A Question Generator for an Online Tutoring System

Jungsoon Yoo       Sung Yoo  Stefan Rusek
csyoojp@mtsu.edu  cssung@mtsu.edu  swr2c@mtsu.edu
Computer Science Department
Middle Tennessee State University, Murfreesboro, Tennessee, USA

Abstract: It is often necessary for a teacher to prepare a test over information for which students have been previously quizzed or to administer multiple versions of the same test. When this situation occurs, teachers are required to spend extra time creating equivalent questions. The major difficulty with this problem lies in efficiently creating sets of equivalent questions. To solve this problem of efficiency, instead of creating the question set, the teacher may create a template of each question. Once a template of a question is created for each question in a test, multiple equivalent tests can be generated as necessary. This paper describes a system that aids teachers in creating question templates and in generating equivalent questions for an online tutoring system. The system provides a user friendly question template editor with a GUI and the question generation module. The system employs a robust language so that a generalized problem description can be represented.

1. Introduction

Teachers spend a significant amount of time creating a good test set. However, it is often necessary for a teacher to prepare a test over information for which students have been previously quizzed or to administer multiple versions of the same test. When this situation occurs, teachers are required to spend extra time creating equivalent questions. This task is not only time consuming but is also a very tedious non-productive task.

When the teacher wants to make multiple question sets over the same material, the main problem lies in efficiently creating a set of equivalent questions that addresses the same concepts. This paper proposes the creation of a description of a question, called a template, from which equivalent questions can be generated automatically as necessary. We have developed a system, called the question generator that can be used as an aid to teachers in creating a template for a question and in generating equivalent questions. The question generator has been successfully used for an online tutoring system for CS I and II laboratories (Yoo et al. 2004).

In the following sections, the system design and the question template is described. Also discussed are the features of the system along with the corresponding issues addressed. A sample run of the system with illustrations is presented. The paper concludes with related works, discussions, and an evaluation of the system.

2. System Design

The goal of the question generator is to provide a tool that aids teachers in creating multiple question sets. The system was developed to be used for an online tutoring system. The design goals of the system are:

- The system should provide a template editor with a convenient graphical user interface (GUI).
- The system should support a language that can be used in a question template which represents a set of questions that covers an identical concept.
- The system should provide a question generation module that can produce a set of questions from a template on demand.
- The components of the system should be modular.

A question is designed to effectively identify the student’s level of understanding for a certain concept. However, there are many questions which are conceptually identical but cosmetically different. A question template can be used to represent a generalized form of a set of questions which are conceptually identical. In addition to the generalized form of the question itself, a question template stores other information regarding the set of questions such as answers, topics, the level of difficulty, and any additional explanation to help students understand the
A teacher creates a template using the (question) template editor provided by the question generator. The question generator uses the template to create similar questions that cover the identical concept. The details of a question template will be discussed in the following section.

Figure 1 shows the architecture of the system. The template editor interfaces with the concept base as well as a template file if it exists. Once a template is created, questions may be previewed for any corrections or generated for future use. The concept base contains concepts to be covered in a test or a lab. The concepts are organized in a rule base so that the dependencies and relationships between sub-concepts are represented. Each template created using the template editor is associated with a concept(s) in the concept base. Associating the concept base with question templates will help teachers maintain the test bank.

Once a template is completed, it can be exported to a file or a number of questions can be generated using the (question) generation module. Even though, the question generator can be completely modular, it was designed as a component of an online tutoring system. The question generator is built using Microsoft C# .NET, while the rest of the tutoring system is built on a Linux Platform. Among the components of the question generator, the generation module needs to be interfaced with the tutoring system to provide questions as needed. Mono (www.mono-project.com) was used to interface the generation module with the rest of the tutoring system.

