?i1  where { ?x a ?C1.
    ?x ?O ?i1. ?O1 a owl:ObjectProperty. }
```

These queries when used to retrieve tuples from ontologies, may generate a large result set. Last column of Table 3 lists the total count of tuples that are generated using the fourteen patterns from a set of sample ontologies (where Mahabharata ontology is developed by our research team and the other

Ontology	Instances	Concepts	Object properties	Datatype properties	Total tuple count
Mahabharata	181	17	24	9	25,550
Geography	713	9	17	11	61,861
Restaurant	9747	4	9	5	15,261,063
Job	4138	7	7	12	17,889,466

Table 3: Specifications of the sample ontologies and the respective tuple counts.

ontologies by Mooney's research group[2]). These tuple counts represent the possible generic factual-questions that can be generated from the respective ontologies.

An MCQ based exam is mainly meant to test the wider domain knowledge with a fewer number of questions. Therefore, there is a need to select a small set of significant tuples from the large result set, to create a good MCQ question-set. But, the widely adopted method of random sampling can result in poor question-sets (we verify this in the Experimental Evaluation section). In the next section, we propose three screening heuristic techniques to choose the most appropriate set of tuples (questions) from the large result set.

Significant Question-Set Selection Heuristics

In this section, we introduce three screening techniques: (1) Property based screening (2) Concept based screening (3) Similarity based screening. We consider a movie related ontology (addressed as Movie ontology) as an example ontology for illustrating the screening methods.

Property based screening

In this screening method, tuples generated using each of the fourteen question patterns will be given as input. We first group these tuples based on the properties they contain. We call the properties which we use for grouping as the *property sequence* of each group. Figure 1 shows a sample tuple-set generated using Pattern-5 from Movie ontology. The property sequences corresponding to each of the grouped tuples are also shown in the figure (as P_1, P_2, and P_3). If the count of the resultant tuples w.r.t. a property sequence is comparatively high, we can omit that property sequence from question generation. This is based on the intuition that the questions based on common properties of a set of objects appear to be trivial. These questions can be categorized as routine questions and are less likely to be chosen by a human to conduct a test. For example, in Movie ontology, the questions formed using the properties like `isDirectedBy, starring` are trivial questions when compared to the questions framed using the predicates like `wonAward, isBasedOn`. This is because, the former predicate combination is present for all movie instances and the latter is present only for selected movie instances.

To compare the significance of property sequences, we assign a triviality score (*Property Sequence Triviality Score* (denoted as $PSTS$)) to each of the property sequences, such that a lower triviality score means high significance and vice

[2]https://files.ifi.uzh.ch/ddis/oldweb/ddis/research/talking-to-the-semantic-web/owl-test-data/ (last accessed 27/01/2015)

x	O_1	i_1	O_2	i_2	*Property Sequence (P)*
1. braveheart	isDirectedBy	melGibson	starring	sophieMarceau	
2. titanic	isDirectedBy	jamesCameron	starring	kateWinslet	
3. braveheart	isDirectedBy	melGibson	starring	patrickMcGoohan	P_1 ={isDirectedBy,
4. rush	isDirectedBy	ronHoward	starring	chrisHemsworth	starring}
...					
n. deja_vu	isDirectedBy	tonyScott	starring	denzelWashington	
1. argo	wonAward	oscar_academy_12	isBasedon	the_great _escape	
2. a_beautiful_mind	wonAward	oscar_academy_01	isBasedon	a_beautiful_mind_novel	P_2 ={wonAward,
...					isBasedon}
m.forrest_gump	wonAward	oscar_academy_94	isBasedon	forrest_gump_novel	
1. new_zealand	hasCapital	wellington	largestCity	auckland	
2. india	hasCapital	new_delhi	largestCity	mumbai	P_3 ={hasCapital,
...					largestCity}
l. sweden	hasCapital	stockholm	largestCity	stockholm	

Figure 1: Shows the set of tuples generated using SPARQL query for Pattern 5. We grouped the tuples w.r.t. the properties (O_1 and O_2). Here, $n \gg m > l$.

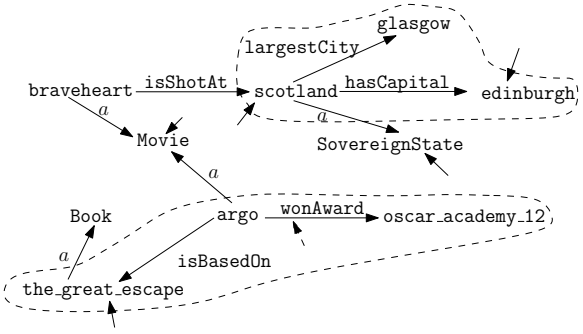

Figure 2: A portion of the RDF graph of Movie ontology, where a denotes `rdf:type`.

