Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)

Dimensionally Guided Synthesis of Mathematical Word Problems

Ke Wang **Zhendong Su**
Department of Computer Science
University of California, Davis
{kbwang, su}@ucdavis.edu

Abstract

Mathematical Word Problems (MWPs) are important for training students' literacy and numeracy skills. Traditionally MWPs have been manually designed; an effective automated MWP generator can significantly benefit education and research. The goal of this work is to efficiently synthesize MWPs that are *authentic* (*i.e.*, similar to manually written problems), *diverse* (*i.e.*, covering a wide range of mathematical tasks), and *configurable* (*i.e.*, varying difficulty levels and solution characteristics). This is challenging because a generated problem needs to both exhibit a well-founded mathematical structure and also an easily understood natural language story. Our key insight is to leverage the important role that dimensional units play in MWPs, both textually and symbolically. We first synthesize a dimensionally consistent equation and then compose the natural language story via a bottom-up traversal of the equation tree. We have realized our technique and extensively evaluated its efficiency and effectiveness. Results show that the system can generate hundreds of valid problems per second with varying levels of difficulty. More importantly, we show, via a user study with 30 students from a local middle school, that the generated problems are statistically indistinguishable from actual textbook problems for practice and examination.

1 Introduction

A *Mathematical Word Problem* (MWP) comprises several sentences to communicate a short narrative, which includes some numerical information and asks for some unknown quantities to be calculated. MWPs are interesting exercises because they challenge a student from multiple perspectives: (1) reading and understanding the problem (literacy skills); (2) modeling the narrative via symbolic equations (analytical skills); and (3) obtaining an answer by solving the equations (numerical skills). Figure 1 depicts an example MWP.

Given their significant role in education, our goal is to automatically construct well-designed MWPs. This automation has some important benefits. First, it frees teachers from manually designing MWPs and provides students with a rich source of practice problems. Second, it benefits AI research that develops algorithms for solving MWPs by providing a large supply of diverse benchmark problems.

Example 1. ***A Mathematical Word Problem****: Joseph has been running at a constant speed for 75 minutes. Eric completed 50 laps around a 400-meters track. Joseph and Eric have traveled the same distance. What's Joseph's speed measured in meters per hour?*
Equation*:* $X \times 1.25 = 400 \times 50$
Solution*:* $X = 16,000$

Figure 1: An example MWP with its equation and solution.

However, automatic generation of MWPs is challenging because an MWP needs to have a well-formed mathematical structure and a clear natural language story. Our insight is to utilize *dimensional units* to tackle the challenges, both textually and mathematically. Indeed, our careful study of popular MWP corpuses shows that most MWPs concern physical quantities expressed in their corresponding dimensional units. In the literature, there also exist many discussions on the significant role that dimensional units play in MWPs [Erickson, 1999; Adam, 2006; Singley and Bennett, 2002]. Relying on dimensional units, our generation procedure operates in two main steps. First, it synthesizes an equation where each quantity (either a variable or constant) is assigned a dimensional unit such that the equation is dimensionally consistent (*i.e.* both sides have the same dimension). In the second step, it translates the generated equation into a multi-sentence narrative; the key is to leverage the (binary) expression tree representation of the synthesized equation. In particular, the expression tree both provides the skeleton of the story and also allows the full narrative to be constructed recursively.

We have implemented our MWP generation algorithm and extensively evaluated it. Our evaluation focuses on two aspects: (1) performance of the generation procedure, and (2) authenticity of the generated problems (in terms of their conformity to actual textbook problems). Results show that our system is efficient, taking one second to generate hundreds of MWPs, and more importantly the generated problems closely resemble textbook problems for practice and examination.

The rest of the paper is structured as follows. We first detail our MWP generation methodology (Section 2). Then, Sec-

