

Prompt Engineering

Presented by Team 2

Soneya Binta Hossain (sh7hv)

Jessie Chen (hc4vb)

Ali Zafar Sadiq (mzw2cu)

Jeffrey Chen (fyy2ws)

Minjae Kwon (hbt9su)

Presentation Outline

Paper 1: Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review

Paper 2: Skeleton Of Thought: Prompting LLMs For Efficient Parallel Generation

Paper 3: Topologies of Reasoning: Demystifying Chains, Trees, and Graphs of Thoughts

Paper 1

Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review

Presenters

Minjae Kwon (hbt9su)

Overview

- 1. Basic of Prompt Engineering:** Role Playing, Few-Shot
- 2. Advanced:** CoT, Self-consistency, Knowledge Generation, ToT, GoT, CoVe
- 3. Assess:** Subjective and Objective, Comparing Different Prompt Methods
- 4. Applications:** Education, Content Creation and Editing, Computer Programming, Reasoning, Dataset Generation
- 5. Future Direction:** Understanding of Structure, AIGC
- 6. Conclusion**

Basic of Prompt Engineering

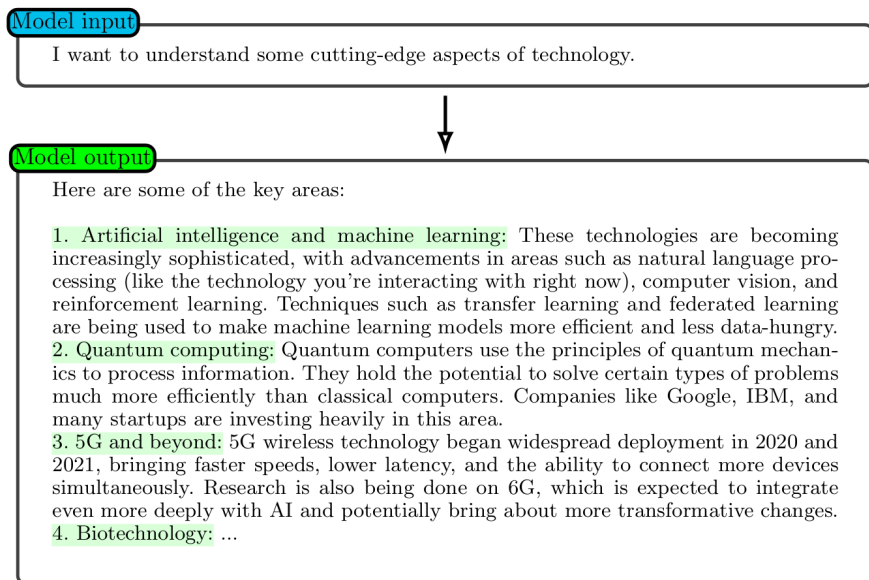


Fig. 1 Giving instructions without extra description.

1. Basic Prompt

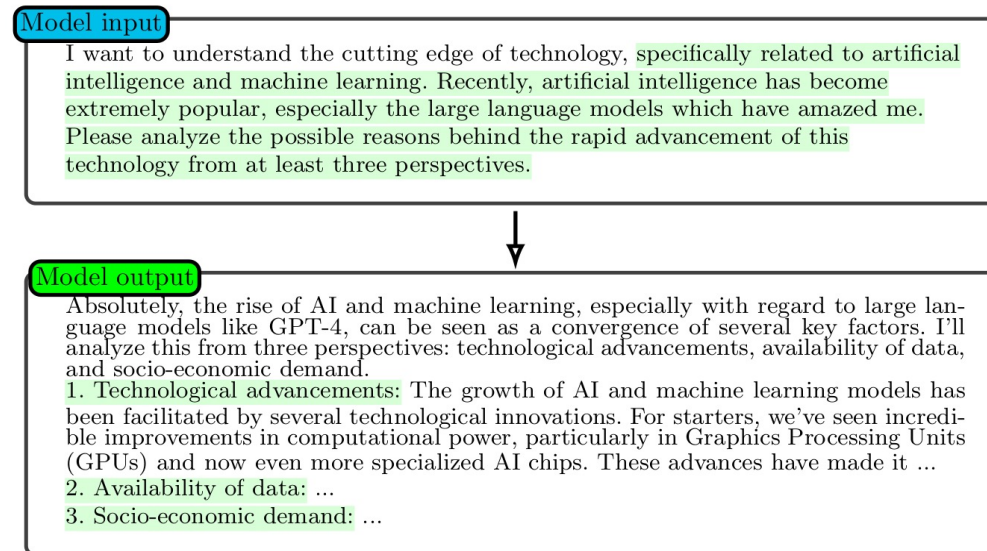


Fig. 2 A clearer and more precise prompt.

2. Clear and Precise

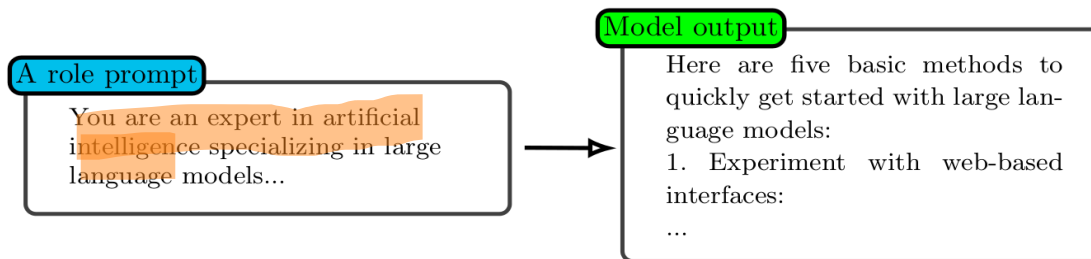


Fig. 3 Role prompting example.

3. Role Playing

Basic of Prompt Engineering

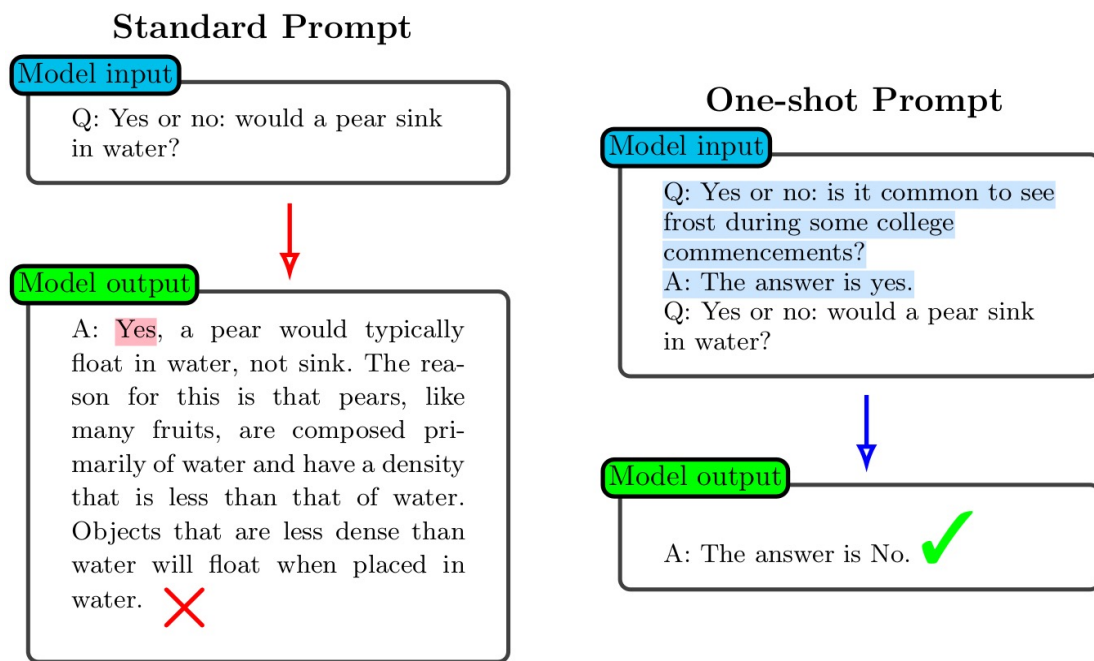


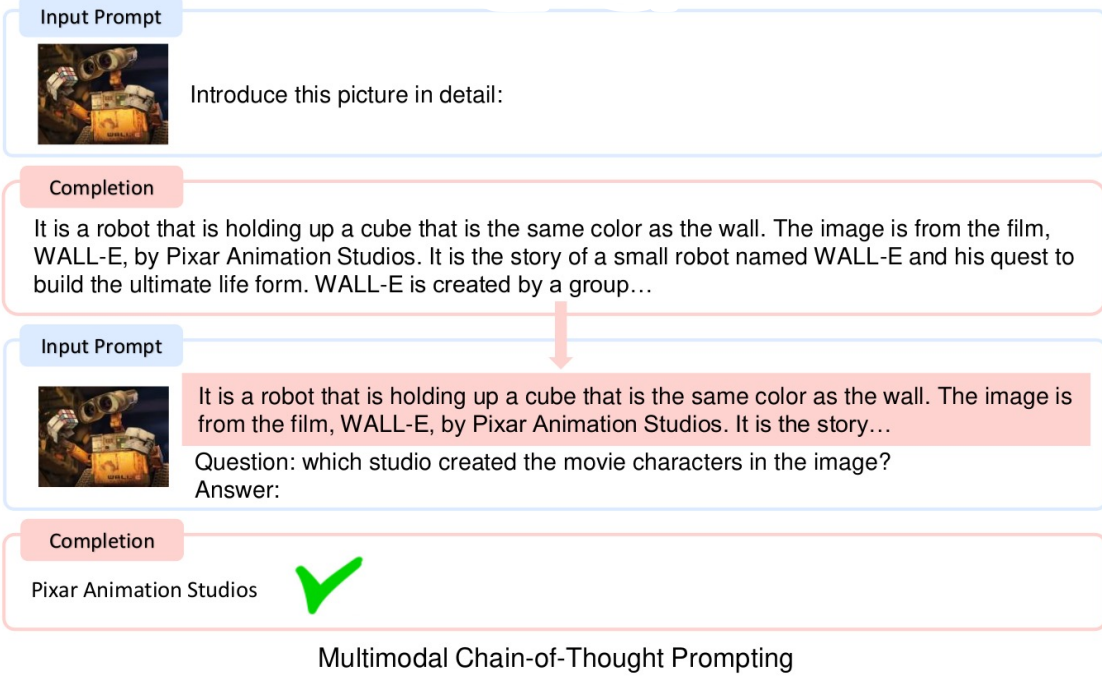
Fig. 4 Comparison of standard prompt and one-shot prompt.

