

國立交通大學

資訊科學與工程研究所



以分眾分類法為基礎建造國小數學應用問題

A Folksonomy-Based Approach to Constructing Elementary

School Arithmetic Word Problems

研 究 生：游士緯

指導教授：曾憲雄 博士

中 華 民 國 九 十 八 年 六 月

以分眾分類法為基礎建造國小數學應用問題

A Folksonomy-Based Approach to Constructing Elementary
School Arithmetic Word Problems

研究生：游士緯

Student : Shih-Wei You

指導教授：曾憲雄

Advisor : Dr. Shian-Shyong Tseng

國立交通大學
資訊科學與工程研究所
碩士論文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

In partial Fulfillment of the Requirements

for the Degree of

Master

In

Computer Science

June 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

以分眾分類法為基礎建造國小數學應用問題

研究生：游士緯

指導教授：曾憲雄博士

國立交通大學資訊學院
資訊科學與工程研究所

摘 要

讓學生解數學應用問題可以訓練學生的算數應用能力，然而，要建造新的數學應用問題需要耗費時間以及成本，因為要以合理、多變的問題情境來描述應用題的數學算式是很複雜的。本篇論文，我們以分眾分類法收集老師的知識，這些知識包含問題情境以及問題結構。我們提出了階層式數學應用問題框架式知識表達法來表示問題結構，因為框架的繼承性質適合表示數學應用問題的分類結構。我們提出的數學應用問題文法可促進維護數學應用問題以及問題的演化，包含自我演化以及合作式演化。自我演化改變問題的邏輯來產生不同的題目。合作式演化將兩個問題結構整合成一個新的問題結構。此外，框架中的規則可將不同的問題情境套用在問題結構上；以此大量產生數學應用問題。我們建構了以分眾分類法為基礎的題庫管理系統，來評估系統的使用滿意程度，實驗結果顯示，大多數的國小數學老師都很滿意我們的系統。

關鍵字：分眾分類法、框架式知識表達法、自動產生、前後文無關文法、
數學應用問題

A Folksonomy-Based Approach to Constructing Elementary School Arithmetic Word Problems

Student: Shih-Wei You

Advisor: Dr. Shian-Shyong Tseng

**Institute of Computer Science and Engineering
Nation Chiao Tung University**

Abstract

The arithmetic word problem enables students to apply their arithmetic skills to solve the problem in application level arithmetic of Bloom's taxonomy. However, it is costly and time consuming to construct new arithmetic word problems due to the complexity of combining the different arithmetic questions with the various and reasonable scenario. In this thesis, we aim to apply **Folksonomy-Based approach** to collecting teachers' knowledge consisting of **Question Scenario** and **Question Structure**. We proposed an **Arithmetic Word Problem Frame Hierarchy** knowledge representation to represent these question structure because inheritance property of frames is appropriate to represent the category structure of arithmetic word problems. The proposed **Arithmetic Word Problem Grammar** can facilitate the maintenance of arithmetic word problems and their evolutions including **Self-Evolution** and **Collaborative-Evolution**. Self-evolution changes the arithmetic logic to generate different problems. And collaborative-evolution integrates two question structures into a new question structures. Besides, the rules embedded in the frame can be used to apply different scenario to the question structure; thus a large amount of arithmetic word problems can be generated. We have also constructed a **Folksonomy-Based Item Bank management system (FIB)** to evaluate the satisfaction degree of FIB. The evaluation results show that most of elementary school teachers satisfy with our proposed system.

Keywords: Folksonomy, Frame-based knowledge representation, Automatic generation, Context-free grammar, Arithmetic word problem

致謝

首先誠摯的感謝指導教授：曾憲雄博士。曾教授在這兩年碩士生涯中，對我們的細心教導以及諄諄教誨，不管在學習研究方面抑或是做人處事，時常與我們分享他過去寶貴的經驗，也在我感到迷惘時給予當頭棒喝，真的非常感謝曾教授給予我們的耐心、用心及苦心，讓學生獲益匪淺，特別要在這對曾教授獻上十二萬分的感謝。同時也要非常感謝孫春在教授、葉耀明教授以及賀嘉生教授在學生口試時特別撥空來參加審查，也給予我非常多的建議及鼓勵，讓本篇論文更加豐富、更加有價值。

在研究生涯中，也非常感謝曲衍旭學長，謝謝您這段期間的指導及幫忙，學術研究上給予很多鼓勵和建議，在這邊也要深深的獻上我的感激！同時也感謝實驗室的所有學長姊們，尤其是林喚宇學長、李宗儒學長，感謝兩位學長對我的指導以及幫助，在工作繁忙的期間願意撥空指導我的論文並且給予相當多的建議，如果沒有兩位學長的幫忙，這篇論文一定不可能及時完成，在此讓我致上最大的感謝，非常感謝兩位鼎力相助。對於兩年同窗的好友：小雅、惠君、啟琿、祖淵，還有實驗室的學弟妹們，金龍、紹宜、嘉祥、嘉榕、國彰、杰峰，特別謝謝你們這段期間的互相鼓勵及支持，我才能堅持努力完成這篇碩士論文，謝謝你們。特別是小雅，我實驗室最好的朋友，這段期間內我們互相吐露心事以及互相支持鼓勵，因為有妳，我的碩士生活才更加的多采多姿，這段期間有任何得罪或者是不周全的地方，請容我在這邊跟妳致歉，這兩年內真的很感謝有妳。

最後要謝謝我最愛的家人以及身邊的好朋友們(威男、Win 兄、魚魚、其他的親朋好友)，在我最無助最失落時總有你們陪著我，不管身在何方，真的非常謝謝你們。

士緯

2009 年 7 月于新竹

Table of Contents

摘 要.....	ii
Abstract.....	iii
致謝.....	iv
Chapter 1 Introduction.....	1
Chapter 2 Related Works.....	4
2.1 Automatic Question Generation for Math Word Problems.....	4
2.2 Automatic Question Generation Approach for Program Learning	5
2.3 Automatic question generation approach for software design assessment.....	5
2.4 Dynamic test generation over ontology-based knowledge Representation.....	6
Chapter 3 Folksonomy-Based Arithmetic Word Problems Collection and Arithmetic Word Problem Analysis.....	8
3.1 Folksonomy-Based Problems Collection.....	8
3.2 Arithmetic Word Problem Analysis.....	10
3.3 Arithmetic Word Problem Frame Hierarchy.....	11
3.4 Spiral Model for Folksonomy-Based Problems Collection.....	13
3.5 Grammar and Question Structure Evolution.....	14
3.6 User Shared Content Management.....	31
Chapter 4 Folksonomy-Based Item Bank Management System.....	34
4.1 System Architecture	34
4.2 Question Collection Phase	36
4.3 Generation Problem Phase	39
Chapter 5 Implementation and Experiment.....	42
5.1 System Implementation.....	42
5.2 Experiment	43
5.3 Discussion	47
Chapter 6 Conclusions and Future Works.....	48
References.....	50

List of Definition

Definition 1: Arithmetic Word Problem Grammar..... 15



List of Examples

Example 1: Production rule of CP	17
Example 2: Production rule of BP	18
Example 3: Production rule of QSP	18
Example 4: User provide a perimeter problem Question Structure	19
Example 5: Production rule of QRP	19
Example 6: Production rule of SSP	20
Example 7: Mapping Scenario to Question Structure	22
Example 8: Applying scenario to QR^V	23
Example 9 : Self-Evolution	25
Example 10: Collaborative-Evolution	28
Example 11: Scenario Repository	32



List of Figures

Figure 1 Property of Arithmetic Word Problem	10
Figure 2 An arithmetic word problem frame hierarchy	11
Figure 3 Slot attributes of arithmetic word problem frame	12
Figure 4 Spiral model for folksonomy-based problems collection .	13
Figure 5 Scenario Slot of QR₁	21
Figure 6 Scenario mapping to Question Structure	22
Figure 7 Question requirement ontology and scenario slot ontology	24
Figure 8 Question requirement ontology and scenario slot ontology	26
Figure 9 Question requirement ontology and scenario slot ontology	31
Figure 10 Scenario Repository	33
Figure 11 System architecture of folksonomy-based item bank management system	34
Figure 12 Question Editor Interface	36
Figure 13 The Steps of editing new question requirement	38
Figure 14 Arithmetic problem category	39
Figure 15 Problem Generation Scope	40
Figure 16 Generation problem list	41
Figure 17 Question editor	42
Figure 18 Teacher select problem categories	43
Figure 19 Arithmetic word problems generated by FIB	43

CHAPTER 1 INTRODUCTION

In elementary school education, learning of arithmetic is very important to develop students' science and mathematics basic ability. According to the cognitive domain of Bloom's taxonomy[1], the knowledge of a domain consists of six cognitive levels: Knowledge, Comprehension, Application, Analysis, Synthesis, and Evaluation. The pure arithmetic questions can just train and evaluate students' arithmetic skills in knowledge and comprehension level. Thus, the arithmetic word problem with embedded arithmetic question and the corresponding real scenario can enable students to apply their arithmetic skills to solve the problem in application level.

In elementary school arithmetic skill learning, the drill and practice approach [2, 3] is usually used to make students master the arithmetic word problem solving, so a large amount of arithmetic word problems are needed to prevent students from just memorizing the answers of the used arithmetic word problems. However, it is costly and time consuming to construct new arithmetic word problems due to the complexity of combining the different arithmetic questions with the various and reasonable scenario. Therefore, how to assist teachers to construct more arithmetic word problem and reduce the constructing cost is an important issue.

Previous researches aim to use predefined question template to automatically generate different questions to reduce the question constructing cost, where the examined background knowledge, such as ontology or activity diagram, is required during the generation of the questions about the background knowledge. The various structures of arithmetic word problems, named Question Structure,

are used to examine various arithmetic skills. Previous researches do not consider the possibilities of generating various arithmetic word problems from predefined question templates; moreover, they do not take scenario effects into account. In other words, for a given question template, our goal is to generate different questions from different scenarios.

In this thesis, we aim to apply folksonomy-based approach to collecting teachers' knowledge about arithmetic word problem because a large amount of ideas of arithmetic question structures and question scenarios generated by different teachers can be shared and reused in order to reduce teachers' effort to construct arithmetic word problem. In order to facilitate managing and reusing the shared arithmetic word problem knowledge, frame-based knowledge representation is used to represent these problems because the questions of the same question structure are stereotyped and the inheritance property of frames is appropriate to represent the category structure of arithmetic word problems. Besides, the rules embedded in the frame can be used to apply different scenario to the question structure; thus a large amount of arithmetic word problems can be generated.

