

Higher Cognitive Items Generation Algorithms

Ebenezer Aggrey[1], Maiga Chang[1], Rita Kuo[2], and Xiaokun Zhang[1]

[1] School of Computing and Information Systems, Athabasca University, Canada
[2] Department of Computer Science and Engineering, New Mexico institute of Mining and Technology, USA
aggreyeb@shaw.ca, rita.mcsl@gmail.com, maiga@ms2.hinet.net, xiaokunz@athabascau.ca

Abstract. The main goal of this research is to design item generation algorithms which can be integrated into the Online Test System developed earlier by the authors. The algorithms will be capable of generating items belong to higher cognitive level based on Bloom Taxonomy from a knowledge map created by a teacher (or co-created by a group of teachers). With the help of such integrated system teachers can reduce the time and effort they spend to prepare tests for assessing students' mastery and understanding level of what they taught in class. This paper discusses the proposed algorithms in details and explains the experiment design in the end.

Keywords: Knowledge Map. Item Generation. Hierarchical Concept Map. Concept Schema. Test Items generation Algorithms.

1. Introduction

Data structure is one of the foundation courses which is widely taught in science and engineering in many higher institutions. Chris, is a teacher who teaches undergraduate computer science. He taught his class data structure interfaces and their implementations concepts and wished to use the data structure concept hierarchy he presented to prepare higher and lower cognitive multiple choice test items to assess his students' cognitive abilities.

So what did Chris do? He spent days and a lot of efforts to prepare the test items and mark students' answer sheets manually, then he wondered whether or not there is a way to generate the test items from the data structure concepts and to mark students' answer sheets automatically. The answer is Yes, many researchers in literature have used different techniques: template-based [4], Natural Language Processing-based [6], and Knowledge map-based [7], to develop automatic item generation systems with the aim to reduce the time and effort that Chris and other teachers may need to spend on preparing test items. However, most of these systems can only generate items for lower cognitive levels.

© Springer Nature Singapore Pte Ltd. 2018

M. Chang et al. (eds.), *Challenges and Solutions in Smart Learning*, Lecture Notes in Educational Technology,
https://doi.org/10.1007/978-981-10-8743-1_8

The purpose of this research is to design item generation algorithms for existing Online Test System (OTS) research which allows teachers to create knowledge map for their own course and manage tests for their students. The research makes the OTS capable of generating items for both lower and higher cognitive level automatically based on the knowledge maps the teachers created. With the proposed algorithms' help, teachers will be able to generate items as their item bank for creating different tests with few effort.

The rest of this paper is organized as follows. Section 2 reviews the relevant literature for models applied in this research. Section 3 analyzes the conceptual model such as the Knowledge Map, Test item model. In addition, the general algorithm architecture will be presented in this section. In Section 4 the data to be used as inputs of the item generation algorithms and the algorithm design will be explained. Furthermore, the example demonstrating how major and minor algorithms working together to generate the items with outputs will be presented and explained. Section 5 explains the evaluation plan this research intend to verify the usability of the Online Test System and the effectiveness of the item generation algorithms in terms of whether or not the cognitive level each generated item belongs to is expected. Finally, Section 6 summarizes the research and discusses future works that can done later.

2. Relevant Research

Template-based [1, 11] and Knowledge Map approach [2, 7] are some of the techniques reported in literature to generate multiple choice items. While most of these approaches can generate items for lower cognitive levels and aims to assist teachers to reduce the time and effort they use to prepare test items, they cannot generate higher cognitive items. However, Template-based approach is cost effective in the sense that it can be uses to generate large amount test items by manipulating stimulus, stem, and options placeholder only. Moreover, common errors in developing multiple choice item such as omissions and additions of words, phrases, spelling, punctuations, capitalization, item structure, typeface, formatting can be avoided [5] because only the stimulus, stem, and options of the questions are being changed during question generation [10].

On the other hand Knowledge map approach has the ability to store additional information about the concept under consideration in its concept schema [2, 3] hence it can assist in the generation of higher cognitive items. This research investigate how to generate items for both higher and lower cognitive level from a knowledge map of Data Structure in the computer science domain.

