World Models for Math Story Problems

Author(s):
Opedal, Andreas; Stoehr, Niklas; Saparov, Abulhair; Sachan, Mrinmaya

Publication date:
2023-07

Permanent link:
https://doi.org/10.3929/ethz-b-000653696

Originally published in:
https://doi.org/10.18653/v1/2023.findings-acl.579

World Models for Math Story Problems

Andreas Opedal[⊗,±] **Niklas Stoehr**[⊗] **Abulhair Saparov**[÷] **Mrinmaya Sachan**[⊗]
[⊗]ETH Zürich [÷]New York University
[±]Max Planck ETH Center for Learning Systems
andreas.opedal@inf.ethz.ch niklas.stoehr@inf.ethz.ch as17582@nyu.edu mrinmaya.sachan@inf.ethz.ch

Abstract

Solving math story problems is a complex task for students and NLP models alike, requiring them to understand the world as described in the story and reason over it to compute an answer. Recent years have seen impressive performance on automatically solving these problems with large pre-trained language models and innovative techniques to prompt them. However, it remains unclear if these models possess accurate representations of mathematical concepts. This leads to lack of interpretability and trustworthiness which impedes their usefulness in various applications. In this paper, we consolidate previous work on categorizing and representing math story problems and develop MATHWORLD, which is a graph-based semantic formalism specific for the domain of math story problems. With MATHWORLD, we can assign world models to math story problems which represent the situations and actions introduced in the text and their mathematical relationships. We combine math story problems from several existing datasets and annotate a corpus of 1,019 problems and 3,204 logical forms with MATHWORLD. Using this data, we demonstrate the following use cases of MATHWORLD: (1) prompting language models with synthetically generated question-answer pairs to probe their reasoning and world modeling abilities, and (2) generating new problems by using the world models as a design space.

https://github.com/eth-nlped/mathworld

1 Introduction

Math story problems (MSPs) are short narrative texts that describe a dynamic situation in the world consisting of entities, actions and states, followed by a quantitative question about the world, as displayed in Fig. 1. The task of automatically solving MSPs has received much research attention

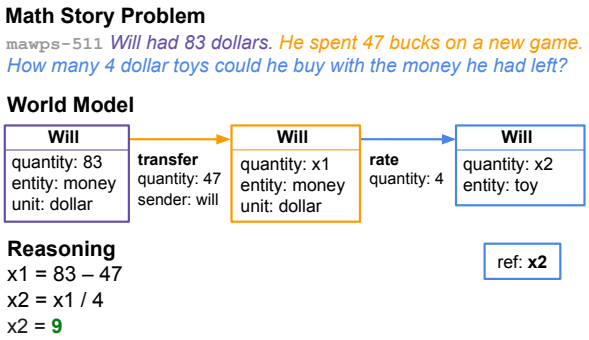


Figure 1: An example of a world model in MATHWORLD. MATHWORLD can be used to develop interpretable MSP solvers, to study the reasoning of LLMs and as a design space for generation of new MSPs.

in NLP. While earlier models for solving MSPs (Hosseini et al., 2014; Kushman et al., 2014; Roy and Roth, 2015) focused on extracting various features from text to learn probabilistic models, recent efforts have used pre-trained large language models (LLMs) (Yang et al., 2021; Drori et al., 2022; Lewkowycz et al., 2022, *inter alia*). Although they display high performance on benchmarks, it has been shown that such neural models tend to rely heavily on shallow heuristics, raising questions about whether the models can indeed "understand" MSPs and robustly solve them (Patel et al., 2021; Stolfo et al., 2023).

From the human side, solving MSPs requires a wide set of skills. A student must not only perform a set of given computations, but first be able to process the text and map it into a corresponding world model that represents the situation described in text (Cummins et al., 1988; Stern, 1993). Inspired by this, we take a step towards developing more interpretable solvers and introduce MATHWORLD, a semantic world model framework for MSPs.

MATHWORLD can be viewed as a formalism for reasoning in dynamical problem settings (McCarthy, 1963; Reiter, 1991), specific to the domain of MSPs. It represents each problem as a directed graph called a *world model* (§ 3). The

Findings of the Association for Computational Linguistics: ACL 2023, pages 9088–9115
July 9-14, 2023

nodes in a world model are containers (§ 3.1) representing entities' possession of some quantity (Hosseini et al., 2014) and the edges represent various types of mathematical relations between the quantities (§ 3.2). The relations correspond to mathematical concepts that have been previously shown to cover a vast majority of MSPs (Mitra and Baral, 2016; Roy and Roth, 2018). We annotate a MATHWORLD dataset consisting of $1,019$ English MSPs from various widely-used datasets (Koncel-Kedziorski et al., 2016b; Miao et al., 2020; Patel et al., 2021), which we make publicly available.

There are several potential use cases of MATHWORLD, of which we discuss three. First, one natural application is that of developing interpretable MSP solvers. A solver using MATHWORLD follows two steps: (i) semantic parsing and (ii) reasoning. The semantic parser takes an MSP text and outputs a world model based on the explicit information in the text. The reasoner then takes the world model and solves the problem based on the quantities and their relations. Our experiments show that LLMs struggle to build accurate and well-formed world models; we encourage future work to develop stronger semantic parsers for MATHWORLD.

Another use case of MATHWORLD is as a tool to study the reasoning capabilities of existing solvers. For instance, we can use the world model annotations to automatically generate synthetic subquestions for the MSPs. Using such subquestions, we give empirical evidence that GPT-3 (Brown et al., 2020) benefits from the structured knowledge derived by world models in its ability to solve MSPs. We further use our synthetic questions to understand if GPT-3 can indeed answer these intermediate questions about the world described in the MSPs, and not just the final question. We find that for problems where GPT-3 answers the final question correctly, it can only answer 64% of the intermediate questions. This suggests that GPT-3 is not accurately building world models for these problems but might be relying on reasoning shortcuts.

Finally, MATHWORLD can be considered as a design space for generating interesting new MSPs. We illustrate the usefulness of MATHWORLD for the task of generating MSPs by prompting an LLM using the world model annotations.

2 Related Work

Math story problems in NLP Although the problem of automatically solving MSPs has gathered substantial interest in NLP (Roy and Roth, 2015; Kushman et al., 2014; Huang et al., 2017; Amini et al., 2019; Xie and Sun, 2019; Drori et al., 2022), the focus has traditionally been on improving answer accuracy rather than providing didactic human-interpretable solutions (Shridhar et al., 2022). Some approaches map the text to expression trees (Koncel-Kedziorski et al., 2015; Yang et al., 2022; Roy and Roth, 2017) or explicitly model arithmetic concepts (Mitra and Baral, 2016; Roy and Roth, 2018). However, few if any computational works have attempted to solve MSPs by using mental models (Johnson-Laird, 1983), which is a common framework for analyzing how humans solve MSPs (Kintsch and Greeno, 1985). Taking inspiration from mental models of MSPs, we offer MATHWORLD as a computational model (fully expressible in first-order logic, App. D) which represents reasoning steps, arithmetic concepts and fictional elements in a human-readable graph format. We hope that such an approach can support intelligent tutoring systems (Anderson et al., 1995), e.g., by delivering feedback and hints (Zhou et al., 1999; Fossati, 2008) or generating new MSPs (Polozov et al., 2015; Koncel-Kedziorski et al., 2016a; Srivastava and Goodman, 2021).

In particular, we draw inspiration from Hosseini et al. (2014), who propose a symbolic approach that maps the text to container-based states. However, their symbolic representation is purely extracted from syntactic rules without human annotation. Further, their approach only covers problems that involve a transfer of some quantity between some actors (although they do not use that terminology), requiring addition and/or subtraction. In contrast, MATHWORLD is more closely tied to the MSP's semantics. It covers a strictly larger set of problem types, involving more concepts and all four basic arithmetic operators $(+, -, \times, \div)$. See Table 1 for a comparison between MATHWORLD and Hosseini et al. (2014), as well as Mitra and Baral (2016) and Roy and Roth (2018) from which we adopt the taxonomy over arithmetic concepts.

Reasoning with large language models LLMs have displayed impressive performance on numerical reasoning tasks (Brown et al., 2020; Chowdhery et al., 2022), particularly by the help of careful prompt engineering (Wei et al., 2022; Shridhar et al., 2023; Zhou et al., 2023). While language models have been argued to be intrinsically limited in their ability to perform human-like rea-

	Arithmetic coverage	Conceptual coverage	Semantic granularity	Annotations?	Mapping to formal logic?
MATHWORLD	$(+, -, \times, \div)$	Transfer Rate Comparison Part-whole	World model	Yes	Yes
Hosseini et al. (2014)	$(+, -)$	Transfer	World model	No	No
Mitra and Baral (2016)	$(+, -)$	Transfer Comparison (add) Part-whole	Concepts & equations	Yes	No
Roy and Roth (2018)	$(+, -, \times, \div)$	Transfer Rate Comparison Part-whole	Concepts & equations	No	No

Table 1: Comparison between MATHWORLD and other MSP works that use a more fine-grained symbolics than equations alone. "Annotations" refers to whether those symbolics are explicitly provided in the dataset.

soning (Bender and Koller, 2020), the mechanism by which they find answers in complex reasoning tasks is currently an active area of research (Tafjord et al., 2021; Saparov and He, 2023). MATHWORLD provides ground truth world model annotations, which is valuable in such studies (as demonstrated in § 5.2). One other aspect of LLMs that may limit them when applied to reasoning is that they produce natural language text, which may be ambiguous and diverse. These considerations motivate us to study MSPs as structured representations of meaning, which can in turn be used to generate natural language (Saparov and Mitchell, 2022).

Semantic parsing MATHWORLD can be viewed as a domain-specific semantic formalism. Our work thus also relates closely to semantic parsing, particularly of graph-based structures (Banarescu et al., 2013; Cai and Lam, 2019; Zhang et al., 2019; Bai et al., 2022). However, while most other formalisms consider meaning only at the sentence level, our world model graphs span the meaning across multiple sentences.

3 MATHWORLD

In this section, we present our world model formalism MATHWORLD. We formalize an MSP as a sequence of n sentences $\mathbf{s} = s_1 \circ \cdots \circ s_n$. It can be separated into a **body** $\mathbf{b}$ and a **question** q, such that $\mathbf{s} = \mathbf{b} \circ q$. The body is further partitioned into a sequence of $n - 1$ declarative sentences $\mathbf{b} = s_1 \circ \cdots \circ s_{n-1}$ and the question consists of a single interrogative sentence $q = s_n$.

World models in MATHWORLD are directed and labelled graphs, denoted g.[1] We refer to the nodes of the graph as **containers** (§ 3.1) and the edges of the graph as **relations** (§ 3.2). Each container and relation is labelled with a set of properties. One such property is the **quantity**, which may be either an explicit number mentioned in text or a variable representing an unknown number. The containers and relations along with their properties specify the equations induced by the MSP. In addition, each g is associated with a **reference variable** r, which points to the variable in g that holds the correct answer to the question as stated in q. We consider each $\mathbf{s}$ to be associated with some structure (g, r).

We say that g is **faithful** if it represents the semantics of the problem text according to the framework of MATHWORLD. Further, g is **complete** if r can be solved with the equations induced by g. A complete world model is **correct** if, when evaluated, r gives the correct answer to the problem. See Fig. 1 for an example of a world model.