Teachers often construct various arithmetic word problems via transforming the arithmetic logic embedded in one arithmetic word problem. By our observation, production rules in context-free grammar are appropriate to describe this kind of transformation. Besides, applying context-free grammar can facilitate maintenance of arithmetic word problems and their evolutions including self-evolution and collaborative-evolution. Self-evolution changes the arithmetic

logic to generate different problems. And collaborative-evolution integrates two question structures into a new question structures. More specifically, the arithmetic word problem knowledge provided by multiple teachers can be fused and evolved to generate more questions by applying some specific production rules.

The expert-designed questionnaires are used to evaluate the degree of usage satisfaction of the proposed **Folksonomy-Based Item Bank management system** (FIB). The FIB is constructed based on the folksonomy-based approach we have mentioned above. 100 elementary school teachers are involved in evaluating FIB. The evaluation results show that most of elementary school teachers satisfy with FIB.

The remainder of the article is organized as follows. In Chapter 2, we will introduce previous researches related to automatic generation domain. In Chapter 3, we will introduce a folksonomy-based approach and the context-free grammar for describe and evolve the collected knowledge. The architecture and main functions of proposed Folksonomy-Based Item Bank will be described in Chapter 4. In Chapter 5, we will discuss the experiment result. Finally, Chapter 6 gives the conclusion and future work.

CHAPTER 2 RELATED WORKS

The previous researches aim to reduce question construction cost and generate a large amount of questions using automatic question generation approaches. We will introduce the related researches as follows.

2.1 AUTOMATIC QUESTION GENERATION FOR MATH WORD PROBLEMS

The Math Test Creation Assistant (Math TCA)[4, 5] which is a simple template-based natural language generation system was designed to provide a general tool for constructing questions. In Math TCA, question template consists of a series of fixed verbal template and variable slot, as illustrated below:

A car traveled ___miles in ___ hours. On average, how fast did the ___ move during this time period?

Values of variable slots in the question template can be filled by any numeric number. However, the verbal template cannot be modified unless system manager creates a new question template, and it is time-consuming.

Deane [6] built a natural language generation (NLG) system for automatic general verbal items, which can transform a concept to verbal content. They defined an arithmetic word problem consisting of generic concepts, for example, the “rate” word problem consists of generic concepts, *VEHICLE*, *MOVE*, *DISTANCE*, *RATE*, or *TIME*. The generic concept is the abstract expression of a verbal content, which can be described by corresponding verbal; for example, *VEHICLE* could be a car, a bus or a track. The NLG system can generate

arithmetic word problem which is described by different generic concepts. However, the generating methodology of NLG has not been mentioned; thus, it might not be able to be implemented.

2.2 AUTOMATIC QUESTION GENERATION APPROACH FOR PROGRAM LEARNING

Aravind K Krishna [7] proposed a problem generator named **problett**, which could be part of a courseware of teaching C++ programming language course to undergraduate students. Students can practice and learn about the concepts of operator precedence and associativity by solving the problems generated by problett. The problett generates expression as a binary tree, where terminal nodes contain operands and non-terminal nodes contain operators. The problett can generate a large amount of expressions automatically, which consists of operands and operators. The operands are generated using a random number generator, and the operators are generated randomly from the operator types, which are selected by the user. By using the problett, the student's performance of programming language is thus improved. However, the problett can only generate the pure arithmetic questions.

2.3 AUTOMATIC QUESTION GENERATION APPROACH FOR SOFTWARE DESIGN ASSESSMENT

An automatic generation of online assessments, which was proposed by Imran A. Zualkernan [8], is used to judge a software engineer's level of comprehension of artifacts created during software design. Activity diagrams are a

type of model commonly used in software design, which typically captures control flow in a situation. In this research, the question level in the generating assessments is based on Bloom's taxonomy, and a typical assessment consists of a number of questions from each level of understanding. The questions are all about the misunderstood patterns which are commonly occurring differences between the activity diagrams. This research uses the Hazard Operators (HAZOP)[9] to generate classed misunderstandings, which are multiple choice questions for software engineers in order to assess their understanding of activity model. This research proposed an online assessment to judge an engineer's level of understanding of a software design and reduce the cost to design the exams.

2.4 DYNAMIC TEST GENERATION OVER ONTOLOGY-BASED KNOWLEDGE REPRESENTATION

Branko [10] built a dynamic test generation system whose knowledge is formalized by Web Ontology Language (OWL). The idea of dynamic test generation system is to apply some question/answer template and use reasoning algorithm to generate question and answers. Every concept in OWL has specific elements. The decision algorithm will determine which question/answer templates match the specific elements, and choose these templates to generate multiple choice questions. This ontology-based dynamic test generating approach can be applied to different domain, but it is time-consuming to construct an OWL.

The automatic question generation approaches, proposed by previous researches, are not appropriate for generating arithmetic word problem because the fixed question templates cannot generate arithmetic word problems with various question structures to evaluate different arithmetic concepts. In addition, the scenarios are necessary to be embedded in arithmetic word problems to test whether students can apply mathematical abilities to solve practical problems, but the previous researches did not take question scenarios into consideration. Thus, two issues to automatically generate arithmetic word problem are described as follows:

1. How to generate and collect arithmetic word problem with various question structures.
2. How to reuse and share scenarios from previous arithmetic word problems.



CHAPTER 3 FOLKSONOMY-BASED ARITHMETIC WORD PROBLEMS COLLECTION AND ARITHMETIC WORD PROBLEM ANALYSIS

3.1 FOLKSONOMY-BASED PROBLEMS COLLECTION

The arithmetic word problem enables students to apply their arithmetic skills to solve the problem in application level arithmetic of Bloom's taxonomy. However, it is costly and time consuming to generate arithmetic word problems because the good arithmetic questions with embedded reasonable scenarios are difficult to be designed. Thus, how to provide more problems automatically to reduce the arithmetic word problem constructing cost is an important issue. Traditionally, many teachers generated various ideas of different arithmetic questions and different embedded scenarios, but these ideas are difficult to be shared without a good management technique. With rapid growth of Web2.0, one of the emerging visions is the "harnessing the collective intelligence" of a users community to contribute their knowledge. The folksonomy means the user-generated classification, emerging through bottom-up consensus. In this thesis, we aim to apply folksonomy-based approach to collecting teachers' knowledge consisting of **Question Structure** and **Question Scenario**. Question structure is the structure of this word problem, and with an embedded question scenario to describe the arithmetic word problem. Because a large amount of ideas of question structures and scenarios generated by different teachers can be shared

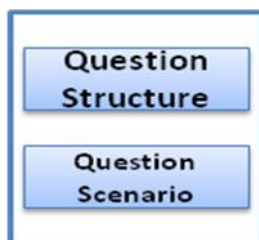
and reused in order to reduce teachers' effort to construct arithmetic word problem. In order to facilitate managing and reusing the shared arithmetic word problem knowledge, a proposed **Arithmetic Word Problem Frame Hierarchy** knowledge representation is used to represent these question structure because the arithmetic word problem of the same question structure are stereotyped and the inheritance property of frames is appropriate to represent the category structure of arithmetic word problems. Teachers often construct various arithmetic word problems via transforming the arithmetic logic embedded in one arithmetic word problem. By our observation, production rules in context-free grammar are appropriate to describe this kind of transformation. The proposed **Arithmetic Word Problem Grammar** can facilitate maintenance of arithmetic word problems and their evolutions including **Self-Evolution** and **Collaborative-Evolution**. Each arithmetic word problem has its question structure in which a propositional logic statement is embedded; every atomic proposition can be selected as a question statement. There are two kinds of evolutions. In order to generate different problems, self-evolution selects different atomic propositions from the original propositional logic statement as the question statement. Collaborative-evolution integrates two question structures with the same atomic proposition to generate a new question structure. More specifically, the arithmetic word problem knowledge provided by multiple teachers can be fused and perform problems' evolutions to generate more questions by applying some specific production rules. Besides, the rules embedded in the frame can be

used to apply different scenario to the question structure; thus a large amount of arithmetic word problems can be generated.

In Section 3.2, we will analyze the properties of arithmetic word problems which we focus on. And the corresponding knowledge representations and knowledge evolutions, which are performed via new knowledge provided by teachers, will be described in Section 3.3.

3.2 ARITHMETIC WORD PROBLEM ANALYSIS

As shown in Figure 1, according to our observation, an arithmetic word problem consists of a **Question Structure** with an embedded **Scenario** to describe the arithmetic word problem. This kind of arithmetic word problem can evaluate whether students can apply the arithmetic skills in real environment, which is the application level knowledge in Bloom's cognitive domain. Thus, when an arithmetic word problem is provided by a teacher, the problem's question structure is managed as various arithmetic problems to examine different arithmetic concepts, and its scenario can be managed in a scenario repository and applied to other question structures to generate more various arithmetic word problems with the same embedded scenario.



Arithmetic word problem

Figure 1 Property of Arithmetic Word Problem

3.3 ARITHMETIC WORD PROBLEM FRAME HIERARCHY

In this section, we will introduce our knowledge representation, the arithmetic word problem frame hierarchy. The question structures which have a stereotyped description are provided by teachers. In order to share and reuse question structures, the frame based representation is used to represent each question structure, and the frame's inheritance property can facilitate managing the categories of question structures.