versa. We define $PSTS$ of a property sequence P as follows, where the *Potential-set* of P is the set of those instances which may *possibly* satisfy all the properties in P.

$$PSTS(P) = \frac{\#\textit{ Instances satisfying all the properties in P}}{\#\textit{ Instances in the Potential-set of P}}$$

Potential-set of P is specified by the expression $Type(q, P, r)$, where q is question-pattern and r denotes a pattern element. $Type(q, P, r)$ is derived by considering the intersection of the class constraints and, the domain and range of those properties in P which are associated with r. For example, consider the property sequence $P = \{p_1, p_2\}$ in the pattern $q = i_2 \overrightarrow{p_1}\, x \overrightarrow{p_2}\, i_1$ and $r = x$. Then, $Type(q, P, x)$ is taken as $Range(p_1) \cap Domain(p_2)$. Similarly for $P = \{C_1, p_1\}$ and $q = C_1 \overleftarrow{a}\, x \overleftarrow{p_1}\, i_1$, $Type(q, P, x) = C_1 \cap Range(p_1)$. For the same q and P, if r is i_1, we take $Type(q, P, i_1)$ as $Domain(p_1)$. In $PSTS$ calculation, we use r as the pivot-instance x for finding the potential-set.

The property sequences which are satisfied by more number of instances can be avoided by considering an appropriate (dataset specific) maximum triviality score threshold (T_p). Depending on the number of questions (or tuples) to be obtained, we can choose the T_p. Continuing our example, if the calculated $PSTS$ of our property sequences P_1, P_2 and P_3 are 0.9, 0.33 and 0.82 respectively, by considering a T_p of 0.85, we can omit P_1 from question framing.

'#' stands for number of or cardinality of

Concept based screening

As mentioned in the previous section, the generic factual-questions that are generated from a domain ontology may contain questions which are irrelevant to the domain. For example, consider the following questions corresponding to the tuples generated from Movie ontology: (1) *Choose the movie which is based on The Great Escape and won an Oscar-award*; (2) *Choose the Sovereign state with capital Edinburgh and having largest city Glasgow.* Even though these questions are generated using property sequences selected in the previous screening method (P_2 and P_3), the first question is more related to the domain of the ontology than the other. So, in this screening method, our heuristics for determining the importance of a question is by cross checking the type information of the main instance in the tuple with the important classes of the domain ontology. If the instance belongs to any of the important classes, we conclude that the question is significant. In this work, we consider the pivot-instance in each tuple as the main instance.

The portion of the RDF graph corresponding to our example questions is shown in Figure 2. The first question is important because the pivot-instance `argo` is a member of the class `Movie`, which is an important concept of Movie ontology. But the class `SovereignState` in the second question is not an important class of the domain. Therefore, we can categorize the second question as a less significant one.

Potentially important class selection In the literature, there exist a number of efforts (Peroni, Motta, and dAquin 2008; Wu et al. 2008), to identify important concepts of an ontology. Among the different works, the approach by Peroni et al. (2008) gained much attention. This is because, the important concepts are identified by considering topological measures such as *density* and *coverage*, as well as statistical lexical measures (*popularity*), and cognitive criteria (*natural categories*). Their approach is experimentally proved to select important concepts which have excellent degree of correlation with those concepts chosen by human experts. In our implementation, we used their method for choosing the important classes of the sample ontologies.

Similarity based screening

The tuple-set (represented as S) selected using the above two methods may not be sufficiently small for conducting an assessment test. For example, consider that there exist 1000 movie instances in the `Movie` class (an important concept) of our Movie ontology, which satisfy all the properties in the property sequence P_2. Then, questions about all these 1000 instances will appear in our filtered result set. We can possibly remove some of these questions, since all these questions are useful only to test a learner's knowledge on a specific portion of the domain knowledge. Therefore, in this third level of screening, by considering the similarity of the tuples in the set S, we select a small set of representative tuples to achieve this task.

Graphical representation and coverage Consider the tuple-set S as $\{t_1, t_2, ..., t_h\}$, where h is the count of the filtered tuples. Each of these tuples $t \in S$ can be considered as a set of triples. A triple in t is in either of the two forms: $< s,\ p,\ o >$, $< s,\ \texttt{rdf:type},\ C >$, where p is a property, s is a subject, o is an object and C is a class name.