In order to allow for incremental parsing, we segment the world models into sentence-level logical forms m_i, $i = 1, \ldots, n$. The logical form is a sequence that represents the containers and/or relations associated with the corresponding sentence.[2] We can convert $(m_1, \ldots, m_n)$ to a world model graph, and vice versa. The two representations are nearly equivalent, with the exception of a few caveats (see App. F for details). There is no bound on the problem length and, by extension, the number of logical forms. MATHWORLD is thus able to represent problems of any arbitrary number of

[1] The graphs may be cyclic. Although in practice, they tend to be acyclic.

[2] A logical form may be empty. Such will be the case for text outside the coverage of MATHWORLD.

reasoning steps. The assignment of logical forms may be ambiguous in the sense that there may be multiple faithful logical forms for a given sentence (discussed in App. B).

We consider subgraphs g_i, for sentence i, of the final graph g. A subgraph g_i corresponds to the logical forms up to sentence i, i.e., $(m_1, \ldots, m_i) \mapsto g_i$. We refer to the subgraph for some sentence index i as the **state** of i. As an example of how world models are built incrementally with states, consider Fig. 1. The first sentence maps to the container for label *Will* holding the entity *money* of quantity 83 with unit *dollar*. The second sentence provides information on an update to Will's possessed money, a TRANSFER relation (§ 3.2.1). Finally, the question sentence introduces rate information, a RATE relation (§ 3.2.2), between money and toys.

In the next sections, we describe the details of containers and relations in depth.

3.1 Containers

We adopt and modify the containers described in the model of Hosseini et al. (2014). Semantically, containers represent containment/possession. We refer to the possessor in the text as the **label** of the container.[3] In Fig. 1, the container label is *Will* for all containers (although in general the label can vary across containers). The label must be a noun plus any associated noun adjuncts (like *elementary school*). In addition to label, a container may have the following four properties:

Entity: The entity is *what* is possessed in the container. It is a noun, for which there may be an associated count. When expressed in a problem text, it must be the head of a noun phrase. In Fig. 1, *money* and *toy* are entities.[4]

Quantity: The quantity is the number associated with the entity. It may be known, in which case it will be a positive real number, or unknown, in which case it will be a variable.

Attribute: The attribute is a modifier for the entity. It is often an adjective, but may take other forms as well. The attribute is an optional property.

Unit: The unit is the unit of measurement for the entity. A unit property must exist if the entity is a mass noun, but may exist in other cases as well. For example, "liter of water" and "kg of apples" will both be assigned to containers with units. The unit is an optional property.

Entity, attribute and unit are written in their lemmatized forms. The label is not, in order to be able to distinguish between a set (plural: *friends*) and an element of some set (singular: *friend*).

Note that the containers take a variable number of properties; having arity 3, 4 or 5. Two containers are **equal** if they have the same arity and the same properties. We refer to a container's **structure** as its container label, entity, attribute (if exists) and unit (if exists). Two containers are **structurally equal** if they have the same structure.

3.2 Relations

Relations are the edges in g. They represent the interactions between the various parts of the world model, from which the equations of the MSP are induced. The relations are directed, and the direction encodes semantics of the relation depending on the type of relation. Like containers, relations have properties. The properties and their arity also depend on the type of relation.

There are four types of relations: TRANSFER, RATE, COMPARISON and PARTWHOLE. Together they span all four basic arithmetic operators $(+, -, \times, \div)$. Next, we give a detailed description of each of these relation types. Examples of world models with each relation type are provided in App. A.

3.2.1 TRANSFER

TRANSFER relations model that a transfer of some quantity of an entity has occurred. A given container structure will either gain or lose quantity from a TRANSFER relation. For example, "Alice ate 3 apples" will correspond to a TRANSFER with a loss of 3 apples for the container labeled Alice. A TRANSFER is always between two containers of the same structure. The direction of the edge describes order: The source container will hold the quantity *before* the transfer event occurred, and the target container will hold the quantity *after* the transfer event occurred.

In addition to quantity, TRANSFER takes the following two properties:

Recipient: The label of the container structure where the quantity of the given entity is *gained*.

[3] There may not always be an explicit possessor expressed in text. In such cases, we use the label *World*.

[4] Note how the term *money* is not actually expressed in the problem text. Similarly, the word *time* will seldom be expressed in MSPs involving reasoning about time.

Sender: The label of the container structure where the quantity of the given entity is *lost*.

A recipient, a sender or both must exist. TRANSFER thus has arity 2 or 3. The TRANSFER relation either adds or subtracts the relation quantity to/from the source container quantity, depending on whether the relation connects the recipient containers or sender containers.

3.2.2 RATE

The RATE relation models mathematical rate between two quantities. These two quantities are held in two separate containers with the same label, and the ratio quantity of the rate is given as a property to the relation. RATE has this one single property. The direction of the edge determines the relationship: The source container holds the numerator of the rate, and the target container holds the denominator of the rate. In the example in Fig. 1, the source container holds the entity *money* and the target container holds the entity *toy*, indicating that the rate quantity concerns *money per toy*. Mathematically, RATE implies that the source quantity divided by the relation quantity equals the target quantity.

3.2.3 COMPARISON

COMPARISON is invoked when there is an explicit relationship between two quantities in the MSP. For example, "Alice is twice as old as Bob". The COMPARISON relation may be either between containers with different labels, such as "Alice has 3 more apples than Bob", or between containers with the same label, such as "Alice has 3 more red apples than she has green apples". It takes two properties; quantity and type:

Type: The arithmetic operation type COMPARISON. It can take one of the two values; *add* (indicating addition) or *mul* (indicating multiplication).

The quantity held in the source container is the one that is combined with the quantity of the COMPARISON relation under the arithmetic operator, the output of which will be the quantity held in the target container.

3.2.4 PARTWHOLE

PARTWHOLE relations model set partitions. The set represented by some container is partitioned into subsets, each of which is represented by another container. For each of the subset containers (the parts), there is an outgoing edge to the container with the superset (the whole). Thus, PARTWHOLE implies that for a given container that has ingoing PARTWHOLE edges, the sum over the quantities in the source containers of those edges equals the quantity in the target container. Note that PARTWHOLE differs from the other relations in that it requires multiple edges to induce an equation.[5] In most cases, all containers involved in a PARTWHOLE relation will have the same label. The relation can then be viewed as a relation between entities possessed by a specific label. For instance, "Alice has 3 red apples and 6 green apples, how many apples does she have in total?" would be represented by PARTWHOLE. PARTWHOLE relations have no properties.

PARTWHOLE relations may represent meaning that is not explicit in text. Parsing the text of a problem that requires PARTWHOLE might thus lead to an incomplete (§ 3) world model, which may require additional assumptions. In addition, orienting PARTWHOLE relations might require commonsense knowledge. For instance, a problem might introduce a quantity for tables and a quantity for chairs, and ask about the total number of furniture.

3.3 World model equivalence and similarity

One of the principal utilities of MATHWORLD is to allow for evaluating models on their reasoning ability. For that we need consistent equivalence notions and similarity metrics between world models, which we provide here.

Let g and g' be **isomorphic** if there exists an isomorphism on the underlying graphs that additionally preserves relation types. We consider two forms of equivalence notions between world models, which we call strong and weak equivalence. Weak equivalence deems two world models to be equal if they are isomorphic. Strong equivalence additionally requires all properties of the containers and relations to be equal.[6] In addition, we create two similarity scores based on the AMR metric *smatch* (Cai and Knight, 2013): Weak smatch considers graph topology in the same way as our isomorphism equivalence, and strong smatch additionally considers all properties of the world models. We give details on these similarity scores in App. C.

[5] Note that a PARTWHOLE relation can be equivalently represented as a hyperedge.

[6] In practice, we lemmatize all properties before performing this equivalence check.

3.4 Comparison to other logical formalisms

MATHWORLD can be fully expressed in first-order logic (FOL). We provide a constructive proof in the form of a conversion in App. D, which enables comparison of the expressive power of MATHWORLD with that of other formalisms. Both AMR and MATHWORLD restrict the expressivity of full FOL in different ways. AMR provides a way to express negation (the `polarity` relation) but does not provide a way to directly express universal quantification[7] (Bos, 2016). MATHWORLD represents sets of objects as containers and enables universal quantification over those sets. This is restricted, however, as MATHWORLD does not allow the definition of sets of sets, or nested universal quantification.[8] Negation is not directly expressible in MATHWORLD, as it is designed for the domain of MSPs where negation is quite rare.

MATHWORLD is more comparable to *situation calculus* (McCarthy, 1963), where each relation can be modeled as an action that changes the state of the world. Like situation calculus, the changing world state over time is implicitly represented in MATHWORLD (via the TRANSFER relation), whereas in FOL, an explicit description of the time of each event is necessary.

4 Data Collection

In order to study how models are able to answer MSPs, convert them to logical form, perform world modeling, and reason mathematically to find the answer, we require a diverse dataset of labeled MSPs that spans all concepts covered by MATHWORLD. To ensure diversity and wide variety in the examples, we collect them from numerous sources:

1. The math word repository MAWPS (Koncel-Kedziorski et al., 2016b) gathers several datasets (Hosseini et al., 2014; Kushman et al., 2014; Koncel-Kedziorski et al., 2015; Roy and Roth, 2015), thus providing a wide variety of MSPs.
2. To complement with more challenging problems, we also adopt problems from ASDIV-A (Miao et al., 2020), which was designed for linguistic diversity and math concept diversity.

	Train		Test	
	MSPs	LFs	MSPs	LFs
ASDIV-A	328	1,052	83	272
MAWPS	312	936	79	235
SVAMP	173	563	44	146
TOTAL	813	2,551	206	653

Table 2: Size of annotated dataset in terms of number of MSPs and number of sentence-aligned logical forms (LFs), stratified by dataset of origin and split.

3. We also annotate a subset of the SVAMP dataset (Patel et al., 2021), which was introduced as a challenge set to test robustness to data artifacts. This enables future work to test the robustness of MATHWORLD parsers.

We randomly sample a subset from each of these three datasets,[9] and annotate them with world models. We obtain $1,019$ MSPs, which corresponds to $3,204$ logical forms, which we partition into $80/20$ train/test splits. Table 2 provides more details.

We hire external workers for annotation. Annotation follows three phases: A first training phase where annotators are given several small sets at a time with follow-up discussion sessions, an agreement phase in which all annotators are given the same problems and a final scale-up phase. We use an annotation tool created specifically for this work (shown in App. E.2). The problems are annotated incrementally sentence-by-sentence, in order to match logical forms to sentences as described in § 3. Questions are hidden from annotators until all preceding sentences are completed, in order to avoid bias stemming from having read the question—MATHWORLD is meant to capture the world model of the problem irrespective of what is asked in the question. Within sentences, we ask annotators to add containers and relations according to the order in which they occur in text. This allows us to write the logical forms according to within-sentence order when creating training data for semantic parsing. We maintain this order with integer IDs that are incremented automatically in

[7] It is possible to do so indirectly, as in $\neg\exists x.\neg\phi(x) \equiv \forall x.\phi(x)$, but this can only be done once per sentence.

[8] This disallows higher-order expressions, e.g., COMPARISON relations between quantities expressed in TRANSFER relations. It also disallows nested possession outside of what is made possible under RATE, e.g., structures like "Alice has a house that has a shelf that has a book that has 200 pages."

[9] We also considered the larger GSM8K dataset (Cobbe et al., 2021), which contains problems with more reasoning steps. However, although we found MATHWORLD to cover many of its MSPs, annotation workers were unable to reliably annotate these problems. Future work may aim to augment the data to assign ground truth world model structures to longer MSPs, using techniques similar to those demonstrated in § 5.3.

the annotation tool.