4. Few Shot

| Use Case | Temperature | Top_p | Description |
|--------------------------|-------------|-------|--|
| Code Generation | 0.2 | 0.1 | Generates code that adheres to established patterns and conventions. Output is more deterministic and focused. Useful for generating syntactically correct code. |
| Creative Writing | 0.7 | 0.8 | Generates creative and diverse text for storytelling. Output is more exploratory and less constrained by patterns. |
| Chatbot Responses | 0.5 | 0.5 | Generates conversational responses that balance coherence and diversity. Output is more natural and engaging. |
| Code Comment Generation | 0.3 | 0.2 | Generates code comments that are more likely to be concise and relevant. Output is more deterministic and adheres to conventions. |
| Data Analysis Scripting | 0.2 | 0.1 | Generates data analysis scripts that are more likely to be correct and efficient. Output is more deterministic and focused. |
| Exploratory Code Writing | 0.6 | 0.7 | Generates code that explores alternative solutions and creative approaches. Output is less constrained by established patterns. |

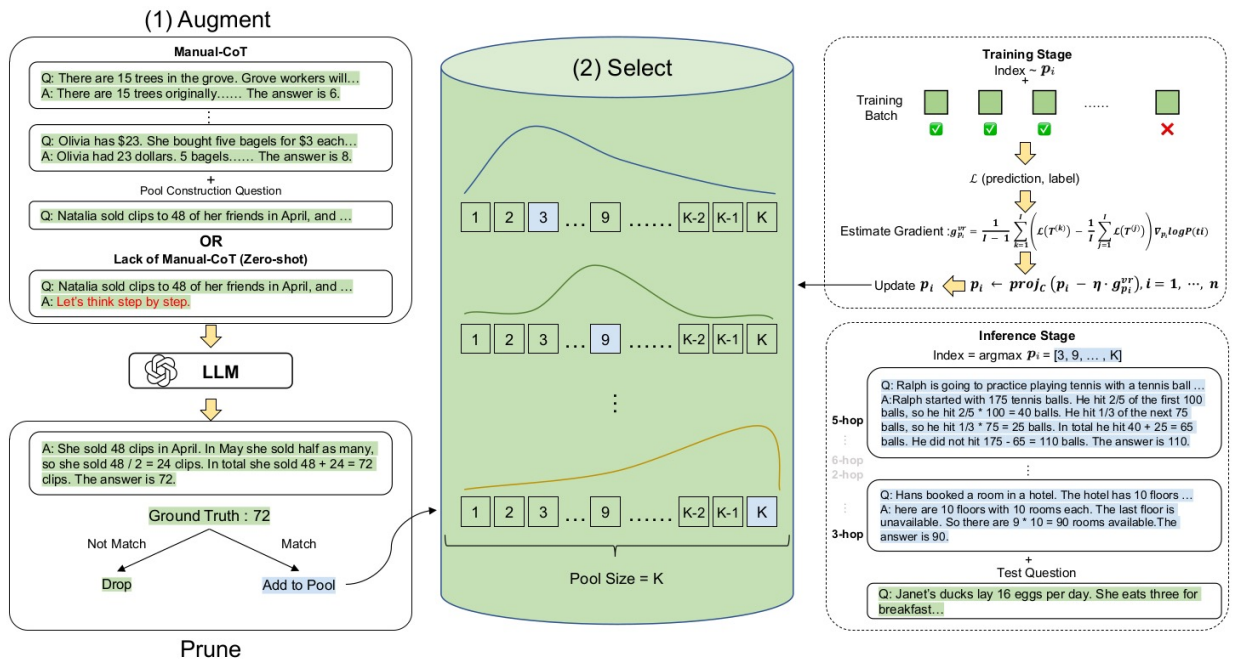
5. LLM settings: temperature and top-p

Advanced Methodologies



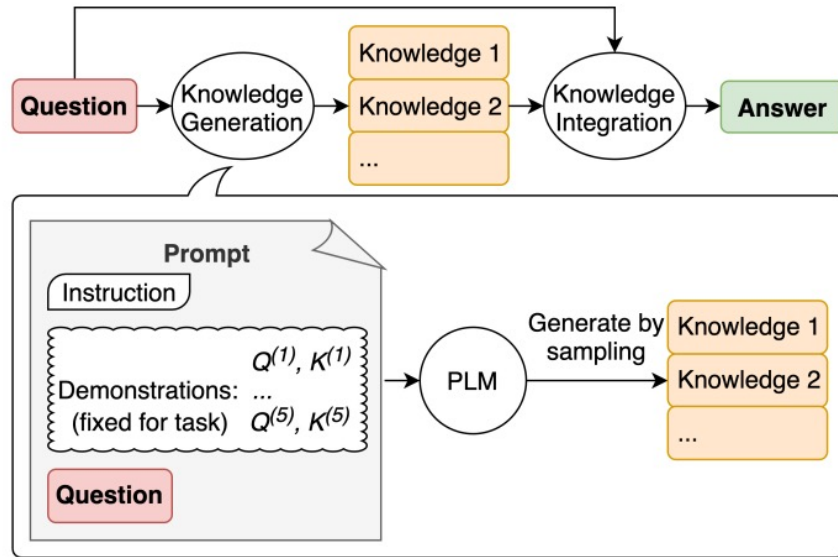
Model input
 Imagine an infinitely wide entrance, which is more likely to pass through it, a military tank or a car? **Let's think step by step.**

1. Chain of Thought, Golden Chain of Thought



2. Self-Consistency:
 Step1 - CoT prompting
 Step2 - Sampling
 Step3 - Marginalize the reasoning path

Advanced Methodologies



| Dataset | Question / Knowledge | Prediction | Score |
|------------|--|---------------------------------|---------------------|
| NumerSense | the word children means [M] or more kids. | one | 0.37 0.35 |
| | The word child means one kid. | two | 0.91 |
| CSQA | She was always helping at the senior center, it brought her what? People who help others are usually happier. | feel better happiness | 0.97 0.02 0.98 |
| CSQA2 | Part of golf is trying to get a higher point total than others. The player with the lowest score wins. | yes no | 1.00 0.00 1.00 |
| QASC | Sponges eat primarily Sponges eat bacteria and other tiny organisms. | cartilage krill and plankton | 0.95 0.00 0.99 |

3. Knowledge Generation

Stage 1: Decompose Question into Subquestions

Q: It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?

Language Model

A: To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"

Stage 2: Sequentially Solve Subquestions

Subquestion 1

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Language Model

A: It takes Amy 4 minutes to climb and 1 minute to slide down. $4 + 1 = 5$. So each trip takes 5 minutes.

Append model answer to Subquestion 1

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Q: How long does each trip take?
A: It takes Amy 4 minutes to climb and 1 minute to slide down. $4 + 1 = 5$. So each trip takes 5 minutes.

Language Model

A: The water slide closes in 15 minutes. Each trip takes 5 minutes. So Amy can slide $15 \div 5 = 3$ times before it closes.