We can consider an undirected graph $G = (V, E)$, with vertex set $V = \{t \mid t \in S\}$, and edge set $E = \{\ (t_1, t_2)\ |\ t_1, t_2 \in S\ and\ Similarity(t_1, t_2) \geq c\ \}$, where $Similarity(.)$ is a symmetric function which determines the similarity of two tuples w.r.t. their pivot-instances and c is the minimum similarity score threshold.

$$Similarity(t_1, t_2) = \frac{1}{2}\Bigg(\frac{\#(\ X(P(t_1)) \cap X(P(t_2))\ )}{\#(\ X(P(t_1)) \cup X(P(t_2))\ )} + \frac{\#Triples\ in\ t_1\ Semantically\ Equivalent\ to\ triples\ in\ t_2}{Max(\#Triples\ in\ t_1, \#Triples\ in\ t_2)}\Bigg)$$

In the equation to calculate similarity of tuples, $P(t)$ represents the property sequence of t, and $X(P(t))$ denotes the set of instances (in the ontology) which satisfies the properties in $P(t)$. We calculate the similarity score of two tuples based on the relationship between (unary and binary) predicates in one tuple to their counterparts in the other tuple, and the number of the semantically similar triples in them.

A score based on the matching predicates in the respective tuples, is given by the first part of the equation. $X(P(t_1))$ and $X(P(t_2))$ become equal when there is a one-to-one correspondence of predicates in the tuples. In the second part of the equation, semantic equivalence of triples $< s_1,\ p_1,\ o_1 >$ and $< s_2,\ p_2,\ o_2 >$ (in t_1 and t_2 respectively) is calculated by considering sub-property, symmetric and inverse relationships between p_1 and p_2 while matching. For example, $<$ `jacky, hasFriend, bobby` $>$ is equivalent to $<$ `bobby, isFriendOf, jacky` $>$, if `hasFriend` is the inverse of `isFriendOf`.

Once a graph G is created based on a minimum similarity score threshold, we can select a representative set of tuples (or vertices) based of the connectivity of the graph. A question-set, containing dissimilar questions, tends to check a wider knowledge than a question-set containing similar questions. To make a question-set small enough to check the same knowledge, we can remove questions of similar types, after maintaining a representative question from among them. In this regard, a question-set created using representative tuples (denoted as as Rep_Set) can indeed be considered as a good question-set. We observe that the properties of a DOMINATING SET (graph theory) satisfy the required characteristics of a Rep_Set. To recall, a dominating set D for a graph $G(V, E)$ is a subset of V such that every vertex not in D is connected to at least one vertex in D. Since we are interested in the smallest set of representative tuples, we find MINIMUM DOMINATING SET of vertices from G.

Finding the minimum dominating set of vertices from a graph is an NP-hard problem; so we use a modified version of the minimum vertex-cover approximation algorithm available in Java graph-theory library (http://jgrapht.org/), to find the minimum dominating set.

Distractor Generation

Distractors (or distracting answers) are one of the main components which determines the quality of an MCQ item (Woodford and Bancroft 2005). Closeness of distractors with the

correct answer is one factor which helps in determining the difficulty level of an MCQ.

In this work, we adopt a simple SPARQL based method to find the distractors of a selected question tuple. For each significant tuple, we modify the corresponding pattern's SPARQL template, and assign values to the query variables, except for the variable that corresponds to the key. For example, consider `[argo][wonAward][oscar_academy_12][isBasedon][the_great_escape]` as an important tuple (from Figure 1) of Pattern-5 (i.e., $i_2 \overleftarrow{O_2}\, x\, \overrightarrow{O_1}\, i_1$), with variable x denoting the key. The query after replacing the pattern's SPARQL template variables with values is:

```
select ?x where { ?x wonAward oscar_academy_12.
                  ?x isBasedon the_great_escape. }
```

Let W be the result set of the above query. Now, the distractors of a tuple t with k as the *key* and q as the corresponding question-pattern is defined as:

$$Distractor(t, k, q) = Poten.Set(t) - W$$

In the above equation, $Poten.Set(t)$ denotes the potential-set that corresponds to a tuple t, and is defined as $Type(Q(t), P(t), k)$, where $Q(t)$ and $P(t)$ denote the question-pattern and the property sequence respectively of t (see the section Property based screening). If this equation gives a null set or a lesser number of distractors when compared to the required number of options, we can always choose any instance or datatype value other than those in $Poten.Set(t)$ as a distractor. This is represented in the following equation, where U is the whole set of instances and datatype values in the ontology. The distractors generated using the following equation are considered to be farther from the key than those generated using the above equation.

