Geometry Question Generator: Question and Solution Generation, Validation and User Evaluation

Rahul Singhal and Martin Henz

School of Computing
National University of Singapore (NUS)
Singapore
rahulsinghal@nus.edu.sg
henz@soc.nus.edu.sg

Abstract. Current massively open online courses (MOOCs) are providing several technical challenges for educators. One of these challenges is automated generation of questions, along with the solutions, in order to deal with a large number of students. Geometry is an important part of the high school curriculum. Hence, in this paper, we have focused on the high school geometry domain. We have proposed a framework that combines a combinatorial approach, pattern matching and automated deduction to generate and solve geometry problems for high school mathematics. The system would help teachers to quickly generate large numbers of questions on a geometry topic and may also support the setting of standardized tests such as PSLE, GMAT and SAT.
Our novel methodology uses (i) a combinatorial approach for generating geometric figures from the user input, (ii) a pattern matching approach for generating questions, and (iii) automated deduction to generate new questions and solutions. By combining these methods, we are able to generate questions involving finding or proving relationships between geometric objects based on a specification of the geometry objects, concepts and theorems to be covered by the questions. We propose several algorithms to avoid generation of repeated questions and to avoid questions having redundant information, which increases the effectiveness of our system. We have tested our generated questions on an existing geometry question solving software JGEX, verifying the validity of the generated questions. A survey with the real users such as high school teachers and students on generated questions and solutions shows that our system is effective and useful.

Keywords: Automated Deduction, Graph-based Knowledge Representation, Pattern matching, High school geometry, Axiomatic approach, Constraint handling rules (CHR)

1 Introduction

Geometry, the study of space and spatial relationships, is an important and essential branch of the mathematics curriculum at all grade levels. The study

of geometry develops logical reasoning and deductive thinking, which helps us expand both mentally and mathematically. Children who develop a strong sense of spatial relationships and who master the concepts and language of geometry are better prepared to learn number and measurement ideas, as well as other advanced mathematical topics [13].

Euclidean geometry is a branch of mathematics which deals with the study of plane and solid figures on the basis of axioms and theorems employed by the Greek mathematician Euclid. It is important to understand Euclidean geometry when studying a course because geometry does not follow any set pattern. In Euclidean geometry, one can only learn the axioms and results proven from these axioms. The student must apply these axioms with no set pattern or list of steps for solving such questions. Therefore, a question may have (possibly infinitely) many solutions. To practice the required problem solving skills, students require a large number of different types of geometry questions on various concepts. Generally, textbooks and online sites provide a limited predefined number of questions for each topic. Once practiced, these questions lose their purpose of enhancing student thinking. The tedious and error-prone task of generating high-quality questions challenges the resources of teachers. Hence, there is a need for software which assists both teachers and students to generate geometry questions and solutions.

The software can also act as a personalized instructor. It can generate questions that cover the required topic and meet the required level of student proficiency. Apart from helping users, the framework of generating questions has scientific contributions to other research areas, such as Intelligent tutor systems (ITS) and Massive Online Open Courses (MOOCs).

Various research has been performed in automated deduction of theorems at high school level in the geometry domain, although none with the goal of automatic question generation. Instead, they mainly demand users to generate the question with the help of tools. In addition, they mainly focus on solving and assessment of the questions. Our survey shows that the currently available geometry systems, such as JGEX, Geogebra, Cinderella and Sketchpad, are not able to automatically generate questions of user specified geometry topics.

The aim of this paper is to develop a framework that can be used to generate geometry questions based on specific inputs, such as geometry objects and theorems to be involved in their solution. For a given set of geometry objects, the algorithm can generate a large set of questions along with their solutions. The solutions will involve user desired theorems directly or indirectly. Hence the framework can generate questions to test the theorem on various geometry objects and concepts.

The main contributions of this paper are as follows:

1. Our geometry question generator combines the complementary strengths of a combinatorial approach, pattern matching and deductive reasoning. It can generate geometry questions which were not possible previously.

2. A knowledge representation is described for geometry objects and predefined theorems. This representation helps in applying theorem information within the generated questions.
3. The system is made more effective by including algorithms for avoidinig repreated questions and handling redundant data.
4. A substantial evaluation is provided that demonstrates the effectiveness of our generator. It can generate various categories of the questions covered in the textbooks and questions asked in SAT and GMAT. A survey is done with the real users (teachers and students) which shows the usefulness of our system.