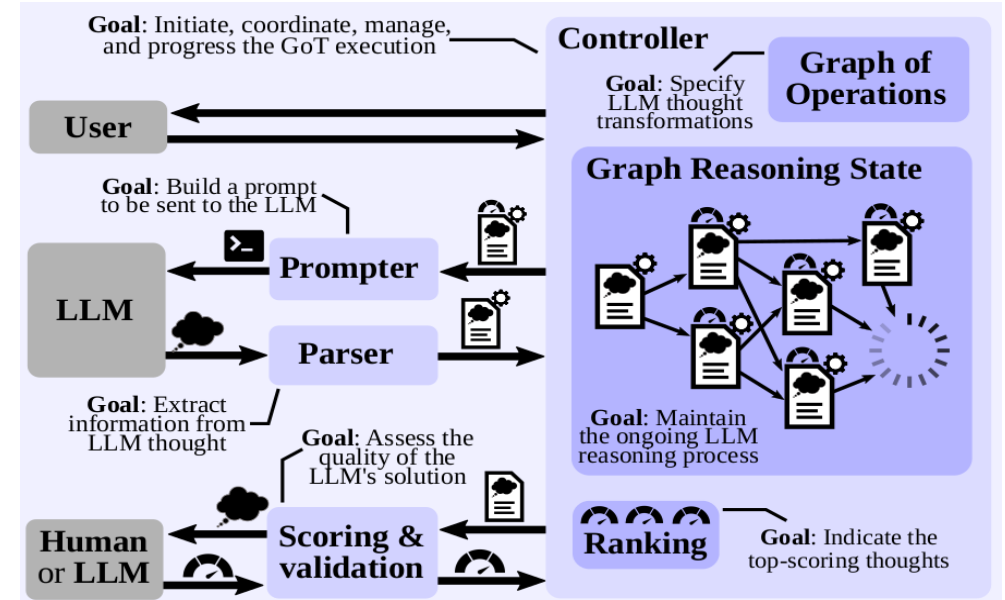
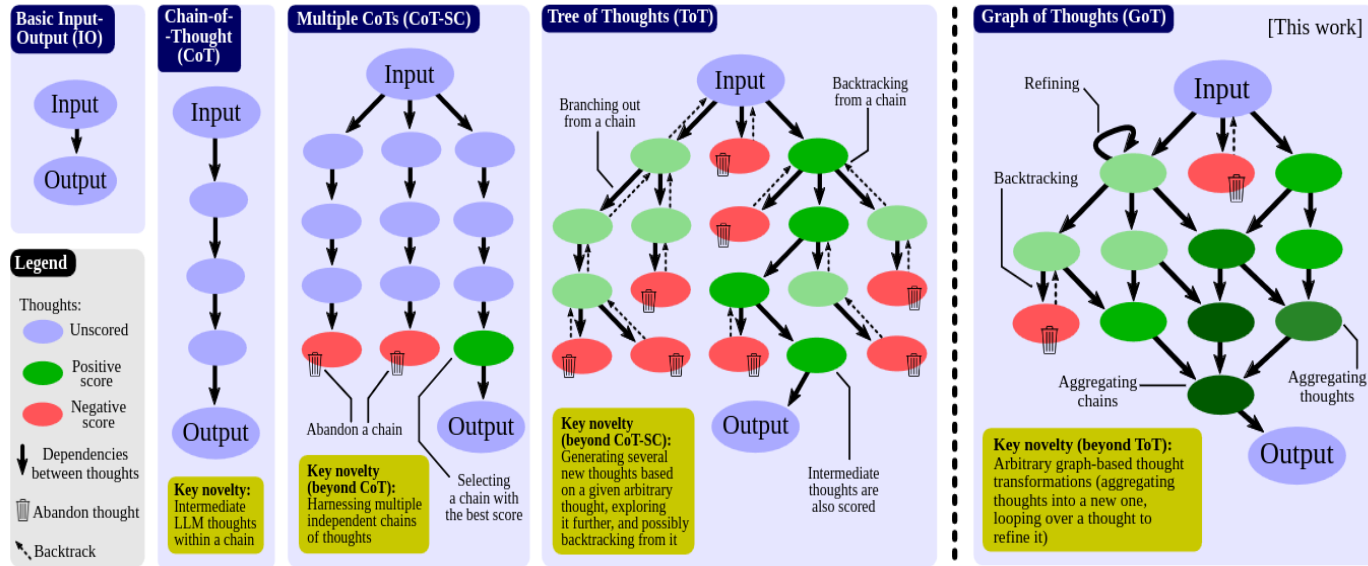
Subquestion 2

Q: How many times can she slide before it closes?

Figure 1: Least-to-most prompting solving a math word problem in two stages: (1) query the language model to decompose the problem into subproblems; (2) query the language model to sequentially solve the subproblems. The answer to the second subproblem is built on the answer to the first subproblem. The demonstration examples for each stage's prompt are omitted in this illustration.

4. Least-to-most Prompting

Advanced Methodologies



Tree of thoughts prompting

Imagine three different experts answering this question. All experts will write down 1 step of their thinking, then share it with the group. Then all experts will go on to the next step, etc. If any expert realizes they're wrong at any point then they leave. The question is...

Fig. 9 A sample ToT prompt.

```
# Initialization of the Controller with the language model, operations graph, pr
controller = Controller(
    lm=ChatGPT(), # Instance of the language model
    graph=graph_of_operations, # Operations graph composed of Generate, Validat
    prompter=PortfolioPrompter(), # Custom prompter for portfolio-related queri
    parser=PortfolioParser(), # Custom parser to interpret the language model's
    problem_parameters={
        'available_projects': [{'value': 3, 'budget': 2}, {'value': 1, 'budget':
        'budget_limit': 2,
        'generated_portfolio': None
    }
)
```

[Image source](#)

5. Tree of Thoughts (ToT)

6. Graph of Thoughts (GoT)

Advanced Methodologies

Query
Name some politicians who were born in NY, New York.

1. Baseline Response
Here are some politicians who were born in NY, New York:
1. Hillary Clinton - former secretary of state and former Democratic presidential nominee
2. Donald Trump - former president of the United States
3. Michael Bloomberg - former Mayor of New York City and former Democratic presidential candidate
... <list continues..>

2. Plan Verifications
- Where was Hillary Clinton born?
- Where was Donald Trump born?
- Where was Michael Bloomberg born?
... <questions continue..>

3. Execute Verifications
Hillary Clinton was born in **Chicago, Illinois**, United States on October 26, 1947.
Donald Trump was born on June 14, 1946, in **Queens, New York City, New York**, United States
Michael Bloomberg was born on February 14, 1942, in **Boston, Massachusetts**, United States.
...









4. Final Verified Response
Here are some politicians who were born in NY, New York:
1. Donald Trump - former president of the United States
2. Alexandria Ocasio-Cortez - Democratic member of the U.S. House of Representatives
... <list continues..>

7. Chain of Verification

Prompt examples

Explore what's possible with some example prompts

Q Search... All categories

| | |
|--|---|
|  Grammar correction Convert ungrammatical statements into standard English. |  Summarize for a 2nd grader Simplify text to a level appropriate for a second-grade student. |
|  Parse unstructured data Create tables from unstructured text. |  Emoji Translation Translate regular text into emoji text. |
|  Calculate time complexity Find the time complexity of a function. |  Explain code Explain a complicated piece of code. |
|  Keywords Extract keywords from a block of text. |  Product name generator Generate product names from a description and seed words. |

8. Plugins

Assessing the Efficacy of Prompt Methods

1. Subjective evaluations

Pros: Fluency, Accuracy, Novelty, and Relevance

Cons: Inconsistency Problem, Expensive, Time Consuming

- Human Evaluator

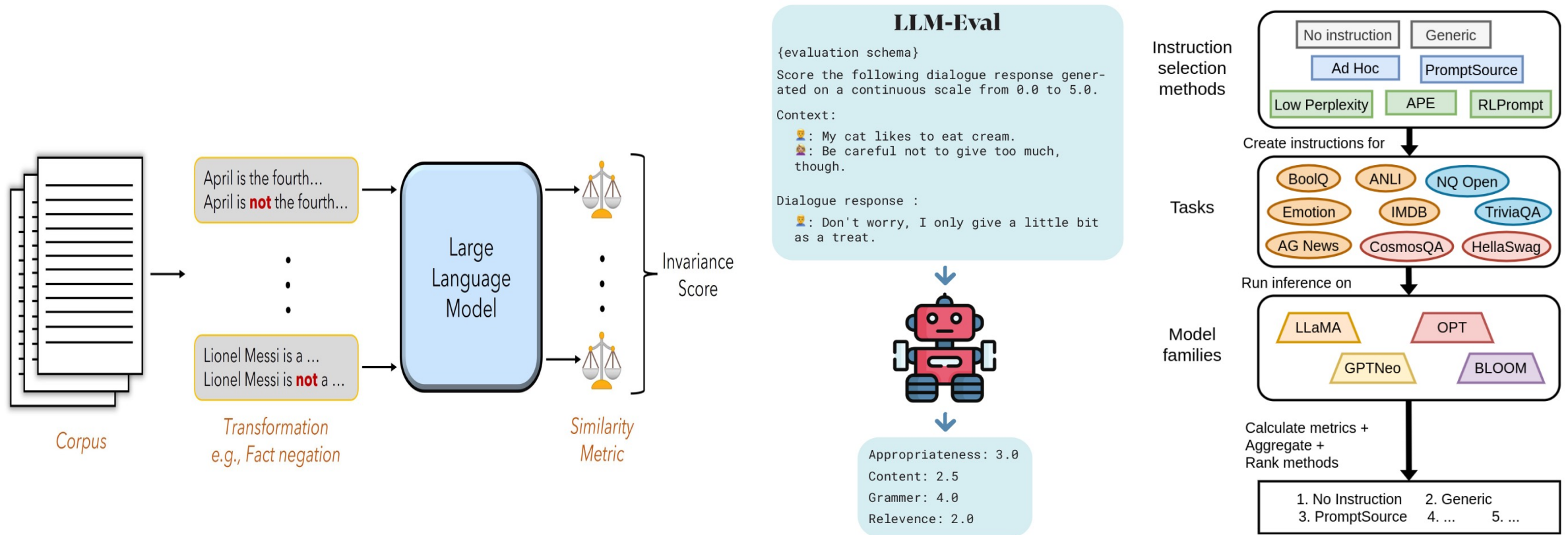
2. Objective evaluations

Pros: Automatic Evaluation, Cheap, Quick

Cons: Alignment Problem

- BLEU: BiLingual Evaluation Understudy
- ROUGE: Recall-Oriented Understudy for Gisting Evaluation
- METEOR: Metric for Evaluation of Translation with Explicit ORdering
- BERTScore: BERT Model used for metric.