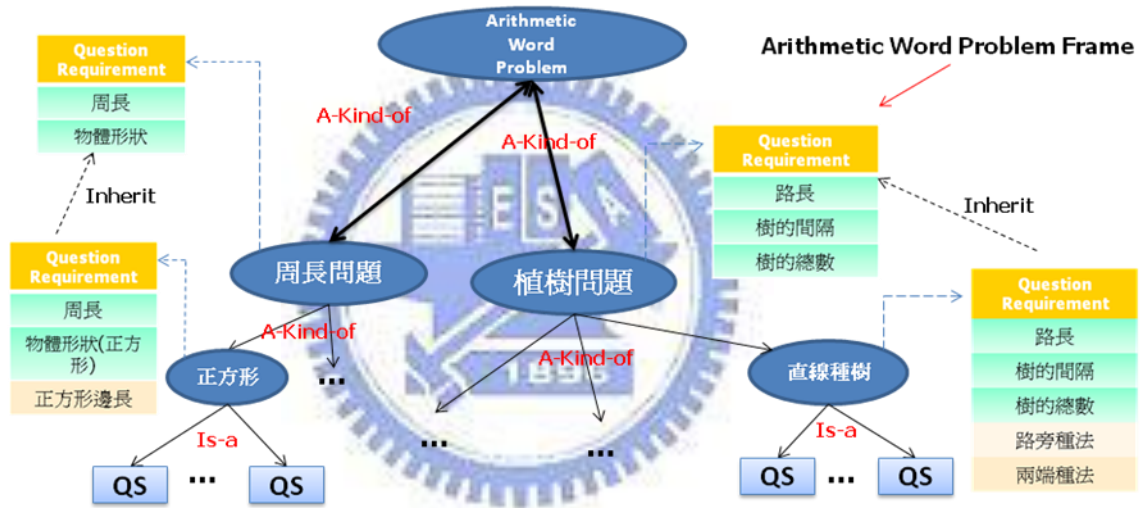


Figure 2 An arithmetic word problem frame hierarchy

As shown in Figure 2, all the question structures of arithmetic word problems are maintained as an **arithmetic word problem frame hierarchy**, where the root is a general frame whose properties must be inherited by all kinds of question structures. The second level frames, the major problem categories, consist of different kinds of arithmetic problem with their specific requirements, named **Question Requirement**. For example, “perimeter problem” is a kind of arithmetic problem, and the corresponding frame slots “perimeter” and “object shape” are

defined to represent the necessary requirements in this kind of problem. All frames of sub-problem category have to inherit the necessary requirements from the frame of their corresponding major problem category, e.g. “square perimeter problem” and ”triangle perimeter problem” are two sub-problem categories inherit frame slots “perimeter” and “object shape” from the major problem category “perimeter problem”. The inherited frames from sub-problem category frames are question structure frames, which can generate real arithmetic word problem by applying question scenarios.

周長問題 Frame Attributes	Question Requirement (QR)	Scenario Slot (SS)	QR Type	SS Type
	Slot name	Slot value	Facet	
	周長 : 周長	“周長”	Variable	Changeable
	周長 : 周長的長度	“100”	Variable	Unknown
	周長 : 周長的單位	“公尺”	Variable	Changeable
	物體形狀 : 形狀	“正方形”	Fix	Changeable
	物體形狀 : 物體	“禮盒”	Fix	Changeable
	正方形邊長 : 邊長	“邊長”	Variable	Changeable
	正方形邊長 : 邊的長度	“5”	Variable	Unknown
	正方形邊長 : 邊的單位	“公尺”	Variable	Changeable
正方形周長問題 Frame Attribute	Required Question		If needed	

Self-Evolution rule
Collaborative- Evolution rule
Scenario Mapping rule

Figure 3 Slot attributes of arithmetic word problem frame

Figure 3 shows the slot attributes of arithmetic word problem frame which is a “square perimeter problem” frame of sub-major problem category. The slot attributes consist of slot attributes from “perimeter problem” frame and specific slot attributes of “square perimeter problem” frame. The slot name is combining

with question requirement name and scenario slot name, the left hand side of colon is question requirement name and the right hand side is corresponding scenario slot name. Slot value stores the value of scenario slot. Furthermore, facet defined the type of question requirement and scenario slot, the left hand side of colon is question requirement type and the right hand side is scenario slot type. More important, there are rules which are self-evolution rule, collaborative-evolution rule and scenario mapping rule embedded in every arithmetic word problem frame, these rules will be triggered while generating arithmetic word problems

In Section 3.4, we will introduce how we manage question structure of arithmetic word problems and perform different kinds of evolution.

3.4 SPIRAL MODEL FOR FOLKSONOMY-BASED PROBLEMS COLLECTION

We apply folksonomy-based approach to collecting a large amount of question structures, thus more and more various arithmetic word problems can be generated.

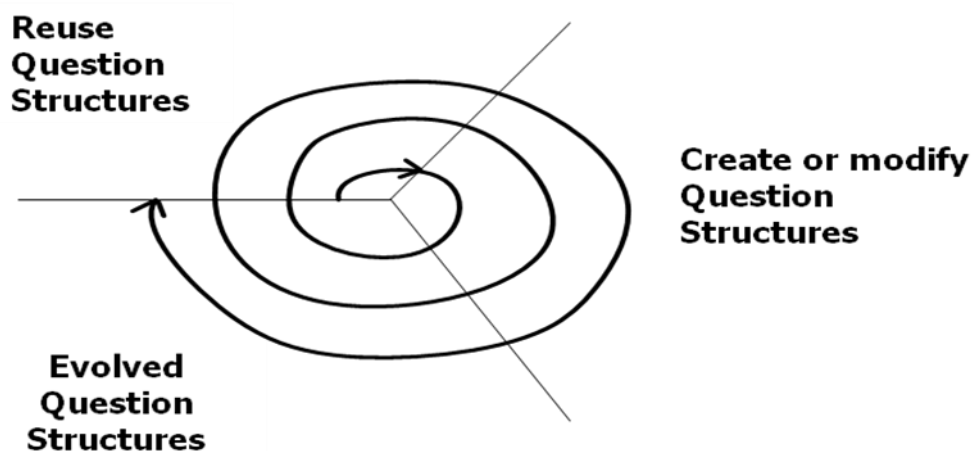


Figure 4 Spiral model for folksonomy-based problems collection

As shown in Figure 4 **Spiral model for folksonomy-based problems collection**, users reuse or reference the existing question structures and as they familiar with the folksonomy-based approach, they will create new question structures or modify existing question structures. In addition, these question structures will evolve into new question structures, thus these question structures can generate a large amount of various arithmetic word problems.

3.5 GRAMMAR AND QUESTION STRUCTURE EVOLUTION

3.5.1 ARITHMETIC WORD PROBLEM GRAMMAR AND APPLYING QUESTION SCENARIO

Teachers often construct various arithmetic word problems via transforming the arithmetic logic embedded in one arithmetic word problem. By our observation, production rules in context-free grammar are appropriate to describe this kind of transformation. Besides, applying context-free grammar can facilitate maintenance of arithmetic word problems and their evolutions. A context-free grammar is a grammar that the left side of every production rules is always a single nonterminal symbol and the right side is a string of terminals and/or nonterminals. Based on the previous definition of context-free grammar, we can use different nonterminals to denote different major problem categories and sub-problem categories. The production rules from nonterminals of major problem categories only derive the nonterminals of their corresponding sub-problem categories. Hence, this grammar property can help us easily manage arithmetic word problems. Moreover, the similar new arithmetic word problems

can be generated from an arithmetic word problem by adding new production rules, evolving from the production rules of the original question structure. Besides, two questions can be integrated to generate new question structures by combining production rules of two question structures. These two question evolution and fusion approaches are named self-evolution and collaborative-evolution, respectively. The following chapters will show how to model an arithmetic word problem by context-free grammar, and how to perform self-evolution and collaborative-evolution based on the grammar. For providing proper mapping from question structures to question scenarios, our context-free grammar also contains production rules which drive question structures through question requirements into scenario slots, the terminal symbols.

Definition 1: Arithmetic Word Problem Grammar

- $G = (V, T, P, S)$ is the arithmetic word problem grammar where $V = \{S\} \cup C \cup B \cup QS \cup QR$, is a finite set of non-terminal symbols, $T = SS^U \cup SS^C$, is a finite set of terminal symbols, P is the set of production rules, and S is the start symbol.
- $V = \{S\} \cup C \cup B \cup QS \cup QR$
 - S is the start symbol.
 - C is a set of major problem category non-terminal symbols
 - B is a set of sub-problem category non-terminal symbols
 - QS is a set of question structure non-terminal symbols.
 - $QR = QR^V \cup QR^F \cup QR^{V'}$ is a set of question requirement non-terminal symbols.

- QR^V is a set of question requirement which contains numeric variable, is called “*variable type*” question requirement.
 - QR^F is a set of question requirement which not contains no numeric variable, is called “*fix type*” question requirement.
 - QR^V is a set of question requirement which is a question statement, and is variable type question requirement too.
- $T = SS^U \cup SS^C$ is a set of terminal symbols,
 - SS^U is a set of scenario slot which contains variable, is called “*unknown type*” scenario slot.
 - SS^C is a set of scenario slot which does not contain variable, is called “*changeable type*” scenario slot.
 - $P = CP \cup BP \cup QSP \cup QRP \cup SSP$ is a set of production rules
 - CP is a set of major problem category production rules, which is $S \rightarrow C_1 | C_2 | \dots | C_p$, where $C_i \in C$, $1 \leq i \leq p$, p is the number of major problem categories.
 - BP is a set of sub-problem category production rules, which is $C_i \rightarrow B_{i1} | B_{i2} | \dots | B_{iq}$, where $B_{ij} \in B$, $1 \leq j \leq q$, q is the number of sub-problems in major problem category C_i
 - QSP is a set of question structure production rules, which is
 - $B_{ij} \rightarrow QS_{ij1} | QS_{ij2} | \dots | QS_{ijr}$, where $QS_{ijk} \in QS$, $1 \leq k \leq r$, r is the number of question structures in sub-problem category B_{ij} .
 - QRP is a set of question requirement production rules, which is

$QS_{ijk} \rightarrow QR_{ijk1}^X QR_{ijk2}^X \dots QR_{ijks}^V$, where $QR_{ijkm} \in QR$, $1 \leq m \leq s$, s is the number of question requirements in question structure QS_{ijk} , $X \in \{V, F\}$.

- SSP is a set of scenario slot production rules, which is

$QR_{ijkm} \rightarrow SS_{ijkm1}^X SS_{ijkm2}^X \dots SS_{ijkmt}^X$, where $SS_{ijkmn} \in SS$, $1 \leq n \leq t$, t is the number of scenario slots in question requirement QR_{ijkm} , $X \in \{C, U\}$.

The number of production rules in **P** is depending on the corresponding instances which are provided by users. For example, if users add a new sub-problem category, then a new production rule corresponding with this new sub-problem will be added to **BP**, and if users provided a new question structure, then a new production rule corresponding with this new question structure will be added to QSP.

The major problem category production rules (CP) are defined to represent the rules of generating the major problem categories in arithmetic word problem frame hierarchy. The usage of CP is shown in Example 1.

Example 1: Production rule of CP

$S \rightarrow \text{植樹問題} \mid \text{周長問題} \mid \dots$

The initial start symbol can generate all the major problem category non-terminal symbols, such as “tree-plant problem” and “perimeter problem”.

Each major problem category non-terminal symbol can be transformed to a set of sub-problem category non-terminal symbols using sub-problem category production rules (BP).