2 Related Work

In this section, we provide a general review of related works. Computational research in the geometry domain started in the 19th century. However, lack of question generating in geometry in the literature required a principled ab-initio approach in our work. Researchers mainly focused on proving geometry theorems. We are mainly interested in the non-algebraic methods. Non-algebraic methods for automatic discovery and proof of geometry theorems can be further divided into three approaches: coordinate-free methods, formal logic methods and search methods.

2.1 Coordinate-Free Methods

These methods are applicable to constructive geometry statements of equality. Various methods have been proposed under this category, such as the area method [12, 3], the full-angle method [2, 18], the complex number method, the vector method for Euclid plane geometry [2], the volume method for Euclidean solid geometry [1] and the argument method for non-Euclidean geometry [1]. The area method was further improved and developed into a computerized algorithm [2, 1]. These methods can only be applied in constructive geometry, which is outside the scope of this paper.

2.2 Formal Logic Methods

Theorems in Tarski's geometry were proven using Interactive Theorem Prover (ITP) [10], albeit limited to several trivial theorems. ITP is an interactive theorem prover based on the resolution principle, which generates resolution style proofs that resemble traditional proofs. In 1989, Quaife continued the work of McCharen with Otter [14]. Otter is an automated theorem prover based on resolution. A series of tactics such as Hyper-resolution, UR-resolution, paramodulation, support set and clause weight can improve the resolution efficiency. In 2003, Meikle and Fleuriot developed Hilbert's geometry with the theorem prover Isabelle/Isar [11], an interactive and/or semi-automated theorem prover. The greatest disadvantage of formal logic methods is their low reasoning efficiency [9], caused by combinatorial explosion of their search space.

2.3 Search Methods

Fundamentally, the search method is used in a rule-based expert system, which includes a rule database, a fact database and a reasoning engine. The inference rules stored in the rule database include axioms, theorems, lemmas, formula, definitions and algebraic operation rules in geometry. Geometric facts stored in the fact database include geometry predicates such as angle bisector, equidistant points, parallel lines and perpendicular lines. The reasoning engine deduces new geometric facts by applying inference rules to the facts database. There are three ways of performing deduction search, namely forward chaining [19, 18], backward chaining [18], and bidirectional chaining [4].

In 1995, Chou, Gao, and Zhang described an integration of a deductive database into the search methods [2, 6, 6, 1]. The deductive database method can find a fixpoint for a given geometry diagram, containing all properties of the geometry diagrams that can be deduced using a fixed set of geometry rules. They effectively controlled the size of the facts database with structural deductive database techniques.

Each search method has different advantages and disadvantages. Forward chaining is always feasible, but does not have an explicit reasoning goal. Backward chaining has an explicit reasoning goal, but sometimes lacks feasibility. The bidirectional chaining method is feasible and has an explicit reasoning goal, but is hard to implement. We are using forward chaining in our framework as it is most suited for generating previously unknown quantities.

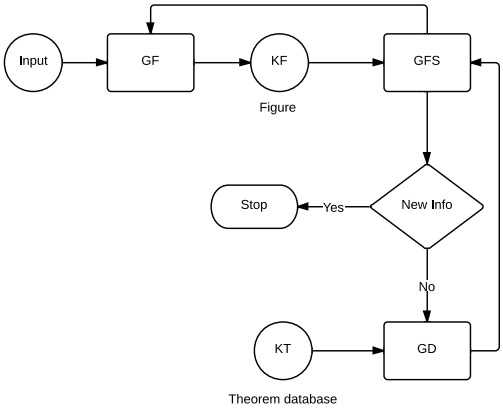
3 Geometry question specification

Mathematically, a geometry question Q generated by our system can be represented by a quintuple (Object O, Concept C, Theorem T, Relationship R, Query type qt) where:

- O $\in$ (lines, triangles, square, circle, ...)
- C $\in$ (perpendicular, parallel, midpoint, angle-bisector, circumcircle, ...)
- T $\in$ (Pythagorean theorem, similarity theorem, ...)
- R $\in$ (syntactic, quantitative)
- qt $\in$ (syntactic, quantitative)

In order to generate geometry questions, the user has to provide a set of geometry objects O such as triangles, squares, etc., and a set of concepts C which the user wants the generated question to cover. Optionally the user may select a set of theorems T to be tested by the question. The relationship R can be either syntactic such as perpendicular, parallel, etc., or quantitative such as the length of an object, the ratio of two quantities etc. The query type qt is the type of generated question that can be asked to find the hidden relationship which can be calculated from the given information.