Assessing the Efficacy of Prompt Methods



3. Comparing different prompt methods: Self-Supervised Evaluation, LLM-Eval, InstructEval

Dataset Generation

Principles of Accounting: Course Outline

Duration: 10 weeks

Class Structure:

- Lecture: 1 hour per week
- Workshop/Tutorial: 2 hours per week

Week 1: Introduction to Accounting

- Lecture: The Importance of Accounting in Business
- Workshop: Introduction to Financial Statements

Week 2: Basics of Bookkeeping

- Lecture: The Accounting Cycle
- Workshop: Recording Transactions using Double Entry System

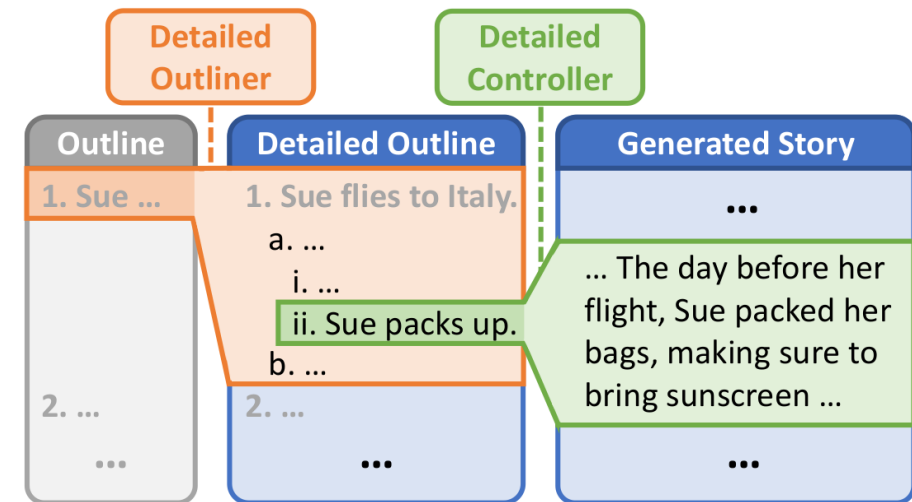
.....

Group Assignment: Company Financial Analysis

•**Description:** Groups of 4 students will select a public company and perform a financial statement analysis ...

•Rubrics:

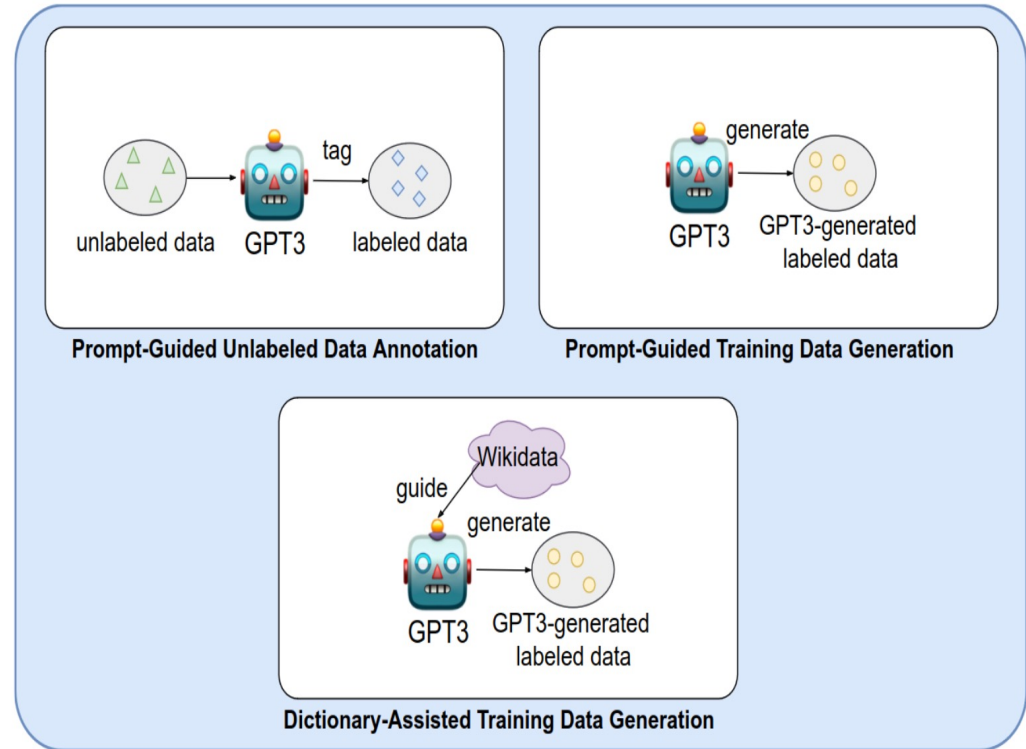
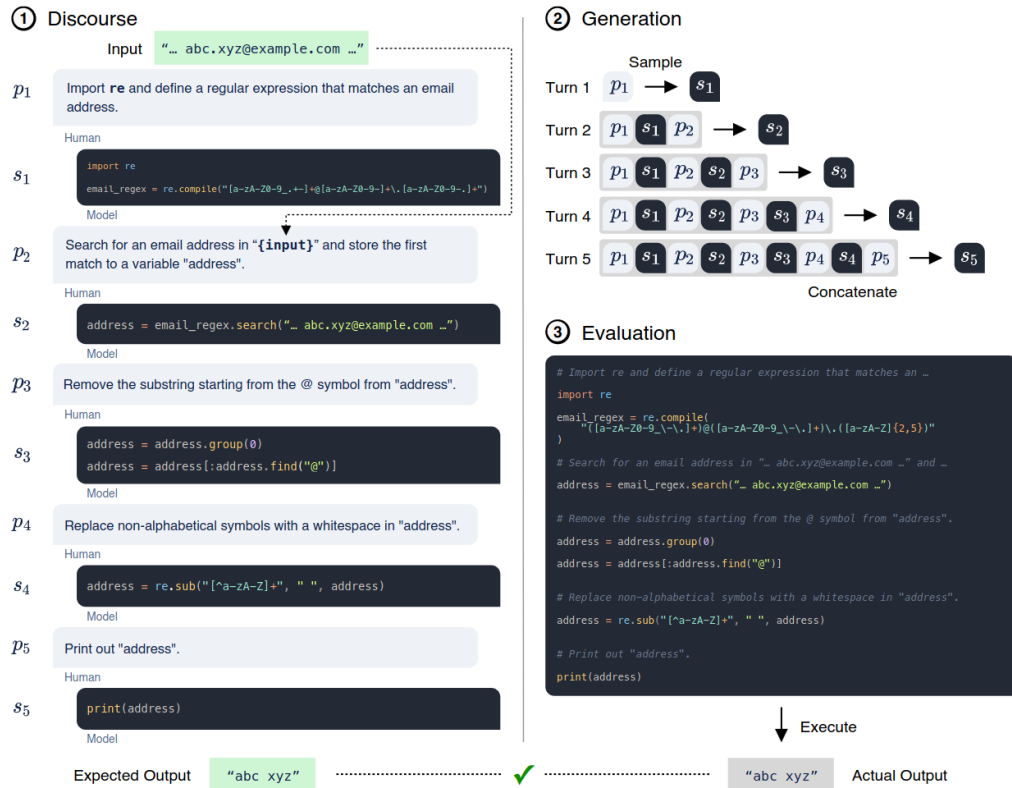
- **Research and Data Collection (20%)**
 - Correct annual report selected
 - All relevant data extracted properly
- **Analysis (40%)**
 - ...
- **Presentation (20%)**
 - ...
-



1. Assessment in teaching and learning: Provide rubric about a course, automated grading

2. Content Creation and Editing: Pathways LM (PaLM), Dynamic Prompting, Detailed Outline Control (DOC)

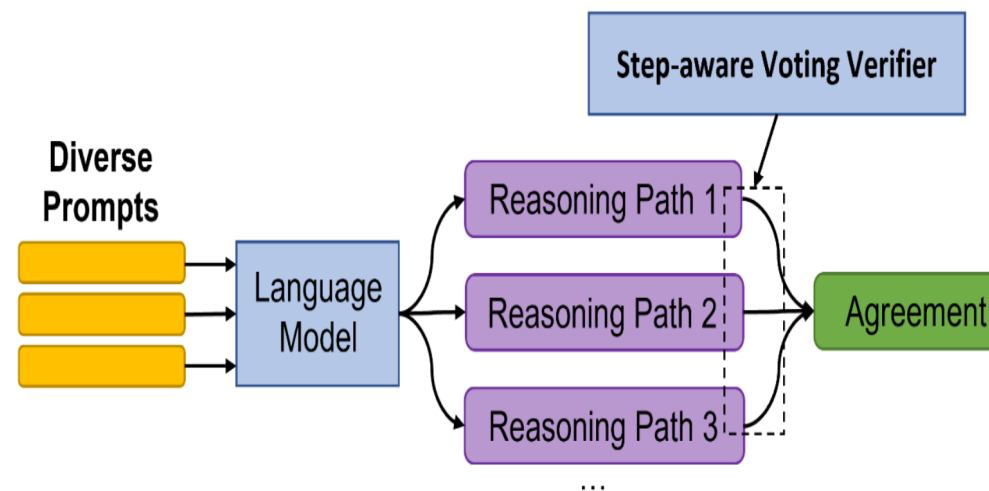
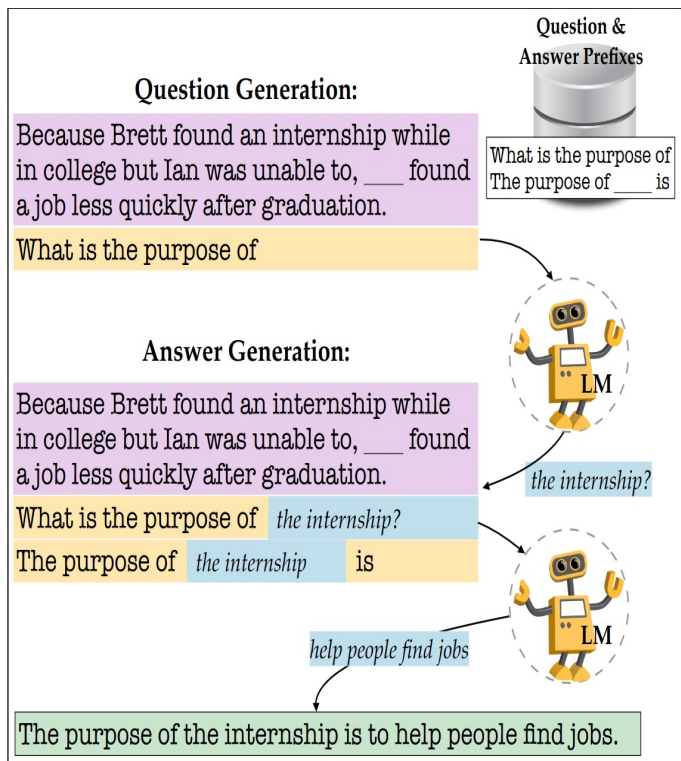
Dataset Generation