$$Distractor_{appro.}(t, k, q) = U - Poten.Set(t)$$

Experimental Evaluation

We consider two ontologies for our detailed evaluation.

- *Mahabharata* (or Mahabh.) *ontology*: based on the characters of the epic story of Mahabharata[4].
- *Geography* (or Geo.) *ontology*: based on geographical data of United States, was developed by Ray Mooney and his research groupat the University of Texas.

The entity counts of the ontologies are detailed in Table 3. We carefully developed Mahabharata ontology with a sole intention to find out the effectiveness of our proposed approaches. Domain experts of Mahabharata provided knowledge for modeling the ontology.

Outline of the evaluation process

Domain experts prepared three question-sets (Set-A, Set-B and Set-C) of sizes 25, 50 and 75 from each of the two test ontologies. These question-sets are considered to be the benchmark-sets (abbreviated as BM-sets). In case of Geo. ontology, domain experts created BM-sets of the required sizes from the ontology-data-related questions collected by Mooney's research group.We then generated three question-sets which correspond to the cardinalities of the BM-sets, using our screening techniques. These automatically generated sets are denoted as AG-sets. All the question-sets (both BM-sets and AG-sets) and the test ontologies that are involved in our experiments are available at our web page[5].

Ontology	# Significant Qns.		
	$Count_{Req} = 25$	$Count_{Req} = 50$	$Count_{Req} = 75$
Mahabh.	44	81	118
Geo.	28	61	93

Table 4: Number of tuples in the AG-sets corresponding to different $Count_{Req}$ values.

For a comparison of question-sets, and to study the effectiveness of the proposed approaches, we calculated the *precision* and *recall* of each AG-set with the corresponding BM-set.

Automated question-set generation

In the screening methods that we discussed before, there are three parameters which help in controlling the final question count: T_p (max. triviality score threshold), I (number of important concepts) and c (min. similarity score threshold). Appropriate values for each of these parameters are determined in a sequential manner; the T_p limits the use of common property patterns; then, the I helps in selecting only those questions which are related to the most important domain concepts; the parameter c is used to choose a dispersed set of questions that spans the domain knowledge.

Question-sets of required sizes ($Count_{Req} = 25, 50$ and 75) are generated by finding suitable values for each of the three ontology specific parameters, using the following approximation method.

The parameters T_p and I are not only ontology specific but also specific to each of the 14 patterns. For each pattern, we choose a suitable value for T_p (T'_p) such that the first screening process will generate a tuple-set whose cardinality is relatively larger than the required count. In our experiments, we choose a T'_p which can generate (nearly) thrice the required count ($Count_{Req}$). Considering a higher T'_p can increase the variety of property combinations in the final tuple-set. In the second level of screening, we choose an I value (I'), which reduces the tuple-set to the required size. Since we are repeating this procedure for all 14 patterns, we can expect a total question count of approximately 14×25 (for $Count_{Req} = 25$) or 14×50 (for $Count_{Req} = 50$) or 14×75 (for $Count_{Req} = 75$). Therefore, in the next level of screening we choose a c value (c') which can generate a tuple-set of cardinality approximately equal to $Count_{Req}$. Table 4 shows the count of question tuples filtered using suitable parameter values from our two test ontologies.

Comparison of *AG* and *BM* question-sets

Comparison of the two question-sets involves finding the semantic similarity of questions in one set to their counterpart

[4]http://en.wikipedia.org/wiki/Mahabharata

[5]https://sites.google.com/site/ontomcqs/research

$Count_{Req}$	Ontology	Our approach		Random selectn.	
		Prec.	*Rec.*	*Prec.*	*Rec.*
25 (Vs. Set-A)	Mahabh.	0.72	0.80	0.17	0.04
	Geo.	0.82	0.52	0.14	0.10
50 (Vs. Set-B)	Mahabh.	0.91	0.55	0.11	0.11
	Geo.	0.91	0.47	0.19	0.11
75 (Vs. Set-C)	Mahabh.	0.92	0.62	0.24	0.04
	Geo.	0.93	0.43	0.21	0.13

Table 5: Precision and recall of AG-sets against BM-sets.

in the other. To make the comparison precise, we converted the questions in the BM-sets into their corresponding tuple representation. Since, AG-set is already available in the form of tuple-set, the similarity measure which we used in the previous section is adopted to find the similar tuples across the two sets. For each of the tuples in AG-set, we find the most matching tuple in the BM-set, thereby establishing a mapping between the sets. We considered a minimum similarity score of 0.5 (ensuring partial similarity) to count the tuples as matching ones.