Figure 1 Connection of the components and knowledge representations

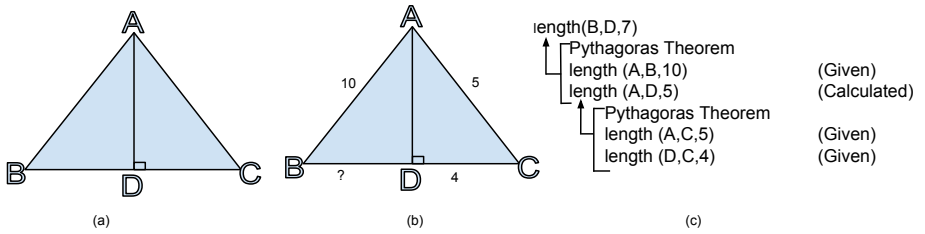


4 Framework

Our framework comprises of three major components along with the knowledge databases used for storing input, geometry figures and a set of predefined theorems. Figure 1 shows the connection of these components. The input consists of geometric objects, concepts and theorems selected by the user. The input is fed into the first component, *Generating Figure (GF)*. This component is used for generation of geometric figures from the input. Each figure constitutes a diagrammatic schema (DS) [8] and a set of unknown variables representing the relationship between geometric objects. The geometric figure is passed to the second component, *Generating Facts and Solutions (GFS)*. This component is used to find values of the unknown variables representing possible relationships to be asked by the generated question. GFS makes use of the predefined knowledge database of axioms. It results in the formation of a *configuration* (Cfg) containing known values for some relationships between its objects.

A question is considered suitable if it covers the essential information such as a new fact and a proper reasoning for the generated fact. Currently, the decision of suitability is taken manually by the user. If the suitability conditions for the generated configuration (Cfg) are not met then configuration is fed into the last component, *Generating data for the figure (GD)*. GD assigns values to unknown variables of relationships. Repeated processing by GD makes the questions generated from Cfg easier and easier, because the values assigned by GD appear as given facts in the generated questions. GD makes use of a predefined set of theorems and makes sure that the assignment results in successful generation of geometry questions. FC generated from this component is again passed to GFS component and this loop continues until a question is found which meets suitability condition.

Figure 2 (a) The figure is generated by using *triangle* and *perpendicular* as the geometry objects using the GF component. (b) The data is generated for (a) using the GD component. (c) The new fact and its derivation using the GFS component is generated from the figure shown in (b).



Along with these components, two knowledge databases are used in the framework, namely the Knowledge database for the generated figures *(KF)* and the knowledge database for predefined theorems *(KT)*.

4.1 Algorithm

Figure 2 shows the step by step execution of the algorithm. We select the following input in our example.

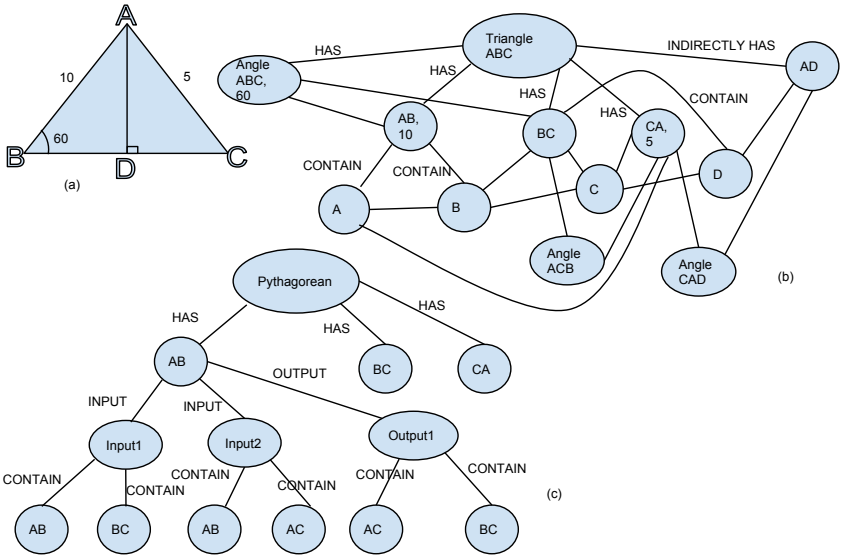
- Object: triangle and line segment
- Concept: perpendicular
- Theorem: Pythagorean Theorem

In the next subsections, we will describe each knowledge database in detail, followed by the three components and their interaction with these databases.