3. Computer Programming: Self-Debugging, Multi-step prompts, Repo-Level Prompt

4. Dataset generation: Is GPT-3 a Good Data Annotator?

Dataset Generation



5. Reasoning Task: Self-Talk, Diverse Prompts, Re-ranking

Future Direction

1. Need Better Understanding of Model Structure

2. Agent for AI-generated content (AIGC) tools

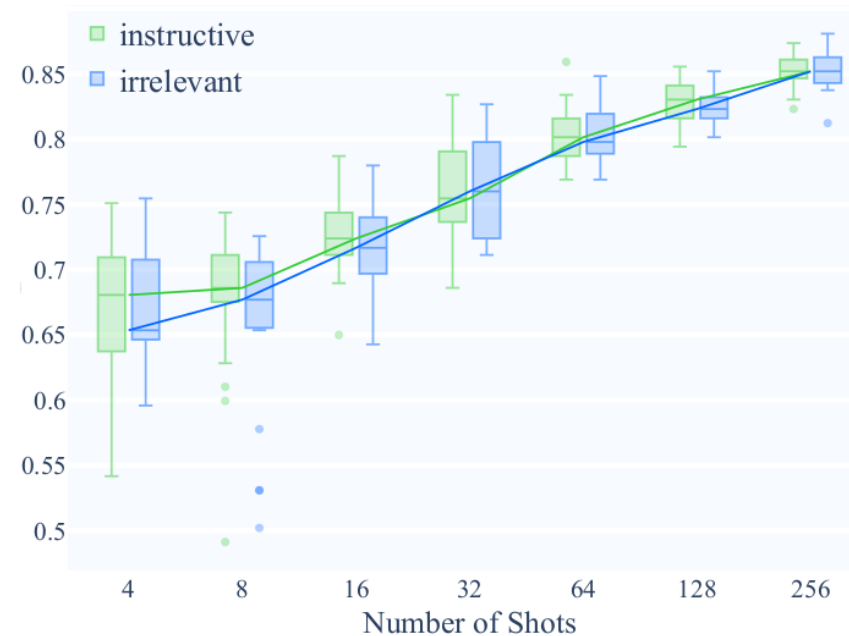


Figure 2: T0 (3B) on RTE. There is no practical difference between the performance of the models trained with instructive templates vs. those trained with irrelevant templates at any number of shots.

Conclusion

- Prompt Technique as to guiding and optimizing LLMs
- Understanding structure of LLMs is important for further Prompting Technique

Paper 2

Skeleton Of Thought: Prompting LLMs For Efficient Parallel Generation

Presenters:

Soneya Binta Hossain (sh7hv)

Jessie Chen (hc4vb)

Skeleton Of Thought: Prompting LLMs For Efficient Parallel Generation



Outline:

- Motivation
- High-level Overview
- Method
- Evaluation
- SoT with Router (SoT-R)
- Conclusion, Limitations and Future Work

Motivation

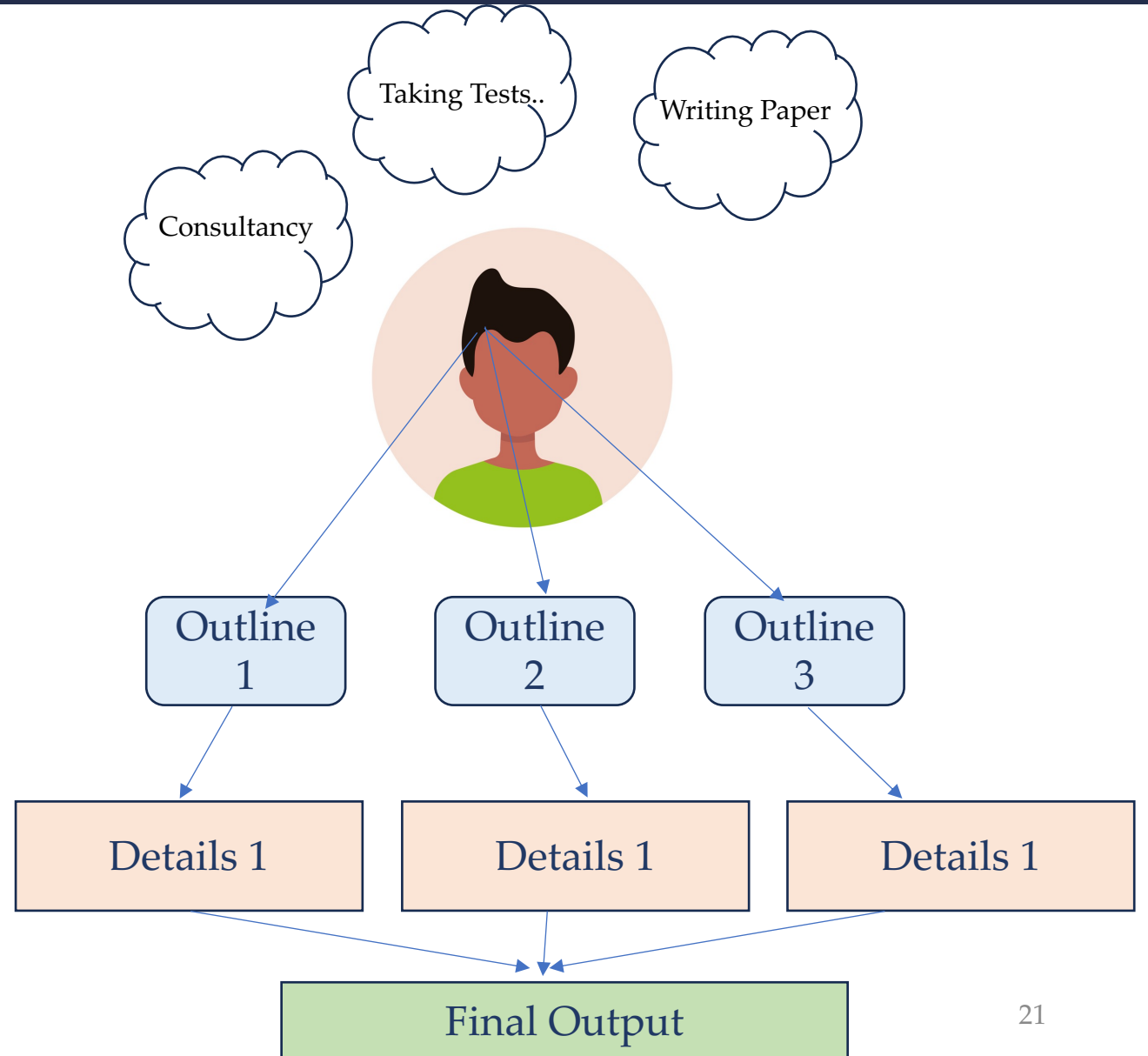
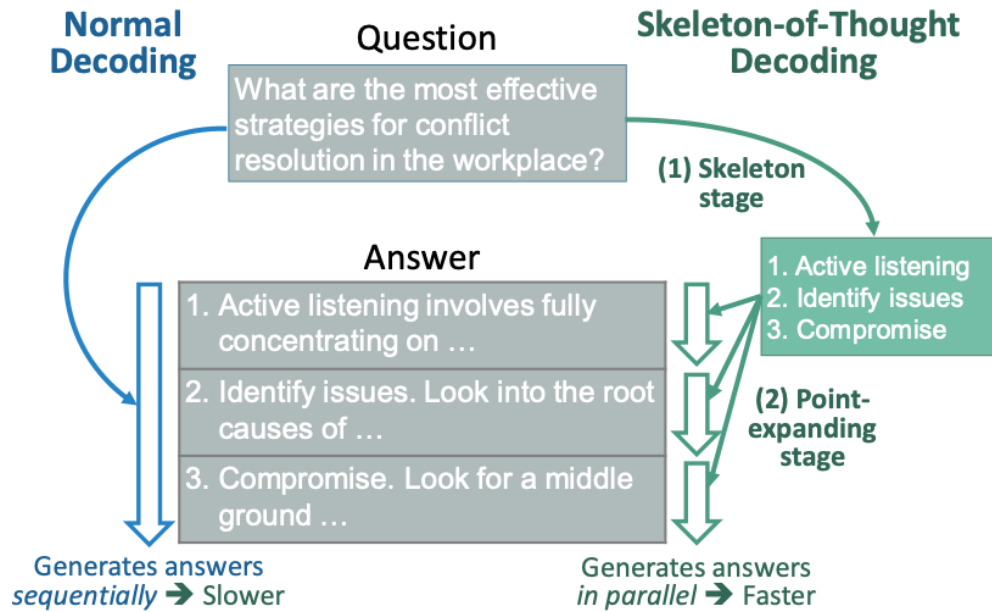
LLMs are great!!!



