Semantic Graphs for Syntactic Simplification: A Revisit from the Age of LLM

Peiran Yao and **Kostyantyn Guzhva** and **Denilson Barbosa**
Department of Computing Science
University of Alberta
{peiran, denilson}@ualberta.ca

Abstract

Symbolic sentence meaning representations, such as AMR (Abstract Meaning Representation) provide expressive and structured semantic graphs that act as intermediates that simplify downstream NLP tasks. However, the instruction-following capability of large language models (LLMs) offers a shortcut to effectively solve NLP tasks, questioning the utility of semantic graphs. Meanwhile, recent work has also shown the difficulty of using meaning representations merely as a helpful auxiliary for LLMs. We revisit the position of semantic graphs in syntactic simplification, the task of simplifying sentence structures while preserving their meaning, which requires semantic understanding, and evaluate it on a new complex and natural dataset. The AMR-based method that we propose, AMRS^3, demonstrates that state-of-the-art meaning representations can lead to easy-to-implement simplification methods with competitive performance and unique advantages in cost, interpretability, and generalization. With AMRS^3 as an anchor, we discover that syntactic simplification is a task where semantic graphs are helpful in LLM prompting. We propose AMRCoC prompting that guides LLMs to emulate graph algorithms for explicit symbolic reasoning on AMR graphs, and show its potential for improving LLM on semantic-centered tasks like syntactic simplification.

1 Introduction

Frameworks for symbolic sentence meaning representations, exemplified by UCCA (Abend and Rappoport, 2013), Abstract Meaning Representation (AMR) (Banarescu et al., 2013), and UMR (Gysel et al., 2021), provide varying levels of abstraction away from the lexical and syntactical structures of natural language sentences, commonly in the form of *semantic graphs* (Oepen et al., 2020). Compared to dense representations such as semantically meaningful embeddings (Reimers and Gurevych, 2019), representing the meaning of a sentence as a graph allows for the use of classical (and explainable) algorithms (e.g. traversal and partition) to ease the development of more controllable and interpretable methods for semantic-focused NLP applications, including but not limited to text simplification (Sulem et al., 2018), question answering from knowledge bases (Kapanipathi et al., 2021), and text-style transfer (Shi et al., 2023).

Meanwhile, large language models (LLMs), representatively the ChatGPT (Ouyang et al., 2022; OpenAI, 2023) and Llama (AI@Meta, 2024) families, have demonstrated prevailing performance in the aforementioned applications. Their instruction following capability (Ouyang et al., 2022) enables training-free adaptation to specific tasks, which, in terms of the burden for implementation, is at a similar level to that of writing graph algorithms on top of semantic graphs. This prompts researchers to rethink the role of symbolic meaning representations in the era of LLMs, and to explore the potential of combining the two paradigms, with the negative findings that directly appending AMR to the input of LLMs is not beneficial, if not harmful, in many tasks (Jin et al., 2024).

Along these lines, we study the task of syntactic simplification and aim to answer two research questions: **RQ1** (§4): Can state-of-the-art meaning representing semantic graphs provide a light-weight, easy-to-implement, and interpretable alternative to LLMs for this task? **RQ2** (§5): Can it be helpful to supply semantic graphs as auxiliaries to LLMs to improve their performance on this task?

Syntactic simplification, including variants like Split and Rephrase (Narayan et al., 2017), sentence splitting (Niklaus et al., 2019) and Gao et al. (2021), is a type of text simplification task that aims to rewrite sentences to reduce the syntactic complexity while preserving its meaning, typically operationalized by converting a complex text into a

Code, models, and data are available at `https://github.com/U-Alberta/AMRS3`.

Proceedings of TextGraphs-17: Graph-based Methods for Natural Language Processing, pages 105–115
August 15, 2024 ©2024 Association for Computational Linguistics

set of atomic sentences with simpler structures. It has practical applications in improving text accessibility for less-proficient readers (Watanabe et al., 2009), improving weaker NLP pipelines (Niklaus et al., 2023), and detecting hallucination in complex statements (Hou et al., 2024). Despite modifying syntactic structures as the outcome, the task is inherently semantic-focused, as sentences are expected to be atomic in meaning and semantically equivalent to the original complex sentence, making semantic graphs a natural choice as an intermediate.

To answer RQ1, we propose AMRS^3 (shorthand for **A**bstract **M**eaning **R**epresentation for **S**yntactic **S**entence **S**implification), a simple yet effective graph-based algorithm that breaks down the AMR graph of a complex sentence into a set of subgraphs, each corresponding to a semantic unit. The subgraphs then guide the generation of simpler sentences which form the final output. AMR is chosen as it is the meaning representation that receives more attention in recent developments of treebanks (Knight et al., 2020), parsing (Xu et al., 2023), text generation (Bai et al., 2022), and cross-lingual adaptation (Wein and Schneider, 2024), and it reflects the state-of-the-art of graph-based meaning representation. We demonstrate that with a well-developed semantic graph like AMR, a syntactic simplification system can be derived from simple rules as a lightweight alternative to LLMs. Evaluations on the synthetic WebSplit (Narayan et al., 2017) dataset and real-world complex sentences from a Humanities corpus (Brown et al., 2022) show that AMRS^3 yields simplifications that are comparable to those of complex existing systems and LLMs in terms of both syntactic simplicity and meaning preservation, while enjoying, in principle, the merits of simplicity, interpretability, and language-neutrality.