4.2 Knowledge Database for the Figure/ Figure Configuration (KF)

KF contains the question figure and configurations using a graph-based knowledge representation. The nodes of our graph represent the geometric objects. Two nodes can have multiple labeled arcs between them, representing multiple relationships. In addition, an arc can be bidirectional or unidirectional depending on the relationship between the nodes. Following the example of the previous section, Figure 3(a) shows the geometric figure: a triangle ABC is given and AD is perpendicular to BC. Figure 3b shows its partial representation in a graph format where we focus on the most important relations. Relationships such as "equal angle" and "equal length", which require multiple objects/nodes, are represented by nodes, e.g. , the relationship *"$\angle ABC$"* in Figure 3(b). In addition, nodes of our graph store the value of quantitative relationships. For example, the node named *"Angle ABC"* stores the value of an angle and is connected to the two nodes representing the sides AB and BC.

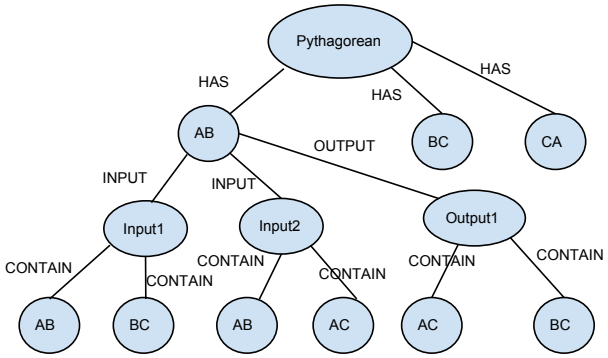
Figure 3 (a)The geometry figure and (b) Partially drawn knowledge representation of (a). (c) Representation used for generating data for the given figure.



4.3 Knowledge Database for Predefined Theorems (KT)

KT is a knowledge database which contains the predefined database of theorems and is used for assigning values to the unknown variables of configurations generated by the GF component. It uses a graph-based representation similar to KF. However the difference lies in the timing of generation. In order to use KT in the algorithm, some properties of theorems need to be known. Each theorem can be applied to a particular geometry configuration. Each theorem has certain inputs and outputs corresponding to inputs in the geometry configuration. Hence the usage of a theorem requires assigning the variables of input and/or output. In KT, the information of input and output for each theorem is stored along with the geometric configuration in which the theorem can be applied. Nodes represent objects whereas arcs represent the input and output relationships. Input and output are decided offline before the execution of the algorithm and later used for assigning unknown variables to get a useful question. To illustrate the process, the Pythagorean Theorem is taken as an example in Figure 3c. Given a triangle ABC, where $\angle ABC$ is $90°$, Figure 4 partially shows its usage of KT in generating data. The node representing the side AB has three arcs, two of which represent an input and one of which represents an output relationship. To use the side AB as an input, the length of the sides AB and BC or AC need to be assigned. On the other hand, to use side AB as an output, the length of sides AC and BC needs to be given. By giving this as input or output, we can be sure that the Pythagorean Theorem will give a consistent result.

Figure 4 Representation used for generating data for the given figure.



4.4 Generating figure configuration from the user input (GF)

This is the first step executed by the framework mentioned in the Section 4.1. This component generates a figure configuration through the combination of a predefined number of ways to combine geometric objects. Currently, we are focusing on triangles and line segments. Hence our algorithm includes combinations in which various triangles and lines can intersect.

Configuration generation involves the addition of geometry objects, concepts and optionally, user-desired theorems. The detailed algorithm is mentioned here [15].

4.5 Generating facts and solutions from the configuration (GFS)

This component is responsible for finding the values of unknown variables of the generated figure/configuration from the other components. This component acts as question generator and solver. The unknown variables whose values have been found represent the generated questions. The steps that leads to finding the unknown variables represent the solution. There can be many ways for finding the values of the unknown variables. In such cases, this component shows all solutions. For generating new facts, it uses a predefined database of theorems.