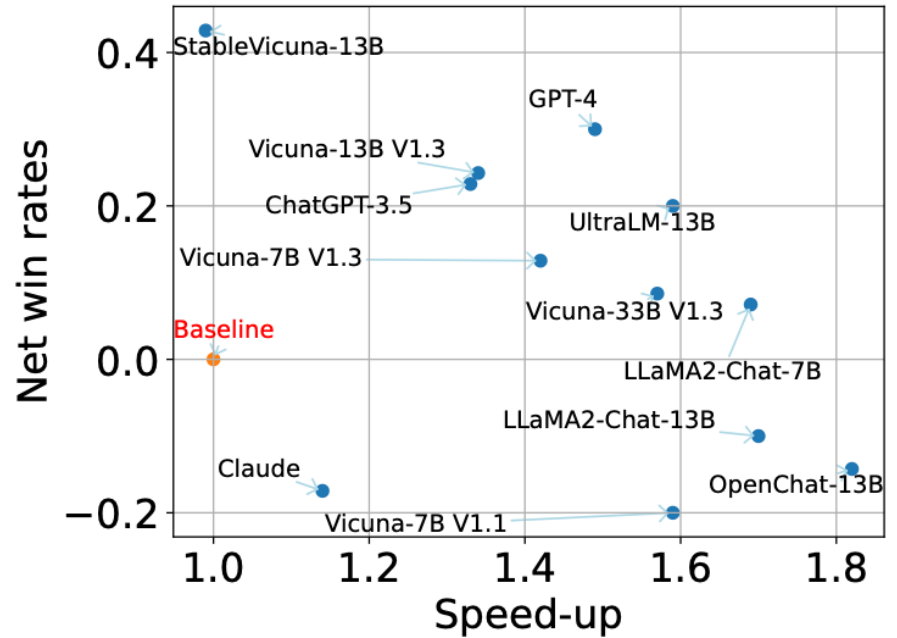
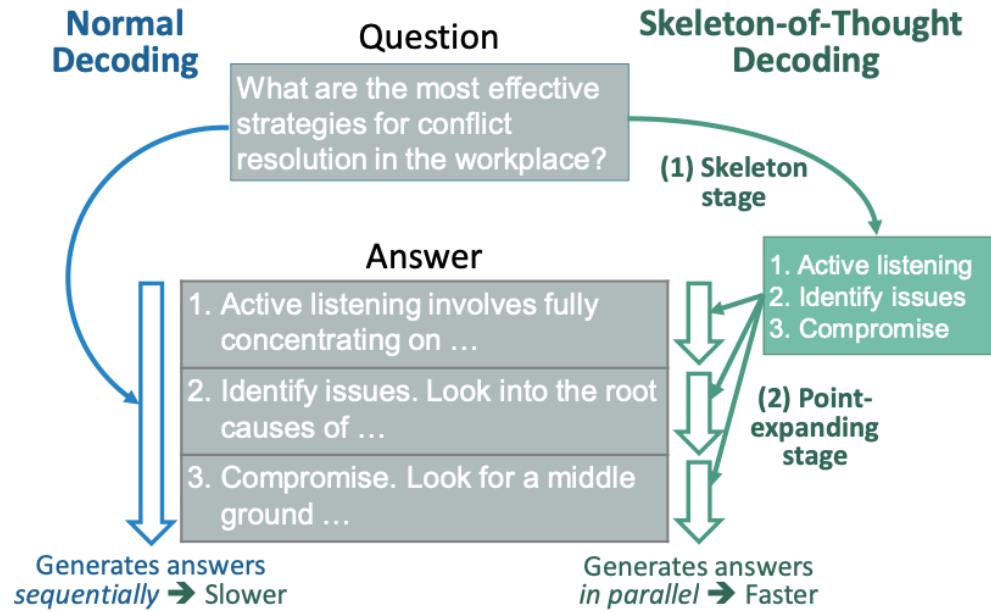
But inference is slow

- Reason for LLMs' slow inference
 - A large model size
 - Expensive attention operation
 - The sequential decoding approach
- Existing work either compress/redesign the model, serving system, hardware
- **This work instead focus on the 3rd axis and propose Skeleton Of Thought for efficient parallel decoding**

High-level Overview



High-level Overview



Left: SoT first prompts the LLM to give out the skeleton, then conducts batched decoding or parallel API calls to expand multiple points in parallel, and finally aggregates the outputs to get the final answer.

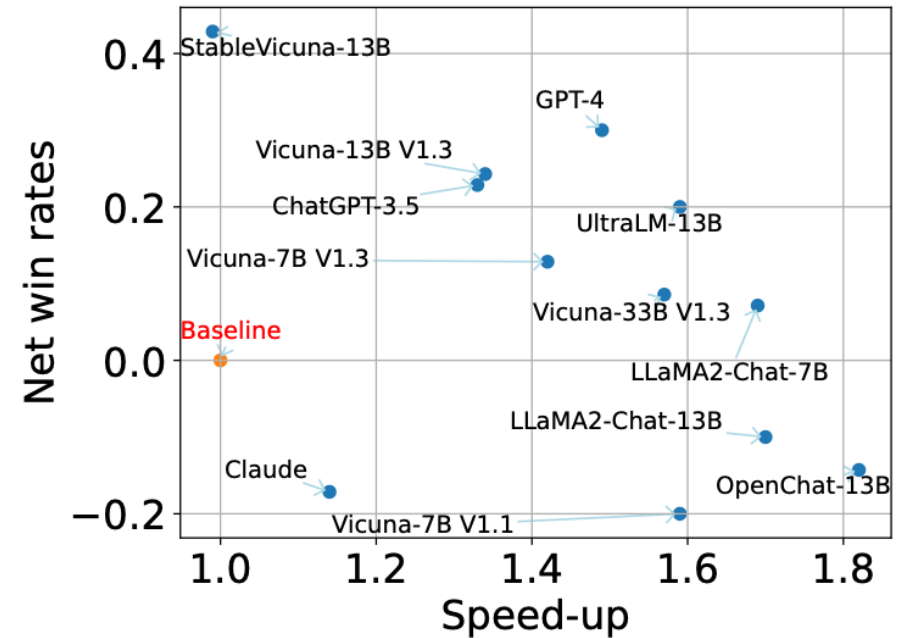
Right: The net win rate is the difference between the fraction of questions that SoT-R has better and worse answers than normal generation. The speed-up is the ratio between the latency of normal and SoT-R generation.

High-level Overview

- From 22 s to 12 s (1.83× speed-up) with Claude.
- From 43 s to 16 s (2.69× speed-up) with Vicuna-33B V1.3 on an NVIDIA A100.

- Suitable for questions requiring a long answer, can be planned ahead.
- Not suitable for questions requiring step-by-step reasoning or need a short answer.
- SoT with router (SoT-R), employs a router to only trigger SoT for suitable questions.

- Tested on 12 recent LLMs.
- Up to 2.39x speed up, also improves answer quality.



Right: The net win rate is the difference between the fraction of questions that SoT-R has better and worse answers than normal generation. The speed-up is the ratio between the latency of normal and SoT-R generation.

Method

- LLM generates the skeleton first

Prompt 1. Skeleton Prompt Template T^s

[User:] You're an organizer responsible for only giving the skeleton (not the full content) for answering the question. Provide the skeleton in a list of points (numbered 1., 2., 3., etc.) to answer the question. Instead of writing a full sentence, each skeleton point should be very short with only 3~5 words. Generally, the skeleton should have 3~10 points. Now, please provide the skeleton for the following question.

{question}

Skeleton:

[Assistant:] 1.

- Each key point from the skeleton is expanded in parallel

Prompt 2. Point-Expanding Prompt Template T^{pe}

[User:] You're responsible for continuing the writing of one and only one point in the overall answer to the following question.

{question}

The skeleton of the answer is

{skeleton}

Continue and only continue the writing of point {point index}. Write it ****very shortly**** in 1~2 sentence and do not continue with other points!

[Assistant:] {point index}. {point skeleton}

Method

Proprietary models with only API access:

- Multiple parallel API calls
- More cost for an increased number of API requests and tokens.

Prompt 2. Point-Expanding Prompt Template T^{pe}

[User:] You're responsible for continuing the writing of one and only one point in the overall answer to the following question.

{question}

The skeleton of the answer is

{skeleton}

Continue and only continue the writing of point *{point index}*. Write it **very shortly** in 1~2 sentence and do not continue with other points!

[Assistant:] *{point index}*. *{point skeleton}*

For open-source models:

- Batched processing is used, paddings are added to the left of each request
- Decoding latency is mainly due to weight loading rather than activation loading or computation
- Increased batch size does not increase per token latency much
- SoT allows Bx more token decoding within same amount of time

Evaluation



- Vicuna-80, 80 questions from nine categories: coding, math, writing, roleplay, and so on
- WizardLM, 218 questions spanning more categories and diverse difficulties



- 12 models (9 open-source models and 3 API-based models)
- The weights of all the open-source models from Hugging Face.