Operator	First Operand	Second Operand	Result	Interpretation
Addition	$Q_1(du)$	$Q_2(du)$	$Q_3(du)$	Q_3 is the addition of Q_1 and Q_2
Addition	$Q_1(du)$	$\Delta(du)$	$Q_2(du)$	Q_2 is greater than Q_1 by Δ
Addition	$\Delta_1(du)$	$\Delta_2(du)$	$\Delta_3(du)$	Δ_3 is the combined offset of Δ_1 and Δ_2
Addition	R_1	R_2	R_3	Q_3 is the combined ratio of R_1 and R_2
Subtraction	$Q_1(du)$	$Q_2(du)$	$Q_3(du)$	Q_3 is the difference between Q_1 and Q_2
Subtraction	$Q_1(du)$	$Q_2(du)$	$\Delta(du)$	Q_1 is greater than Q_2 by Δ
Subtraction	$Q_1(du)$	$\Delta(du)$	$Q_2(du)$	Q_1 is greater than Q_2 by Δ
Subtraction	$\Delta_1(du)$	$\Delta_2(du)$	$\Delta_3(du)$	Δ_3 is the offset equaling Δ_1 minus Δ_2
Subtraction	R_1	R_2	R_3	Q_3 is the ratio equaling R_1 minus R_2
Multiplication	$Q_1(du_1)$	$Q_2(du_2)$	$Q_3(du_1 * du_2)$	Q_3 is the multiplication of Q_1 and Q_2
Multiplication	$Q_1(du_1)$	R	$Q_2(du_1)$	Q_2 is R times as much as Q_1
Multiplication	R_1	R_2	R_3	Q_3 is the product ratio of R_1 and R_2
Division	$Q_1(du_1)$	$Q_2(du_2)$	$Q_3(du_1/du_2)$	Q_3 is the division of Q_1 and Q_2
Division	$Q_1(du)$	$Q_2(du)$	R	Q_1 is R times as much as Q_2
Division	$Q_1(du)$	R	$Q_2(du)$	Q_1 is R times as much as Q_2

Table 1: Semantics of binary operands *w.r.t.* dimensional units.

tion 3 presents the evaluation setup and results. Finally, we survey related work (Section 4) and conclude (Section 5).

2 Methodology

This section details the design of our MWP generation algorithm. It first presents a formalization of MWPs. Then it describes the two key modules for MWP generation: the equation generator and the narrative generator. Throughout this section, we will use the example MWP in Figure 1 to illustrate the operation of our algorithm.

2.1 Formalization

We define an MWP both syntactically and semantically. First syntactically, an MWP refers to its textual representation, which we define as follows.

Definition 1 (**Syntax of MWPs**). *An MWP is a set of sentences T containing two kinds of numerical information: the set of given values, denoted by C, and the set of unknown values, denoted by $\mathcal{X}$. The former comes directly from T, while the latter is to be determined by T and C.*

Semantically, an MWP refers to its corresponding symbolic representation, which we define as follows.

Definition 2 (**Semantics of MWPs**). *An MWP is a set of equations (in this work, we focus on linear equations) E, whose variables are denoted by $V \subseteq \mathcal{X}$ and the parameters denoted by $P \subseteq C$.*

Next, we formalize the notion of *valid MWPs*. A valid MWP clearly should have one unique solution to E. In addition, each equation $e \in E$ should be dimensionally consistent, *i.e.*, its two sides should have the same dimensional unit. The following properties model this intuition.

Focusing on only the semantic definition E of an MWP regardless of its dimensional units assignment, we specify the first condition, *correctness*, for an MWP to be well-formed.

Definition 3 (**Correctness**). *An MWP E is correct iff E is (1) independent,* i.e., *$\nexists e \in E.E \setminus \{e\} \rightarrow e$, where $\rightarrow$ denotes algebraic derivation, (2) consistent,* i.e., *E is satisfiable, and (3) exactly determined,* i.e., *$|E| = |V|$.*

Under the assignment of dimensional units, we specify the other condition, *well-typedness*.

Definition 4 (**Well-Typedness**). *An MWP E is well-typed iff for each equation $e \in E$, the expressions on both sides of e evaluate to the same dimensional unit.*

Finally, we can define the validity of an MWP.

Definition 5 (**Well-Formedness**). *An MWP E is well-formed iff it is both correct and well-typed.*

2.2 Equation Generator

The purpose of the equation generator is to (1) synthesize an equation, and (2) assign a dimensional unit to each quantity in the equation such that well-typedness is maintained.

Before giving the details of generating well-typed equations, we define the semantics of binary arithmetic operations when dimensional units are concerned.

Semantics of Binary Operations As Table 1 shows, we consider three kinds of values: *quantity* (denoted by Q_{1-3}), *offset* (denoted by Δ_{1-3}), and *ratio* (denoted by R_{1-3}). This classification of the values and dimensional units facilitates narrative generation (to be discussed in Section 2.3).

Equation Generation Procedure Our procedure consists of four steps: (1) *semantic instantiation*: instantiate the operational rules listed in Table 1 with dimensional units; (2) *seed equation synthesis*: synthesize a random equation whose two sides have a single variable each that is of the same dimensional unit; (3) *variable unrolling*: randomly select a variable in the equation and substitute it with two fresh variables *w.r.t.* the instantiated rules of the dimensional units; and (4) *repeat/stop*: repeat step (3) or terminate as needed (*e.g.*, when having reached a desired complexity of arithmetic operations).