We performed an agreement analysis of 125 overlapping MSPs, revealing a high agreement rate considering the complexity of the annotation task. Concretely, 61 out of these 125 were strongly equivalent (§ 3.3) across annotators, and 107 were weakly equivalent (§ 3.3). Many of the only weakly equivalent annotations were due to ambiguity in the properties (App. B.1), and almost half of the 18 non-agreed problems were due to ambiguity in relation type (App. B.2). The strong and weak smatch scores were 0.91 and 0.97 respectively. These can be interpreted as approximate upper bounds on the smatch scores achievable by any model, due to the ambiguity in the dataset. Many of the annotation errors, also outside of the overlapping set, could be either corrected or discarded *ex post*. Further details on annotation are given in App. E.

5 Applications of MATHWORLD

In this section we showcase some applications of MATHWORLD: solving (§ 5.1), probing of reasoning (§ 5.2) and generation of new MSPs (§ 5.3).

5.1 Parsing and Reasoning

We spell out a framework for solving MSPs using MATHWORLD. The framework consists of two components: A *parser* and a *reasoner*. The parser is tasked with assigning a faithful world model g to an input problem $\mathbf{s}$, along with a reference variable r. The reasoner is then queried with r and computes an answer based on the induced equations of g. We also present a set of initial experiments, meant to introduce the task of MATHWORLD parsing to the community.

5.1.1 Parser

Given an MSP $\mathbf{s}$, the task is to assign a world model g. The first step is to predict the sequence of logical forms $m_1, \ldots, m_n$. We model this as a conditional distribution

$$p(m_1, \ldots, m_n \mid \mathbf{s}) = \prod_{i=1}^{n} p(m_i \mid s_1, \ldots, s_i). \quad (1)$$

With this factorization, we can parse the graph incrementally one sentence at a time. The factorization is based on two assumptions: $m_i \perp s_j, \forall i < j$ and $m_i \perp m_j, \forall i \neq j$. Both are aligned with MATHWORLD as outlined in § 3: the first assumption means that a logical form is independent of the sentences in subsequent steps, and the second assumption means that logical forms are independent of each other. Dependencies of logical forms on preceding sentences are kept due to coreferences, elliptical constructions and other inter-sentence dependencies.

As explained in § 3, the logical forms are linearized representations of the world model graphs. Thus, our pipeline (as well as applications like those demonstrated in § 5) requires that we are able to convert from one representation to the other: World model graphs must be converted to logical forms in order to create training data for a semantic parser, and the predicted logical forms must be converted to world model graphs and reference variables for visualization and reasoning. The details of this conversion are given in App. F.

5.1.2 Reasoner

Once we have a world model graph, we apply a reasoning algorithm over the graph to compute an answer. The reasoner takes a world model and a reference variable, and outputs a numeric value for the reference variable r. Our implementation is deterministic and follows two steps. First, it extracts all equations induced by the world model (as described in § 3.2 and illustrated in App. A). Second, it solves for r using a recursive algorithm. Full pseudocode along with a discussion is presented in App. H.[10]

5.1.3 Baseline solving experiments

We demonstrate our proposed modeling framework with a baseline semantic parser, in the form of a large language model that is supervised in-context. We use Codex (Chen et al., 2021), as language models trained on code have been previously shown to perform well on structured prediction tasks (Madaan et al., 2022; Drozdov et al., 2023). The prompt contains 50 ground truth examples from MAWPS and ASDIV-A, and we evaluate the model on the test sets of MAWPS, ASDIV-A and SVAMP. We also implement a rule-based baseline system, based on Hosseini et al. (2014).

Our results corroborate that this is a challenging task; for the least difficult dataset the model gets roughly one third of the problems correct, and predicts a complete world model for only slightly more than half of the problems. The rule-based baseline gets nearly no problems correct. Indeed, a model

[10] We note that annotated world models are not necessarily complete (def. in § 3). Annotators were requested to only build world models that represent what is made explicit in the text. Some problems may require additional background knowledge to build a complete world model.

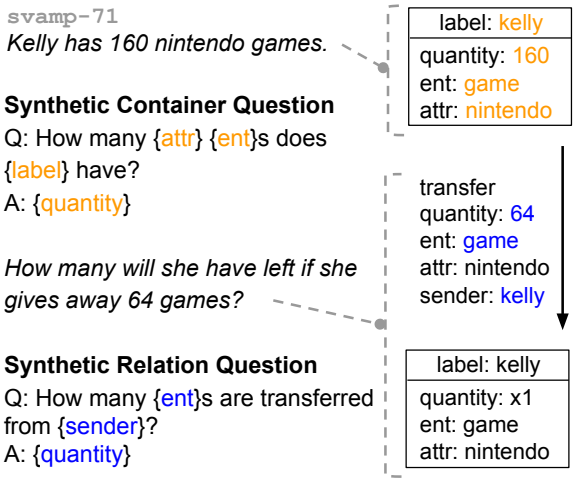


Figure 2: Synthetically created question-answer pairs based on templates. Note that the quantity in the container or relation does not need to be expressed in text, but could be a variable. Such cases test the model's ability to reason over intermediate quantities.

must, for each sentence, produce well-formed logical forms that exhaustively and correctly capture the semantics in MATHWORLD, combine these into a world model and query the reasoner with the correct reference variable. One mistake in any of these steps may lead to an incorrect answer. With much progress and research interest in semantic parsing in recent years (Shin et al., 2021; Qiu et al., 2022) there are several promising directions for improvement, and we invite the research community to help in developing stronger semantic parsers for this challenging task. Further details on the setup and results can be found in App. I.1.

5.2 Probing LLMs' partial knowledge

World models enable us to study the reasoning ability of LLMs: Beyond just testing whether a model outputs the correct solution to an MSP, we can test whether the model follows a correct reasoning path and accurately builds world model representations.

Setup We design question and answer templates that are automatically filled based on information in the world model. Two examples of such templates are given in Fig. 2 and a list of all templates is given in App. I.3. By courtesy of the world model we know the true answer to each of these synthetic questions, enabling us to create prompts with question-answer pairs.

We experiment with three types of prompts, all displayed with full-length examples in Table 8: *(1) synth QA (all at once)*. We first include the complete problem text, followed by synthetic question and answer pairs related to some part of the text. We randomly sample two such pairs; *(2) synth QA (sentence-by-sentence)*. We again sample two question-answer pairs at random, but in this setting they are imputed right after the sentence in which the answer to the question is given; *(3) original MSP QA*. Under this setting we do not include any synthetic question-answer pairs, only the original text. All prompts end with the MSP question that we aim to solve followed by "A:". We study both whether the synthetic questions help the model answer the MSP correctly, and how well the model answers the synthetic questions themselves.

	from x MSPs	
QA type	0	1
(1) synth QAs (all at once)	70.8	71.8
(2) synth QAs (sent by sent)	71.3	**78.6**
(3) original MSP QAs	69.4	70.8

Table 3: Results obtained by GPT-3 in answering math story problems reported in accuracy percent. A larger increase in performance is observed when the synthetic question-answer pairs are presented at the relevant part of the text, rather than at the end.

Results We report results obtained by GPT-3 (Brown et al., 2020) on the combined test set of all three datasets in Table 3. The number of in-context examples is either 0 or 1. We observe increased performance when including synthetic question-answer pairs, particularly in setting (2) where the questions are imputed at the relevant part of the MSP text. We hypothesize that doing so helps guide the reasoning trace of the model, in a similar vein as chain-of-thought prompting (Wei et al., 2022). Further, we find that GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020), Codex (Chen et al., 2021), T5 (Raffel et al., 2020) and NT5 (Yang et al., 2021) overall perform poorly, but benefit from an increase in performance when synthetic question-answer pairs are provided.

We further compare the ability of GPT-3 to answer the intermediate synthetic questions to its ability to answer the original final question. For each MSP, we first select a container or relation uniformly at random and then create a synthetic question. We then ask both the synthetic question and the original question at the end of two separate prompts in a zero-shot setting. Table 4 displays the results. Interestingly, in more than one third of the

Original Question	Synthetic Question Correct	Wrong
Correct	46.0%	25.7%
Wrong	11.0%	17.3%

Table 4: We test whether the model gets synthetic questions about parts of the world model right and compare it against its performance on the original question.

cases that the model gets the original question right (top row), it gets the intermediate synthetic question wrong (top right cell). Overall it also shows a higher accuracy on the original questions (top row) than the synthetic intermediate questions (left column). While some of these results could be explained by the nature of the templated questions, it does seem to indicate that the model makes use of heuristics rather than human-like reasoning when solving MSPs (Patel et al., 2021).

5.3 Generation of MSPs

MATHWORLD can be considered as a space under which a practitioner can design new MSPs with certain desired features. For instance, a teacher may be interested in generating variations of an MSP to test a specific mathematical concept with a specific unknown variable. To demonstrate the potential for such applications we provide a small proof-of-concept experiment.

Setup We use GPT 3.5 Turbo (Ouyang et al., 2022) with a prompt of 30 examples from the train sets of MAWPS and ASDIV-A. One example consists of the logical forms for a full MSP world model (source) followed by the text of the MSP (target). We separate sentence-aligned logical forms in the source as well as the sentences in the target by a marker, so that the model can pick up the alignment patterns. The ground truth examples are sampled randomly. To generate a new MSP conditioned on a world model, we append the logical form corresponding to the world model to the end of the prompt. We try generating new MSPs both based on (i) world models present in our annotated test sets (paraphrasing) and (ii) manual augmentations of annotated world models. We perform evaluation for setting (i) using SacreBLEU (Post, 2018) and BERTScore (Zhang et al., 2020), comparing all MSPs in the test sets to their paraphrases.[11]

[11] More details on the generation setup are given in App. I.2.

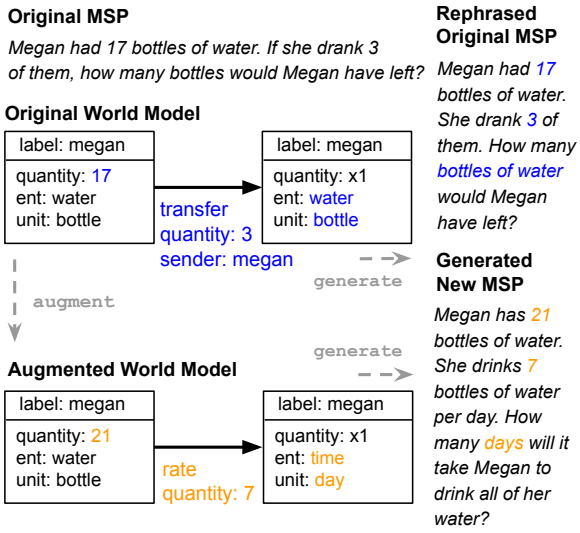


Figure 3: Example MSPs generated by GPT 3.5 Turbo.

Results We obtain SacreBLEU scores of 66.73, 40.86 and 26.02 and F1 BERTScores of 0.933, 0.930 and 0.931 for MAWPS, ASDIV-A and SVAMP respectively. Qualitatively we observe that the generated MSPs mostly stay faithful to the logical forms but tend to be shorter and less linguistically complex than the original problems, which would explain the comparatively low SacreBLEU scores in comparison to the BERTScores. Further, we give the first six examples we generated according to the described setup. One of them is shown in Fig. 3. The model generates an output MSP very similar to the original, having only accessed the original's ground truth logical forms. We further augment the original world model by changing the TRANSFER to a RATE. Note how the generated MSP is faithful to the augmented world model. The other five examples are shown in Table 6.