- Efficiency (for models and different type of questions)
- Overall answer quality
- SoT-R evaluation

Evaluation

Evaluation of Efficiency:



API-based Model:

```
Start = time.time(); ...; Elapsed_time = time.time() - Start
```

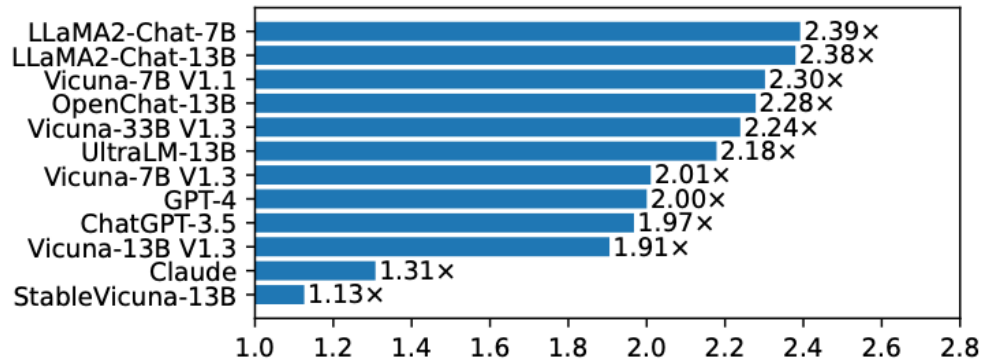
SoT Latency = latency of the skeleton API call + the slowest point-expanding API

Open Source Model:

- LLaMA 7B, 13B, or 33B architectures
- Latency profiling table for each LLaMA architecture on NVIDIA A100
- Latency for
 - 1) prefilling sequences of length 1 to 700 across batch sizes 1 to 16.
 - 2) decoding one token with context lengths of 1 to 1024 across batch sizes 1 to 16.
- SoT latency estimated based on number of points B , token lengths of the requests and responses of the skeleton and point-expanding stages.

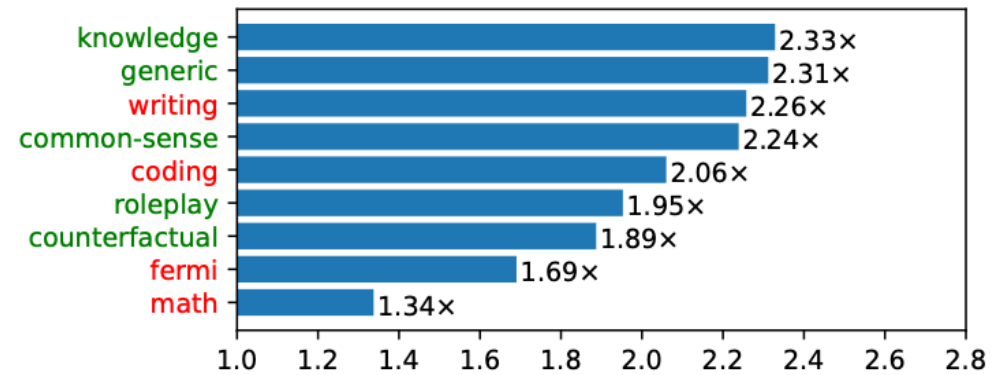
Evaluation – Efficiency

SPEED-UP BREAKDOWN: MODELS



(a) Different models.

SPEED-UP BREAKDOWN: QUESTION CATEGORI

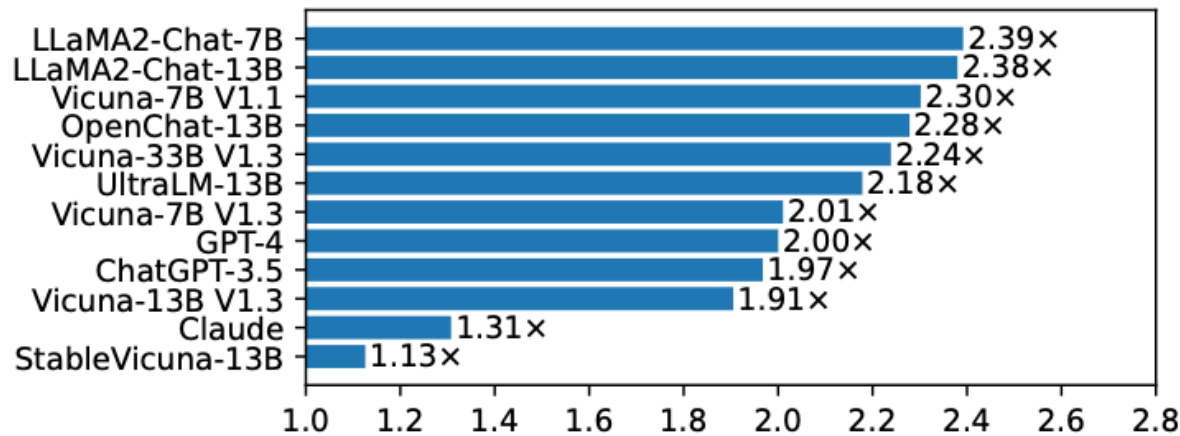


(b) Different categories.

Figure 2: Average speed-ups of SoT on different models and question categories.

Evaluation – Efficiency

SPEED-UP BREAKDOWN: MODELS



(a) Different models.

Findings:

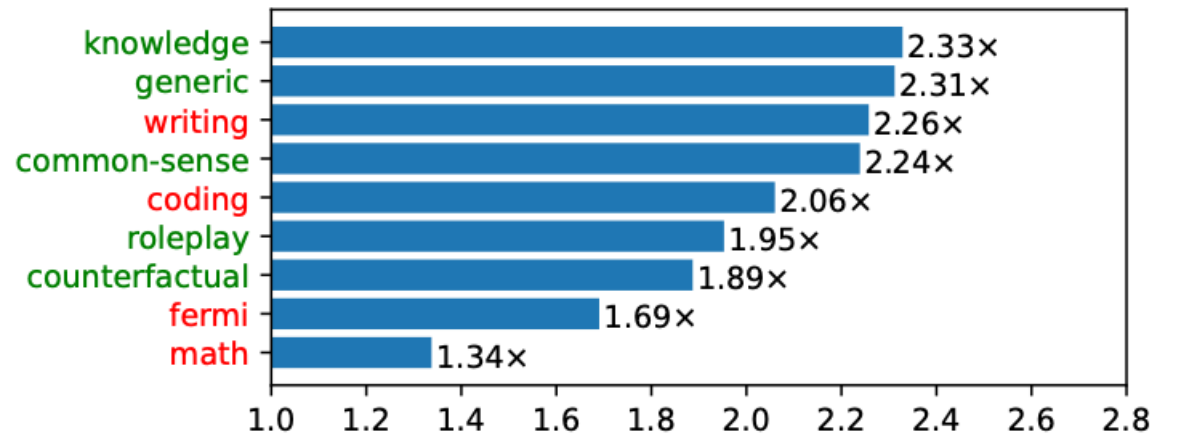
- **#Points:** LLaMA2, Vicuna-7B V1.1, Vicuna-7B V1.3, and ChatGPT-3.5 (<6), GPT-4 and StableVicuna-13B (≈ 9)
- **PE response length:** API-based model follow the pe request better with shorter responses than the open-source model
- **Length Balance:** LLaMA2 and the API-based models generate more balanced point-expanding responses
- **Overall Length:** SoT generated answers are on average, 1~2x longer than normal generation

Evaluation – Efficiency

Findings:

- SoT obtains speed-ups for all question categories
- SoT speeds up the overall answer generation process by 1.89× to 2.33× for the 5 categories.

SPEED-UP BREAKDOWN: QUESTION CATEGORI



(b) Different categories.

Jessie Chen (hc4vb)

Evaluation – Overall Quality

- **Evaluation Process**
 - present a question and a pair of answers to an LLM judge.
- **LLM-based evaluation frameworks:**
 - FastChat: general metric
 - LLMZoo: general metric plus 5 detailed metrics - coherence, diversity, immersion, integrity, and relevance.
- **Extensions to avoid evaluation bias**
 - Running the evaluation twice with either ordering of the two answers
 - For each run, a score is assigned: 1 – win; 0 – tie; -1 – lose
 - Sum the two scores to get the final score
- **Net win rates**
 - $(\#win - \#lose) / \text{total number of questions}$

Evaluation – Evaluation of Answer Quality

- **Overall Quality**

- There is a discrepancy between the two metrics on win rates
- SoT is not worse than the baseline in around 60% of the cases
- The lose rates are also pretty high.