Example 2: Production rule of BP

植樹問題 → 直線種樹 | 多邊形種樹 | ...

This sub-problem category production rule shows that “直線種樹” and “多邊形種樹” are two sub-problem categories of “tree-plant problem”.

Question structures are included in sub-problem categories and the question structure production rules (QSP) can show this grammar.

Example 3: Production rule of QSP

直線種樹 → 兩端種樹 | 一端種樹 | ...

Assume there are two question structure “兩端種樹” and “一端種樹” in category “直線種樹”, QS_1 , the production rules of QSP are used to generate all the question structure non-terminal symbols.

An arithmetic word problem consists of a set of statements, named question requirement, to describe the question’s constraints and requirements, and one of the question requirements have to be described as a question statement. For example, a problem of “square perimeter” consists of at least three statements to describe “object shape”, “square side” and “square perimeter”, and the “square

perimeter” or *“square side”* can be described as a question to ask students. The question requirements, which contain numeric variables and can be described as a question statement, are named variable type question requirement (QR^V), and others are named fix type question requirements (QR^F).

Example 4: User provide a perimeter problem Question Structure

An arithmetic problem “perimeter problem” can be represented as $QS_{perimeter} \rightarrow QR^F_1 QR^V_2 QR^V_3$, where QR^F_1 is “object shape”, QR^V_2 is “square side”, and QR^V_3 is “square perimeter”. In this question, the variable type question requirements need to be identified, such as “square side” and “square perimeter”, and one of the variable type question requirements needs to be determined as the question statement by user, e.g., QR^V_3 .

Example 5: Production rule of QRP

Assume an arithmetic word problem of “兩端種樹植樹問題” is “榕樹 100 顆，種在公路一旁，每隔 5 公尺種一棵，兩端都種，公路 500 公尺”。The question structure “兩端種樹” can be represented as five question requirements: ”樹的總數”，“路旁種法”，”樹的間隔”，“兩端種法”，and “路長”，so the question requirement production rule can be defined as follows:

兩端種樹 \rightarrow 樹的總數, 路旁種法, 樹的間隔, 兩端種法, 路長:

Among the question requirements, “榕樹 100 棵” and “公路 500 公尺” are variable type question requirements because 100 and 500 are numeric variables.

“種在公路一旁” and “兩端都種” are fix type question requirements, which cannot be transformed to the question statement. “公路500公尺” is determined as the question statement by user. Thus, folksonomy-based item bank management system will generate the sentence of question requirement “請問公路多少公尺”.

Every Question Requirement consists of a set of the corresponding description statements, named scenario slot, to describe each requirement in different words or sentences. For example, a requirement of “樹的總數” consists of at least three words to describe “種的物品”, “數量”, and “物品單位”. The “種的物品” can be described by different words or sentences like “花”, “樹” or “電線桿”, etc., and the “物品單位” must be related with the “種的物品”. The “數量” can be any numeric variable like 100,150 or 20, etc. The scenario slots, which contains only number, are named unknown type scenario slot (SS^U), and others are named changeable type scenario slot (SS^C). Each tag of scenario slot like, “種的物品” or “數量”, etc. are provided by users, in other words, we are not focus on dealing with the natural language processing in this thesis.

Example 6: Production rule of SSP

Assume QR_1 is one of “兩端種樹” question requirement consist of three scenario slot, which are “種的物品”, “數量”, and “物品單位”. So the scenario slot production rule can be defined as follows:

兩端種樹 → 種的物品, 數量, 物品單位.

The scenario slots are involved in question requirement will be applied to corresponding value in scenario repository, thus the sentence of question requirement will be diverse. Question scenarios provided by users will be stored in **Scenario Repository**. We will introduce Scenario Repository in more detail in Section 3.6.2.



Figure 5 Scenario Slot of QR_1

Figure 5 shows that each scenario slot in QR_1 will be applied to the corresponding value in scenario repository.

If the scenario p1 be applied to the scenario slots, which are involved in QR_1 , then the sentence of QR_1 will be:

“電線桿 200 根”.

If the scenario p2 applied to the scenario slots, then the sentence of QR_1 will be:

“旗子 50 支”.

Among the scenario slots, “數量” is unknown type scenario slot because it will be applied to every number 200, 50,100 in scenario repository. “種的東西” and “物品單位” are changeable type scenario slots, because each of them applied to the values in Scenario Repository like “榕樹” and “顆” are not numbers.

Example 7: Mapping Scenario to Question Structure

The rules embedded in arithmetic word problem frame can control the question structure to apply different scenario as shown in Figure 6.

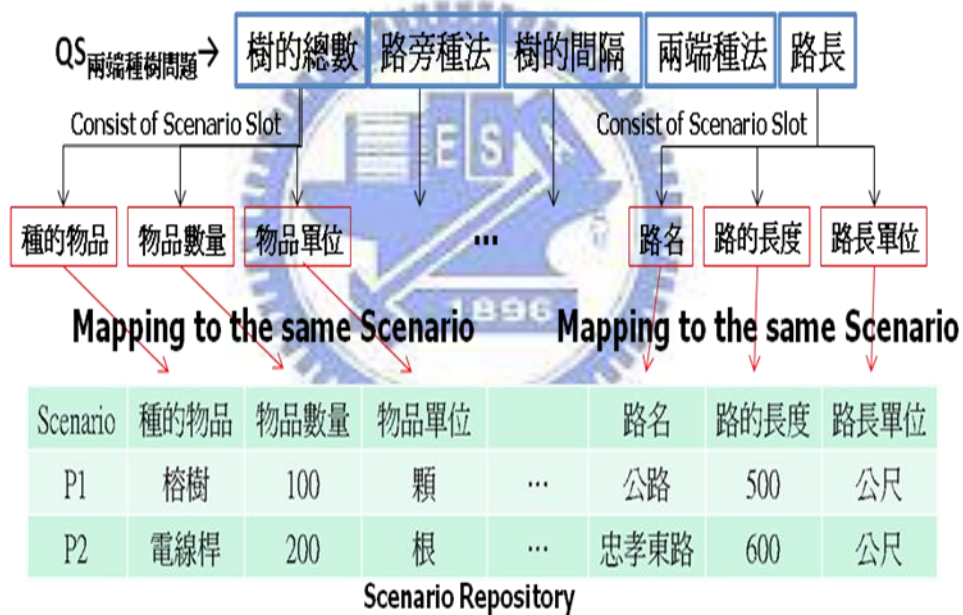


Figure 6 Scenario mapping to Question Structure

QR^V is the question statement as we mentioned before.

The rule of applying scenario to QR^V is:

Add "請問" in front of QR^V first and change the scenario slot value to "多少" if scenario slot type is unknown in QR^V . Finally, the sentence of QR^V will be shifted and become the last sentence in the generating arithmetic word problem.

Example 8: Applying scenario to QR^V

Assume the question requirement of "兩端種樹問題" is :

兩端種樹問題→樹的總數,路旁種法,樹的間隔,兩端種法,路長.

If we apply scenario P1 to the "兩端種樹問題",the sentences are:

"公路 300 公尺，每隔 3 公尺種一棵，種在公路一旁，兩端都種，請問樹多少顆？".

And if the question requirement of "兩端種樹問題" is :

兩端種樹問題→樹的總數,路旁種法,樹的間隔,兩端種法,路長.

If we mapped scenario P1 to the "兩端種樹問題",the sentences are:

"樹 100 顆，每隔 3 公尺種一棵，種在公路一旁，兩端都種，請問公路多少公尺？".

We have introduced how to model an arithmetic word problem by context-free grammar. In next section we will introduce how to evolve question structure by production rules. We have two kinds of evolution methods to evolve question structure, one is Self-Evolution and another is Collaborative-Evolution.

3.5.2 SELF-EVOLUTION

Each arithmetic word problem has its question structure in which a propositional logic statement is embedded. Only one atomic proposition, which is a variable type question requirement, of the statement will be chosen as the question statement. However, this selection is not unique. Every atomic proposition can be selected as the question statement. In arithmetic word problems, we only focus on the atomic propositions correlated to variable type question requirements. In order to generate different problems, self-evolution is to select different atomic propositions from the original propositional logic statement as the question statement. Therefore, we can add new production rules generated via self-evolution into QRP, which contains production rules drive question requirements from question structure. **Figure 7** illustrates the concept of self-evolution.



Figure 7 Self-evolution

Assume there is a multiplication problem, " $A \times B = C$ ". Usually, teacher will test students with the question: " $A \times B = ?$ ", or with the questions " $A \times ? = C$ " and " $? \times B = C$ " which are obviously different questions from the previous one. These two questions can be generated by self-evolution. Assume user provided a question structure QS_I .

The original production rule QRP of QS_I is:

$$QS_1 \rightarrow QR^{V_1'} QR^{V_2} QR^{V_3}$$

The new production rule QRP of QS_1 after applying Self-Evolution:

$$QS_1 \rightarrow QR^{V_1'} QR^{V_2} QR^{V_3} / QR^{V_1} QR^{V_2'} QR^{V_3} / QR^{V_1} QR^{V_2} QR^{V_3'}$$

The two production rules, which are $QR^{V_1} QR^{V_2'} QR^{V_3}$ and $QR^{V_1} QR^{V_2} QR^{V_3'}$ will be added in production rule automatically after applied self-evolution.

Assume QS_1 is a multiplication problem, its production rule QRP is $QS_1 \rightarrow ABC'$, A , B and C are variable type question requirement. There are three variable type question requirements, so we add two production rules after production rule QRP of QS_1 . The production rule QRP of QS_1 will be $QS_1 \rightarrow ABC' | AB'C | A'BC$, that ABC' , $AB'C$ and $A'BC$ is questions:

$$A \times B = ?.$$

$$A \times ? = C.$$

$$? \times B = C.$$



Example 9 : Self-Evolution

Assume user provide a “乘法加總問題” question structure.

Original production rule QRP of “乘法加總問題” is:

乘法加總問題 \rightarrow 樹的總數, 樹的價錢, 樹的總價’, where “樹的總數” and “樹的價錢” are variable type , and “樹的總價” is question statement.

New production rule QRP of “乘法加總問題” after self-evolution be applied is :

乘法加總問題 \rightarrow 樹的總數, 樹的價錢, 樹的總價’ | 樹的總數’, 樹的價錢, 樹的總價 | 樹的總數, 樹的價錢’, 樹的總價.

