Introduction
Word problems are notoriously difficult for children and adults alike. Many people find them much more difficult than the equivalent symbolic representations (see comparison on the right). This phenomenon is caused by language understanding, conceptual knowledge, discourse comprehension, and other aspects required to build a mental representation of a word problem. Moreover, many students find word problems unconnected to their lives and artificial. This perception can be altered with problem personalization: individual interest raises understanding and engagement in a problem solving process (which, in turn, increases students’ performance). However, personalizing word problems in a textbook is expensive, and unreasonable burden on teachers (who would need to be aware of every student’s interests).

Logic generation
Our technique uses answer set programming (ASP) in steps:
1. Equation Generation
a. Guess an equation tree $E$.
b. Deduce whether math requirements are covered by $E$.
c. Forbid invalid trees that do not cover the requirements.

2. Plot Generation
Generates a logical graph $g$, which represents a word problem that models the equation $E$.

Definition. A logical graph $g$ is a tuple $(F, C)$, where:
- $E$: a set of entities. Every entity $e \in E$ has a corresponding ontology type $t$. $F$ forms a hierarchy tree, denoted $F \leq t$.
- $F$: a set of facts. Every fact $f \in F$ has a corresponding ontology relation $R \subseteq relation(F)$. Every relation $R$ has a set of named arguments $arg(C)$. For each fact $f \in F$, every argument $a \in arg$ is associated with an entity $e \in E$ such that $e_a \in e$. For example, the Relation for $e_a \in e$ is shown as $f = R(e_1, e_2, e_3)$.
- $C$: a set of temporal (t) or causal (c) fact connectives. A connective $C$ is a tuple $f \times f$, where tag $t \in \{t, c\}$.

Example. $F$:
- $own(a, c_1), own(c_1, c_2), slays(c_2), Acquires(c_2), TotalCount(c_2, c_3), own(c_3, unk_{1}), unk_{1}$

Natural language generation
4. Sentence ordering
a. Convert each fact $f \in F$ to a sentence using a database of primitive templates.

b. Temporal and connective connectives $C$ define a partial ordering between sentences $B$ Build a linear narrative.

Evaluation
Goal: evaluate generation techniques by assessing comprehensibility and solubility of the word problems’ content.

Study design:
Sample 25 generated word problems with sufficient variability.
Match with 25 equivalent Math word problems.
Conduct 2 Amazon Mechanical Turk studies (1000 subjects each):
A. Evaluate the word problem text with respect to given questions on a forced-choice Likert scale [1−5].
Q1: How comprehensible is the problem? How well did you understand the problem?
Q2: How logical and natural is the sentence order?
Q3: When the problem refers to an actor (e.g. a pronoun or a name) which is being mentioned?
Q4: Do the numbers in the problem fit its story (e.g. it would not make sense for a knight to be 5 years old?)
B. Solve the word problem. Correctness and solving time were recorded.

Findings
✓ Generated problems are rated equally or slightly less comprehensible than the textbook problems ($p = 1.9352, p < 0.001, V = 0.44$).
✓ Generated problems are generally comprehensible ($p = 0.35 - 0.55$).
✓ Solvability of generated problems is indistinguishable from textbook. After removing 4 spiders with unclear language.

References