6 Conclusion

In this work, we have presented a novel formalism, MATHWORLD, for expressing the semantics of math story problems. We have annotated a MATHWORLD corpus consisting of 1,019 problems and 3,204 logical forms. A world model derived from MATHWORLD exposes the structure of the reasoning process needed to solve the problem, which benefits several applications as we have demonstrated in § 5. As such, we hope that MATHWORLD will promote use cases beyond just improved MSP solving, ranging from automated chain-of-thought prompting to math problem generation.

Limitations

MATHWORLD is limited to cover math story problems using the four basic arithmetic operators. Furthermore, within the space of such problems, it does not cover "second-order" MSPs (as discussed in § 3.4). Neither does it cover negation nor inequalities.

We only consider datasets with MSPs written in English in this work. However, MATHWORLD should in principle be able to cover the same type of problems formulated in other languages as well.

An obvious limitation of this work is the low performance on the task of solving MSPs. The focus of this work is to introduce the world model formalism and its use cases, and we leave for future work to build stronger MATHWORLD parsers.

Ethics Statement

We foresee no major ethical concerns with this work. The introduction of MATHWORLD is aimed at improving the interpretability and robustness of existing and future models for math story problem solving. On this account, we hope to contribute to identifying (and hopefully reducing) existing biases in pre-trained language models, or any future alternatives. However, we would like to caution that the formalism could be used to generate inappropriate math story problems.

Acknowledgements

We thank Arnav Mishra, Aryaman Kolhe, Devraj Thakur, Gaurav Saini and Soham Bopardikar for help with annotation work. We further thank Jakub Macina, Kumar Shridhar and Menna El-Assady for input in the early stages of the project, Ethan Wilcox and Ying Jiao for helpful feedback, and Yixiong Wang for help in implementation of a symbolic baseline solver. Andreas Opedal is partially supported by the Max Planck ETH Center for Learning Systems. Niklas Stoehr acknowledges funding from the Swiss Data Science Center.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

John R. Anderson, Albert T. Corbett, Kenneth R. Koedinger, and Ray. Pelletier. 1995. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4(2):167–207.

Xuefeng Bai, Sen Yang, Leyang Cui, Linfeng Song, and Yue Zhang. 2022. Cross-domain generalization for AMR parsing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10907–10921, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Johan Bos. 2016. Squib: Expressive power of abstract meaning representations. *Computational Linguistics*, 42(3):527–535.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3799–3809, Hong Kong, China. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: An evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Denise Dellarosa Cummins, Walter Kintsch, Kurt Reusser, and Rhonda Weimer. 1988. The role of understanding in solving word problems. *Cognitive Psychology*, 20(4):405–438.

Iddo Drori, Sarah Zhang, Reece Shuttleworth, Leonard Tang, Albert Lu, Elizabeth Ke, Kevin Liu, Linda Chen, Sunny Tran, Newman Cheng, Roman Wang, Nikhil Singh, Taylor L. Patti, Jayson Lynch, Avi Shporer, Nakul Verma, Eugene Wu, and Gilbert Strang. 2022. A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level. *Proceedings of the National Academy of Sciences*, 119(32):e2123433119.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2023. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*.

Davide Fossati. 2008. The role of positive feedback in Intelligent Tutoring Systems. In *Proceedings of the ACL-08: HLT Student Research Workshop*, pages 31–36, Columbus, Ohio. Association for Computational Linguistics.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. 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, Doha, Qatar. Association for Computational Linguistics.

Danqing Huang, Shuming Shi, Chin-Yew Lin, and Jian Yin. 2017. Learning fine-grained expressions to solve math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 805–814, Copenhagen, Denmark. Association for Computational Linguistics.

Philip Nicholas Johnson-Laird. 1983. *Mental models : towards a cognitive science of language, inference and consciousness*. Cognitive science series 6. Harvard University Press, Cambridge, Massachusetts.

Walter Kintsch and James G. Greeno. 1985. Understanding and solving word arithmetic problems. *Psychological Review*, 92(1):109–129.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Rik Koncel-Kedziorski, Ioannis Konstas, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2016a. A theme-rewriting approach for generating algebra word problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1617–1628, Austin, Texas. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016b. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Robert Koons. 2022. Defeasible reasoning. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Summer 2022 edition. Metaphysics Research Lab, Stanford University.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281, Baltimore, Maryland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. pages 7871–7880.

Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language models of code are few-shot commonsense learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1384–1403, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

John McCarthy. 1963. Situations, actions, and causal laws. In *Stanford Artificial Intelligence Laboratory and Memo (Stanford Artificial Intelligence Laboratory)*.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Arindam Mitra and Chitta Baral. 2016. Learning to use formulas to solve simple arithmetic problems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2144–2153, Berlin, Germany. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Oleksandr Polozov, Eleanor O'Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popovic. 2015. Personalized mathematical word problem generation. In *IJCAI*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Linlu Qiu, Peter Shaw, Panupong Pasupat, Pawel Nowak, Tal Linzen, Fei Sha, and Kristina Toutanova. 2022. Improving compositional generalization with latent structure and data augmentation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4341–4362, Seattle, United States. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Raymond Reiter. 1991. *The Frame Problem in Situation the Calculus: A Simple Solution (Sometimes) and a Completeness Result for Goal Regression*, page 359–380. Academic Press Professional, Inc., USA.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2017. Unit dependency graph and its application to arithmetic word problem solving. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3082–3088. AAAI Press.

Subhro Roy and Dan Roth. 2018. Mapping to declarative knowledge for word problem solving. *Transactions of the Association for Computational Linguistics*, 6:159–172.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *International Conference on Learning Representations (ICLR)*.

Abulhair Saparov and Tom M. Mitchell. 2022. Towards general natural language understanding with probabilistic worldbuilding. *Transactions of the Association for Computational Linguistics*, 10:325–342.

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kumar Shridhar, Jakub Macina, Mennatallah El-Assady, Tanmay Sinha, Manu Kapur, and Mrinmaya Sachan. 2022. Automatic generation of socratic subquestions for teaching math word problems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4136–4149, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada.

Josep M. Sopena, Agusti LLoberas, and Joan L. Moliner. 1998. A connectionist approach to prepositional phrase attachment for real world texts. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 1233–1237, Montreal, Quebec, Canada. Association for Computational Linguistics.

Megha Srivastava and Noah Goodman. 2021. Question generation for adaptive education. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 692–701, Online. Association for Computational Linguistics.

Elsbeth Stern. 1993. What makes certain arithmetic word problems involving the comparison of sets so difficult for children? *Journal of Educational Psychology*, 85:7–23.

Alessandro Stolfo, Zhijing Jin, Kumar Shridhar, Bernhard Schölkopf, and Mrinmaya Sachan. 2023. A causal framework to quantify the robustness of mathematical reasoning with language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, Toronto, Canada.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. volume 2201.11903. arXiv.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5299–5305. International Joint Conferences on Artificial Intelligence Organization.

Peng-Jian Yang, Ying Ting Chen, Yuechan Chen, and Daniel Cer. 2021. NT5?! Training T5 to perform numerical reasoning. *arXiv*, 2104.07307.

Zhicheng Yang, Jinghui Qin, Jiaqi Chen, Liang Lin, and Xiaodan Liang. 2022. LogicSolver: Towards interpretable math word problem solving with logical prompt-enhanced learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1–13, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations (ICLR)*.

Yujian Zhou, Reva Freedman, Michael Glass, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1999. Delivering hints in a dialogue-based intelligent tutoring system. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, page 128–134, USA. American Association for Artificial Intelligence.

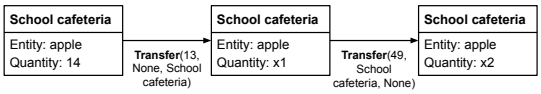


Figure 4: Example of a world model using TRANSFER.

A MATHWORLD Examples

A.1 TRANSFER

Consider the following problem:

> The school cafeteria had 14 apples. If they used 13 to make lunch for the students and then bought 49 more, how many apples would they have?

We display the corresponding world model in Fig. 4. The first sentence will correspond to a container for *school cafeteria* that holds 14 of entity *apple*. The second sentence describes two transfers: a first one where the school cafeteria is the sender of 13 apples, and a second one where the school cafeteria is the recipient of 49 apples. We get two equations:

$$14 - 13 = x_1 \quad (2)$$

$$x_1 + 49 = x_2 \quad (3)$$

The question asks for how many apples the school cafeteria has in the end, which matches the container holding the variable x_2 in the world model.

Although the TRANSFER relation always connects two containers of the same structure in the graph, a transfer event may occur between two containers of different structure. For example, "Alice gives 3 apples to Bob" describes a transfer event with Alice losing 3 apples and Bob gaining 3 apples. In these cases, we need two edges with the same properties in the world model; one for Alice's containers and one for Bob's containers (see Fig. 5). Consider the following problem with a transfer event occurring between two different possessors:

> Alice has 7 apples and Bob has 4 apples. Alice gives 3 apples to Bob. How many apples does Bob have now?

We show the corresponding world model in Fig. 5. *Alice* and *Bob* are represented by two separate containers, which are both updated by the same transfer event.

A.2 RATE

Consider the following problem:

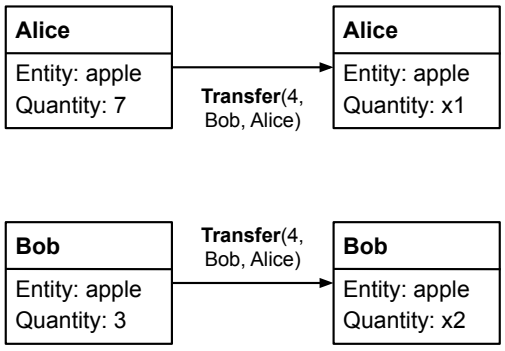


Figure 5: Example of a world model using TRANSFER.

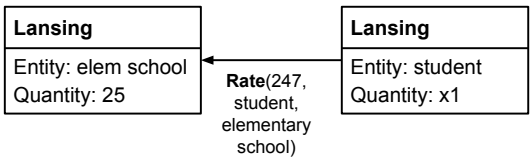


Figure 6: Example of a world model using RATE.

> Lansing has 25 elementary schools. There are 247 students in each school. How many elementary students are there altogether in Lansing?

This is a rate problem, as we get a rate on the number of students per elementary schools in the second sentence. The relation induces the following equation:

$$\frac{x_1}{25} = 247 \quad (4)$$

The question asks for the total number of students in Lansing, which corresponds to the quantity in the container that holds the entity *student*.

A.3 COMPARISON

Consider the following problem:

> James has 232 balloons. Amy has 101 balloons. How many more balloons does James have than Amy?

The first two sentences will correspond to two containers, representing the number of balloons possessed by James and Amy respectively. In the

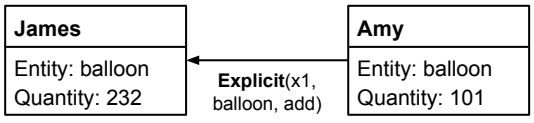


Figure 7: Example of a world model using COMPARISON.

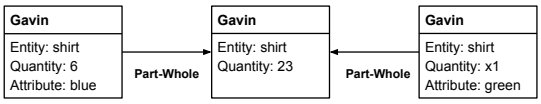


Figure 8: Example of a world model using PARTWHOLE.

question sentence, we get information about an COMPARISON relation between these two containers, with properties x_1 and *add*. Since we need to add the balloons in Amy's container to get the number of balloons in James' container, the edge is directed outwards from Amy's container. This relation induces the following equation:

$$101 + x_1 = 232 \tag{5}$$

The world model is displayed in Fig. 7.