3. The Template Editor

The system provides a template editor with a graphical user interface which has the following capabilities: create and/or save a template using the template editor, open an existing template and make any changes to the template, preview a question to examine how the question will be displayed online, provide a convenient GUI to edit attributes of a template such as answers, and provide convenient editing features such as copy and paste. Figure 2(a) shows a snapshot of the template editor that provides an interface to a user (teacher) to form a question template. A question template represents a description of a question or a set of questions. To represent a description of a question, a template consists of the following attributes:

- Question body: contains the problem statement(s).
- Answer: contains the solution or the expression of the solution to the problem.
- Wrong answers: contains the wrong choices for a multiple choice question.
- Explanation: contains the explanation of the solution which may help students understand the concept.
- Variables: contains the variables used to represent the generalization in the template.
- Script: contains user-defined functions.
- Topics: contains the concept and sub-concept(s) that the question is addressing.
- Level: contains the level of the question difficulty.
- Question Type: contains the type of the question (multiple choice, T/F, or short answer).
In Figure 2(a), the text area in the bottom half is used to enter the question body, the top text area is used to edit the selected tab; in this case the General tab is selected. The top text area has associating tabs that correspond with each attribute except the General tab which associates with the topics, level, and question type attributes. For convenience, the editor provides three additional buttons (add, edit, and delete) for the Variables and Wrong Answers tabs. The template editor is equipped with a menu bar and shortcut keys as in a typical Microsoft Windows text editing application. In addition, it also contains an intelliSense to provide easy access to variables and syntax highlighting for convenient editing. The template editor provides a menu bar which contains file, edit, and help menus. The file menu contains new, open, save, save as, load from Web, save to Web, generate preview, generate question file, and exit. The edit menu contains undo (Ctrl+Z), redo (Ctrl+Y), cut (Ctrl+X), copy (Ctrl+C), paste (Ctrl+V), delete, select all (Ctrl+A), find and replace (Ctrl+H), and goto (Ctrl+G).

4. Representation of a Question Template

We have chosen to use XML to represent a template with a subset of regular expressions in Perl (Wall et al. 2000). By using XML, it is convenient to interface the template with an online tutoring system that contains a database, PHP scripts in the server, and the HTML pages.

**Tags:** Each of the attributes in a template is represented as a top level XML element. A tag was created for each attribute used in a template. Some of the attributes such as the `<wrong-answers>` contains sub-elements. For example, the `<wrong-answers>` tag has `<choice>` sub-elements which are wrong answers to the question. There is no limit to the number of wrong answers that may be entered in a template, however, a maximum of five will be randomly selected for a generated question.

**Variables:** A question template represents a generalized form of a set of questions which are conceptually identical. A template uses variables to represent the generalization of a concept or a set of values. The type of a variable can be integer, extended integer, float, extended float, or string. The values of the extended variables are guaranteed to be unique in a template. A user has to assign the name and the type of the variable being created. If a user does not specify the range of a numeric variable, the default range will be used. However, a variable of the string type should have a range or a set defined by the user. The value of a variable will be randomly selected at question generation time or at preview time.

**Expression:** Regular expressions in Perl are used to specify the range or a set of values for a variable. The variable definition dialog has a preview capability to check the validity of the regular expression. An arithmetic expression...
can be used in a template. The evaluation of an expression is performed when the actual question is generated on demand.

**User friendly editing tools:** The format of XML is not very friendly to human readers. For example in XML, &lt; and &gt; are used instead of ‘<’ and ‘>’, and &lt;item value="$loop" /&gt; is used to represent the variable “loop”. To provide convenient editing, the template editor allows a user to use @[, <, and >. For example, instead of entering &lt;item value="$loop" /&gt;, a user may enter @[$loop] manually or using the intelliSense list. At any point in editing, the intelliSense list displays the current set of the defined variable list from which a variable can be selected and added to the template. The template editor will convert the expression into the XML format. Thus, if a user enters @[$loop] > 0, it will be converted to &lt;item value="$loop" /&gt; 0 by the editor. The system also provides a scripting capability which will be explained next.

**Scripting:** There are many questions that cannot be converted to a useful template without a scripting capability. For example, if a teacher wants to create a template from the following question:

*Given the following array declaration, assign the value 45 in the last element of the array “score”.*

```c
float      score [100];
```

it is necessary to represent the dependency between the answer and the array size. Typically, the array size is represented as a variable in a template to generalize the problem. The teacher may add a script that represents the relationship between the size of the array and the corresponding index in the solution of the problem.

The question generator allows a user to provide C# functions to support scripting. Figure 2(b) shows the partial view of the script provided for the above problem and the problem body that uses the script. The template editor provides @[script:] syntax for a convenient function call. For example, the script call @[$name][@[script::nVal($nth, $number*5)]] in the answer field generates “score[99]” as the correct answer.