After the mapping process, we calculated the *precision* and *recall* of AG-sets to measure the effectiveness of our approach. The precision ($Prec.$) and recall ($Rec.$) are calculated in our context as:

$$Prec. = \frac{\#\textit{Mapped AG-set tuples}}{\#\textit{AG-set}} \qquad Rec. = \frac{\#\textit{Mapped BM-set tuples}}{\#\textit{BM-set}}$$

It should be noted that, according to the above equations, a high precision does not always ensure a good question-set. The case where multiple generated questions matching the same benchmark candidate is such an example. Therefore, the recall corresponding to the AG-set (which gives the percentage of the number of benchmark questions that are covered by the AG-set) should also be high enough for a good question-set.

Table 5 shows the precision and recall of the question-sets generated by the proposed approach as well as the random selection method, calculated against the corresponding benchmark question-sets: Set-A, Set-B and Set-C.

The evaluation shows that, in terms of precision values, the AG-sets generated using our approach are significantly better than those generated using random method. The recall values are in an acceptable range ($\approx 50\%$).

Conclusion

An ontology based factual-MCQ generation method is presented with a set of heuristics to choose a desired number of significant questions. To the best of our knowledge, the proposed screening methods are the first of their kind and, result in a practical method for automatically generating a good quality question-set from a given ontology.

In this paper, we focus on question patterns with at most two predicates. Techniques that involve more number of predicates need to be investigated. Another factor which needs to be considered is controlling the hardness level of an MCQ. A further investigation on the pedagogical value of the generated questions and, a study on the *Ontology specific factual questions* also need to be done.

References

Abacha, A. B.; Silveira, M. D.; and Pruski, C. 2013. Medical ontology validation through question answering. In *AIME*, 196–205.

Al-Yahya, M. M. 2011. Ontoque: A question generation engine for educational assesment based on domain ontologies. In *ICALT*, 393–395. IEEE Computer Society.

Al-Yahya, M. 2014. Ontology-based multiple choice question generation. *The Scientific World Journal* Vol 2014, page 9, ID: 10.1155/2014/274949.

Alsubait, T.; Parsia, B.; and Sattler, U. 2012. Mining ontologies for analogy questions: A similarity-based approach. volume 849 of *CEUR Workshop Proceedings*. OWL Experiences and Directions.

Alsubait, T.; Parsia, B.; and Sattler, U. 2014. Generating multiple choice questions from ontologies: Lessons learnt. volume 1265 of *CEUR Workshop Proceedings*. OWL Experiences and Directions.

Barbara Gross, D. 1993. *Tools for Teaching*. Jossey-Bass Inc., San Francisco, California.

Bloom, B. S.; Engelhart, M. D.; Furst, E. J.; Hill, W. H.; and Krathwohl, D. R. 1956. *Taxonomy Of Educational Objectives: Handbook 1, The Cognitive Domain*. Boston: Allyn & Bacon.

Cubric, M., and Tosic, M. 2010. Towards automatic generation of e-assessment using semantic web technologies. In *Proceedings of the 2010 International Computer Assisted Assessment Conference*.

M.Tosic, and M.Cubric. 2009. SeMCQ- Protege Plugin for Automatic Ontology- Driven Multiple Choice Question Tests Generation. In *Proceedings of the 11th International Protege Conference*.

Peroni, S.; Motta, E.; and dAquin, M. 2008. Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. In Domingue, J., and Anutariya, C., eds., *The Semantic Web*, volume 5367 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 242–256.

Sidick, J. T.; Barrett, G. V.; and Doverspike, D. 1994. Three-alternative multiple-choice tests: An attractive option. *Personnel Psychology* Vol 47, Issue 4, pages 829835.

Simon, M.; Ercikan, K.; and Rousseau, M. 2013. *Improving Large Scale Education Assessment: Theory, Issues, and Practice*. Routledge, New York.

Woodford, K., and Bancroft, P. 2005. Multiple choice questions not considered harmful. In *Proceedings of the 7th Australasian Conference on Computing Education - Volume 42*, ACE '05, 109–116. Darlinghurst, Australia, Australia: Australian Computer Society, Inc.

Wu, G.; Li, J.; Feng, L.; and Wang, K. 2008. Identifying potentially important concepts and relations in an ontology. In *International Semantic Web Conference*, 33–49.

Zoumpatianos, K.; Papasalouros, A.; and Kotis, K. 2011. Automated transformation of swrl rules into multiple-choice questions. In Murray, R. C., and McCarthy, P. M., eds., *FLAIRS Conference*. AAAI Press.