All theorems are represented in an axiomatic format. Forward chaining is used to generate new facts. The rules representation and detailed algorithm is mentioned here [15].

4.6 Removing the redundant data

Redundant data are values assigned to properties of the geometric objects which are not required and can be derived from the previously given data and predefined theorems. It may happen that redundant data is provided in the figure configuration.

Figure 5 Figure explaining redundant case

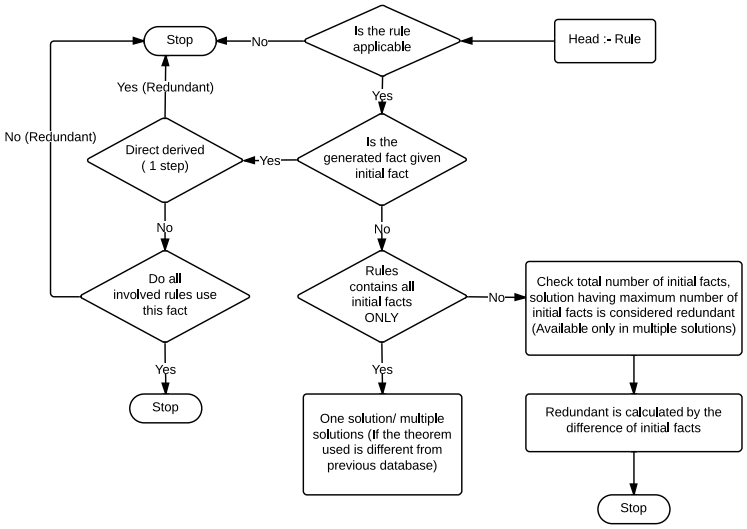


Figure 5 shows a flow diagram of an algorithm for detecting and removing redundant data. Figure 6 explains cases with the help of examples. Figure 6a shows an instance of new configuration without redundant data. In Figure 6b, the length w is redundant, since, by applying the similarity concept, it can be derived from the other two lengths, x and y. In addition, the rules used for generation of w do not previously involve w. However, y is not redundant data, as, the rules used for generating y implicitly requires y. The length AD can be calculated via two ways, Pythagorean Theorem and similarity theorem. Both rules used only initial facts. Hence, both are considered as multiple solutions. Furthermore, the length AB can be derived from two ways. However, in this case, w is consider as a redundant data as w can be generated with other initial facts. Figure 6c shows that w can be obtained from y using the trigonometry rules or vice versa. Hence, one of them can be considered as a redundant data and removed.

Figure 6 An example in which redundant data is provided in the given figure.

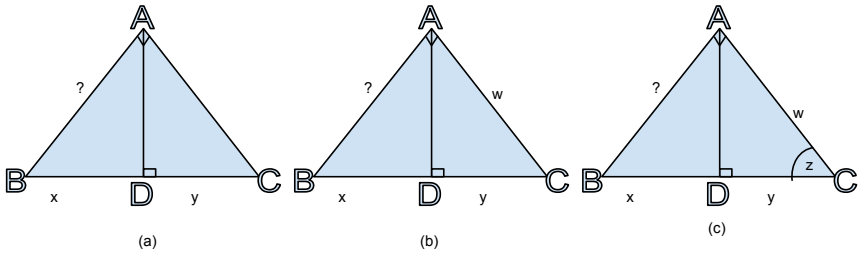
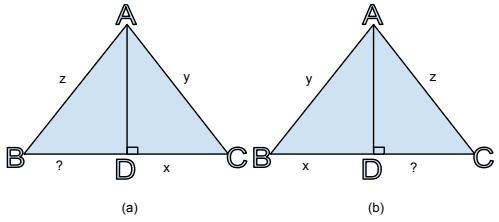


Figure 7 Figure explaining isomorphic solution



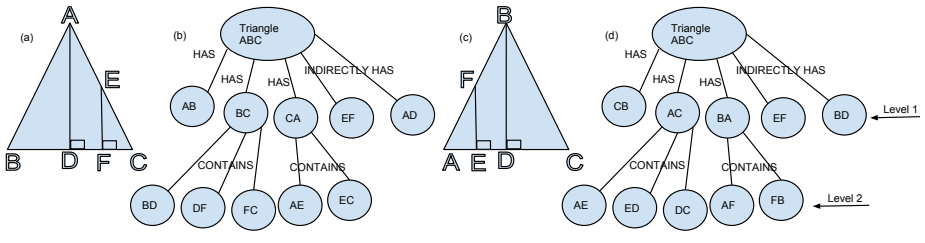
4.7 Generating New Configurations from an Existing Configuration (GD)