**Interface for the concept base:** Figure 3 shows a screenshot of the rule base browser for concepts used in the template editor. Each template has an assigned topic(s) from the rule base of concepts. This can be done conveniently using the rule base browser in the template editor. The rule base browser displays a TreeView of the concept base in the bottom window. When a user selects a node (topic), the related rule(s) will be displayed in the top window. When the user is done, the selected topics will be inserted into the topics attribute of the template. As shown in the figure, the browser contains editing capabilities such as inserting a child node, deleting a node, and creating an empty concept base. It also allows a user to import and export a concept base.

**Previewer:** Figure 4 shows a question generated by the question generator using the template shown in Figure 2(a). As shown in Figure 4, the variable “loop” is substituted with ‘b’ and the variable “loopinit” is replaced with 6.
These replacements are done in all attributes of the template. These values are randomly selected from the specified range each time a question is generated from the template.

5. Related Works and Discussion

There are many popular course management systems such as WebCT (www.webCT.com) and Blackboard (www.blackboard.com), which support test bank management and provide a question editor. There also are a few systems (www.alivetek.com, web.uvic.ca/hrd/halfbaked, www.respondus.com) that provide a user friendly question editor that generates a test bank for popular course management systems such as WebCT and Blackboard. For example, the Respondus system (www.respondus.com) provides a question editor that makes it easier to generate matching questions than WebCT, and which can be imported to WebCT for test bank management.

The Hot Potatoes (web.uvic.ca/hrd/halfbaked) authoring suite provides a question editor in which a style sheet can be altered by a teacher for a customized presentation. However, this system only generates a single question. WebCT (www.webCT.com) provides a question type called “calculated question” which is similar to a numeric variable type. WebCT stores a fixed number of questions generated from a calculated question in a test bank, then questions are selected at random from the test bank. On the other hand, the system discussed here is designed to store each template in a database with associated topics and generate questions on demand.

Our system is inspired by Krishna and Kumar’s system which provides syntax for a generalization of a variable in a C++ program and for the range of numeric values (Krishna & Kumar 2001, Kumar 2000). However, their system does not provide a template editor and does not support regular expressions or scripting capabilities. Furthermore, the question generator described here associates each question with topics, a level, more complex multiple choice support, scripting, and a rule base of concepts to associate questions with related topics covered in class.

The question generator uses a robust template representation language. It is equipped with a template editor which has a user friendly graphical user interface, and contains a portable question generation module. However, there is trade-off for using the system. The overhead of using the system is the time for generating a template for a question. The obvious advantage of the system is its ability to generate many if not infinite number of questions from a template. These questions can be used to reinforce the question’s concept(s) to students. It is anticipated that the savings will out weigh the overhead.
Figure 4 shows the online lab on array concepts that was developed using the question generator. First, a rule base of the array concepts was created. For each of the most specific concepts in the rule base hierarchy, one or more templates were created and stored in a database so that a template could be retrieved using a keyword (topic) and used for actual question generation as needed. A student could solve as many question as he/she wished for each template. The result of the student survey shows that 83% strongly agree with “solving similar questions multiple times helped understanding the concepts,” while 74% liked “solving the similar questions multiple times.” Most of those who did not like to solve similar questions were those who are expecting high grades for the course. The survey shows that this approach helps weaker students more than advanced students. Students were also tested with questions from the array exercises on an exam; most of them performed better on array questions than other topics, and the performance of weaker students improved more than others.

6. Conclusions

A web-based tutoring system has been developed that contains a question tutor, a program tutor, a course management component, and a question generator. In general, learning can be improved by drilling on problem solving. This also applies to an online tutoring system. However, creating multiple questions over the same concept is a tedious, time-consuming task even with a good authoring system. A question generator may help to alleviate this problem.

This paper describes a modular question generator that has a user friendly template editor with a GUI. The system employs a robust representation language so that a generalized problem description can be represented. An experiment shows that students benefit from the exercises using the question generator. Most students felt that the system was usable and helped their understanding. The study shows that the system tends to help weak students more effectively than strong students.

7. References


8. Acknowledgments

This research was supported by the NSF grant DUE-0311367.