Example 2. *Below illustrates how to generate the underlying equation for our running example:*

(1) Synthesize a seed equation:

$$X(du) = Y(du)$$

(2) Assign *du* the dimensional unit *meters*:
$$X(\textit{meters}) = Y(\textit{meters})$$
(3) Unroll X via $X_1(\textit{meters/hour}) \times X_2(\textit{hour}) = X(\textit{meters})$:
$$X_1(\textit{meters/hour}) \times X_2(\textit{hour}) = Y(\textit{meters})$$
(4) Unroll Y via $Y_1(\textit{meters/lap}) \times Y_2(\textit{lap}) = Y(\textit{meters})$:
$$X_1(\textit{meters/hour}) \times X_2(\textit{hour}) = Y_1(\textit{meters/lap}) \times Y_2(\textit{lap})$$

2.3 Narrative Generator

Given the synthesized equation, the narrative generator translates the equation into an MWP in four steps:

Step 1: Apply the binary expression tree First, the synthesized equation is converted to a binary expression tree (BET). We represent the equation using the postfix notation [Burks *et al.*, 1954], where "=", and variables X_1, X_2, Y_1 and Y_2 are respectively the root and leaf nodes of the tree. The dimensional units of the intermediate nodes can be retrieved from their corresponding variables in the equation generation process.

Step 2: Supplement the keywords In general, the high-level strategy we adopted for generating the narrative is to (1) traverse each Atomic Expression Tree (AET) within the given BET; (2) produce a sub-story for each AET independently, and (3) concatenate the sub-stories into the complete narrative. An AET, as its name suggests, refers to the smallest tree units that can be segmented from the entire BET — each AET consists of a parent node (the operator) and two child nodes (the variables) in the synthesized equation.

The purpose of assigning keywords to nodes in a BET is to help produce a sub-story from each AET. Specifically, given the BET depicted in Figure 2a, there are three AETs — α, β and γ — as annotated by their respective bounding boxes. It is clear that, for each of the AETs, the generated dimensional units alone are insufficient to express its sub-story. Thus, the keywords assigned to each node in an AET are designed to complement the existing information in the AET and make its semantics complete.

This is where the benefits of incorporating dimensional units become evident. Because each dimensional unit is in one-to-one correspondence with the type of quantity it designates, *e.g. mile/hour* specifies speed, *mile* specifies distance, *minutes* specifies time, *etc.* Consequently, the dimensional units of all three nodes within an AET collectively establish the skeleton of the corresponding sub-story (we defer the detailed discussion to Step 4).

For example in the α-AET, *mile/hour*, *mile* and *minutes* constitute the core description of a running activity. A simple approach to complete its sub-story is to add a subject, *i.e.*, whoever took the activity of running. Additional descriptive words may be incorporated to make the narrative more interesting.

Formally, the sequence of keywords assignments for each AET is similar to a postorder tree traversal. An AET will not be processed unless both of its child nodes are either leaf nodes of the entire BET or intermediate nodes that have already been assigned with keywords. For the example BET, its three AETs will be traversed in the following order: α, β and γ. The assignment of each type of AET is directed by a predefined rule according to the instantiation of the corresponding operational rule listed in Table 1. For example, within the α-AET, it is meaningless to multiply *John's* running speed by *David's* running time — the subjects of *mile/hour* and *hour* should be consistent.

Note that the keywords are not randomly assigned, but rather selected from the existing vocabulary classified by their types. The vocabulary for keywords of subjects, in particular, is further split into sub-categories such as people's names, animals, fictional characters, *etc.*, which we will explain in detail in Step 3. Figure 2b shows the output of the current step in generating the running example.

Step 3: Perform auxiliary tasks An additional task in generating a sub-story for an AET concerns numerical values, *i.e.*, known values that serve as the conditions or a designated unknown value that serves as the question of an MWP. In other words, the goal of this step is to assign values to all the leaf nodes, except one that is left for the student to answer.

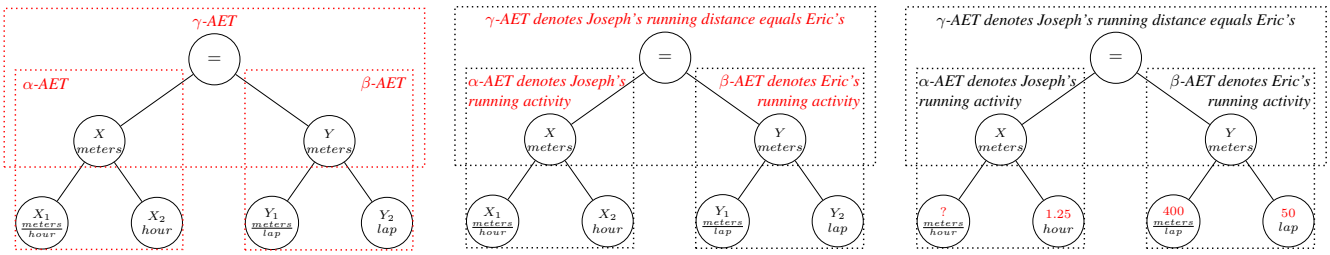
One challenge is how to generate values that fit the context (*e.g.*, it is not sensible for a person to run a hundred miles per hour). Our approach is to associate a valid range of values (*w.r.t.* dimensional unit) for each sub-category where a subject keyword is chosen. The intuition is that each sub-category aggregates elements having common characteristics.