It is unsurprising that LLM outperforms symbolic methods in syntactic simplification (Ponce et al., 2023). We aim to answer RQ2 and see whether AMR still has merits as an auxiliary to LLMs (namely GPT-3.5 and Llama-3-8B) in this task. Contrary to Jin et al.'s (2024) report that directly adding AMR to the input is harmful in many tasks, we find syntactic simplification slightly benefits from the auxiliary AMR inputs. We investigate what elements of AMR are helpful to LLMs in our case, and find that prompting in Chain-of-Code (Li et al., 2023) style allows LLMs to emulate the execution process of AMRS^3 and perform reasoning over AMR graphs, providing insights on how AMR can be made a useful auxiliary for LLMs in this and other semantic-centered tasks.

We contribute a LLM-era's perspective on graphical approaches toward the long-standing task of syntactic simplification: the task is benchmarked on a hard and natural complex sentence dataset that we construct; we offer a reference point of the latest developments in symbolic meaning representations for the task; and finally, we provide insights on the role of symbolic meaning representations in the era of LLMs that complement recent work.

2 Related Work

Text Simplification. Syntactic simplification is a subtask of automated text simplification, the problem of improving text readability and understandability while retaining information, that has a wide spectrum of forms (Al-Thanyyan and Azmi, 2022): complementing syntactic simplification, lexical simplification focuses on replacing complex words with simpler synonyms (Paetzold and Specia, 2017). Meanwhile, summarization is another form of simplification that removes superfluous information or unnecessary details (Nenkova et al., 2011). Given the difference in focuses, general text simplification benchmarks and evaluations such as those of Maddela et al. (2023) and Alva-Manchego et al. (2021) do not directly apply to syntactic simplification in isolation.

Syntactic Simplification. Prior to LLMs, syntactic simplification was commonly modeled as a sequence-to-sequence task where systems are trained on parallel corpora synthesized from knowledge graphs (Narayan et al., 2017), mined from Wikipedia (Botha et al., 2018) and translations (Kim et al., 2021), or crowd-sourced (Gao et al., 2021). These specialized models struggle to generalize to unseen data, which our work demonstrates is solvable with simple rule-based methods combined with a powerful semantic representation (AMR). This combination is admittedly not a new idea: DisSim (Niklaus et al., 2023) is a performative simplification system, yet it relies on a larger set of expert-crafted lexical rules that is not as simple and transferrable as our approach. DSS (Sulem et al., 2018) uses UCCA as the semantic representation, and we inherit its idea and build on AMR which is more powerful. Ponce et al. (2023) evaluates fine-tuning LLMs on a split-and-rephrase dataset, while our analysis on LLM focuses on the zero-shot instruction-following setting.

Symbolic Reasoning for LLM. Jin et al. (2024) suggest that adding serialized AMR graphs to the input of LLM in a direct manner is not effective in prompting LLM to perform implicit reasoning over the AMR graph. This is consistent with the observation that LLM needs guidance on task decomposition to perform complex reasoning (Wei et al., 2022) such as manipulating AMR. However, symbolic data, such as code and AMR, likely has the potential to benefit LLM (Yang et al., 2024). Our work investigates whether methods prompting LLM to perform explicit symbolic reasoning, such as Chain-of-Code (Li et al., 2023), can be more helpful than direct prompting as in Jin et al. (2024). An alternative to prompting, which is beyond the scope of this work, is to fine-tune the LLM across symbolic reasoning tasks including AMR to improve its reasoning ability Xu et al. (2023).

3 Task Setting

In our studies, we consider only the hard cases of syntactic simplification (Niklaus et al., 2019) where a complex sentence needs to be simplified into multiple ones (typically more than two). To the best of our knowledge, there is a lack of high-quality benchmarking datasets for this task. Synthetic and mined datasets such as WikiSplit (Botha et al., 2018) and BiSECT (Kim et al., 2021) come with reference simplifications, but they only focus on binary splits, with WebSplit (Narayan et al., 2017) being an exception. The manually labeled DeSSE dataset (Gao et al., 2021) is in the domain of student essays where the sentences are relatively simple. The usefulness of the provided reference simplifications is limited, as they are often not of high quality and the granularity of the splits is predefined by the dataset generation process. This motivates us to use reference-less evaluation metrics to assess the quality of the generated splits from the aspects of simplicity and meaning preservation separately (Cripwell et al., 2024), and create a natural and realistic dataset of complex sentences.

Datasets. As an instance of traditionally used benchmark datasets, we use WebSplit's test set (WEBSPLIT), with the caveat that it is unnatural. Meanwhile, we mine for sentences with high word and entity mention counts from the Orlando bibliography corpus (Brown et al., 2022), which results in a set of structurally-complex realistic sentences expressing rich relations, written by digital humanists (ORLANDO). Table 1 provides a summary of the size and nature of the two datasets.