A.4 PARTWHOLE

Consider the following problem:

> Gavin has 23 shirts. 6 are blue the rest are green. How many green shirts does Gavin have?

The first sentence will correspond to a container for Gavin holding the quantity of his shirts. The part-whole information is introduced in the second sentence, in which the 6 refers to shirts in the previous sentence (via an elliptical construction), and "the rest" tells us we have an additional complementing part of green shirts. Hence, the second sentence is assigned two new containers with attributes *blue* and *green*, as well as PARTWHOLE relations from both of these containers to the whole container introduced in the first sentence. This leads to the following equation:

$$6 + x_1 = 23 \tag{6}$$

The reference variable is the quantity in the container holding Gavin's green shirts. See Fig. 8 for the world model.

B Ambiguity

Ambiguity occurs when the same problem text may be assigned multiple correct and faithful world models. We distinguish between two types of ambiguity for MATHWORLD: **property ambiguity** and **structural ambiguity**.

B.1 Property ambiguity

Property ambiguity concerns cases where there are multiple possible properties to containers and/or relations that yield a semantically faithful world model. For instance, it is ambiguous whether "carrot sticks" is to be interpreted as an entity *carrot stick*, as entity *carrot* with unit *stick*, or as entity *stick* with attribute *carrot*. Property ambiguity may also follow from syntactic ambiguity in the problem text.

B.2 Structural ambiguity

Structural ambiguity occurs when the topology, including relation types, differs between several correct and faithful world models for a given problem. Consider the following example:

> James ate 22 carrot sticks before dinner and 15 more after dinner. How many carrot sticks did he eat?

This problem could be modeled either with TRANSFER or PARTWHOLE. In the case of TRANSFER, we view James as possessing some quantity of carrot sticks to start with. He then eats 22 of these, which can be viewed as a TRANSFER where James is the sender. This TRANSFER relation will be an outgoing edge into a new updated container for James' carrots. Another TRANSFER occurs for the 15 carrot sticks he ate after dinner. The reference variable would then be the variable held in the first container – how many carrot sticks James had initially. See Fig. 9 for the world model. Note that such a world model is not sufficient for solving the problem without further assumptions, it requires defeasible reasoning (Koons, 2022). We must assume that James had no carrot sticks after having eaten the ones post dinner, corresponding to the third container holding quantity 0, in order for the world model to be complete.[12]

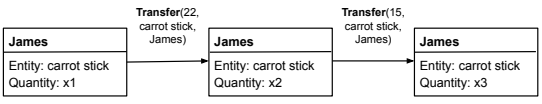


Figure 9: World model with a Transfer interpretation.

Another possibility would be with PARTWHOLE. With PARTWHOLE, we take the static view of James possessing 22 carrot sticks before dinner and 15 carrot sticks after dinner, assigning a container for each. The question statement gives us the information that we are asking for the total number of carrot sticks, which would be parsed with

[12] An alternative would be to augment r to handle expressions, giving $r = 22+15$. This would involve a more complex linarization scheme than that described in App. F.1 however.

PARTWHOLE to a container with the total. The reference will refer to the variable in this latter container. In contrast to the TRANSFER interpretation, the PARTWHOLE interpretation does not require additional assumptions to create a complete world model. See Fig. 10.

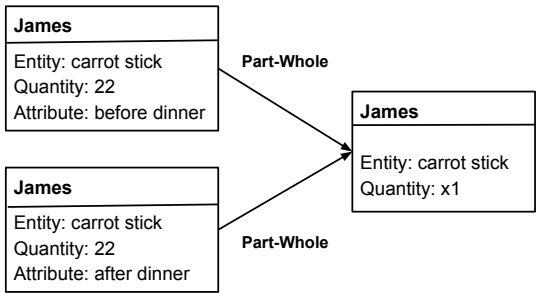


Figure 10: World model with a part-whole interpretation.

C Similarity Scores

In this section, we describe how we adapt smatch (Cai and Knight, 2013) for measuring similarity between world model graphs. We express the world models as conjunctions over logical triples. We label all containers and relations with a unique variable, and denote that such a variable is an instance of a container or one of the five relation types with the triple `instance(variable, type)`. Containers are represented as arguments to the relations in the form of `source` and `destination`, which are non-core roles in AMR.[13] For instance, a container `c` being the source node of relation `r` is represented as `source(r, c)`. The topology smatch score of two world models is then computed by taking the maximum f-score over one-to-one variable mappings between the two world models, as in Cai and Knight (2013).

The full semantic smatch score is computed in the same way, with the addition of logical triples for all the container and relation properties. We define core argument roles for the containers and each of the relation types. For instance, `ARG0` of a container will be its entity. The entity *apple* belonging to container `c` will be represented by two logical triples `instance(e, apple)` and `ARG0(c, e)`.

D Conversion to First-order Logic

In this section, we define a function to convert world model graphs into an equivalent FOL expression.

D.1 Describing quantities

Before introducing the conversion function, we first present a way in which quantities are described in FOL, as a preliminary. We define the `Measure` predicate, which is used to describe the "size" of a set. The set may contain countable entities such as "8 balloons" or uncountable entities such as "10 grams of coffee," and `Measure` is used to specify both types of quantities.

We introduce axioms to enable mathematical reasoning over the `Measure` predicate. If the measure of a set is a cardinal number (as in "8 balloons"), then it is the cardinality of that set:

$$\forall x \forall m(\texttt{Measure}(x, m) \wedge m \in \{0, 1, \ldots\} \leftrightarrow \texttt{Cardinality}(x, m)).$$

For example, if a set x contains 8 elements, we write $\texttt{Measure}(x, 8)$. We also define the additivity of measures:

$$\forall x \forall y \forall m_x \forall m_y (x \cap y = \varnothing \wedge \texttt{Measure}(x, m_x) \wedge \texttt{Measure}(y, m_y) \rightarrow \texttt{Measure}(x \cup y, m_x + m_y)).$$

That is, for any disjoint sets, the measure of their union is equal to the sum of their measures. To describe the size of sets containing uncountable entities (as in "10 grams of coffee"), we use the `Quantity` predicate. For example, if a set x contains 10 grams, we write $\texttt{Measure}(x, \texttt{Quantity}(10, \texttt{Gram}))$. To enable reasoning over such measures, we define the following axiom:

$$\forall x \forall y \forall u(\texttt{Quantity}(x, u) + \texttt{Quantity}(y, u) = \texttt{Quantity}(x + y, u)).$$

That is, quantities may be summed if they share the same units. Subtraction of quantities is defined similarly. Further axioms can be defined to allow conversions between units, such as:

$$\forall x(\texttt{Quantity}(x, \texttt{Milliliter}) = \texttt{Quantity}(x/1000, \texttt{Liter})).$$

D.2 Conversion function

Let $g = (V, E)$ be world model graph consisting of a set of containers V (i.e. vertices) and relations E (i.e. edges). Let $\bar{E} \subseteq E$ be the subset of relations

[13] We refer to the AMR guidelines for more information: https://github.com/amrisi/amr-guidelines

that do not have type PARTWHOLE (for which the semantics of the edges are not independent and thus need to be treated separately). Recall that the world model may also contain variables, which represent unknown quantities. Let U be the set of these variables. We can define a function $\|g\|$ that converts g into an equivalent FOL expression.

$$\|g\| = \exists v_1 \ldots \exists v_{|V|} \exists e_1 \ldots \exists e_{|\bar{E}|} \exists u_1 \ldots \exists u_{|U|} (\|V_1\| \wedge \ldots \wedge \|V_{|V|}\| \wedge \|\bar{E}_1\| \wedge \ldots \wedge \|\bar{E}_{|\bar{E}|}\|).$$

D.2.1 Converting containers

Recall that each container in the world model $V_i \in V$ is labeled with a set of properties: the label (denote as $\mathcal{L}_i$), entity ($\mathcal{E}_i$), quantity ($\mathcal{Q}_i$), attribute ($\mathcal{A}_i$), and unit ($\mathcal{U}_i$). Note that the unit property is optional depending on whether the entity $\mathcal{E}_i$ is countable or not. If the entity is countable, the container is mapped to a definition of a set:

$$\begin{aligned}\|V_i\| = {} & \texttt{Owner}(v_i, \mathcal{L}_i) \\ & \wedge \texttt{Measure}(v_i, \|\mathcal{Q}_i\|) \\ & \wedge \forall x \in v_i(\mathcal{E}_i(x) \wedge \mathcal{A}_i(x)) \wedge \|E^{\texttt{PW},i}\|,\end{aligned}$$

where $E^{PW,i} \subseteq E$ is the set of edges of type PARTWHOLE whose target vertex is i. Otherwise, if the entity is uncountable:

$$\begin{aligned}\|V_i\| = {} & \texttt{Owner}(v_i, \mathcal{L}_i) \wedge \mathcal{E}_i(v_i) \wedge \mathcal{A}_i(v_i) \\ & \wedge \texttt{Measure}(v_i, \texttt{Quantity}(\|\mathcal{Q}_i\|, \mathcal{U}_i)) \\ & \wedge \|E^{\texttt{PW},i}\|.\end{aligned}$$

Note that the attribute and unit properties may be omitted, and if the container v_i is missing a property, the corresponding conjunct is omitted as well (e.g., if the container is missing an attribute property, the conjunct $\mathcal{A}_i(\cdot)$ is omitted). Each quantity $\mathcal{Q}_i$ is mapped as follows:

$$\|\mathcal{Q}_i\| = \begin{cases} \mathcal{Q}_i, & \text{if } \mathcal{Q}_i \in \mathbb{R}, \\ u_j, & \text{if } \mathcal{Q}_i = x_j \text{ for some } x_j \in U. \end{cases}$$

Unlike other relations, the semantics of PARTWHOLE edges are not independent of each other, and so we define them here as a special case:

$$\|E^{\texttt{PW},i}\| = \texttt{PartWhole}(\{v_{s_1}, v_{s_2}, \ldots\}, v_i),$$

where s_j is the index of the source vertex of the edge $E_j^{\texttt{PW},i}$, and so $\{v_{s_1}, v_{s_2}, \ldots\}$ is the set of the source vertices of the PARTWHOLE edges with target vertex i. In section D.3, we provide axioms that define the semantics of each relation, including `PartWhole`.

D.2.2 Converting relations

Each relation $\bar{E}_i \in \bar{E}$ is also converted into a conjunction. Let s_i be the index of the source vertex of $\bar{E}_i$, and similarly let t_i be the index of the target vertex.