Additional constraints are synthesized to restrict all variables to be positive, the minuend to be greater than the subtrahend, *etc.*, and prefer integers over decimals or fractions. Finally, we employ Z3 [De Moura and Bjørner, 2008], an off-the-shelf state-of-the-art SMT solver, to resolve the synthesized constraints and discover appropriate values for each of the variables. Among all assigned variables, one will be kept as unknown to generate an MWP's question. Figure 2c shows the outcome of this step in generating the running example.

Step 4: Put everything together At this point, keywords have been assigned to each node and numerical information has been distributed to leaf nodes of a BET. We can now generate each AET's sub-story and synthesize the entire narrative.

As briefly introduced in Step 2, the dimensional units for all three nodes within an AET suffice to comprise a skeleton of the sentences expressing its sub-story. The skeleton can be viewed as a verbal template containing slots to be filled with items, which are keywords and values in this case. In order to increase the diversity of the story expressions, we design multiple verbal templates *w.r.t.* each type of AET, *i.e.*, the combination of dimensional units of all three nodes.

To decide which template to instantiate at runtime, we consider the combination of keywords assigned to each node within an AET. In particular, each blank in a template needs to match the keywords assigned to each of the respective nodes within an AET. If there are multiple suitable candidates, one will be randomly chosen among them. Utilizing the techniques for generating sub-stories for AETs, our system traverses the entire BET in the order that it assigns keywords and finally concatenates the sub-stories into a complete story.



(a) 1st: Generate the BET from equation. (b) 2nd: Assign the BET with keywords. (c) 3rd: Fill the BET with values.

Figure 2: The first three steps of narrative generation.

Example 3 outlines the concrete steps that the system takes to produce the narrative of the MWP shown in Example 1.

Example 3. *We illustrate how to first generate a sub-story for each AET using templates (where ⇓ denotes instantiation by filling the slots with keywords, values, or both), and then how to concatenate the sub-stories into an entire narrative for our running example.*

(1) Generate the sub-story of α-AET using the template:

_____ has been running at a constant speed for _____ hours.

⇓

Joseph has been running at a constant speed for 75 minutes.

(2) Generate the sub-story of β-AET using the template:

_____ completed _____ laps around a _____-meters track.

⇓

Eric completed 50 laps around a 400-meters track.

(3) Generate the sub-story of γ-AET using the template:

_____ and _____ have traveled the same distance.

⇓

Joseph and Eric have traveled the same distance.

(4) Generate the question sentence using the template:

What's _____'s speed measured in meters per hour?

⇓

What's Joseph's speed measured in meters per hour?

(5) Concatenate the sub-stories in the order of α, β and γ.

2.4 Analysis and Discussions

In this section, we first show that the generated problems are well-formed. Then, we discuss how to control the difficulty levels of the generated MWPs, the capability of our generation procedure, and the configurability of the generated problems.

Well-Formedness of the Generated Problems According to Definition 5, an MWP is well-formed iff it is correct and well-typed. The generated problems are correct because (1) each problem has a single equation, which is clearly independent; (2) each has a solution guaranteed by the SMT solver and thus consistent; and (3) the number of equations and the number of variables are both one, thus the problem is exactly determined. The generated MWPs are also well-typed because given the well-typed seed equation, whenever a variable is unrolled, well-typedness is preserved according to the operational rules listed in Table 1. Because the generated MWPs are both correct and well-typed, they are well-formed.

Remark. It is possible for dimensional units to have multiple story interpretations. We cope with this by associating each subtree with a sub-story theme. Our current implementation supports example themes such as people running, swimming and climbing, and vehicles moving between an origin and destination for the same set of subtrees in Example 1. During the generation process, our system ensures that all subtrees adopt the same theme, thus all generated sub-stories are compatible.

Also note that equations are randomly generated, but their associated dimensional units are not. Rather, they are predefined. We do not use nonsensical dimensional units in problem generation. It is also possible for dimensional units to have multiple interpretations. Our system has full control of the generation process. Whenever the variable unrolling step introduces new dimensional units, the system controls the precise interpretation of each, even when a unit may have multiple interpretations.