Assessing Simplicity. We measure the opposite of simplicity, the syntactic complexity of sentences, by L2SCA (Lu, 2010), a widely adopted set of features that highly correlate with human judgments of syntactic complexity. It measures 14 features from five syntactic aspects. For the clarity of presentation, from each aspect, we choose one feature with the highest correlation with human judgments.

Assessing Meaning Preservation. Following recent work (Ponce et al., 2023; Cripwell et al., 2024), we use BERTScore Recall (Zhang et al., 2020) computed with DeBERTa-NLI[1] (He et al., 2021) to assess whether the meaning of the original sentence is preserved in the simplification. We do not follow previous work relying on BLEU as its lack of semantic understanding is criticized for being particularly unsuitable for simplification tasks (Sulem et al., 2018; Alva-Manchego et al., 2021).

4 AMR for Rule-based Simplification

We argue that Abstract Meaning Representation (AMR) is suitable for syntactic simplification, as its abstraction away from surface strings and syntactic structures (Oepen et al., 2020) allows us to define concise and interpretable rules for simplification, and its well-developed resources for parsing and generation provide a guarantee for high-quality conversions between text and graphs. This leads to the development of AMRS^3 , an AMR-based system for syntactic simplification that is driven by a handful of simple and interpretable rules.

4.1 Rule Set

As illustrated in Figure 1, AMRS^3 at a high-level projects a complex sentence to the space of AMR graphs using a semantic parser, and then breaks down the AMR graph of a complex sentence into a set of subgraphs, each corresponding to a semantic unit, which are then realized into simpler sentences using an AMR-to-text model.

An AMR graph (as in Fig. 1) is a rooted directed acyclic graph where nodes represent *concepts* and edges represent *relations* between concepts (Banarescu et al., 2013). Non-leaf nodes in AMR are usually *core concepts* (highlighted nodes in Fig. 1)

[1] microsoft/deberta-xlarge-mnli as suggested by latest BERTScore guidelines.

Dataset	Size	Example
WEBSPLIT	938	*Addiction journal is about addiction and is published by Wiley-Blackwell which has John Wiley & Sons as the parent company .*
ORLANDO	1,104	*She covers several British trials on sexual matters and on what might be described as trumped-up evidence: the prosecution of Penguin Books for publishing Lawrence's Lady Chatterley's Lover, 1960, the trial of ex- Liberal Party leader Jeremy Thorpe for conspiracy to murder, and the trial of Stephen Ward (described by the Oxford Dictionary of National Biography both as osteopath and scapegoat and as the British Dreyfus) for living on immoral earnings in the wake of the resignation of Minister John Profumo on 4 June 1963.*

Table 1: Summary the two datasets of complex sentences, where WEBSPLIT is synthesized and unnatural while ORLANDO contains natural sentences of *absurd* complexity similar to the examples.

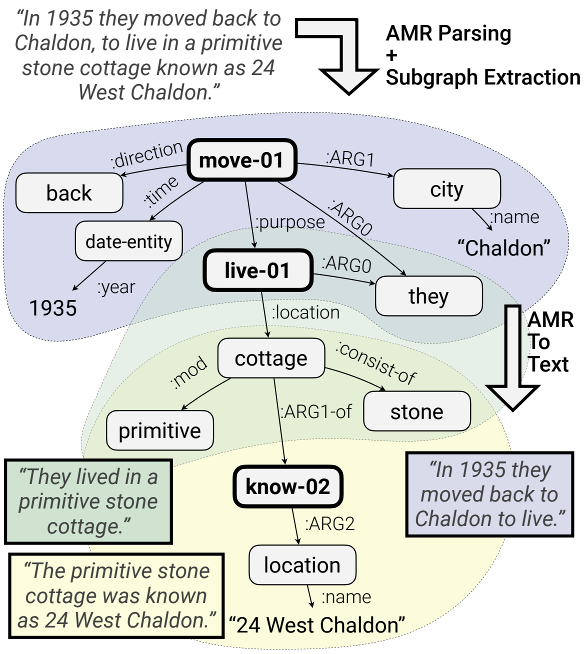
"In 1935 they moved back to Chaldon, to live in a primitive stone cottage known as 24 West Chaldon."
AMR Parsing
+
Subgraph Extraction
move-01
:direction
back
:time
date-entity
:year
1935
:ARG1
city
:name
"Chaldon"
:purpose
:ARG0
live-01
:ARG0
they
:location
cottage
:mod
primitive
:consist-of
stone
:ARG1-of
know-02
:ARG2
location
:name
"24 West Chaldon"
AMR
To
Text
"They lived in a primitive stone cottage."
"The primitive stone cottage was known as 24 West Chaldon."
"In 1935 they moved back to Chaldon to live."

Figure 1: Three stages of AMRS[3] : (1) Complex input sentence (top) is parsed into an AMR graph. In the AMR graph, core concepts are highlighted. (2) Subgraphs (three encircled graphs) that correspond to simpler sentences are identified using the subgraph extraction algorithm. (3) The subgraphs are realized into text (three boxes at the bottom) using an AMR-to-text model.

that map to predicates in OntoNotes (Pradhan et al., 2007) semantic roles, and the remaining nodes are arguments of the core concepts such as (named) entities. AMR concepts are not anchored to words, and a core concept captures an event even if the word that realizes it is a noun, adjective, or is of another part-of-speech. This allows us to simplify the sentence by focusing on and only on the core concepts and their arguments:

Rule 1 (Core Concept): If a node is a core concept and has more than σ arguments, it is considered a semantic unit, and the subgraph rooted at this node is extracted as a subgraph.