The basic algorithm involves performing pattern-matching of the generated configuration with the predefined set of rules to check which rule can be applied in the given configuration. It makes use of predefined set of rules represented in KR. Each theorem consists of input and output nodes for each variable. Depending on the selected variable and corresponding input and output nodes, the values of unknown relationships are assigned in the configuration. The detailed algorithm is given in [15].

5 Uniqueness

Uniqueness refers to generate non-repeating and non-isomorphic questions. Repetition refers to the same question. Isomorphic refers to the mirror-image of an exiting question (see Figure 7). Uniqueness plays an important role in terms of the effectiveness of our system. We proposed an algorithm to generate unique question different from all the previously generated questions. The main concept lies in storing all the previously generated configurations and performing efficient configuration matching to maintain uniqueness.

Figure 8 Figure explaining isomorphic configurations



Providing uniqueness in the question's figure/configuration

Uniqueness in the figure configuration involves two cases—the generation of the same figure and the generation of a mirror-image of the previously generated figure. Algorithm 1 describes an algorithm for providing uniqueness in the generated questions. The algorithm is based on checking the equality of configurations at each level. We explain the Algorithm 1 with the help of an example. Figure 8 shows two isomorphic configurations. They have the same number of nodes and relationship links and types at each level. In addition, the number of nodes attached to nodes below and above with equal number of relationship type are the same in both configurations. For example, at level 1, both the figure configuration have three nodes having "HAS" relationships and two nodes having "Indirectly has" relationships. Similarly, at level 2, three nodes are attached to one node at level 1 via the "Contain" relationship. In addition, two other nodes are attached to one node at level 1 via the "Contain" relationship. This equality is same through out each level. Hence, in this case, the algorithm will return true.

Data: A new geometric figure configuration, existing database of previously generated figure configurations
Result: True or false depending on the repetition of new figure configuration with the existing database

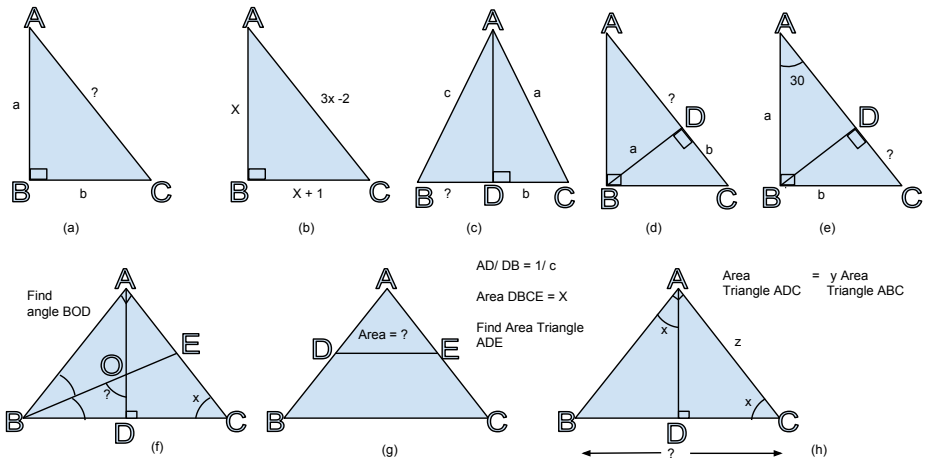
1. Check for the number of objects and concepts in the new figure configuration with the existing database. If not same, terminate with false else goto next step.
2. Check the number of nodes at each level to be same. If not same, terminate with false else goto next step.
3. Check the number of relationships at each level to be same. If not same, terminate with false else goto next step.
4. Check the number of nodes and relationships at each level to be same. If not same, terminate with false else goto next step.

Algorithm 1: Algorithm for removing repetition in geometry figure

Removing repetition in the question data

This algorithm is used only when the above algorithm declares the two figure configurations to be isomorphic. The algorithm is nearly same as Algorithm 1, except, for the additional checking which includes the values of known relationships at each level. Figure 7 shows an example where the two geometric questions would be considered equal. The figure configurations are considered equal following the Algorithm 1 and the values of known relationships are equal at each level. For example, AB of (a) with AC of (b) and AC of (a) with AB of (b).