The two production rules, which are 樹的總數', 樹的價錢, 樹的總價 and 樹的總數, 樹的價錢', 樹的總價 will be added in production rule *QRP* automatically after self-evolution be applied.

The arithmetic word problem generated from production rule 乘法加總問題→ 樹的總數, 樹的價錢, 樹的總價' is:

“樹 100 顆，每顆樹 50 元，請問總共多少元?”.

The arithmetic word problem generated from production rule 乘法加總問題→ 樹的總數', 樹的價錢, 樹的總價 is :

“每顆樹 50 元，總共 5000 元，請問樹多少顆?”.

The arithmetic word problem generated from production rule 乘法加總問題→ 樹的總數, 樹的價錢', 樹的總價 is :

“樹 100 顆，總共 5000 元，請問每顆樹多少元?”.

3.5.3 COLLABORATIVE-EVOLUTION

It is possible that the same atomic proposition in different propositional logic statements; in other words, there exists a variable type question requirement involved in different question structures. The different propositional logic statements can be integrated into a new propositional logic statement, which means two arithmetic word problems can be integrated into a new arithmetic word problem. Collaborative-evolution is to integrate two question structures with the same variable type question requirement into a new question structure. We can

add new production rules generated via collaborative-evolution into QSP of these two question structure. Figure 8 illustrates the concept of Collaborative-evolution.



Figure 8 Collaborative-evolution

If a variable type question requirement appeared in two question structures of different sub-problem categories then add new production rules in production rule QSP of these two question structures.

Assume there two question structures, $QS_1 \rightarrow QR^V_1 QR^V_2 QR^V_3$ in Sub-problem category B_1 and $QS_2 \rightarrow QR^V_3 QR^V_4 QR^V_5$ in sub-problem category B_2 , Then add new production rule in production rule QSP of these two question structures.

$$B_1 \rightarrow QS_{new}$$

$$B_2 \rightarrow QS_{new}$$

$$QS_{new} \rightarrow QR^V_1 QR^V_2 QR^V_4 QR^V_5$$

QR^V_3 appears in both QS_1 and QS_2 will be discarded first, and then can be integrated into a QS_{new} .

Assume QS_1 is a multiplication problem “ $A \times B = C$ ” from sub-problem category B_1 , its production rule is $QS_1 \rightarrow QR^V_1 QR^V_2 QR^V_3$. And the other, QS_2 is an addition problem “ $C + D = E$ ” from sub-problem category B_2 , its production rule is $QS_2 \rightarrow QR^V_3 QR^V_4 QR^V_5$. Because of QR^V_3 appears in both question structures, so we discard QR^V_3 and integrate QS_1 and QS_2 to generate a new

question structure. Because QS_1 and QS_2 belong to different sub-problem categories, we add new production rules, which are:

$$B_1 \rightarrow QS_{new}$$

$$B_2 \rightarrow QS_{new}$$

$$QS_{new} \rightarrow QR^V_1 QR^V_2 QR^V_4 QR^V_5$$

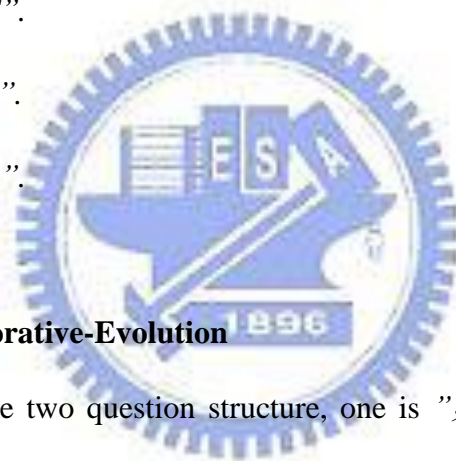
QS_{new} , which is integrated with QS_1 and QS_2 . After QS_{new} apply Self-Evolution, the problems of QS_{new} are :

$$“? \times B + D = E”.$$

$$“A \times ? + D = E”.$$

$$“A \times B + ? = E”.$$

$$“A \times B + D = ?”.$$



Example 10: Collaborative-Evolution

Assume these are two question structure, one is “乘法加總問題” question structure from sub-problem category B_1 , its production rule is :

乘法加總問題 \rightarrow 樹的總數, 樹的價錢, 樹的總價, where “樹的總價” and “樹的價錢” are variable type , and “樹的總數” is question statement.

And the other, is “兩端種樹問題” question structure from sub-problem category B_2 , its production rule is :

兩端種樹問題 \rightarrow 樹的總數, 路旁種法, 樹的間隔, 兩端種法, 路長, where “樹的總數”, “樹的間隔” are variable type question requirements, “路旁種法”, “兩端種法”, is fix type question requirements, and “樹的總數” is question statement.

Because of the variable type question requirement, "樹的總數" appears in both question structures, so we discard "樹的總數" and integrate "乘法加總問題" and "兩端種樹問題" to generate a new question structure. Because of "乘法加總問題" and "兩端種樹問題" is belong in different sub-problem category, so we add new production rules, which are :

$$B_1 \rightarrow QS_{new}$$

$$B_2 \rightarrow QS_{new}$$

$$QS_{new} \rightarrow \text{樹的價錢, 樹的總價, 路旁種法, 樹的間隔, 兩端種法, 路長}$$

We could generate QS_{new} with this new production rule, which is integrated with QS_1 and QS_2 . After QS_{new} apply Self-Evolution, the production rule QRP of QS_{new} is :

$$QS_{new} \rightarrow \text{樹的價錢', 樹的總價, 路旁種法, 樹的間隔, 兩端種法, 路長} \mid \text{樹的價錢, 樹的總價', 路旁種法, 樹的間隔, 兩端種法, 路長} \mid \text{樹的價錢, 樹的總價, 路旁種法, 樹的間隔', 兩端種法, 路長} \mid \text{樹的價錢', 樹的總價, 路旁種法, 樹的間隔, 兩端種法, 路長}'$$

Here we list two arithmetic word problems generated by production rules.

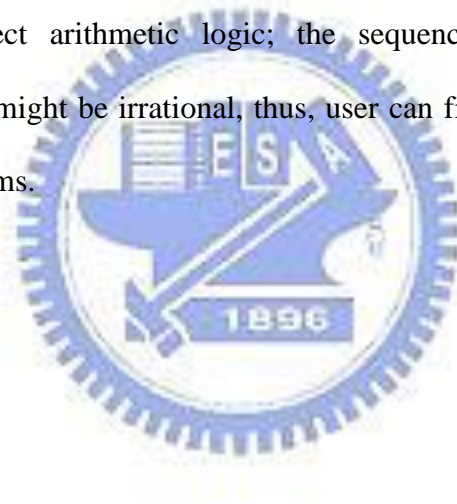
The arithmetic word problem generated from production rule $QS_{new} \rightarrow \text{樹的價錢', 樹的總價, 路旁種法, 樹的間隔, 兩端種法, 路長}$ is :

"總共要花 3000 元，種在公路一旁，每隔 3 公尺種一棵，兩端都種，公路 300 公尺，請問每顆樹多少元？".

The arithmetic word problem generated from production rule $QS_{new} \rightarrow$ 樹的價錢, 樹的總價', 路旁種法, 樹的間隔, 兩端種法, 路長 is :

“每顆樹 30 元，種在公路一旁，每隔 3 公尺種一棵，兩端都種，公路 300 公尺，請問總共要花多少元?”.

Moreover, the QS_{new} can be applied collaborative-evolution again and integrated with another question structure to generate a new question structure; thus, more and more various arithmetic word problems can be generated from the new question structure. In this thesis, we focus on generating the arithmetic word problems with correct arithmetic logic; the sequence of sentences in the generating problems might be irrational, thus, user can fine-tune the sentences in the generating problems.



3.6 USER SHARED CONTENT MANAGEMENT

3.6.1 QUESTION REQUIREMENT ONTOLOGY AND SCENARIO SLOT ONTOLOGY

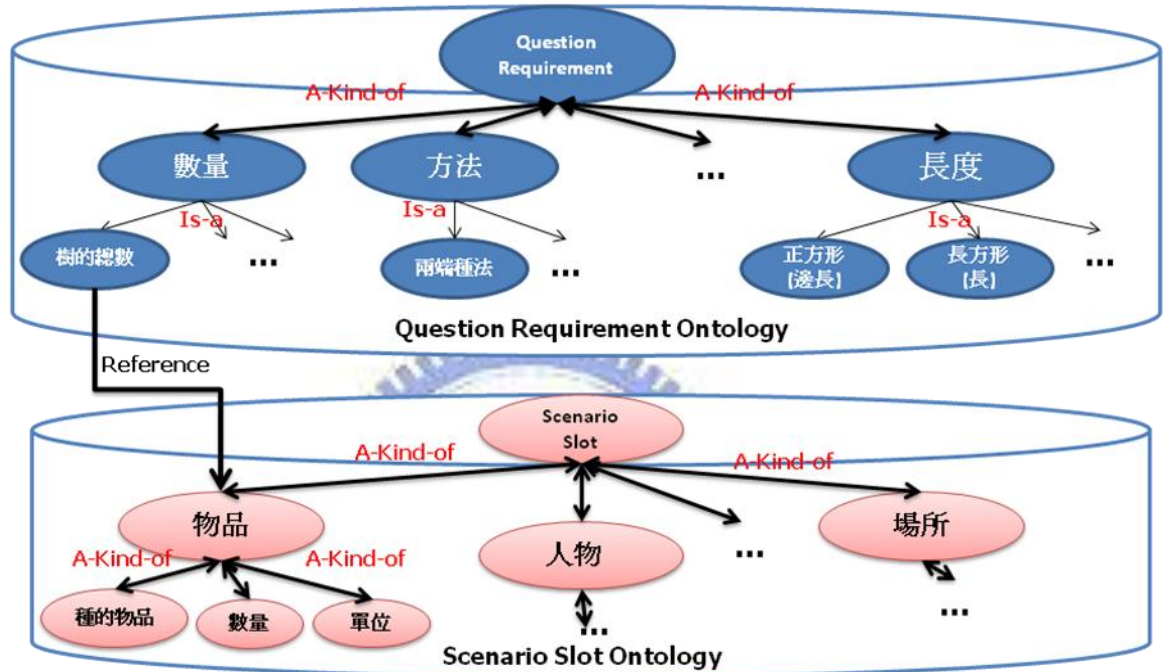


Figure 9 Question requirement ontology and scenario slot ontology

Figure 9 shows the question requirement ontology and scenario slot ontology. We use question requirement ontology and scenario slot ontology to maintain the relation of question requirement and every scenario slot contains in question requirement. The structure of both question requirement ontology and scenario slot ontology is similar. There are many concept usually be tested in elementary school arithmetic word problem like “數量”, “長度”, “形狀” and “面積”, etc., hence these concepts are maintain in layer 2 in question requirement ontology. Layer 3 maintains all of Question Requirements provided by users. In an arithmetic word problem, some words or statements could be changed like

“Object”, “Place” or “Character”, etc. Hence these categories are maintained in layer 2 in scenario slot ontology, and layer 3 maintains all the attributes provided by user related with the correspond category in layer 2.