3. Analysis

In order to define the specification for the item generation algorithms the research team performed the following iterative system analysis tasks.

3.1 Selecting Taxonomy to classify the items to be generated

The items to be generated has to be classified as higher or lower cognitive levels hence Blooms Taxonomy [8] was selected. It provides a useful guidelines to classify knowledge as well as cognitive processes to demonstrate learning. Many researchers in literature have leveraged it successfully to evaluate cognitive abilities of students and also prepare learning objectives [2].

Blooms Taxonomy uses verbs "Remember", "Understand" to signify lower cognitive process and "Apply", "Analyze", "Evaluate" as higher cognitive activities. Whiles lower cognitive process expects students to recall facts about concepts and classify concepts based on their characteristics and behavior, higher cognitive activities expect students to know how a concept under consideration works in real world situation. In context of Data Structure in computer science domain, for example given application of specific data structure the student will be able to select the suitable data structure for that particular task. Another example will be given a software components which utilizes the same data structure with their respective time complexity or order of growth of the implemented algorithms students will be able to select the best software component if running time is required specification or the memory usage is concern.

3.2 Selecting Knowledge Structure and defining conceptual model

For teachers to create and manage the knowledge structure – a conceptual model which defines the concepts and their properties, relationships, and constraints has to be develop. The research selected knowledge map for this task. Knowledge Map is a graphical representation of knowledge. Conceptually, it consist of two parts namely concept hierarchy and concept schema. Whiles the concept hierarchy presents the relations between concepts of interests, the concept schema store related information associated with the concepts [7]. Fig. 1 shows an example of concept hierarchy for the interface and its implementation in Data Structure course.

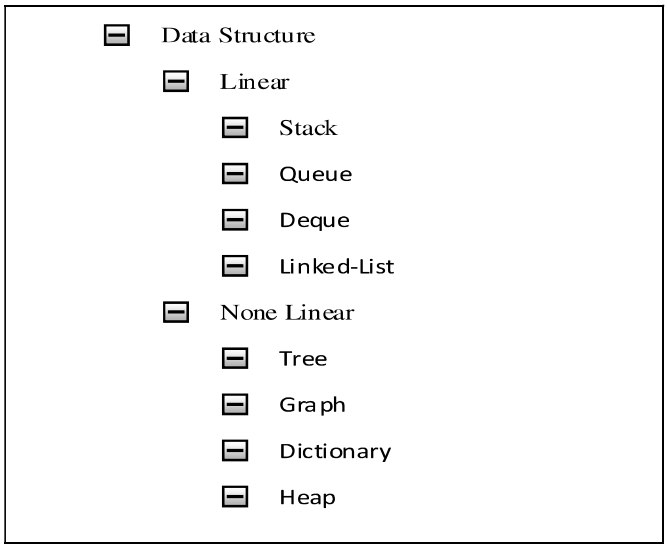


Fig. 1. A concept hierarchy example

The concepts are organized in a tree like structure. The root node is "Data Structure" and Linear and none linear are type of data structure. Stack, Queue, Deque, Linked-List are type of linear data structure and Tree, Graph, Dictionary, and Heap are type of None Linear data structure.

Formally, Knowledge Map is defined as KM = (H, S) where H is concept hierarchy and S is its concept schema. Concept hierarchy is a 3- tuple defined as H = (D, R, K) where D = {d1, d2…, dn} is a set of domain concepts, R = {r1, r2…, rn} is a set of relation between concepts and their parent. e.g. {"Type of", "Part of"}, and K = {k1, k2…, kn} is a set of constraints that should hold to create the concept or relations to be established.

Constraints for concept hierarchy are: IF H is a concept hierarchy THEN (1) H should have special node n, called Root with no parent node. (2) Each node referred in this research as Concept Node (N) such that $N \neq n$ has a unique parent node. And, (3) each concept node may have concept schema. It can inherit concept schema from parent node.