Control of Difficulty Levels A simple approach for adjusting the difficulty levels is to vary the complexity of the generated equations, *i.e.*, the more arithmetic operations, the more involved the generated MWP. We also employ another method to increase the difficulty level of an MWP by manipulating how to choose the unknown quantity for students to calculate. Traditionally, the unknown quantity is selected among the leaf nodes of a BET, and the reason for this phenomenon is to make every other nodes necessary in the process of solving the MWP. In other words, if the unknown quantity is not selected from the leaf nodes, some nodes in the BET will become irrelevant. Take the BET discussed in Section 2.3 for example. If we make the node X as the unknown quantity instead of one of X_1, X_2, Y_1 and Y_2, the sub-story generated from α-AET will become redundant in the complete narrative, therefore creating a distraction that may further challenge students. We will show in Section 3

units_of(X_1,X_2,Y_1,Y_2) = \$/Lb, Lbs, \$/Gal, Gals	*units_of(X_1,X_2,Y_1,Y_2) = L/hr, hr, L/hr, hr*	*units_of(X_1,X_2,Y_1,Y_2) = J/s, s, J/°C, °C*	*units_of(X_1,X_2,Y_1,Y_2) = m, m, m, m*	*units_of(X_1,X_2,Y_1,Y_2) = m/hr, hr, m, N/A*
Rachel bought 4 pounds of **shrimp** at the price of 8\$ per pound in a supermarket. **Kyle** pumped 10 Gallons of gas into his car at a gas station. **Rachel** and **Kyle** happened to spend the same amount of money. What is the price of gas at the gas station where **Kyle** pumped gas?	**A water pump** can fill up a **tank** at a rate of 6000 liters per hour for 10 hours. **Another pump** can draw all the water out from **another tank** at full capacity for 8 hours. The **two tanks** have the equivalent capacity. How much water can the **second pump** draw out per second?	**A heater** which can transfer 60J heat per second is set to operate for 5 seconds. **A kettle** boils **water** (heat capacity: 4179 J/°C for 1kg) within minutes. **The heater** and **the kettle** generate the same amount of energy during the respective time span. What's the temperature increase of the **water**?	A square-shaped **dining table** has a side of 6 meters. A rectangular **rug** has a length of 9 meters. The **dinner table** covers the same area as the **rug**. What's the width of the **rug**?	**Joseph** has been running at a constant speed for 75 minutes. **Michael** who has finished 5000 meters only ran a quarter of the distance **Alexander** has done. **Joseph** and **Alexander** have traveled the same distance. What's **Joseph's** speed measured in meters per hour?

Table 2: A series of independently generated MWPs, all of which are synthesized from the same equation in Example 1. For each problem narrative, we highlight in bold the keywords of each node in the AETs.

that error rates for problems with more arithmetic operations and distractions are higher than those without.

Capability of Our MWP Generation Procedure First, the generation power is determined by the capability of our equation generator. Next for each generated equation, we discuss the universe of the corresponding synthesized MWPs. Considering an equation in the representation of a binary expression tree, clearly for a random equation generation algorithm, the sample space of the equations that can be generated is exponential in the number of the arithmetic operators. Furthermore, the universe of MWPs that can be generated from an equation is determined by two factors: the assignment of the dimensional units and the selection of the arithmetic operation type.

For example, in Table 2 the first four columns show the MWPs that are generated by adopting distinct sets of dimensional units on the variables in the same example equation. Although the MWP in the last column also has different assignment of the dimensional units from the example MWP, more importantly it chooses a separate instantiated operational rule to unroll variable Y, in particular the second row of the "Multiplication" section in Table 1, leading to Y_2 being a ratio.

Now, for a fixed arrangement on the dimensional units or arithmetic operational types of a generated equation, we discuss the space of the MWPs. Assume the generated equation has N AETs, each of which has the set of pre-defined templates TEM_N, the set of MWPs that can be generated is $S = \{E \mid X_1 \in TEM_1, X_2 \in TEM_2 ... X_N \in TEM_N \wedge E = X_1 \cap X_2 \cap .... \cap X_N\}$ (excluding the influence caused by the different keyword assignments). In addition, assume there are M ways of arrangements on a generated equation, then the universe of MWPs that can be generated from one equation equals to $M \times |S|$ approximately. According to the discussion on the capability of the equation generator, the total number of MWPs that can be synthesized from our generation algorithm is $4^{N-1} \times M \times |S|$. Take the four-step equation MWPs for example, assigning the value of M and $|TEM_N|$ to be around ten will result in the total number of MWPs being in the range of millions.

Configurability of the Generated Problems The adjustable difficulty levels of the generated problems together with the capability of the generation procedure creates a platform that can enable personalized workflow for each student. If a student solves a problem correctly, the student may be presented with a problem that is more difficult than the last problem. Or if a student fails to solve a problem, the student may be presented with simpler problems to reinforce the core concepts. Whenever the student wants to challenge a harder problem again, she may be presented with similar problems to the one she had trouble with earlier (by varying the assignment of dimensional units or the selection of the instantiated operational rule on the same equation) to reassess her understanding.