Every attributes of arithmetic word problem frame will reference to question requirement ontology to keep consistence between each question requirement of different arithmetic word problem category. For example, the question requirement “長方形的長” will appear in different kind of arithmetic word problem. Hence the arithmetic word problem frame in different kind of category contain the same attribute “長方形的長”, which will reference to the “長方形(長)” in the category of “長度” in question requirement ontology. That will make different question structure in different arithmetic word problem category could apply Collaborative-Evolution.

3.6.2 SCENARIO REPOSITORY

We have mentioned **Scenario Repository** above. The question scenario provided by user will be stored in scenario repository; The user provided values of each scenario slots contain in question requirement will be stored into scenario repository..

Example 11: Scenario Repository

Assume there are two question requirements QR_1 and QR_2 . QR_1 consists of scenario slot SS_1 , SS_2 and SS_3 , QR_2 consist of Scenario Slot SS_2 , SS_3 and SS_4 . For QR_1 , assume User assigns values a, b, c to SS_1 , SS_2 and SS_3 respectively, and

assigns values d, e, f to SS₂, SS₃ and SS₄ respectively for QR₂. Then these values will be stored in Scenario Repository shown as Figure 10.

Scenario Slot Tag	Scenario Repository				
	SS ₁	SS ₂	SS ₃	SS ₄	...
Scenario P1	a	b	c		...
Scenario P2		d	e	f	...
...

Figure 10 Scenario Repository



CHAPTER 4 FOLKSONOMY-BASED ITEM BANK MANAGEMENT SYSTEM

4.1 SYSTEM ARCHITECTURE

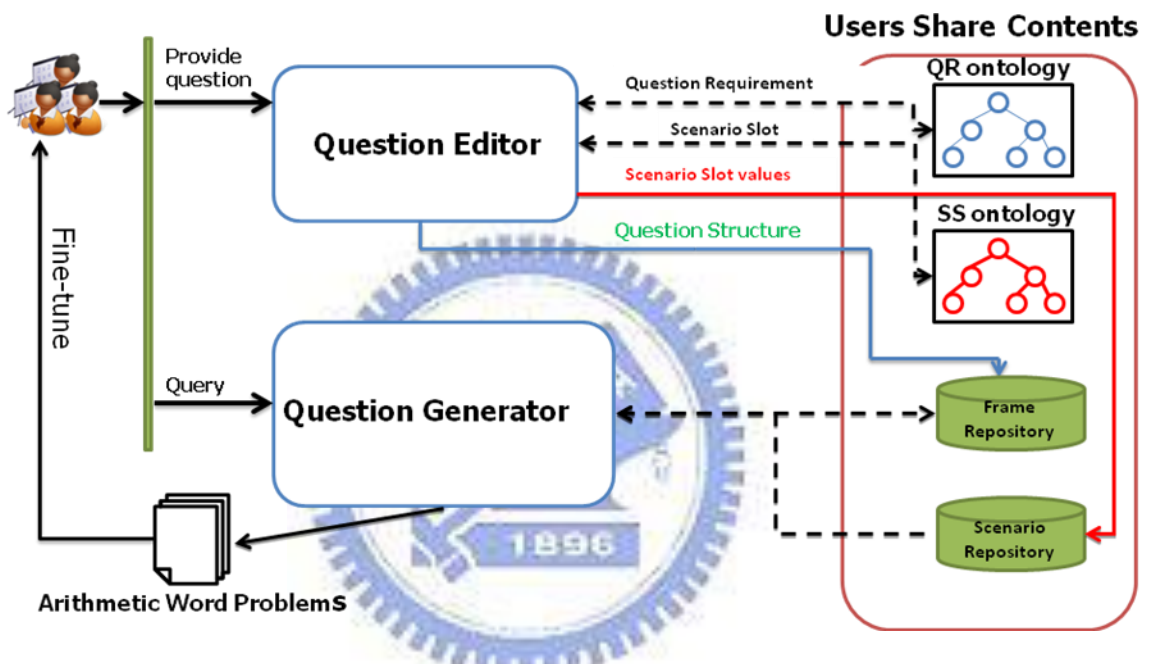


Figure 11 System architecture of folksonomy-based item bank management system

As shown in Figure 11, the **Folksonomy-Based Item Bank management system (FIB)** can assist teachers to share and reuse their designed arithmetic word problems. Like other Web2.0 society system, users could recommend resources and discussed with other community users in FIB, but in this thesis, we focus on the functions of sharing question structures and scenarios.

FIB provides two kinds of functions: question collecting and generating problem to facilitate teachers to provide and reuse arithmetic word problem, respectively. When teachers aim to give questions, they could edit question

structures and scenarios using Question Editor, where the previous question structures and scenarios provided by other teachers can be referred to design more various scenarios. When teachers aim to reuse questions, they can choose an arithmetic word problem category to determine the scope of generating questions. Then, the question structures in the selected arithmetic word problem category will be collected, and self-evolution, collaborative-evolution are applied to these question structures to generate more related question structures. These question structures can apply the previous shared scenarios to generate arithmetic word problems for teachers, and these problems can be fine-tuned and downloaded for teachers to use.



4.2 QUESTION COLLECTION PHASE

4.2.1 TEACHER PROVIDED SCENARIO

In Question Collection phase, teachers could edit question structure and scenario through Question Editor, as shown in Figure 12.

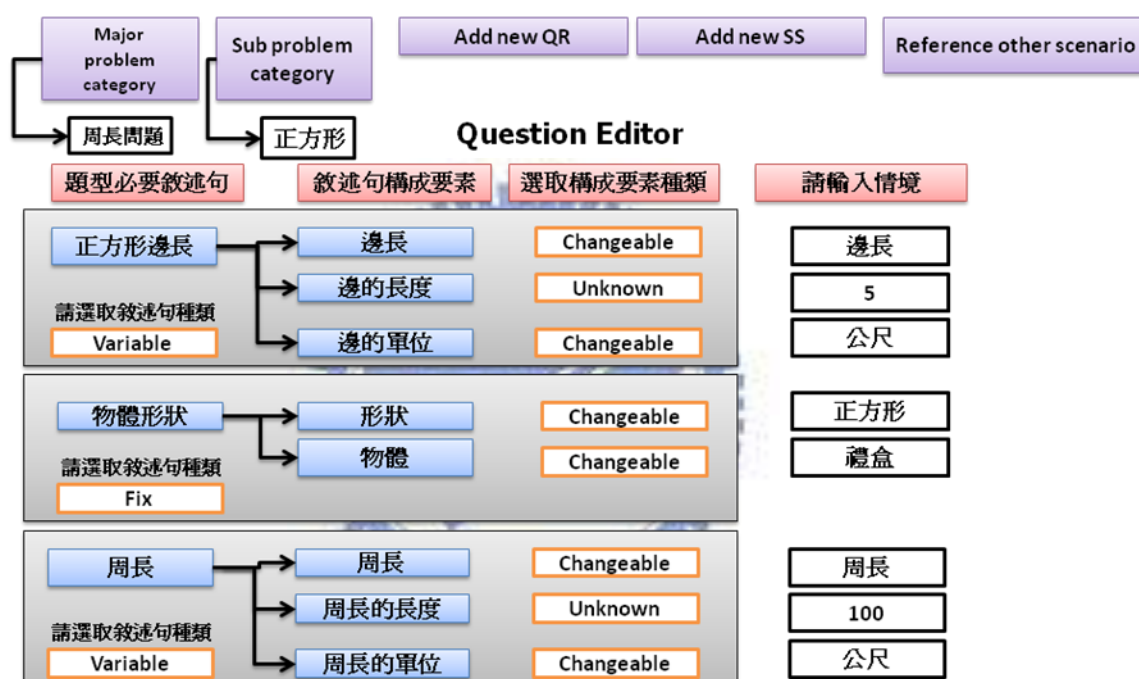


Figure 12 Question Editor Interface

Firstly, assume teachers select an arithmetic word problem category “perimeter problem category” from major problem category list, and then the question requirements of perimeter problem category, such as “周長” and “物體形狀”, will be shown to teachers. After determining major problem category, teachers need to select sub-problem category, e.g. “Square problem”, and the corresponding question requirements “正方形邊長” will be shown to teachers.

Then, they could refer to previous shared question scenarios provided by other teachers and start to assign the values of scenario slot into the textbox. The values of scenario slots provided by teachers will be stored in scenario repository and shared to other teachers, so more and more question scenarios can be generated in the item bank.



4.2.2 TEACHER EDIT QUESTION REQUIREMENT

Teachers could edit question requirement by question editor. The steps of adding a new question requirement is shown in Figure 13.