Concept schema stores additional information (i.e. relation name, concept name, concept action, attribute name, and attribute value). An example partial concept schemas for linear data structure is shown in table 4.

3.3 Defining conceptual model for Item

In this research an item consist of (1) stimulus which can be text or code segment that gives context to the question to being asked, (2) stem which is a question based on the stimulus, (3) answer options which consist of distractors, i.e. incorrect answers, (4) correct answer also known as the key, and, (5) cognitive type. There are four constraints for creating an item: (1) each item must have a stimulus; (2) each

item must have a stem; (3) each item must have at least two answer options and up to four; and, (4) each item must have one correct answer.

For example a typical test item and it components (stimulus, stem, answer options are shown table 1.

Table 1. Typical Apply Cognitive type items

Cognitive Type	Stimulus	Stem	Answer Options
Apply	A programmer was asked to implement Stack data structure to be used for software component that simulate Last In First Out mechanism (LIFO).	Select the suitable functions the programmer has to implement	A. Push ,pop, peek B. peek ,enqueue, dequeue C. addFirst ,addLast, removeFirst D. add ,remove, contains
Apply	A student designed and implemented data structure for software module. Upon uniting testing it was found that the data structure exhibit Last In First Out mechanism (LIFO).	Choose the possible data structure the student implemented.	A. Queue B. Stack C. Deque D. Linked-List

3.4 Designing Item Algorithm Workflow

Figure 5 shows the workflow of item generation algorithm. The workflow has seven steps described as following in details:

1. A teacher can select cognitive type and a concept node. In this case the teacher wants the system to generate items for cognitive type Apply hence he or she choose cognitive type ct_A from the list and concept node c_t from the concept hierarchy. The selections then are sent as inputs of the "Algorithm Selection" function as Step 1 shows.

2. When "Algorithm Selection" function receives the chosen concept node and cognitive type, it select appropriate proper major algorithm accordingly. Since ct_A is received, the "Apply Item Generation" Algorithm is activated (as Step 2 shows) and the c_t and ct_A are passed into it as inputs.

3. The "Apply Item Generation" Algorithm uses ct_A to retrieve item rules of cognitive type Apply (R^A) as Step 3 shows.
4. After the algorithm gets R^A, it enters into a loop enumerating through the rules and perform the following tasks sequentially:

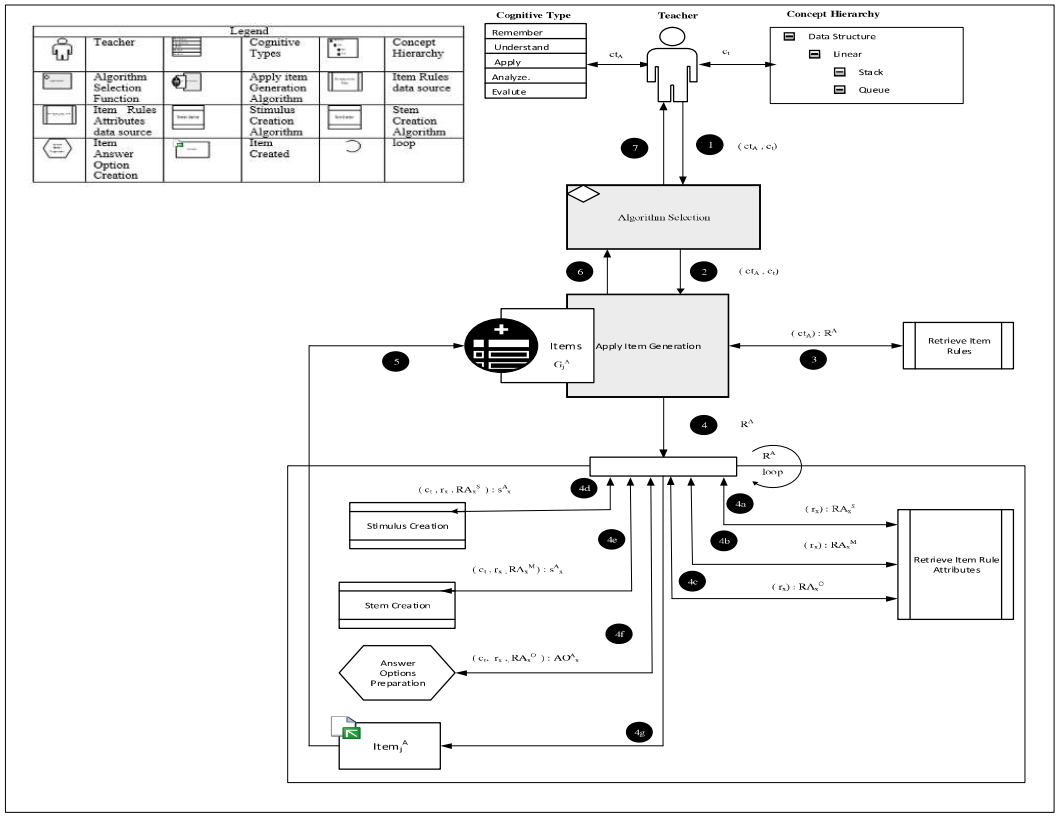


Fig. 5. Apply item algorithm workflow

(4a) Request for stimulus rule attributes (RA_x^S) passing the current rule (r_x) as parameter

(4b) Request stem rule attributes (RA_x^M) passing the current rule (r_x) as parameter

(4c) Request answer options rule attributes (RA_x^O) passing the current rule (r_x) as parameter

(4d) Ask "Stimulus Creation" supporting algorithm to create the item stimulus (s^A_x) by passing cognitive type c_t,, a rule r_x , and the retrieved stimulus rule attributes set RA_x^S.

(4e) Execute "Stem Creation Algorithm" to create the item stem (m^A_x) passing c_t, current rule (r_x) and RA_x^M.

(4f) Call "Answer Options Preparation Algorithm" to prepare answer options and their key (AO^A_x) passing c_t , current rule (r_x), RA_x^O.

(4g) Create an item G with the outputs of Steps 4d to 4f, {s^A_x, m^A_x, AO^A_x} and add to the item generated set I as shown in Step 5.

5. In the end the "Apply Item Generation" algorithm passes back the generated item set G to the caller (as Step 6 shows) and finally to the teacher as Step 7 shows.

4. Algorithm Design

This sections discusses the major and minor algorithms that work together to generate the items. When "Algorithm Selection" function as shown fig.2 it is called, first item generated set G_j and temporary item set Q_j are initialized to empty; then it chooses proper major algorithm base on the cognitive type the teacher selected (at Line #2 to #9 show) . .For instance if the algorithm receives apply cognitive type ct_A then it will invoke "Apply Item Generation" algorithm (as in Line # 4) passing c_t, ct as parameters.

ALGORITHM 1: Algorithm Selection
Input: set of cognitive type. $CT_j = \{ct_{j1}, ct_{j1}, \ldots, ct_{jn}\}$ and $ct \subset CT_j$
c_t , concept node
Output: set of item generated $G_j = \{g_{j1}, g_{j2}, \ldots, g_{jn}\}$
Local : $Q_j = \{q_{j1}, q_{j2}, \ldots q_{jn}\}$, temporary set of item generated
1: $G_j \leftarrow \{ \emptyset \}$, $Q_j \leftarrow \{ \emptyset \}$
2 : case cognitive type (ct) of
3 : Apply:
4 : $Q_j \leftarrow$Apply item Generation (c_t, ct)
5 : Analyze:
6 : $Q_j \leftarrow$ Analyze Item Generation (ct, ct)
7 : Evaluate:
8: $Q_j \leftarrow$ Evaluate Item Generation(ct, ct)
9: end Case
10: $G_j \leftarrow G_j \cup Q_j$

Fig. 2. Algorithm Selection Function

The "Apply Item Generation" shown in Fig.2 generates Apply cognitive items. When the algorithm is called the items generated variable G_j^A is first initialized to empty set (at Line # 1); then it retrieves Apply items rules R^A passing ct_A as parameter (at Line # 2). From (line # 3- #11) R^A is enumerated selecting rule attributes for item stimulus RA_x^S, stem RA_x^M and answer options RA_x^O respectively passing the current rule r_x .