Algorithm 1 Extract subgraphs from an AMR graph G by performing DFS and applying the rules defined in §4.1.

```
procedure SUBGRAPHS($G$)
    $r \leftarrow \varnothing$; $q \leftarrow \{G.root\}$
    for all $e \in G.edges$, $e$ is inverse do
        $q \leftarrow q \cup \{e.from\}$                ▷ Rule 3
        $e.from, e.to \leftarrow e.to, e.from$
    while $|q| > 0$ do                          ▷ Extract from roots
        $g' \leftarrow$ DFSCOPY($q.pop(), q$)
        $r \leftarrow r \cup \{g'\}$
    return $r$
procedure DFSCOPY($n$, $q$)
    if $n$ is leaf return $n$
    if $n$ is core concept, $|n.edges| > \sigma$ then
        $q \leftarrow q \cup \{n\}$                     ▷ Rule 1
        return $n$
    if $n$ was visited then                      ▷ Rule 2
        for all $e \in n.edges$, $e$ is non-core do
            $n.addEdge$(DFSCOPY($e.to, q$))
    else for all $e \in n.edges$ do
            $n.addEdge$(DFSCOPY($e.to, q$))
    return $n$
```

A single concept (e.g. *they* in Fig. 1) can be the argument of multiple core concepts. To avoid redundancy, we only extract all relations of a concept on the first occurrence and only keep non-core relations (names, values, etc. as opposed to subjects and objects) on subsequent occurrences.

Rule 2 (Revisit): If a node has been visited before, only extract non-core relations.

AMR by default is rooted at a single predicate (e.g. *move-01*) as its focus. Non-focused predicates, except for the arguments of the focused predicate, are connected by inverse relations (e.g. *know-02* `:ARG1-of` *cottage* in Fig. 1) that are often realized as relative clauses. Depending on the granularity of simplification, we may choose to extract unfocused concepts as their own subgraphs as well by reversing the direction of the inverse relations and creating a new root.

Rule 3 (Inverse Relations): (Optional) If a node is connected by an inverse relation, reverse the direction of the inverse relation and extract the subgraph rooted at the node.

4.2 Implementation

Using depth-first search (DFS) with the rules above, we extract a set of subgraphs from the AMR graph (Algorithm 1), where σ is heuristically set at 2. We use AMRBART[2] (Bai et al., 2022), a unified model with strong performance in both AMR parsing and AMR-to-text, to parse the complex sentence into AMR graphs and realize the subgraphs into text. During AMR-to-text generation, we adopt the common practice of anonymizing named entities (Konstas et al., 2017).

As suggested by Bai et al. (2022), text-AMR pairs generated by semantic parsers (silver data) can benefit the training of AMR-to-text models. To adapt AMRBART to simple sentences, we leverage this property and finetune AMRBART on silver text-AMR pairs by parsing sentences from Simple English Wikipedia[3] using AMRBART. After finetuning, AMRBART realizations on the held-out set achieve a BLEU of 46.23, compared to the base model's BLEU of 39.53.

4.3 Baselines

We perform a comparison between AMRS^3 and the following existing systems for syntactic simplification using the evaluation methods outlined in §3. The results are reported in Table 2.

DisSim. DisSim (Niklaus et al., 2023) performs a recursive transformation of a sentence based on a set of 35 hand-crafted syntactic and lexical rules related to the sentence's phrase structure.

ABCD. ABCD (Gao et al., 2021) represents a sentence as a graph where edges are dependency and neighboring relations, and trains a neural network to predict actions on the edges. We use its MinWiki-MLP release.

DSS. DSS (Sulem et al., 2018) uses UCCA as the semantic representation, splits the UCCA graph based on parallel and elaborator scenes, and converts the subgraphs into text using a neural model.

LLM. We directly instruct GPT-3.5 (turbo-0125; Ouyang et al., 2022) and Llama-3 (8B-Instruct; AI@Meta, 2024) with Prompt 1.

Prompt 1: Direct Prompting

`[System]` You are a helpful assistant that simplifies syntactic structures.
`[User]` Rewrite the following paragraph using simple sentence structures and no clauses or conjunctions: `{complex sentence}`

4.4 Discussion

AMRS^3 achieves competitive performance without specialized supervised training. Overall, simplifications generated by AMRS^3 are on par with or better than the baselines in terms of meaning preservation on both datasets, as shown by the comparisons in Table 2, despite not being trained on task-specific supervised data. The performance is close to Llama-3, a state-of-the-art LLM. The syntactic simplicity of the generated sentences, measured by L2SCA, is at the same level as the best-performing baselines on WEBSPLIT and better on ORLANDO, suggesting that the good performance of meaning preservation is not achieved by sacrificing syntactic simplicity. The interpretable rule set of AMRS^3 makes the method easily customizable. The comparison between AMRS^3 with and without Rule 3 exemplifies how a compromise between simplicity and meaning preservation can be made by simple adjustments of the rules.