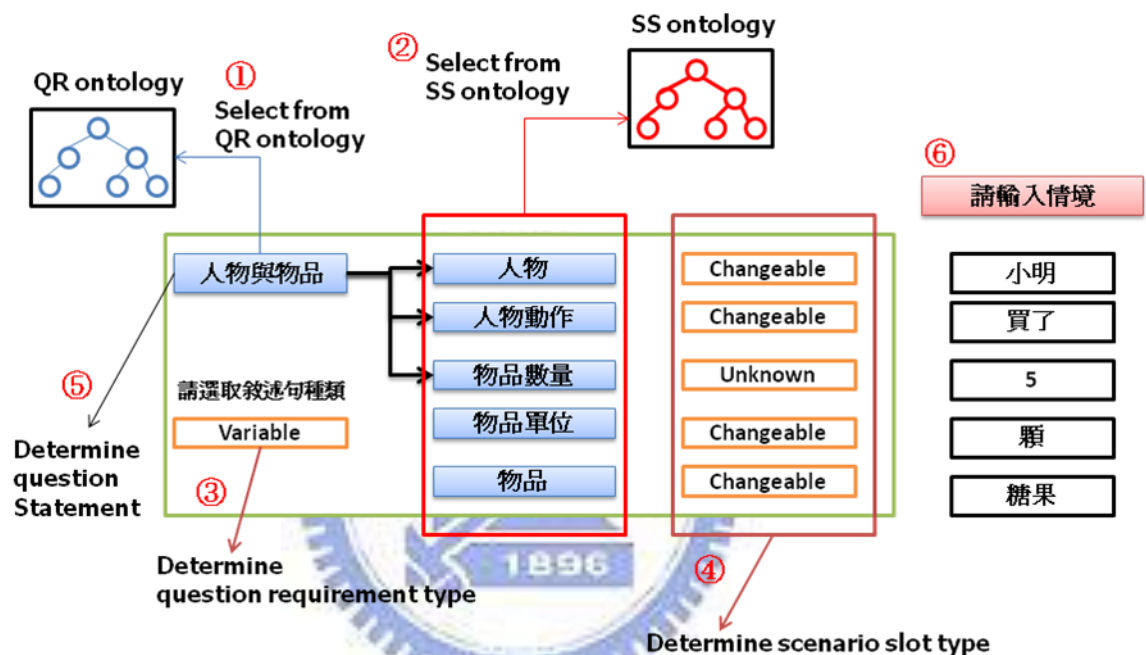


Figure 13 The Steps of editing new question requirement

First, teachers have to select a question requirement from question requirement ontology. If they can't find suitable question requirement, they can create a new one and assign a tag for it. For scenario slots, teachers can select one from scenario slot ontology to describe this question requirement. If they can't find the desire one, they also can create a new scenario slot and assign a tag for it. Then teachers have to determine the question requirement type. After that, while each of the scenario slots are default as changeable type, teacher has to assign which scenario slot is the unknown type. After that, teachers need to determine

which question requirement in the question structure is question statement, and the question requirement will be QR^V in the question structure. Finally, teachers could start to assign the values of scenario slots into the textbox.

4.3 GENERATION PROBLEM PHASE

4.3.1 SELECT PROBLEM CATEGORY

As shown in Figure 14, in FIB, teachers can ask system to generate arithmetic word problems with the same major problem category, sub-problem category, or question structure.

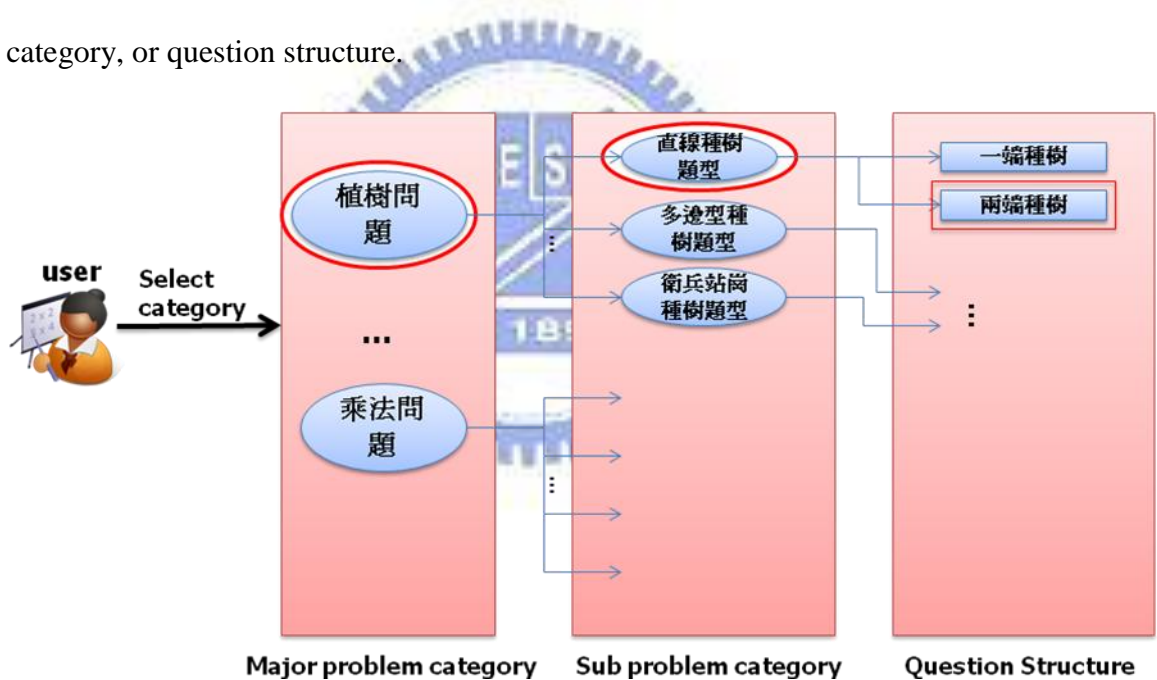


Figure 14 Arithmetic problem category

In Figure 14, the question structure “兩端種樹” is selected.

4.3.2 QUESTION GENERATION CONSTRAIN

After teachers determine their desired scope to generate questions, the corresponding production rules will be fired according to the selected scope.

Example 14: Problem Generation Scope

All question structures in the category chosen by user will be applied by Self-Evolution and Collaborative-Evolution first, and fire production rules related with chosen category. As shown in Figure 15, if a major problem category “植樹問題”, a sub-problem category “直線種樹”, or a Question Structure is chosen by teachers, all the corresponding production rules will be fired to generate arithmetic word problems of the same major problem category, sub-problem category, or question structure, respectively. The problem generation scopes of major problem category, sub-problem category, and question structure are shown as the largest dot square, the medium dot square, and the smallest dot square, respectively.

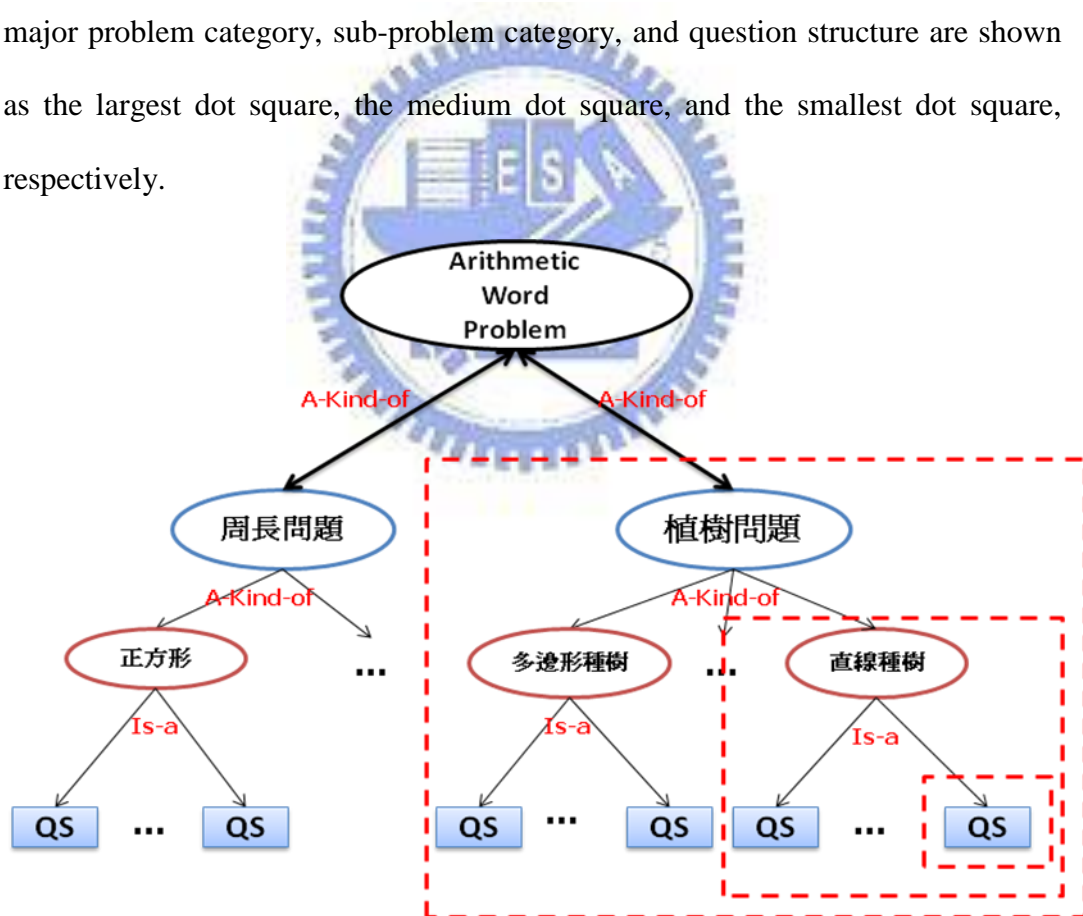


Figure 15 Problem Generation Scope

4.3.3 CHOOSE OR FINE-TUNE ARITHMETIC WORD PROBLEMS

Generated arithmetic word problems will be listed and categorized into Self-Evolution Problem List and Collaborative-Evolution Problem List as shown in Figure 16.

選取

產生的題目列表

Self-Evolution Problem List

樹100顆，每隔3公尺種一棵，種在公路一旁，兩端都種，請問公路多少公尺？

忠孝東路600公尺，每隔3公尺插一根，插在公路一旁，兩端都插，請問樹電線桿多少根？

...

Collaborative-Evolution Problem List

公路300公尺，種在公路一旁，兩端都種，每隔3公尺種一棵，每顆樹30元，總共要花多少元？

...

下載試題

Figure 16 Generation problem list

Teachers could choose problems or fine-tune the sentences of problems, and then download these arithmetic word problems.

CHAPTER 5 IMPLEMENTATION AND EXPERIMENT

5.1 SYSTEM IMPLEMENTATION

We have implemented a Folksonomy-Based Item Bank management system (FIB) as shown in Figures 15~17, where teachers can use question editor to edit question structure and provide scenarios. Teachers can ask system to generate arithmetic word problem and choose or fine-tune them for practices or exams.