ALGORITHM 2 : Apply Item Generation

Input : c_t , a concept node selected ; ct_A, an apply cognitive type

Output : G_j^A is subset of G_j, set of apply item generated

1 : $G_j^A \leftarrow \{ \emptyset \}$
2 : $R^A \leftarrow$ Retrieve Apply Item Rules by ct_A
3 : for each item rule r_x in R^A
4 : $RA_x^S \leftarrow$ Select Rule Attributes for item stimulus (r_x)
5 : $RA_x^M \leftarrow$ Select Rule Attributes for item stem (r_x)
6 : $RA_x^O \leftarrow$ Select Rule Attributes for item Answer Options (r_x)
7 : $s^A_x \leftarrow$ Stimulus Creation (c_t , r_x , RA_x^S)
8 : $m^A_x \leftarrow$ Stem Creation (c_t , r_x , RA_x^M)
9 : $AO_x \leftarrow$ Answer Options Preparation (r_x, c_t ,RA_x^O)
10: $G_j^A \leftarrow G_j^A \cup \{s^A_x , m^A_x , AO_x \}$
11: end for

Fig. 3. Apply item generation algorithm

In Line # 7 "Stimulus Creation" algorithm is called passing concept node c_t selected, rule (r_x), and RA_x^S which then returns formatted item stimulus (s^A_x); Then "Stem Creation" algorithm is called with rule (r_x), which then returns formatted stem (m^A_x) (at Line # 8). At Line # 9, "Answer Options Preparation" algorithm is asked to prepare the answer options accepting r_x, c_t , RA_x^O as parameters and returns set of answer options AO_x . The stimulus, stem, and answer options created from Line # 7, # 8, # 9 respectively is used to create item $\{s^A_x , m^A_x , AO_x \}$and added to item generated set G_j^A. After all the iteration of the rules is completed i.e. from Line # 3 to # 11 the items as shown in table .1 will be generated.

ALGORITHM 5: Stimulus Creation

Input : c_t , concept node selected
Item rule = $r_x \in R^A$.
RA_x^S , set of apply item rule attributes for stimulus

Output : s^A_x , item stimulus created
Local : $SV_j = \{ s_{vj1} , s_{vj2} , \ldots , s_{vjn} \}$

1: $SV_j \leftarrow \{ \emptyset \}$
2: for each rule attribute r_{xa} in RA_x^S
3: case r_{ax}^{source} of
4: supporting:
5: $SP_j \leftarrow$ Retrieve random supporting attribute value pair passing r_{ax}^{name}
6: $SV_j \leftarrow SV_j \cup SP_j$
7: concept schema:
8: $CVj \leftarrow$ Retrieve attribute value pair from concept schema passing c_t c_t , ra,r_{xa}
9: $SV_j \leftarrow SV_j \cup CVj$
10: concept node name:

```
11:        c_n ← Retrieve concept node name passing c_t
12:        SV_j ← SV ∪ { r_ax^name , c_n }
13:      end case
14:    end for
15:    TS^A ← Read stimulus template from item rule r_x
16:    s^A_x ← Format stimulus template passing TS^A and SV_j
```

Fig. 4. Stimulus Creation algorithm

When the algorithm in Fig 4 "Stimulus Creation" is called with concept node selected $c_{t,}$ element of stimulus rule $r_x \in R^A$, item rule attribute RA_x. First, the algorithm initialize stimulus attribute value pair variable SV_j to empty set (at Line #1): then loop through RA_x^S (from Line #2 - # 14) choosing the attribute source value r_{ax}^{source} of rule attribute object (.i.e. r_{xa}). For supporting (Line #4) it retrieves random supporting attribute value pair for the stimulus (SP_j) (at Line #5) and added to SV_j If the case is concept schema (Line #7) it retrieves attribute value pair CVj (Line #8) and add it to SV_j (Line #9) from concept schema passing c_t if the case is concept node name (at Line # 10) the algorithm retrieves concept name with c_t then create { r_{ax}^{name} c_n }and add it to SV_j . Lastly, (at line #15). it reads the apply item stimulus template TS^A from the item rule r_x and format the stimulus template with SV_j (at Line # 16)

5. Evaluation Plan

The research team intends to invite secondary and undergraduate school to have hands on experience with Online Test System. The teachers will be given small task to (co-)create or their own knowledge maps or import from other teachers' creations. Then they need to associate the knowledge maps to one of their courses followed by generating items for different cognitive levels and creating test for their courses.