3 Evaluation

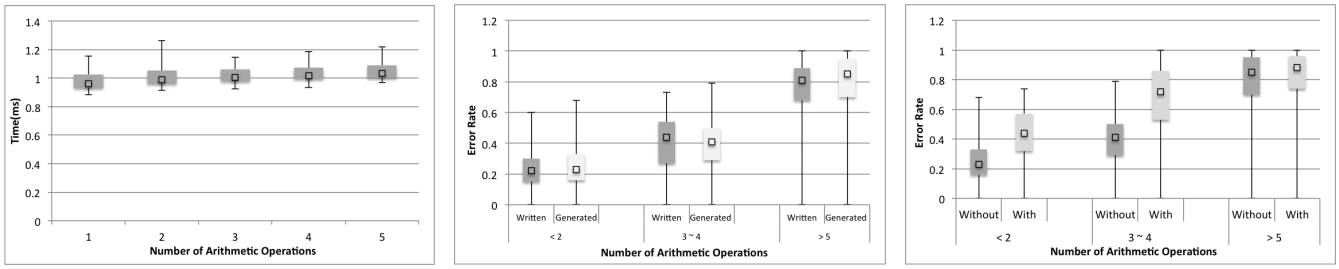
This section presents two experiments to evaluate the performance of our generation algorithm and the similarity of the synthesized MWPs versus actual textbook MWPs.

We have adopted 43 dimensional units that measure speed, price, temperature, area, volume, work rate, *etc.*

3.1 Performance

First, we focus on evaluating the performance of our MWP generation algorithm. We classify the synthesized MWPs into five categories according to the number of primitive arithmetic operations involved in the equation. For each category, we synthesize 100 problems and measure the time taken to synthesize each. We have conducted our experiments on a desktop with a 4th generation Intel Core i7-4770 processor and 16GB RAM, running Ubuntu 12.04 LTS.

Figure 3a shows the measurement results as a boxplot. We adopt the conventional style of plot where the bottom and top of the box are the first and third quartiles, and the marker inside the box denotes the mean. The two ends of the



(a) Performance with varying difficulty. (b) Textbook MWPs vs. generated MWPs. (c) With and without distractions.

Figure 3: Evaluation results.

whiskers represent the minimum and maximum. As shown in Figure 3a, the time taken to synthesize problems increases slightly as the number of the arithmetic steps increases. On average, our algorithm takes around one millisecond to synthesize an MWP.

3.2 Problem Authenticity

This is the more important aspect of our evaluation. We have conducted a pilot study to carefully assess the authenticity of our synthesized MWPs from the actual textbook MWPs. By authenticity, we mean that (1) the generated MWPs should be indistinguishable statistically from the textbook MWPs, and (2) problems that share the same underlying structure (*i.e.*, solution length) should be similar in terms of their difficulty level from the students' perspective.

3.2.1 Study Design

We invited 30 seventh grade students from a local middle school to participate in the study. We selected 24 textbook problems from the Singapore Math curriculums [Publications, 2009a; 2009b; 2009c] and generated the same number of MWPs with an equivalent distribution of complexity. During the study, participants were given a single test containing all 48 problems, mixed randomly. For each problem, participants were not only required to solve but also to guess if it was manually written (*i.e.* textbook MWP) or automatically synthesized (*i.e.* generated MWP). The total duration for this pilot study was one and half hour, occupying two class periods.

3.2.2 Study Results

In this section, we present the details of our results from the following three aspects.

Verification of Statistical Indistinguishability First, for each participant, we gather the numbers of problems that are labeled real (*i.e.* actual textbook problems) and generated (*i.e.* generated problems) conditions respectively. Then we run a paired *t* test on these two numbers across all the 30 participants, and found that there was no significant difference in the scores for real (M=11.13, SD=4.6) and generated (M=11.13, SD=3.7) conditions [$t(29) = 0, p = 1$]. These results suggest that the source of the problems (real versus generated) does not have an effect on human perception of provenance.

To further strengthen our analysis, we have aggregated the data into contingency tables, one for each participant, with rows corresponding to the classification made by each participant (written or generated), and columns to real and generated conditions. Next, to test whether their classifications are independent from the conditions under which the problems are produced, we applied the χ^2 test of independence. Finally, to aggregate the results of the χ^2 tests (one per participant), we applied the Stouffer test using the weighted Z-score method [Whitlock, 2005]. This allows us to lift the results of the individual χ^2 tests to the group level. The Stouffer test statistic Z was calculated as 1.232 and the corresponding p-value was 0.108. These imply that taken together the data indicate that the generated MWPs are statistically indistinguishable from actual textbook MWPs.