AMRS^3 enjoys unique merits beyond empirical performance. Specially trained models such as ABCD suffer from the lack of generalizability to new domains, as seen in its drastic performance drop on ORLANDO texts. In contrast, AMR models that AMRS^3 relies on are trained on a diverse set of data and can be easily improved for new domains by finetuning on silver data. LLMs are powerful and training-free, while AMRS^3 is lightweight and performs similarly well to open-weight LLMs. Admittedly, rule-based DisSim is lightweight and is performant in the evaluation. Compared to models based on semantic representation, DisSim requires a complex set of 35 lexical and syntactic rules,

[2] `AMRBART-large-v2 (AMR3.0)`

[3] Sentences extracted from `simplewiki-20230101` dump, with 5,000 held out as test set.

Method	BERTScore ↑		L2SCA ↓				
	Mean	Median	MLT	C/S	C/T	T/S	CN/T
			on ORLANDO				
AMRS^3	0.73	0.72	12.00	1.02	1.07	0.96	1.22
AMRS^3 (w/o Rule 3)	0.79	0.79	18.17	1.30	1.32	0.98	1.89
ABCD	0.50	0.51	14.99	0.94	1.19	0.80	1.98
DisSim	0.74	0.74	11.15	1.18	1.16	1.02	1.24
DSS†	-	-	-	-	-	-	-
GPT-3.5	0.80	0.82	12.65	1.14	1.13	1.01	1.26
Llama-3-8B	0.74	0.74	7.89	1.07	1.07	1.00	0.70
Exact Copy	1.00	1.00	157.25	2.66	2.18	1.22	4.69
			on WEBSPLIT				
AMRS^3	0.81	0.81	8.92	1.00	1.02	0.99	0.68
AMRS^3 (w/o Rule 3)	0.86	0.86	12.26	1.16	1.16	1.00	1.16
ABCD	0.90	0.91	9.53	1.00	1.10	0.91	0.94
DisSim	0.87	0.87	8.54	1.05	1.05	0.99	0.67
DSS†	0.74	0.74	10.69	0.97	1.19	0.81	1.05
GPT-3.5	0.90	0.90	7.79	1.02	1.02	1.00	0.52
Llama-3-8B	0.84	0.85	6.69	1.01	1.01	1.00	0.38
Exact Copy	1.00	1.00	16.57	1.64	1.50	1.10	1.72

Table 2: Evaluation results of AMRS^3 and baselines on ORLANDO and WEBSPLIT. BERTScore measures meaning preservation (↑ the higher the better), and L2SCA measures syntactic complexity (↓ the lower the better). † We use the output provided by Sulem et al. (2018) on WebSplit only, as no code is available. Five L2SCA metrics correspond to production unit length, overall complexity, subordination, coordination, and phrasal complexity. See Lu (2010) for the exact definition of L2SCA metrics.

while AMRS^3 only needs three simple rules. The rules of DisSim are crafted for English only and are hard to transfer to other languages, while despite AMR not being an interlingua (Banarescu et al., 2013) the rules of AMRS^3 are language-agnostic and can be easily adapted to other languages with AMR parsers. Methods based on other semantic representations, such as UCCA-based DSS, perform worse despite having a similar workflow to AMRS^3 , showcasing the *"free upgrades"* that advances in semantic representation tools can bring.

Takeaways. As AMRS^3 demonstrates, semantic graphs like AMR are mature enough to support the easy development of lightweight and interpretable systems, that still have certain advantages in LLM's age, for tasks like syntactic simplification.

5 AMR for LLM-based Simplification

Given the position of AMR as an expressive and suitable intermediate for syntactic simplification and LLM's strong performance in the task, a natural question arises as to whether AMR can be used as an auxiliary to LLMs to improve their performance in syntactic simplification in the scenario of zero-shot prompting. We investigate this question by designing a set of controlled prompting strategies to examine how the elements of AMR affect LLM. This is an addition to Jin et al. (2024) which tested directly appending AMR to the prompt in a variety of tasks, while syntactic simplification was not included in their study. Extending their work, we explore a new prompting strategy (named AMR Chain-of-Code or AMRCoC) that guides LLMs to perform explicit symbolic reasoning over AMR graphs instead of making implicit inferences as in Jin et al. (2024).

Prompt 2: Direct Full AMR Prompting (Jin et al., 2024)

`[User]` You are given a paragraph and its abstract meaning representation (AMR).
# Paragraph
`{complex sentence}`
# AMR
`{amr}`
Rewrite the paragraph using simple sentence structures and no clauses or conjunctions. You can refer to the provided AMR if it helps you in the rewriting.
The rewritten paragraph:

	Prompting	BERTScore ↑				Prompting	BERTScore ↑		
		Mean	Median	MLT ↓			Mean	Median	MLT ↓
on ORLANDO									
GPT-3.5	*vanilla*	0.80	0.82	12.65	Llama-3	*vanilla*	0.74	0.74	7.89
	direct AMR	0.81	0.82	12.79		*direct AMR*	0.78	0.78	11.74
	subgraphs	0.80	0.81	11.65		*subgraphs*	0.78	0.78	12.45
	entities	0.79	0.80	10.99		*entities*	0.70	0.71	7.75
	predicates	0.73	0.74	7.34		*predicates*	0.70	0.70	7.55
	AMRCoC	0.79	0.81	17.29		*AMRCoC*	0.76	0.77	14.03
on WEBSPLIT									
GPT-3.5	*vanilla*	0.90	0.90	7.79	Llama-3	*vanilla*	0.84	0.85	6.69
	direct AMR	0.88	0.89	8.59		*direct AMR*	0.83	0.85	8.15
	subgraphs	0.87	0.88	8.35		*subgraphs*	0.82	0.84	7.41
						entities	0.78	0.79	6.63
						predicates	0.76	0.77	7.12
	AMRCoC	0.89	0.90	9.15		*AMRCoC*	0.84	0.85	8.27

Table 3: Evaluation results of GPT-3.5 and Llama-3 on ORLANDO and WEBSPLIT with different prompting strategies. Notations are consistent with Table 2. Due to space limit, we only show one L2SCA metric, MLT, that has the highest variance across prompts.

5.1 Direct AMR Prompting

Jin et al.'s (2024) evaluation framework simply supplies linearized AMR in PENMAN format (Matthiessen and Bateman, 1991) in parallel with text, providing only vague instructions to the LLM on how to use the AMR, and requiring the LLM to directly produce the output *without* [4] explicitly producing reasoning steps. To add to their tests, we adapt their framework to the syntactic simplification task as in Prompt 2.

Performance. Interestingly, our evaluations (Table 3) show that the direct AMR prompting does not harm the performance of LLMs in syntactic simplification, and in some cases, it provides improvements especially for more complex inputs. This adds syntactic simplification as a counterexample to the findings of Jin et al. (2024).

Effect of Elements. To isolate the effects of different elements (subgraphs, entities, and predicates) of AMR, we further design a set of controlled prompts following the same format of Prompt 2, where the linearization of complete AMR is replaced by specific parts of the AMR:
(1) Instead of the sole AMR corresponding to the whole complex sentence, we provide a list of AMR graphs extracted with Algorithm 1 for each semantic unit in the sentence (**subgrpahs**);
(2) We provide only a list of predicates in the AMR (**predicates**), e.g. *"move, live, know"* as in Figure 1;
(3) We provide only a list of entities as reflected by the non-core concepts in the AMR (**entities**), e.g. *"date (1935), they, city (Chaldon), location (24 West Chaldon)"* as in Figure 1.

Both predicates and entities provide incomplete information about the events of a sentence, while not requiring LLM's capability to reason over a symbolic graph. However, we find that for the tasks and LLMs in question, LLMs are capable of directly and implicitly using information in the AMR as appropriate, while trading information completeness for the ease of symbolic graph processing offers more harms than benefits (Table 3).

Takeaways. Directly supplying AMR to LLMs is not monochromatically harmful across tasks. Growing the list of tasks benefited and harmed by direct AMR prompting is needed to draw conclusions on the role of meaning representations in the LLM era.

5.2 AMRCoC Prompting

Despite the evidence that LLMs can benefit from direct AMR prompting, it is widely accepted (Wei et al., 2022; Saparov and He, 2023, *inter alia*) that LLM's reasoning capability over complex tasks (e.g. processing AMR) can be improved by explicitly decomposing them into reasoning steps. To remedy the lack of explicit reasoning, we build on Chain-of-Code (CoC) prompting (Li et al., 2023), where pseudocode execution is shown helpful for the LLM to perform explicit algorithmic reasoning in general tasks, and design AMRCoC prompting

[4] Despite having an imprecise name "AMR for Chain-of-Thought" prompting in the original paper.

Prompt 3: AMR-Chain-of-Code Prompting (with simplification and an example)

```
[User]
```

You are given a paragraph and its abstract meaning representation (AMR). AMR captures “who is doing what to whom” in a sentence. Begin your steps with # Steps.

# Functions to process AMR

```
EXTRACT_SUBGRAPH(root: Node) -> AMR
```

extracts the subgraph rooted at ‘root‘.

```
AMR_TO_TEXT(graph: AMR) -> str
```

converts an AMR graph back to text.

# Example program

```
for predicate in amr:
g = EXTRACT_SUBGRAPH(predicate)
print(AMR_TO_TEXT(g))

[Assistant]
```

# Steps

```
g = EXTRACT_SUBGRAPH(flow-01) => (z1 / flow-01
:ARG1(...) ...)
print(AMR_TO_TEXT(g))
=> It flows through the town of Yeovil.
g = EXTRACT_SUBGRAPH(join-01) => (z5 / join-01
:ARG1 (z2 / it) ...)
...
```

# Output

It flows through the town of Yeovil. It joins River Parrett.

Property	**Orlando**	**Websplit**
Following algorithm	99.8%	92.8%
Grammatical AMR	31.3%	67.8%
Node and edge existence	98.6%	99.7%
Node coverage	72.3%	90.0%
Matching algorithm output	52.1%	66.0%

Table 4: Success rates of Llama-3’s Chain-of-Code execution at different stages. Numbers are macro-averaged across all input complex sentences. For the first four rows, higher values are always favored.