要素名稱	要素內文	範例	排序	新增題幹要素
物體的形狀	<input type="text"/> 長方形		請選擇 新增自訂題幹要素	加入現有要素
連接詞(的)	<input type="text"/> 的		請選擇 新增自訂題幹要素	加入現有要素
物體名稱	<input type="text"/> 志孝東路		請選擇 新增自訂題幹要素	加入現有要素
物體的數量	<input type="text"/> 一		請選擇 新增自訂題幹要素	加入現有要素
物體的數量單位	<input type="text"/> 個		請選擇 新增自訂題幹要素	加入現有要素

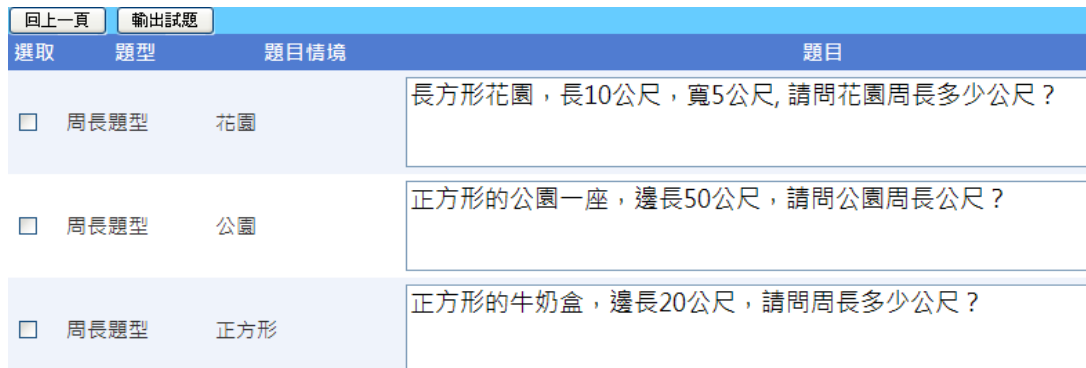
Figure 17 Question editor

Teachers could edit question structure and assign the values of scenario slot into the textbox, and add new question structure or scenario slot by question editor as shown in Figure 17.

請選擇您要取得的題型: 主題型			周長題型	子題型	請選擇子題型	確定	回上一頁
勾選想要的題幹	題幹名稱	此題幹的題目範例					
<input type="checkbox"/>	正方形周長基本問題(算周長)	正方形的盒子一個，邊長5公分，請問盒子的周長多少公分？					
<input type="checkbox"/>	長方形的物體(周長)	長方形的長30公尺，寬25公尺，請問周長是多少公尺？					
<input type="checkbox"/>	正三角形周長基本問題(算周長)	正三角形的一塊土地，邊長73公尺，請問周長是多少公尺？					

Figure 18 Teacher select problem categories

Teachers can ask system to generate arithmetic word problem, they can choose the problem category from the interface of FIB as shown in Figure 18.



The screenshot shows a web interface with a blue header bar containing buttons for '回上一頁' (Back) and '輸出試題' (Export Questions). Below the header is a table with columns for '選取' (Select), '題型' (Problem Type), '題目情境' (Problem Context), and '題目' (Problem Statement). Three rows are visible, each with a checkbox in the '選取' column.

選取	題型	題目情境	題目
<input type="checkbox"/>	周長題型	花園	長方形花園，長10公尺，寬5公尺，請問花園周長多少公尺？
<input type="checkbox"/>	周長題型	公園	正方形的公園一座，邊長50公尺，請問公園周長公尺？
<input type="checkbox"/>	周長題型	正方形	正方形的牛奶盒，邊長20公尺，請問周長多少公尺？

Figure 19 Arithmetic word problems generated by FIB

Teachers can choose and fine-tune arithmetic word problems generated by FIB, and could download these problems through the interface as shown in Figure 19.

5.2 EXPERIMENT

In the evaluation of experiment, we use a questionnaire of system satisfaction, which have been validated by elementary school teachers and education experts. 100 6th-grade math teachers of elementary school in Miao-Li County are involved in evaluating FIB and fill out our Likert-five-point-scale questionnaire to show their opinions about the system. The five-level Likert scale is shown in Table 1.

Degree	Meaning
1	Strongly disagree
2	Disagree
3	Neither agree nor disagree
4	Agree
5	Strongly agree

Table 1 The format of a typical five-level Likert item

The questionnaire contains 21 questions and can be divided into three parts: the first part is to evaluate the satisfaction degree of system usability, the second part is to evaluate the desire for teachers to share their knowledge in FIB, and the third part is to show whether FIB can inspire teachers to generate more various questions. The results of each part are shown in Tables 2~4.

Questionnaire Item List	Average
FIB can assist me to constructing arithmetic word problem.	3.69
FIB can satisfied my requirements to constructing arithmetic word problem.	3.39
FIB can reduce my workload and time to constructing arithmetic word problem.	3.47
I think it is worth using FIB.	3.64
I will keep using FIB.	3.43

I will recommend my colleagues and friends to use FIB.	3.52
I have learned about how to use FIB to assist me to construct arithmetic word problem	3.48
I feel it is easy to operate FIB.	3.27
I feel it is convenient to use FIB to create the content of question.	3.51
I feel it is convenient to use FIB to edit the content of question	3.63
I feel the response of FIB is quick.	3.57
Totally speaking, I feel the FIB interface is friendly.	3.40
Totally speaking, I am satisfied with FIB.	3.54

Table 2 The satisfaction degree of system usability

The results show that most teachers are satisfied with the operation in FIB, and most of them think FIB can assist them to construct arithmetic word problem, so they will keep using FIB to construct arithmetic word problem. Teachers also think it is convenient to create or edit question by using FIB to reduce teachers' workload and time for constructing arithmetic word problem.

Questionnaire Item List	Average
I can find different ideas to construct arithmetic word problem in FIB	3.69
I can enhance my skill of construct arithmetic word problem by observing the questions constructed by other teachers in FIB.	3.78
Via using FIB, I have more creative ideas about constructing arithmetic word problem, and more easily to express my	3.52

knowledge of application problem.	
-----------------------------------	--

Table 3 The results to show whether FIB can inspire teachers to generate more various questions

The results show that most of teachers were inspired to construct arithmetic word problem by using FIB, and most of them think their skill of construct arithmetic word problem can be enhanced by observing other teachers' questions. Moreover, the results show that teachers' idea of constructing question can be enriched by folksonomy-based approach.

Questionnaire Item List	Average
I am willing to share my arithmetic word problem to everyone in FIB.	3.62
I think the question shared by me can promote others' skill of constructing arithmetic word problem.	3.49
The idea of creation and sharing questions by teachers in FIB can also be applied to other domain.	3.74

Table 4 The teachers' desire to share knowledge in FIB

Teachers think the folksonomy-based approach can also be applied to other domain after using FIB. The most important information shown in the results is that teachers are willing to share their knowledge to each other, and the ideas to construct arithmetic word problems can be promoted by this kind of communication.

Some suggestions given by teachers are shown as follows:

- It will be better to let teachers be able to add graph in arithmetic word problem.

- It is a little bit difficult for beginners to construct arithmetic word problem if they are not familiar with the operation of FIB.

5.3 DISCUSSION

The experiment shows that the folksonomy-based approach can reduce the teachers' workload and time for constructing arithmetic word problem, and the teachers' question construction ideas can be inspired others' designed questions. In addition, they think it is easy to create and edit arithmetic word problem by FIB, and they are willing to share knowledge to each other. Overall, FIB is effective to generate arithmetic word problems and share the knowledge of arithmetic word problem.



CHAPTER 6 CONCLUSIONS AND FUTURE WORKS

In this thesis, we use folksonomy-based approach to collecting teachers' knowledge about arithmetic word problem because a large amount of ideas of arithmetic question structures and question scenarios generated by different teachers can be shared and reused in order to reduce teachers' effort to construct arithmetic word problem. In order to facilitate managing and reusing the shared arithmetic word problem knowledge, the proposed **Arithmetic Word Problem frame Hierarchy** is used to represent these problems because the questions of the same question structure are stereotyped and the inheritance property of frames is appropriate to represent the category structure of arithmetic word problems. Besides, the rules embedded in the frame can be used to apply different scenario to the question structure; thus a large amount of arithmetic word problems can be generated. On the other hand, we propose question requirement ontology and scenario slot ontology to share and reuse question requirements and scenario repository to facilitate question structures evolution and various scenario applying. The proposed **Arithmetic Word Problem Grammar** can facilitate the maintenance of arithmetic word problems and their evolutions including **Self-Evolution** and **Collaborative-Evolution**. Self-evolution changes the arithmetic logic to generate different problems. And collaborative-evolution integrates two

question structures into a new question structures. Finally, we apply scenarios shared by teachers to question structures to generate arithmetic word problems.

In the future, systematic ontology consensus constructing approaches can be applied to manage question requirement ontology and scenario slot ontology. Besides, the natural language processing mechanism can be used with the semantic information provided by question structures and scenario slots to improve the description sentences in the generated arithmetic word problems. With these improvements, FIB can be more effective to provide higher quality arithmetic word problems.



REFERENCES

1. Bloom, B.S.E., *Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain*. 1956, New York: Longmans, Green.
2. Polya, G., *How to solve it (2ed.)*. 1957, Princeton: NJ : Princeton University Press.
3. Riley, M.S., Greeno, J. G, & Heller, J. I., *Development of children's probleming-solving ability in arithmetic*. The development of mathematical thinking. 1983, New York: Academic Press. 153-196.
4. Bennett, R.E.I.p., *Automatic item generation : An overview.*: Princeton.
5. Singley, M.K., Anderson,J. R. & Gevins, J. S., *Promoting abstrace strategies in algebra word problem solving.*, in *International Conference on the Learning Sciences : Proceedings of the 1991 Conference*. 1991.
6. Deane, P., Sheehan, Kathleen, *Automatic Item Generation via Frame Semantics : Natural Language Generation of Math Word Problems*, in *Annual Meeting of the National Council on Measurement in Education*. 2003. p. 28.
7. Kumar, A.K.K.A.N., *A PROBLEM GENERATOR TO LEARN EXPRESSION EVALUATION IN CS I, AND ITS EFFECTIVENESS*. *Journal of Computing in Small Colleges*, 2001. 16,4.
8. Zualkernan, I.A., El-Naaj, S. A., Papadopoulos, M., Al-Amoudi, B. K., & Matthews, C. E., *Automatic Generation of Just-in-time Online Assessments from Software Design Models*. *Educational Technology & Society*, 2009. 12 (1),: p. 173–192.
9. Kim, S., J., Clark, A., & Mcdermid, J. A., *The regorous generation of Java mutation using HAZOP.*, in *In Proceedings of the 12 the International Conference on Software and Systems Engineering and Their Applications (ICSSEA '99)*. 1999.

10. Branko Žitko *, S.S., Marko Rosić , Ani Grubišić , *Dynamic test generation over ontology-based knowledge representation in authoring shell*. *Expert Systems with Application* 2009. 36: p. 8185-8196.