Comparison of Error Rates In the following discussion, we split the two problem sets into three categories according to the number of arithmetic steps in an equation (*i.e.* <3, $3 \sim 4$, and >4), and report the results for each category separately. To measure the similarity of the two problem sets within each of the categories, we run a two one-sided test [Schuirmann, 1987] for equivalence on the participants' error rates. An important decision is how to define the zone of "clinical indifference", *i.e.* a range of effects that can be considered clinically trivial. For this purpose, we randomly partitioned the textbook MWPs within each category that a participant has been tested on into two halves and summarized the participants' error rates for each half. Next, we apply the two sided test to compute the threshold error margin — anything less than this value would make the two partitions dissimilar. However, the two randomly partitioned problem sets from textbook MWPs must be similar, so we can use the threshold error margin to compute the similarity between the synthesized and the textbook MWPs.

Figure 3b shows the error rates for the two problem sets within each category. The error margins are computed to be 0.087, 0.091 and 0.118 from the two random partitions on the textbook problem set within each category respectively. Then, a two-sided test was used to compare each participant's error rates across the two sets of problems within each of the categories. These tests show that the error rate conforms significantly across the two sets of problems [$t(58) = -2.672, p = 0.005$] and [$t(58) =$

$1.673, p = 0.049]$, $[t(58) = -2.234, p = 0.015]$ and $[t(58) = 1.792, p = 0.039]$, $[t(58) = -2.020, p = 0.024]$ and $[t(58) = 1.704, p = 0.047]$, indicating that the overall difficulty of the synthesized MWPs is similar to that of the textbook MWPs.

Controlled Measure of Problem Difficulty Apart from the error rates reported from Figures 3b, a paired *t* test was used to compare each participant's error rate across only the generated problems sets of "fewer than three" versus "three to four" arithmetic operations, and "three to four" and "greater than four" arithmetic operations. As expected, the error rate for MWPs of three to four arithmetic operations was significantly higher than that of fewer than three arithmetic operations $[t(58) = 4.223, p < 0.001]$, indicating that the problems of three to four arithmetic operations were more difficult. Similarly, the error rate for the problems of greater than four arithmetic operations was significantly higher than the problems of three to four arithmetic operations $[t(58) = 6.171, p < 0.001]$, indicating that the problems of greater than four arithmetic operations were more difficult.

We have conducted another user study where the same participants were asked to complete another separate test set consisting of 24 synthesized problems. Those problems are randomly sampled from a corpus, where each problem is also synthesized with redundant information in its story line. The purpose of this study is to assess the impact of distractions may have on the generated MWPs. Figures 3c shows that problems created with distractions are indeed more difficult than those created without distractions, using the same number of primitive arithmetic expressions.

A paired *t* test was used to compare each participant's error rates across the two problem sets with or without distractions for the same number of arithmetic operations. For problems with fewer than three versus three to four arithmetic operations, the error rate for problems mixed with distractions was significantly higher than those without $[t(58) = 4.063, p < 0.001]$ and $[t(58) = 5.319, p < 0.001]$, indicating that problems with distractions were more difficult than those without distractions. As for problems of more than four arithmetic operations, the difference on error rates was insignificant $[t(58) = 0.789, p = 0.433]$ because the participants' error rates on MWPs with more than four arithmetic operations were already very high, leaving little room to perceive the increased difficulty.

4 Related Work

This section surveys closely related work, which we group into two categories: (1) automatic MWP generation and (2) automatic MWP solving.

Automatic MWP Generation Singley *et al.* [Singley and Bennett, 2002] developed the pioneer project on automatic item generation of MWPs. Their approach can be viewed as a simple template-based natural language generation system. Deane *et al.* [Deane and Sheehan, 2003] proposed more complex methods based on Frame Semantics [Fillmore, 1976]. More recently, Polozov *et al.* [Polozov *et al.*, 2015] considered the same problem from the personalization angle — they focused on generating problems with engaging story lines.

All the above efforts focus primarily on natural language story generation. In contrast, as we have stated earlier, our key insight is to leverage the important role that dimensional units play in MWPs. Indeed, our work introduces an effective methodology that leverages dimensional units for generating authentic, diverse and configurable MWPs to aid education and research. Our study with 30 students clearly demonstrates the effectiveness of the presented approach.

Automatic MWP Solving Most previous efforts [Liguda and Pfeiffer, 2012; Briars and Larkin, 1984; Fletcher, 1985; Dellarosa, 1986; Bakman, 2007; Yuhui *et al.*, 2010] on automatic MWP solving adopted a symbolic approach. The basic idea was to parse an MWP's natural language text by applying pattern matching rules based on predefined heuristics. Recently, statistical learning methods were proposed in [Hosseini *et al.*, 2014; Kushman *et al.*, 2014]. Hosseini *et al.* [Hosseini *et al.*, 2014] considered solving homogeneous addition and subtraction problems with verb categorization learned from training data, while Kushman *et al.* [Kushman *et al.*, 2014] used learning to solve a broader range of word problems from the equations or simply final answers for the training problems. In comparison, our work considers the orthogonal problem of automatic MWP generation. Besides its educational benefits, our work also facilitates AI research on MWP solving.