Figure 9 (a-e) Generated questions based on "triangle", "perpendicular" and "Pythagorean Theorem" as input. Questions in figure(a-c)can be solved using Pythagorean Theorem only. However, questions of figure (d) requires similarity and figure (e) requires trigonometry for finding the unknown value. (f) uses angle-bisector theorem. (g-h) uses similarity theorem. The details can be seen here [15]



6 Implementation

Each component of our tool is implemented independently, using state-of-the-art languages, libraries and systems. C++ is used for performing calculations and Python is used for implementation of the algorithms used in GF and GD components. The algorithm in GFS component is implemented using Constraint Handling Rules (CHR) [5]. CHR are used for generating new facts from the axioms and the given facts. In our implementation, we use the CHR library provided by K.U.Leuven, on top of SWI-Prolog [16]. The theorems used in GFS component are manually converted in the format used by CHR library. For implementing knowledge representation KF and KT, the graph database Neo4j [17] is used. KF knowledge graph is used and modified by all the components and finally represents the question. Our knowledge databases such KT and predefined ways of intersection of geometric objects are manually generated and stored before the questions generation.

Experimental Results

The system can generate geometry questions using the framework described in Section 4. Currently, our knowledge database of objects contains line segments and triangles. In addition, we have a predefined set of more than 100 theorems. The generated questions cover various categories, e.g. similarity. Figure 9 shows

Users	Has seen Online/Textbook (in %)	Appropriateness of solution (in %)	Input used (in %)	Concept used (in %)	Quality (in %)
High school teachers	30	90	99	99	90
College level teachers	80	95	99	99	80
High school students	10	90	99	99	95
College level students	20	85	99	99	90

Table 1. Survey resultss after real user's testing

various questions generated by our system on selecting "*triangle*" as object, "*perpendicular, angle-bisector*" as concept and "*Pythagorean Theorem*" as a theorem to be covered. The generated questions are tested using the existing geometry solver tool JGEX [7]. For testing in JGEX, the figure configuration is drawn manually by the user and the system is asked to prove/find a certain relationship. The tool is able to prove/solve all the questions generated by our system. Comparing the solutions generated by JGEX with the solutions generated by our system, we found interesting differences that may stem from different representations of geometric knowledge and reasoning techniques and that deserve further investigation.

Evaluation

A pilot evaluation was conducted in order to estimate the feasibility of the whole approach and generated questions. We entered 50 rules in the system, which correspond to specific geometric theorems: Similarity in triangles, Pythagorean Theorem, and Basic Trigonometry formulas.

Different type of inputs are given on order to generate various different problems. Objects include right-angle, equilateral, isosceles and scalene triangles. The covered concepts include triangle perpendicular, median and angle-bisector. Some of the questions have multiple ways of finding the relationships. The prototype generated large number of problems, some of which were "isomorphic", i.e. identical from a pedagogical point of view. All problems were correct, as manually checked by the authors. Ten of the generated questions were selected for evaluation, with the aim of covering a wide range of concepts and objects. The small number of problems was due to evaluator's limited time availability.

The users of the survey were high school teachers, students and professionals involved in standardized exams like GMAT and SAT. The selected problems were given to several experienced High School Mathematics teachers from India, Singapore and US. Hundreds of students, both high school and college level, were involved in the survey.

For each problem, the assumptions, known relationships and unknown relationships to be proved/found as well as generated diagrams were given to the participants. Six survey questions were asked for each generated geometric question, covering aspects such as the quality and the appropriateness of the generated solutions. Table 1 shows the questions asked and the statistics of the

user's response. It can be seen that various teachers have different perception of the quality of the question to be given in the exam. Similarly the case with difficulty of the generated question.

The table shows that half of the users considers these questions as new. However, this depends on the domain knowledge and memory. Several questions were considered new by almost all users. In addition, most of the users are in consensus with the usefulness of the generated questions. Furthermore, most of the users have considered the generated solution of the questions appropriate for high school mathematics. The options for providing objects and concepts of generating these questions matches with those actually provided in the system. Overall, it can be seen that the system is able to fulfill the aim of generating questions quickly from the given input along with the appropriate answer.