If the edge $\bar{E}_i$ is labeled as TRANSFER, it may have the following properties: the sender (denote as $\mathcal{S}_i$), recipient ($\mathcal{R}_i$), entity ($\mathcal{E}_i$), quantity ($\mathcal{Q}_i$), attribute ($\mathcal{A}_i$), and unit ($\mathcal{U}_i$). Similarly to containers, the entities in relations may be countable or uncountable. For brevity, we only show the conversion for the case where the entities are countable, but the conversion of uncountable quantities mirrors that shown for containers above. In this case, the TRANSFER edge is converted:

$$\begin{aligned}\|\bar{E}_i\| = {} & \texttt{Transfer}(e_i) \\ & \wedge \texttt{Source}(e_i, v_{s_i}) \wedge \texttt{Target}(e_i, v_{t_i}) \\ & \wedge \texttt{Sender}(e_i, \mathcal{S}_i) \wedge \texttt{Recipient}(e_i, \mathcal{R}_i) \\ & \wedge \exists r(\texttt{Arg}(e_i, r) \wedge \texttt{Measure}(r, \|\mathcal{Q}_i\|) \\ & \quad \wedge \forall x \in r(\mathcal{E}_i(x) \wedge \mathcal{A}_i(x))).\end{aligned}$$

If the edge $\bar{E}_i$ is labeled as RATE, it may have the following properties: the entity ($\mathcal{E}_i$), quantity ($\mathcal{Q}_i$), attribute ($\mathcal{A}_i$), and unit ($\mathcal{U}$). Then, the edge is converted:

$$\begin{aligned}\|\bar{E}_i\| = {} & \texttt{Rate}(e_i) \\ & \wedge \texttt{Source}(e_i, v_{s_i}) \wedge \texttt{Target}(e_i, v_{t_i}) \\ & \wedge \exists r(\texttt{Arg}(e_i, r) \wedge \forall y \in r(\texttt{Measure}(y, \|\mathcal{Q}_i\|) \\ & \quad \wedge \forall x \in y(\mathcal{E}_i(x) \wedge \mathcal{A}_i(x)))).\end{aligned}$$

Finally, if the edge $\bar{E}_i$ is labeled as COMPARISON, it may have the following properties: the type ($\mathcal{T}_i \in \{\texttt{Add}, \texttt{Mul}\}$), quantity ($\mathcal{Q}_i$), and unit ($\mathcal{U}_i$). Then, the edge is converted:

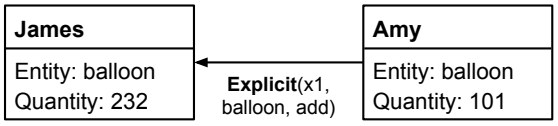
$$\begin{aligned}\|\bar{E}_i\| = {} & \texttt{Comparison}\mathcal{T}_i(e_i) \\ & \wedge \texttt{Source}(e_i, v_{s_i}) \wedge \texttt{Target}(e_i, v_{t_i}) \\ & \wedge \texttt{Arg}(e_i, \|\mathcal{Q}_i\|).\end{aligned}$$

Note that in the above, the sender, recipient, attribute, and unit properties are optional. If the relation is missing any property, the corresponding conjunct is omitted (e.g., if the attribute property is missing, the corresponding term $\mathcal{A}_i(x)$ is omitted).

See Figure 11 for an example application of the above conversion function.

Natural language representation:
"James has 232 balloons. Amy has 101 balloons. How many more balloons does James have than Amy?"

MATHWORLD representation:



First-order logic representation:

$$\begin{aligned}&\exists v_1 \exists v_2 \exists e_1 \exists u_1 (\\ &\ \texttt{Owner}(v_1, \texttt{James})\\ &\ \wedge \texttt{Measure}(v_1, 232) \wedge \forall x \in v_1.\texttt{balloon}(x)\\ &\ \wedge \texttt{Owner}(v_2, \texttt{Amy})\\ &\ \wedge \texttt{Measure}(v_2, 101) \wedge \forall x \in v_2.\texttt{balloon}(x)\\ &\ \wedge \texttt{ComparisonAdd}(e_1)\\ &\ \wedge \texttt{Source}(e_1, v_2) \wedge \texttt{Target}(e_1, v_1)\\ &\ \wedge \exists r(\texttt{Arg}(e_1, r) \wedge \texttt{Measure}(r, u_1)\\ &\quad \wedge \forall x \in r.\texttt{balloon}(x)))\end{aligned}$$

Figure 11: Example of a math story problem with its equivalent representations as a world model graph and in first-order logic.

D.3 Semantics of relations and predicates

We define the semantics of each relation, starting with the RATE relation:

$$\begin{aligned}&\forall e \forall v_s \forall v_t \forall r(\texttt{Rate}(e) \wedge \texttt{Arg}(e, r)\\ &\quad \wedge \texttt{Source}(e, v_s) \wedge \texttt{Target}(e, v_t)\\ &\rightarrow \texttt{Partition}(r, v_s) \wedge\\ &\quad \exists m(\texttt{Measure}(r, m) \wedge \texttt{Measure}(v_t, m))),\end{aligned}$$

where $\texttt{Partition}(r, v_s)$ denotes that r is a *partition* of the set v_s: r is a set of disjoint subsets of v_s such that their union is equal to v_s. More precisely:

$$\forall x \forall y \Big(\texttt{Partition}(x, y) \leftrightarrow \forall z, z' \in x(z \neq z' \rightarrow z \cap z' = \varnothing) \wedge y = \bigcup_{z \in x} z\Big).$$

We also use the notion of a partition to define the semantics of the TRANSFER relation:

$$\begin{aligned}&\forall e \forall v_s \forall v_t \forall r(\texttt{Transfer}(e) \wedge \texttt{Arg}(e, r)\\ &\quad \wedge \texttt{Source}(e, v_s) \wedge \texttt{Target}(e, v_t)\\ &\rightarrow \exists z(\texttt{Owner}(v_s, z) \wedge \texttt{Owner}(v_t, z)\\ &\quad \wedge \texttt{Recipient}(e, z)\\ &\quad \wedge \texttt{Partition}(\{r, v_s\}, v_t))\\ &\vee \exists z(\texttt{Owner}(v_s, z) \wedge \texttt{Owner}(v_t, z)\\ &\quad \wedge \texttt{Sender}(e, z)\\ &\quad \wedge \texttt{Partition}(\{r, v_t\}, v_s))).\end{aligned}$$

We define the semantics of COMPARISONADD:

$$\begin{aligned}&\forall e \forall v_s \forall v_t \forall m_s \forall m_t \forall r(\\ &\quad \texttt{ComparisonAdd}(e) \wedge \texttt{Arg}(e, r)\\ &\quad \wedge \texttt{Source}(e, v_s) \wedge \texttt{Target}(e, v_t)\\ &\quad \wedge \texttt{Measure}(v_s, m_s) \wedge \texttt{Measure}(v_t, m_t)\\ &\rightarrow m_s + r = m_t).\end{aligned}$$

COMPARISONMUL is defined similarly. Finally, we define PARTWHOLE as a simple set partition:

$$\begin{aligned}&\forall v_t \forall X(\\ &\ \texttt{PartWhole}(X, v_t) \leftrightarrow \texttt{Partition}(X, v_t)).\end{aligned}$$

E Annotation Details

E.1 Data preprocessing

We segment all sentences into smaller independent clauses when possible. This is done in order to create simpler units of training data for a semantic parser. We use the Berkeley Neural Parser (Kitaev and Klein, 2018; Kitaev et al., 2019) for this task, splitting sentences recursively at the two coordinating conjunctions *and* and *but*.[14] Over the three datasets we consider, 302 sentences are split in this way. Additionally, some question sentences start with a subordinate clause that introduces new information, like "If Alice bought 3 more apples today, how many apples did she end up with?". We split these into a declarative clause and an interrogative clause, and remove the leading subordinating conjunction.

[14] Some phrases with a trailing preposition are split erroneously in this way, like "Sally picked 7 lemons and Mary picked 9 lemons from the lemon tree" is split into "Sally picked 7 lemons" and "Mary picked 9 lemons from the lemon tree", pointing to the challenges of prepositional phrase attachment in neural constituency parsing (Sopena et al., 1998). We detect and correct such cases manually.

E.2 Annotation scheme and tool

As mentioned in § 3, MATHWORLD considers logical forms at the sentence level. Hence, we must also annotate the world model graphs incrementally, sentence by sentence. This is done via a drag-and-drop annotation tool, ANT-NLP, built specifically for the purpose of this work.[15] When annotating a problem in the tool, annotators get to build the graph incrementally one sentence at a time. Each sentence is given in a separate page, as shown in Fig. 12, and the graph from the previous sentence is carried over to the next. We save all incremental world models, as they set the basis for the sentence linearization described in App. F.1. The incremental world models are stored in json graph format.[16]

We want annotators to include all information included in the text that fits MATHWORLD, irrespective of the relevance to the question. Therefore, in order not to create any bias stemming from the question, we hide the question sentence until all other preceding parts have been annotated.

We ask annotators to follow the ordering that information is given within each sentence when adding containers and relations. For instance, a sentence "Alice has 3 apples and 4 oranges" should first be assigned a container for apples and then be assigned a container for oranges. This allows us to preserve the ordering of the text when linearizing logical forms for training data. To capture this ordering we annotate IDs for the containers and relations. The space of IDs is the set of natural numbers, and is shared between containers and relations. Ids are incremented automatically in the tool as annotators add new containers or relations.

The tool includes options to flag problems that require background knowledge, or where the annotator is uncertain about their annotation. They can additionally add a free text comment about their annotation for a particular problem.

E.3 Procedure

Annotation is performed by external workers, who are taught to be familiar with the semantics of MATHWORLD. We employ two annotators, hired from a small annotation company in India.[17] At the time of annotation, both of them were undergraduate students in technical universities. As support material, annotators are given a comprehensive guideline document, a set of example annotations and a video showcasing how annotation is performed using the tool. We follow three phases for annotation:

1. **Training phase**. This phase is for annotators to learn the formalism. They are given batches of $5-7$ problems at a time to annotate independently. These annotations are then discussed and, if needed, corrected in a follow-up meeting. The initial batches consist of simple problems, both conceptually and linguistically. After the annotators can successfully annotate these simple problems, they are gradually given more challenging ones. This phase ends when all annotators can successfully annotate most problems across all datasets.

2. **Agreement phase**. Here, annotators are given the same set of 90 problems, with 30 from each dataset. They are asked to annotate these independently. This set is used to measure agreement between annotators.

3. **Scale-up phase**. Here, annotators are given separate datasets to annotate on their own. Some of these problems are overlapping in order to allow for agreement analysis.

E.4 Agreement analysis

We give further details on the agreement analysis of the 125 overlapping problems discussed in § 4. As mentioned, there were 18 isomorphic disagreements between the two annotators (i.e., not weakly equivalent). Out of these, 7 were due to structural ambiguity (App. B.2), 1 was due to a type of error that was fixed during annotation check (see below), 8 due to a type of error for which a problem would be discarded during annotation check, and 2 less serious errors that would not be detected during the annotation check. Most errors were attributed to the same annotator. Ground truth data for overlapping annotations were thus taken from the annotated set of the annotator with the higher performance on the overlapping problems.

[15] Although the annotation tool is built specifically for annotating world models in MATHWORLD, we believe it could with relative ease be adapted to annotation of potential world models for other domains as well. The tool will be shared with other researchers by request.

[16] https://jsongraphformat.info/

[17] There was initially a third annotator involved. However, this annotator dropped out during phase 3 as described below. At that time, it would have required a considerable time investment to hire and train yet another annotator, and so instead, we had one of the two other annotators cover up.