After teachers have used the system they will be given a questionnaire which has three parts. Part one is general information which collects their demographical information like gender, subject teaching and academic role. The second part has 43 5-point Likert scale questions [9] for gathering their perceptions toward the usability of the systems and the item generation feature. The third part consists of 40 items generated by the algorithms for higher and lower cognitive levels and the research team asks teachers to tell researchers which cognitive level each item may belong to. With the data collected, the research team can verify the usability of the system and the item generation feature; moreover, the accuracy and effectiveness of the item generation algorithms can be assessed.

6. Conclusion

Automatic item generation algorithms have been designed and developed for existing Online Test System research. The algorithms enhance the system with item generation feature and make it capable of generating items for both higher and lower cognitive levels. The details of the major and minor algorithms have been presented and explained. The research team would like to collaborate with teachers and schools to test the usability of the system as well as obtaining users perceptions and identify the cognitive level the items generated by the system belongs to.

References

[1] Alves, C. B., Gierl, M. J., & Hollis, L.: Using automated item generation to promote principled test design and development. In: the annual meeting of the American Educational Research Association. (2010) http://www.crame.ualberta.ca/files/AERA%202010%20Denver%20Task%20Model%20AIG.pdf

[2] Chang, M., Kuo, R., Chen, S., Liu, T., & Heh, J.: Developing True/False Test Sheet Generating System with Diagnosing Basic Cognitive Ability. In World Conference on Educational Media and Technology, pp. 5740-5748, Vienna, Austria, June 30-July 4, 2008. (2008)

[3] Chang, M., & Kuo, R.: Elementary Level Botanical Item Generation. Learning Technology. Newsletter, 11(3), 7-9 (2009)

[4] Gierl, M. J., & Lai, H.: Instructional Topics in Educational Measurement (ITEMS) Module: Using Automated Processes to Generate Test Items. Educational Measurement: Issues and Practice, 32(3), 36-50 (2013)

[5] Gierl, M. J., Lai, H., Hogan, J. B., & Matovinovic, D.: A Method for Generating Test Items that are aligned to the Common Core State Standards. Journal of Applied Testing Technology, 16(1), 1-18. (2015)

[6] Gütl, C., Lankmayr, K., Weinhofer, J., & Höfler, M.: Enhanced Automatic Question Creator – EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern eEducation. The Electronic Journal of e-Learning, 9(1), 23-38. (2011)

[7] Hsu, C.-K., Chang, J.-C., Chang, M., Jehng, J.-C & Heh, J.-S.: An Approach for Automatic Learning and Inference by Knowledge Map. In the International Conference on Computers in Education (ICCE 2002), pp. 957-958, Auckland, New Zealand, December 3-6, 2002. (2002)

[8] Krathwohl, D. R.: A Revision of Bloom's Taxonomy: An Overview. Theory Into Practice, 41(4), 212-218. (2002)

[9] Likert, R.: A technique for the measurement of attitudes. Archives of Psychology, 22(140), 5-55. (1932)

[10] Schmeiser, C. B., & Welch, C. J.: Test development. In R. L. Brennan (Ed.), Educational measurement (4th Ed.) Westport, CT: Praeger Publishers. (2006).

[11] Stanescu, L., Spahiu, C. S., & Ion, A.: Question generation for learning evaluation. In the International Multi conference on Computer Science and Information Technology (IMCSIT 2008), pp. 509-513, Wisla, Poland, October 20-22, 2008. (2008)