Although the number of participants is very limited and the report of the above data is somewhat anecdotal, we see an agreement among teachers regarding both diversity and quality for testing. Although these results should by no means be generalized, they are hopeful initial indicators of the potential validity of the proposed measures for exercise selection, providing a useful basis for further justification and/or adjustment.

7 Conclusion

In this paper, we provide a framework for the automatic generation and solving of questions for high school mathematics, specifically in the geometry domain. Our system is able to quickly generate large numbers of questions on specific topics. Such a system will help teachers reduce the time and effort spent on the tedious and error-prone task of generating questions. Our work aims to develop an automated geometry question generation system that uses a deductive approach for finding the relations between mathematical concepts and for generating and proving these conjectures about concepts.

Future work can be carried in various directions. One of the major work would be generation of relevant questions via user's feedback. Other major work would be generating questions according to the required difficulty level. Another improvement would be the automation of process of knowledge addition in the system. Lastly, solutions readability could be improved.

Bibliography

[1] S. C. Chou and X. S. Gao. *Handbook of Automated Reasoning.* Elsevier and MIT Press, 2001.

[2] S. C. Chou, X. S. Gao, and J. Z. Zhang. A deductive database approach to automated geometry theorem proving and discovering. *J. Autom. Reason.*, 25(3):219–246, October 2000.

[3] S. C. Chou, X. S. Gao, and J. Z. Zhang. Area method and automated reasoning in affine geometries, 2011.

[4] H. Coelho and L. Pereira. Automated reasoning in geometry theorem proving with prolog. *Journal of Automated Reasoning*, 2(4):329–390, 1986.

[5] Thom Frühwirth and Frank Raiser, editors. *Constraint Handling Rules: Compilation, Execution, and Analysis.* Books On Demand, March 2011.

[6] X. S. Gao, C. C. Zhu, and Y. Huang. Building Dynamic Mathematical Models with Geometry Expert, I. Geometric Transformations, Functions and Plane Curves. In *Proceedings of the Third Asian Technology Conference in Mathematics*, 1998.

[7] Xiao-Shan Gao and Qiang Lin. MMP/Geometer—a software package for automated geometric reasoning. In Franz Winkler, editor, *Automated Deduction in Geometry*, volume 2930 of *Lecture Notes in Computer Science*, pages 44–66. Springer Berlin Heidelberg, 2004.

[8] Greeno, James G., Magone, and Seth Maria E. Chaiklin. Theory of constructions and set in problem solving. *Memory and Cognition*, 7:445–461, 1979.

[9] Jianguo Jiang and Jingzhong Zhang. A review and prospect of readable machine proofs for geometry theorems. *Journal of Systems Science and Complexity*, 25:802–820, 2012.

[10] J. D. McCharen, R. A. Overbeek, and L. A. Wos. Problems and experiments for and with automated theorem-proving programs. *IEEE Trans. Comput.*, 25(8):773–782, August 1976.

[11] L. Meikle and J. Fleuriot. Formalizing Hilbert's Grundlagen in Isabelle/Isar. In *Theorem Proving in Higher Order Logics*, 2003.

[12] J. Narboux. The area method: a recapitulation. *Journal of Automated Reasoning*, 48:489–532, 2010.

[13] National Council of Teachers of Mathematics. Curriculum and evaluation standards for school mathematics, 1989.

[14] A. Quaife. Automated development of Tarski's geometry. *Journal of Automated Reasoning*, 5:97–118, 1989.

[15] Singhal R., Henz M., , and McGee K. Automated generation of geometry questions for high school mathematics. In *Sixth International Conference on Computer Supported Education. Accepted for publication as full paper*, 2013.

[16] Tom Schrijvers and Bart Demoen. The K.U.Leuven CHR System: Implementation and application. In *First Workshop on Constraint Handling Rules*, 2004.

[17] C. Vicknair and M. Macias. *A Comparison of a Graph Database and a Relational Database*, 2010.

[18] S. Wilson and J. D. Fleuriot. Combining dynamic geometry, automated geometry theorem proving and diagrammatic proofs. In *ETAPS Satellite Workshop on User Unterfaces for Theorem Provers (UITP)*, 2005.

[19] J. Z. Zhang, X. S. Gao, and S. C. Chou. The geometric information search system by forward reasoning. *Chinese Journal of Computers*, 19(10):721727, 1996.