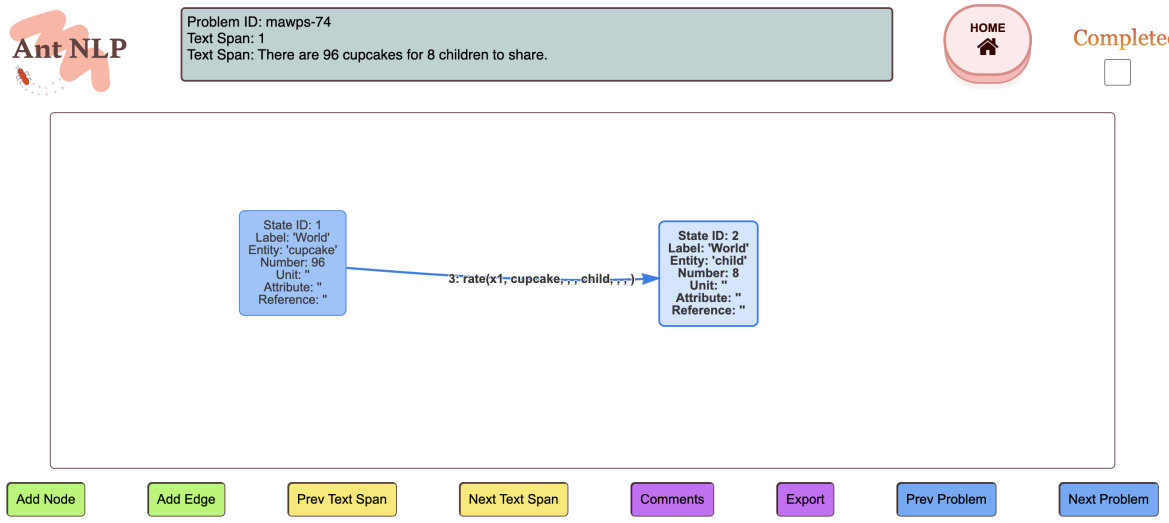


Figure 12: Snapshot of the annotation tool interface of ANT-NLP for a particular sentence. The annotator has the option to add nodes (containers), add edges (relations) between an existing pair of nodes, navigate between text spans, navigate between math story problems, add comments and flags associated with the math story problem, and export the math story problem when its annotation is completed. A problem is considered completed when all text spans have been annotated and a variable referencing the answer has been added for the final text span. Only then can the problem be marked in the top-right corner and be exported to json format. Furthermore, a central dashboard (not shown here) allows the annotator to get an overview over the progress and navigate between problems. Question sentences are not shown in the central dashboard in order to alleviate bias.

There were 46 problems that had a weak equivalence agreement, but not a strong equivalence agreement. Some of these were due to errors and some were due to property ambiguity (App. B.1). The errors were mostly incurred from entering an incorrect property, seemingly by carelessness. Several such cases could be detected and corrected as they led to errors when parsing the world model json file or when applying the reasoner to the world model (App. H). Cases of property ambiguity were often due to the attribute property.

We additionally stratified agreement across relation type. Problems with COMPARISON relations seemed to have the lowest weak equivalence agreement, followed by RATE and TRANSFER. For strong equivalence agreement on the other hand, RATE problems had the lowest agreement, followed by TRANSFER and then PARTWHOLE.

E.5 Annotation check and correction

We performed the following checks of the annotations: whether the json could be parsed into a well-formed world model, whether applying the deterministic reasoner (App. H) would produce the correct answer and whether the annotator had flagged the problem with low confidence or provided a free text comment. Based on these we were able to detect and correct several faulty annotations. Some common errors were: entering the wrong number, entering the wrong reference variable, forgetting to the enter the reference variable, orienting the edge in the wrong direction and misspelling label names. Such errors could easily be corrected. Other more fundamental errors that could not be easily fixed led to discarding the annotation. We also spotted some cases of wrong annotated answers stemming from the original dataset, which were corrected.

F Conversion between World Model Graph and Logical Form

As mentioned in § 5.1, an integral part of our proposed MATHWORLD solver framework, and working with MATHWORLD more generally as in § 5, is the conversion between world models g and logical forms m. In this appendix section we provide details of both directions of this conversion. Both directions of the conversion are lossy to some small degree, as is mentioned in footnote 18.

F.1 World model graph to logical form

Each logical form can be viewed as a incremental graph update that consists of containers and relations based on a sentence in the problem text,

which is represented as a text sequence.

Containers and relations have varying arity, depending on which properties are present. This opens two possibilities. We may either split them into forms for each set of properties and have the property names explicit in the signatures (e.g., containers would have one representation each with arity 3 and 5, and two representations with arity 4), or keep the property ordering consistent and give a default null token for missing properties. We opt for the latter, and set the default null token to be `none`.

We define the following predicates:

- `container(label, quantity, entity, attribute, unit)`
- `transfer(recipient label, sender label, quantity, entity, attribute, unit)`
- `rate(label, quantity, source entity, source attribute, source unit, target entity, target attribute, target unit)`
- `difference(target label, source label, quantity, target entity, target attribute, target unit, source entity, source attribute, source unit)`
- `explicit(target label, source label, quantity, target entity, target attribute, target unit, source entity, source attribute, source unit)`
- `part(whole label, whole entity, whole attribute, whole unit, part1 label, part1 entity, part1 attribute, part1 unit, . . . , partn label, partn entity, partn attribute, partn unit)`

Note that for COMPARISON, the "type" property is lifted out and its value replaces "comparison" as the name of the predicates. We replace "add" and "times" by "difference" and "explicit", respectively, for practical reasons: We do not want the name of the operator that might be required to solve the problem to be confounded with the name of the predicate. Further note that the above predicates are overloaded in comparison to the ones mentioned in § 3. The reason for that is that we require additional information in order to match the linearization to existing incremental graphs (the other direction of the conversion, described in App. F.2). For instance, consider two disconnected containers in a world model graph. If one wished to present them as connected with RATE, it would be sufficient to provide the quantity property to the rate. See, e.g., how in Fig. 1, quantity is provided as the only property in the RATE relation. The other properties given above would be redundant as they are already given in the containers. For a model to be able to orient that rate, however, it needs the additional information to match to the two existing containers.

Note that in the case of TRANSFER, there may be two associated edges in the graph if the properties "recipient label" and "sender label" both take values other than `none`. However, these are both represented by a single `transfer` predicate as above. PARTWHOLE is the only relation whose arity varies, reflecting the number of subsets present in the PARTWHOLE construction. An alternative would have been to have one predicate per edge, but that would have introduced redundancy.[18]

A sentence-level logical form often contains multiple components of the above. In these cases, we follow the ordering as introduced in text, in line with the annotated IDs. If a relation is added together with its source and/or target containers, then the containers must always precede the relation in the ordering. We enforce that the source container always precedes the target container.

As an example, the logical form of the sentence "In a friend group there are 5 football players and 3 tennis players" is:

```
container(friend group, 5, player,
football, none)
container(friend group, 3, player,
tennis, none)
```

Finally, a world model graph may have containers that have not been explicitly introduced in text.

[18] However, a drawback of our PARTWHOLE representation is that it assumes that all the part-whole edges are always introduced together in the same sentence. While this is mostly the case for the data we observe, we found the following exception: "Next on her list are the homeless people where she spent a total of $900.00. She gave $325.00 to the first set of homeless families and $260.00 to the second set of families. How much did she give to the last set of homeless families?". This is one example showing that the conversion is slightly lossy.

For instance, the two sentences "Alice has 5 apples. She ate 2 of them." will be represented by a world model with two containers and a TRANSFER edge, but only the source container is explicitly mentioned in text (in the first sentence). When writing the world model graph as a logical form, we therefore discard the target container in this case. In general, this is done by discarding all containers that do not hold an explicit quantity, unless the sentence is interrogative. For interrogative sentences we want the logical form to represent the reference variable.

F.2 Logical form to world model graph

We now consider the other direction, namely that we have a sequence of logical forms $m_1, \ldots, m_n$ on the form described in the previous section and wish to convert them to a world model graph g.

For m_1, we can trivially convert the logical form to a graph. Note that the relation predicates specify the properties needed to match the relation to containers as well, so if there is a relation predicate in the logical form but no source and/or target container, we can simply create those. For subsequent logical forms, we match the logical form to the graph created from preceding sentences. For relations, we must make sure that we do not create new containers linked by that relation, if any or both of those containers are already existing in the world model. We thus first match the properties corresponding to the source and target containers in the relation predicate to any possibly existing containers, and only create new ones if none are found. In addition, some sentences will just supply an update of an unknown quantity to a known value. In these cases, we do not create a new container, but match the quantity to one already existing so that we can preserve the structural information of that container. We remark that in case that the matching with already existing containers in the world model returns multiple options, we default to the most recently created one. This turns out to work well for most cases, but could be one source of loss.

The reference variable corresponds to the logical form of the last sentence: Interrogative sentences are mapped to logical forms the same way as declarative sentences, and the reference variable is taken as the variable in the container or relation that matches the question's logical form.

Finally, for predicted logical forms, we first check the logical forms for syntactic well-formedness, keeping only the parts of the logical form that are well-formed. An additional (weak) check for semantic well-formedness may match the properties to the vocabulary of the MSP, along with special tokens like "none", "world" etc.

G Difficult Cases to Parse

We estimate a high coverage of our formalism among MSPs. However, although a problem might be within semantic and conceptual coverage of MATHWORLD, the text itself might prove challenging for a parser to interpret. Here, we present two problems that are captured by MATHWORLD that put a high burden on the parser.
First, consider the following problem:

> The teacher distributes 4 candies to 2 students. Student A now has 2 more candies than Student B. Both students had 0 candies to begin with. How many candies does Student A have?

In this problem there is a transfer involved in the first sentence. The recipient of the transfer is not a single independent container however, but a set of two students. We have no information on how many candies these two students have individually, but we know that they collectively got 4 more than they had before. To capture this, we may represent both students as a container with a PARTWHOLE relation to the individual students, which will be the recipient of the TRANSFER. The whole problem is assigned the world model in Fig. 13. This is a faithful and correct world model, but the first sentence puts a high burden on the semantic parser: It must add 8 containers and 6 relations.

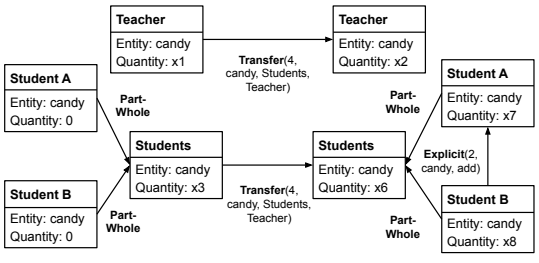


Figure 13: Hypothetical world model associated with a problem text that describes a subset-superset relation over containers.

Next, consider the following problem (adapted from GSM8K):

> Zack decided to give his 3 friends 20 marbles each and kept 5. How many marbles did he initially have?

The first sentence conveys a lot of information. We must add a container for the total number of marbles that Zack possesses, with PARTWHOLE relations representing how many marbles Zack has left and how many his friends have. In addition, we know that there are three friends, which we represent with a RATE. See the world model in Fig. 14. However, the fact that Zack already possesses marbles is implicit from the text, and would be challenging for a parser to detect. As a partial remedy, we could introduce a "TransferEvenly" relation, which would represent a transfer of 20 to each container in a set. In this case, Zack's friends would each be represented in a container.

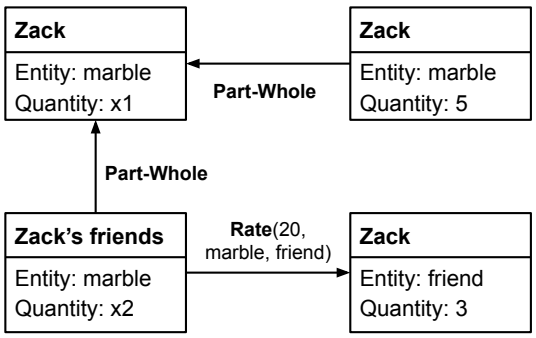


Figure 14: Hypothetical world model associated with a problem text that compresses a lot of information within a single sentence.