5 Conclusion

This paper has introduced an approach to effectively synthesize MWPs. Our evaluation results demonstrate the performance of our synthesis algorithm and strong resemblance of the synthesized MWPs to actual textbook problems. We expect that this work will benefit both general education/training and research on automated MWP solving. Our immediate future work is to reach out to potential user groups that can benefit from our system. Other interesting directions include the support of SI unit conversions and the application of NLP techniques to enrich the generated sentence patterns.

Acknowledgments

We appreciate the anonymous IJCAI reviewers' useful feedback on an earlier version of this paper. This work was supported in part by a Chancellor's Fellowship at the University of California, Davis.

References

[Adam, 2006] John A Adam. *Mathematics in nature: Modeling patterns in the natural world*. Princeton University Press, 2006.

[Bakman, 2007] Yefim Bakman. Robust understanding of word problems with extraneous information. *arXiv preprint math/0701393*, 2007.

[Briars and Larkin, 1984] Diane J Briars and Jill H Larkin. An integrated model of skill in solving elementary word problems. *Cognition and instruction*, 1(3):245–296, 1984.

[Burks *et al.*, 1954] Arthur W Burks, Don W Warren, and Jesse B Wright. An analysis of a logical machine using parenthesis-free notation. *Mathematical tables and other aids to computation*, pages 53–57, 1954.

[De Moura and Bjørner, 2008] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[Deane and Sheehan, 2003] Paul Deane and Kathleen Sheehan. Automatic item generation via frame semantics: Natural language generation of math word problems. 2003.

[Dellarosa, 1986] Denise Dellarosa. A computer simulation of childrens arithmetic word-problem solving. *Behavior Research Methods, Instruments, & Computers*, 18(2):147–154, 1986.

[Erickson, 1999] Ranel Einar Erickson. Method of teaching the formulation of mathematical word problems, May 11 1999. US Patent 5,902,114.

[Fillmore, 1976] Charles J Fillmore. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech*, volume 280, pages 20–32, 1976.

[Fletcher, 1985] Charles R Fletcher. Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods, Instruments, & Computers*, 17(5):565–571, 1985.

[Hosseini *et al.*, 2014] Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, 2014.

[Kushman *et al.*, 2014] Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. Learning to automatically solve algebra word problems. *ACL (1)*, pages 271–281, 2014.

[Liguda and Pfeiffer, 2012] Christian Liguda and Thies Pfeiffer. Modeling math word problems with augmented semantic networks. In *Natural Language Processing and Information Systems*, pages 247–252. Springer, 2012.

[Polozov *et al.*, 2015] Oleksandr Polozov, Eleanor O'Rourke, Adam M Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popovic. Personalized mathematical word problem generation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, page 381388. AAAI Press, 2015.

[Publications, 2009a] Frank Schaffer Publications. *Singapore Math 70 Must-Know Word Problems Level 6, Grade 7*. Singapore Math 70 Must Know Word Problems. Carson-Dellosa Publishing, LLC, 2009.

[Publications, 2009b] Frank Schaffer Publications. *Singapore Math Practice, Level 1A Grade 2*. Singapore Math Practice. Carson-Dellosa Publishing, LLC, 2009.

[Publications, 2009c] Frank Schaffer Publications. *Singapore Math Practice Level 3B, Grade 4*. Singapore Math Practice. Carson-Dellosa Publishing, LLC, 2009.

[Schuirmann, 1987] Donald J Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of Pharmacokinetics and Biopharmaceutics*, 15(6):657–680, 1987.

[Singley and Bennett, 2002] Mark K Singley and Randy E Bennett. Item generation and beyond: Applications of schema theory to mathematics assessment. In *Generating Items for Cognitive Tests: Theory and Practice., Nov, 1998, Educational Testing Service, Princeton, NJ, US; This chapter was presented at the aforementioned conference.* Lawrence Erlbaum Associates Publishers, 2002.

[Whitlock, 2005] MC Whitlock. Combining probability from independent tests: The weighted Z-method is superior to Fisher's approach. *Journal of Evolutionary Biology*, 18(5):1368–1373, 2005.

[Yuhui *et al.*, 2010] Ma Yuhui, Zhou Ying, Cui Guangzuo, Ren Yun, and Huang Ronghuai. Frame-based calculus of solving arithmetic multi-step addition and subtraction word problems. In *Second International Workshop on Education Technology and Computer Science (ETCS)*, volume 2, pages 476–479. IEEE, 2010.