(Prompt 3): LLM is guided to produce explicit reasoning steps over AMR graphs by using functions to process AMR, and an example program that demonstrates the use of these functions. The functions and programs are not formally defined but in the form of function signatures or pseudocode, as we expect LLM to emulate the execution (Li et al., 2023; Chae et al., 2024).

Performance. AMRCoC offers the same level of meaning preservation (last rows of Table 3) compared to direct AMR prompting, although the simplicity of generations degrades to the level of AMRS^3 , which is perhaps unsurprising as we prompt the LLM to follow a similar algorithm. The example program in the prompt may not be optimal, but it is possible to synthesize or improve the program using LLM (Chae et al., 2024).

Emulated Execution. More importantly, the breakdown of AMRCoC execution (Table 4) verifies that LLMs can be prompted to perform explicit algorithmic reasoning over AMR graphs, which is a promising direction for future research. LLM almost always emulates the execution of the example pseudocode program ("Following algorithm" in Table 4). The extracted AMR graphs, although not always grammatically correct especially for complex inputs ("Grammatical AMR"), are not hallucinated and are based on existing nodes and edges in the input AMR ("Node and edge existence"), and mostly match the real execution results of Algorithm 1 ("Matching algorithm output"). When combined, AMR graphs extracted by LLM cover most of the semantic information in the input AMR ("Node coverage"), providing a guarantee for meaning preservation.

Takeaways. Chain-of-Code prompting provides a way for LLM to perform symbolic reasoning over semantic graphs via algorithm emulation. This provides a way to bring algorithmic graph processing to LLMs for semantic-centered NLP applications, to enjoy the benefits of both worlds.

6 Conclusion

In light of recent developments in semantic representations and LLMs, we presented a retrospective view of using semantic representation graphs for syntactic simplification, with refreshed datasets and up-to-date semantic representation models. In prospect, we added to the case studies of the beneficial and harmful effects of using AMR for LLM, and proposed a new AMRCoC prompting strategy with the potential of bridging symbolic and graphical algorithms to LLMs.

Limitations

The proposed AMRS^3 is not the best performing syntactic simplification system in terms of having the highest absolute numbers of BERTScore and L2SCA metrics across the datasets, as is particularly overshadowed by LLMs. The main conclusion is more about the current state of semantic representations: they are still handy in building solutions for semantic tasks, and that solution can have merits that make it a good fit in certain scenarios. Despite that, the design of AMR has some disadvantages that make it less effective to be used out-of-the-box for text simplification, namely the absence of inflectional morphology for tense and number. Banarescu et al. (2013) suggested that this can be remedied by adding these notions to AMR as an extension, which is a direction for future work.

Our evaluation of syntactic simplification is limited to automated methods. Although previous work has shown high correlations between the metrics we use and human judgments on meaning preservation, syntactic complexity, and reading difficulty, we acknowledge that those conclusions might not hold for domains out of their respective evaluations. A systematic evaluation method, tailored to the specific task of syntactic simplification and aligned with human judgments, similar to Alva-Manchego et al. (2021); Maddela et al. (2023), would be beneficial for similar studies but is out of the scope of this work.

Finally, the applicability of AMRCoC prompting is only tested on the single task of syntactic simplification. Although the properties it demonstrates are promising, we have yet to test it on other tasks such as the ones in Jin et al. (2024).

Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC). This work is also supported in part by a gift from Scotiabank. We thank Natalie Hervieux and Dr. Susan Brown for preparing the ORLANDO dataset. We also thank the anonymous reviewers for their valuable feedback.

References

Omri Abend and Ari Rappoport. 2013. UCCA: A semantics-based grammatical annotation scheme. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 1–12, Potsdam, Germany. Association for Computational Linguistics.

AI@Meta. 2024. Llama 3 model card.

Suha Al-Thanyyan and Aqil M. Azmi. 2022. Automated text simplification: A survey. *ACM Comput. Surv.*, 54(2):43:1–43:36.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2021. The (Un)Suitability of Automatic Evaluation Metrics for Text Simplification. *Computational Linguistics*, 47(4):861–889.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. 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.

Jan A. Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from Wikipedia edit history. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 732–737, Brussels, Belgium. Association for Computational Linguistics.

Susan Brown, Patricia Clements, and Isobel Grundy. 2022. Orlando: Women's writing in the british isles from the beginnings to the present. `https://orlando.cambridge.org`. Accessed September 27, 2023.

Hyungjoo Chae, Yeonghyeon Kim, Seungone Kim, Kai Tzu-iunn Ong, Beong-woo Kwak, Moohyeon Kim, Seonghwan Kim, Taeyoon Kwon, Jiwan Chung, Youngjae Yu, and Jinyoung Yeo. 2024. Language models as compilers: Simulating pseudocode execution improves algorithmic reasoning in language models. *CoRR*, abs/2404.02575.

Liam Cripwell, Joël Legrand, and Claire Gardent. 2024. Evaluating document simplification: On the importance of separately assessing simplicity and meaning preservation. In *Proceedings of the 3rd Workshop on Tools and Resources for People with REAding DIfficulties (READI) @ LREC-COLING 2024*, pages 1–14, Torino, Italia. ELRA and ICCL.

Yanjun Gao, Ting-Hao Huang, and Rebecca J. Passonneau. 2021. ABCD: A graph framework to convert complex sentences to a covering set of simple sentences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3919–3931, Online. Association for Computational Linguistics.