H Reasoner

The recursive solver takes as input a target variable and a set of visited equations. It takes all the equations containing the target variable and sorts them in increasing order of number of unknowns. Next, it iterates over the equations in this order. If the equation only has one unknown, that unknown must be the target variable. The function then solves for the target variable and outputs the numeric value. Otherwise, it goes over the other free variables in the equation and applies the recursive function to those as target, with the equation added to the set of visited equations in order to prevent loops. Having solved for the other free variable, it substitutes its numeric value in the equation and solves for the target variable, if possible. We present pseudo-code for the deterministic reasoner in Alg. 1.

Note that this solver assumes a certain structure of the equations, namely, that a solution can be reached by solving a sequence of equations with one unknown. Such is indeed the case for the simple MSPs we consider. However, in the case of a general system of linear equations, this algorithm would fail as it cannot handle equations of more than one unknown. We opt for our recursive solution rather than Gaussian elimination due to runtime gains: for a system of n equations with n unknowns, Gaussian elimination runs in $\mathcal{O}(n^3)$, while our solution has worst-case complexity $\mathcal{O}(n^2)$.

Further note that if we extend r to be a set of variables, we can store the intermediate results in a table and get a dynamic program. This is not necessary in our case as we do not have overlapping sub-problems.

Algorithm 1: Deterministic recursive reasoner.

```
function recursiveReasoner(x, visited)
    /* Prepare equations containing x that
       have not been visited */
    eqs ← {equations containing x} \ visited
    /* Sort in increasing order of number of
       unknowns */
    eqs ← sort(eqs, # of unknowns, increasing)
    for eq ∈ eqs
        /* Go over equations in order */
        if solvable(x, eq)
            /* Solve for x if possible */
            x_val ← solve(x, eq)
        else  /* Otherwise, solve recursively */
            /* Go over other unknowns */
            for x' ∈ eq, x' ≠ x
                x'_val ← recursiveReasoner(x, visited + eq)
                /* Substitute unknown for value */
                eq ← substitute(eq, x', x'_val)
                if solvable(x, eq)
                    x_val ← solve(x, eq)
                    return x_val
    return x_val
```

I Experimental Details

I.1 Solving pipeline

Setup As our LLM we use Codex code-davinci-002. We design a prompt with 50 ground truth examples from MAWPS and ASDIV-A. One example consists of the source sentence, the target linearized logical form, as well as the source and target of the previous sentence in the same MSP, in order to allow the model to account for dependencies between sentences. These examples are handpicked to be representative of MATHWORLD. For every MSP, we then feed each sentence following the same pattern excluding the target as a suffix to the prompt, and sample the target output from Codex. The experiments were performed on the 18th of January 2023. The parameters used for

	MAWPS	ASDIV-A	SVAMP
Answer Acc (%)	33.8	26.9	11.1
Complete WM (%)	50.7	43.3	33.3
Weak Smatch (avg.)	0.76	0.68	0.59
Strong Smatch (avg.)	0.76	0.60	0.38

Table 5: Results on our test sets for the Codex few-shot learning model. Smatch scores (App. C) are averaged across all MSPs, including those where Codex produced an incomplete world model.

sampling were the following: temperature is set to 0, max tokens is 200, frequency and presence penalty are both left at 0 and we add an additional new line stop token (which is used in the prompt to end the ground truth logical forms.

World models are built incrementally using the method described in App. F.2. We apply the deterministic reasoner (App. H) to produce an answer.

Results We show the results in Table 5. Observe that on average less than half of the predicted world models result in an answer (i.e. are complete). The rest of the times the reasoner is either unable to solve for the reference variable (the system of equations induced by the world model is underdetermined) or the world model lacks a reference variable. Incorrect answers are often caused by slight permutations of the correct logical forms (e.g., Codex having swapped the sender and recipient in a TRANSFER relation). If we stratify the problems by relation type, we observe that the model has the highest answer accuracy for TRANSFER and RATE, while PARTWHOLE problems have the lowest answer accuracy. This is to be expected given that the information associated with PARTWHOLE problem is not often made explicit in text (§ 3.2.4).

I.2 Details on constrained generation

The GPT 3.5 Turbo generation experiments were performed on the 24th of May 2023. The model used was gpt-3.5-turbo-0301. The sampling parameters are the same as those used during parsing (App. I.1).

We display the results of the other five MSPs as mentioned in § 5.3 in Table 6. Observe that in all cases, the model is able to generate problems that are faithful to the concept, number and properties of the original world model (comparing the left column and the middle column). Further note that with a temperature parameter of 0, the generated problems are rather conservative. We leave for future work to explore the implications of the sampling parameters for the generated outputs. Finally, consider the right column, where we display the MSPs generated from augmented world models. Three of the generated examples are not completely faithful to how we augmented the world models. In the first example from the top, "Lexie's brother" is provided as the recipient property in the TRANSFER relation, but in the generated example Lexie's brother is the sender. In the third example from the top, we augment the world model with a RATE, but the model instead generates a transfer type MSP. In the last example, Bob is provided as sender while Josh is provided as recipient, but the model generates a problem with these values being swapped. The other two are faithful.

I.3 Details on prompting using synthetic questions

The GPT-3 probing experiments were performed on the 18th of January 2023. The model used was text-davinci-003. The sampling parameters used are the same as those used for Codex during parsing (App. I.1).

In Table 7, we present the templates used to create synthetic question-answer pairs for prompting large language models.

Original MSP	MSP generated from world model	Augmentation	MSP generated from augmented world model
Lexie's younger brother helped pick up all the paper clips in Lexie's room. He was able to collect 81 paper clips. If he wants to distribute the paper clips in 9 boxes, how many paper clips will each box contain?	Lexie's brother has 81 paper clips. He wants to put them in 9 boxes. How many paper clips will he put in each box?	RATE → TRANSFER	Lexie's brother had 81 paper clips. He gave 21 to Lexie. How many paper clips does Lexie's brother have now?
Kevin collected toys to use as prizes at the fair. He collected 14 stuffed animals. He also collected 18 frisbees and several yo-yos. Kevin has 50 prizes in all. How many yo-yos did Kevin collect?	Kevin won 50 prizes at the fair. He won 14 stuffed animals, 18 frisbees, and some yo-yos. How many yo-yos did he win?	PARTWHOLE → TRANSFER	Kevin has 14 stuffed animals. He gave 5 of them to his friend. How many stuffed animals does Kevin have now?
Mrs. Hilt wants to make a border around her garden. She needs 125 rocks to complete the border. She has 64 rocks. How many more rocks does she need to complete the border?	Mrs. Hilt was making a rock border around her garden. She had 125 rocks to use. She used 64 rocks to make the border. How many rocks did she have left?	TRANSFER → RATE	Mrs. Hilt is making a rock border around her garden. She has 125 rocks to use. She has already used 25 rocks. How many rocks does she have left?
Zoe's school sold 620 dollars in raffle tickets. If each ticket cost 4 dollars, how many tickets did they sell?	Zoe spent $620 on raffle tickets. Each ticket cost $4. How many tickets did she buy?	RATE → TRANSFER	Zoe had 620 dollars. She spent 100 dollars. How much money does Zoe have now?
Josh had 16 marbles in his collection. He lost 7 marbles. How many marbles does he have now?	Josh had 16 marbles. He lost 7 of them. How many marbles does Josh have now?	TRANSFER → TRANSFER	Josh has 16 marbles. He gave 7 marbles to Bob. How many marbles does Josh have now?

Table 6: Example of generated math story problems conditioned on world models in MATHWORLD. The left column shows the original math story problem, the middle column shows a math story problem generated conditioned on the ground truth world model of the original problem, and the right column shows a math story problem generated conditioned on a world model that has been created by augmenting the ground truth world model of the original problem. Sentences not faithful to the logical form are colored red.

containers	Q: How many {attr}{ent}s does {label} have? A: {quant} Q: What is the amount of {attr}{ent}s associated with {label}? A: {quant}
TRANSFER	Q: How many {ent}s are transferred from {sour} to {targ}? A: {quant}
COMPARISON (add)	Q: How many more {ent}s does {targ} have than {sour}? A: {quant}
COMPARISON (times)	Q: How much more {ent} does {sour} have than {targ}? A: {quant}
RATE	Q: How many {ent} does {targ} have per {sour}? A: {quant}
PARTWHOLE	Q: How many {sour} are part of {targ}? A: {quant}

Table 7: Templates to automatically create question-answer pairs for prompting. The templates are filled based on the information in the world model.

prompt types	pairs sourced from one MSP (one-shot), i.e., $x = 1$
(1) synth QAs (all at once)	Baker made 43 cakes and 114 pastries. If he sold 154 pastries and 78 cakes. Q: How many cakes does baker have? A: 43 Q: How many sold cakes are associated with baker? A: 78 Q: How many more pastries than cakes did baker sell? A: 76 Bobby had 19 pieces of candy. He ate 2 pieces of candy. Q: What is the amount of candys associated with bobby? A: 19 Q: How many candys are transferred from bobby? A: 2 Q: how many pieces of candy does he still have left? A:
(2) synth QAs (sent by sent)	Baker made 43 cakes and 114 pastries. Q: How many cakes does baker have? A: 43 Baker made 43 cakes and 114 pastries. If he sold 154 pastries and 78 cakes. Q: How many sold cakes are associated with baker? A: 78 Baker made 43 cakes and 114 pastries. If he sold 154 pastries and 78 cakes. Q: How many more pastries than cakes did baker sell? A: 76 Bobby had 19 pieces of candy. Q: What is the amount of candys associated with bobby? A: 19 Bobby had 19 pieces of candy. He ate 2 pieces of candy. Q: How many candys are transferred from bobby? A: 2 Bobby had 19 pieces of candy. He ate 2 pieces of candy. Q: how many pieces of candy does he still have left? A:
(3) original MSP QAs	Baker made 43 cakes and 114 pastries. If he sold 154 pastries and 78 cakes. Q: How many more pastries than cakes did baker sell? A: 76 Bobby had 19 pieces of candy. He ate 2 pieces of candy. Q: how many pieces of candy does he still have left? A:

Table 8: We experiment with three different types of prompts. They are displayed for the one-shot case in which one MSP in addition to the one we are trying to solve is provided in the prompt. In the above case, the model is tasked with making inference on the problem "Baker made 43 cakes and 114 pastries. If he sold 154 pastries and 78 cakes. How many more pastries than cakes did baker sell?".

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Yes, after the Conclusion (sec 7)

☑ A2. Did you discuss any potential risks of your work?
Yes, after the Limitations in the Ethics Statement

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Yes, Abstract (sec 0) and Introduction (sec 1)

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Sec 4 and details in App E

☑ B1. Did you cite the creators of artifacts you used?
Yes, Introduction (sec 1) and Data Collection (sec 4)

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Datasets were published in previous *ACL conferences and are free to be used for research purposes.*

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Our datasets used were created for research purposes and we have not used it for any other purposes.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Our data uses fictional characters.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Our dataset consists of annotations over previously published datasets. We trust the such documentation is given in the original papers.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Sec 4

C ☑ Did you run computational experiments?

Yes, sec 5.3 and sec 6

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
No, we used pre-trained large language models in our experiments

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Yes, in sec 5.3, sec 6 and app I

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not applicable. Left blank.

☒ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
No, but we submitted code in which such details are included.

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Yes, sec 4 and app E

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Yes. We paid a company for annotation, which in turn paid the annotators. We would be happy to give more details on their salaries if necessary for the camera-ready version.

☒ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not in the paper. We did have such discussions with the annotators prior to start of annotation, but not in writing.

☒ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No, we were unaware of this possibility. Although we did make sure that data collection complied with our institution's requirements.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.