Jens E. L. Van Gysel, Meagan Vigus, Jayeol Chun, Kenneth Lai, Sarah R. Moeller, Jiarui Yao, Tim O'Gorman, Andrew Cowell, William Croft, Chu-Ren Huang, Jan Hajic, James H. Martin, Stephan Oepen, Martha Palmer, James Pustejovsky, Rosa Vallejos, and Nianwen Xue. 2021. Designing a uniform meaning representation for natural language processing. *Künstliche Intell.*, 35(3):343–360.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Bairu Hou, Yang Zhang, Jacob Andreas, and Shiyu Chang. 2024. A probabilistic framework for llm hallucination detection via belief tree propagation. *Preprint*, arXiv:2406.06950.

Zhijing Jin, Yuen Chen, Fernando Gonzalez Adauto, Jiarui Liu, Jiayi Zhang, Julian Michael, Bernhard Schölkopf, and Mona Diab. 2024. Analyzing the role of semantic representations in the era of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3781–3798, Mexico City, Mexico. Association for Computational Linguistics.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Gangi Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G P Shrivatsa Bhargav, and Mo Yu. 2021. Leveraging Abstract Meaning Representation for knowledge base question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894, Online. Association for Computational Linguistics.

Joongwon Kim, Mounica Maddela, Reno Kriz, Wei Xu, and Chris Callison-Burch. 2021. BiSECT: Learning to split and rephrase sentences with bitexts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6193–6209, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kevin Knight, Bianca Badarau, Laura Baranescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2020. Abstract meaning representation (amr) annotation release 3.0. In *Linguistic Data Consortium*.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Chengshu Li, Jacky Liang, Fei Xia, Andy Zeng, Sergey Levine, Dorsa Sadigh, Karol Hausman, Xinyun Chen, Li Fei-Fei, and brian ichter. 2023. Chain of code: Reasoning with a language model-augmented code interpreter. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Xiaofei Lu. 2010. Automatic analysis of syntactic complexity in second language writing. *International journal of corpus linguistics*, 15(4):474–496.

Mounica Maddela, Yao Dou, David Heineman, and Wei Xu. 2023. LENS: A learnable evaluation metric for text simplification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16383–16408, Toronto, Canada. Association for Computational Linguistics.

Christian M.I.M. Matthiessen and John A. Bateman. 1991. Text generation and systemic-functional linguistics: Experiences from english and japanese. In *Communication in Artificial Intelligence Series*, pages xxii + 348. Pinter Publisher.

Shashi Narayan, Claire Gardent, Shay B. Cohen, and Anastasia Shimorina. 2017. Split and rephrase. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 606–616, Copenhagen, Denmark. Association for Computational Linguistics.

Ani Nenkova, Kathleen McKeown, et al. 2011. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2023. Discourse-aware text simplification: From complex sentences to linked propositions. *CoRR*, abs/2308.00425.

Christina Niklaus, André Freitas, and Siegfried Handschuh. 2019. MinWikiSplit: A sentence splitting corpus with minimal propositions. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 118–123, Tokyo, Japan. Association for Computational Linguistics.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 1–22, Online. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *Computing Research Repository*, arXiv:2303.08774.

Long Ouyang, Jeffrey 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 F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Gustavo H Paetzold and Lucia Specia. 2017. A survey on lexical simplification. *Journal of Artificial Intelligence Research*, 60:549–593.

David Ponce, Thierry Etchegoyhen, Jesús Calleja-Perez, and Harritxu Gete. 2023. Split and rephrase with large language models. *CoRR*, abs/2312.11075.

Sameer S. Pradhan, Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2007. Ontonotes: A unified relational semantic representation. In *International Conference on Semantic Computing (ICSC 2007)*, pages 517–526.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Kaize Shi, Xueyao Sun, Li He, Dingxian Wang, Qing Li, and Guandong Xu. 2023. AMR-TST: Abstract Meaning Representation-based text style transfer. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4231–4243, Toronto, Canada. Association for Computational Linguistics.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Simple and effective text simplification using semantic and neural methods. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 162–173, Melbourne, Australia. Association for Computational Linguistics.

Willian Massami Watanabe, Arnaldo Cândido Júnior, Vinícius Rodrigues de Uzêda, Renata Pontin de Mattos Fortes, Thiago Alexandre Salgueiro Pardo, and Sandra M. Aluísio. 2009. Facilita: reading assistance for low-literacy readers. In *Proceedings of the 27th Annual International Conference on Design of Communication, SIGDOC 2009, Bloomington, Indiana, USA, October 5-7, 2009*, pages 29–36. ACM.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Shira Wein and Nathan Schneider. 2024. Assessing the Cross-linguistic Utility of Abstract Meaning Representation. *Computational Linguistics*, pages 1–55.

Fangzhi Xu, Zhiyong Wu, Qiushi Sun, Siyu Ren, Fei Yuan, Shuai Yuan, Qika Lin, Yu Qiao, and Jun Liu. 2023. Symbol-llm: Towards foundational symbol-centric interface for large language models. *CoRR*, abs/2311.09278.

Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Heng Ji, and ChengXiang Zhai. 2024. If LLM is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.