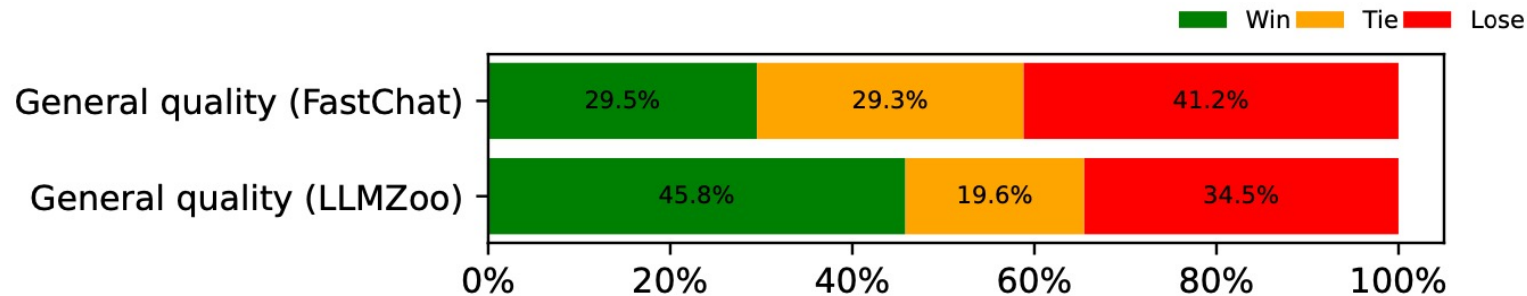
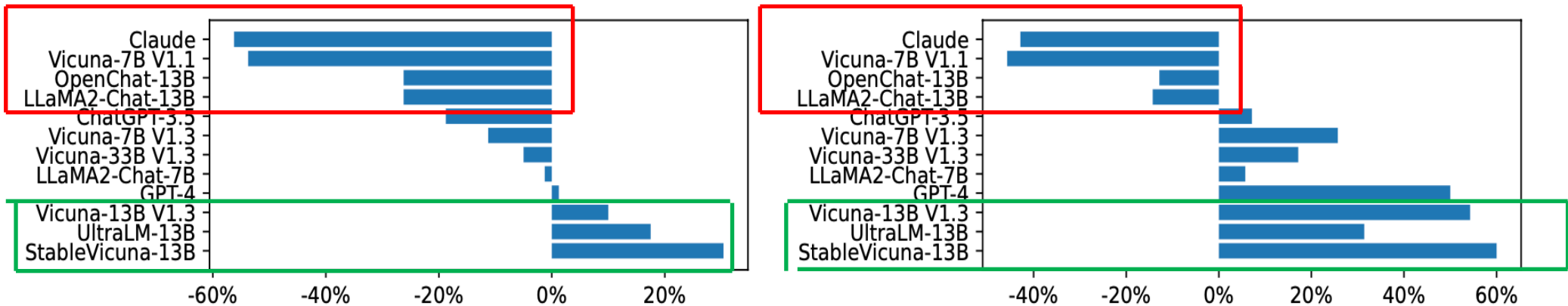


Figure 3: Win/tie/lose rates of SoT v.s. normal generation using “general” metrics from FastChat and LLMZoo. SoT performs better than or equal to normal generation in around 60% cases.

Evaluation – Evaluation of Answer Quality

- **Quality of each Model:**
 - Both metrics agree that OpenChat-13B, Vicuna-7B V1.1, Claude, LLaMA2-Chat-13B have **negative net win rates**
 - Vicuna-13B V1.3, StableVicuna-13B, and UltraLM-13B have **positive net win rates**.



(a) Metric: general quality (FastChat).

(b) Metric: general quality (LLMZoo).

Figure 4: Net win rates of SoT on different models.

Evaluation – Evaluation of Answer Quality

- Reasons for Bad Net Win Rates

Models write the whole answer in the skeleton stage.

Models do not provide details at point-expanding stage.

Some strong models have very high-quality answers that are hard to beat.

Question: How many lightning strikes occur on Earth each day? Try to explain your answer. Your explanation should take the reader through your reasoning step-by-step.

SoT's answer (OpenChat-13B) (Orange texts are the skeleton.)

(Only showing the skeleton)

- 1.Introduction.
2. The Earth's atmosphere.
3. Lightning formation.
4. Electrical discharge.
5. Lightning strikes per second.
6. Calculating the total.

Skeleton:

1. Introduction.

Lightning is a beautiful and awe-inspiring natural phenomenon, but it can also be dangerous. It occurs when positively charged particles in a thundercloud discharge to the negatively charged Earth. The process is still not fully understood, but scientists have been able to estimate how many lightning strikes occur on Earth each day.

Skeleton:

1. The Earth's atmosphere.

The Earth's atmosphere plays a crucial role in the formation of lightning. It consists of different layers, with the troposphere being the lowest and most active. The troposphere is where most of the weather occurs, and it is where lightning primarily forms. The atmosphere's temperature, humidity, and wind patterns all contribute to the likelihood of a lightning strike.

Skeleton:

2. Lightning formation.

When warm, moist air rises, it cools and forms water <more texts are omitted...>

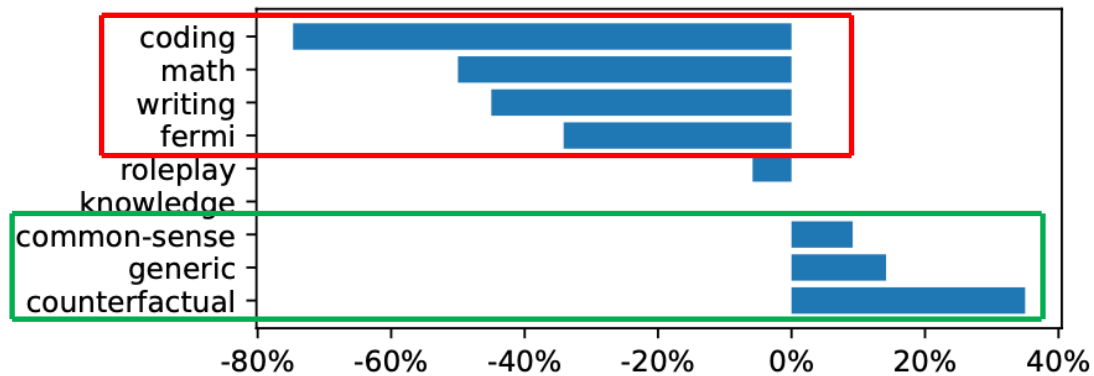
Question: As a superhero, how would you explain your origin story to a curious child?

SoT's answer (Vicuna-7B V1.1) (Orange texts are the skeleton.)

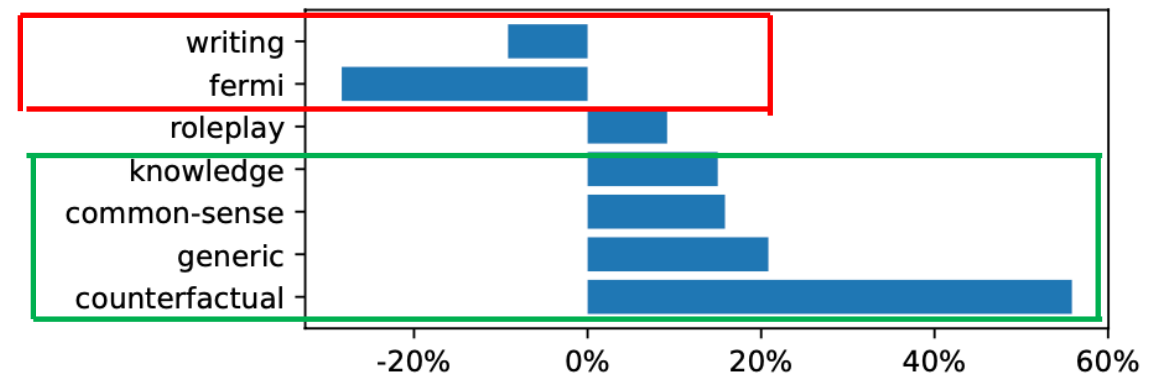
1. Born with extraordinary abilities.
2. Trained to use powers for the greater good.
3. Protecting the innocent and fighting villains.I use my powers to defend the defenseless and bring justice to those who have been wronged.
4. Inspired by courage, justice, and selflessness.

Evaluation – Evaluation of Answer Quality

- **Quality of each Question Category:**
 - SoT performs relatively well on generic, common-sense, knowledge, and counterfactual questions.
 - Relatively poorly on writing, fermi, math, and coding.



(a) Metric: general quality (FastChat).



(b) Metric: general quality (LLMZoo).

Figure 5: Net win rates of SoT on different question categories.

Evaluation – Evaluation of Answer Quality

- **Quality of Detailed Metrics:**
 - SoT improves the diversity and relevance while hurting the immersion and coherence.

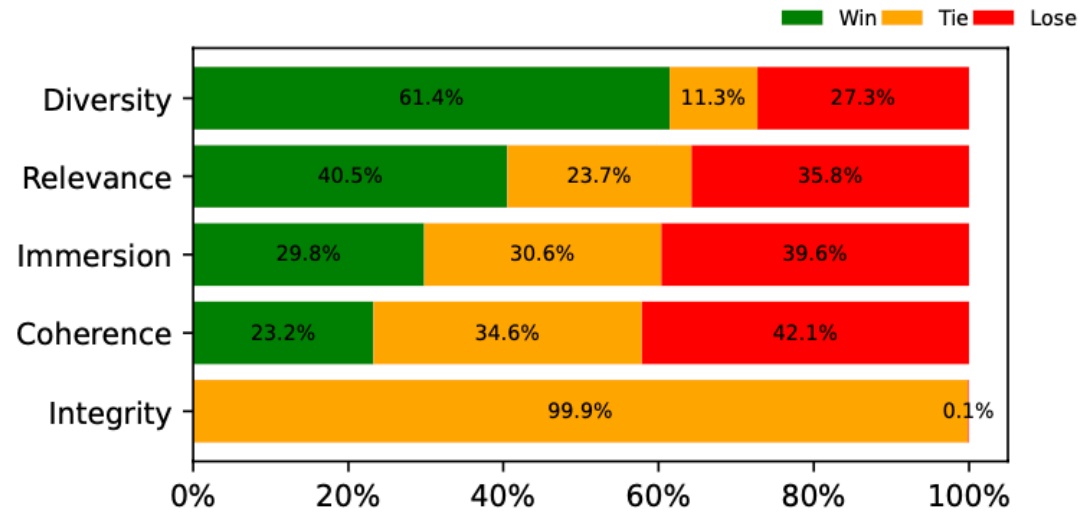


Figure 6: Win/tie/lose rates of SoT v.s. normal generations using metrics from LLMZoo. SoT performs well on diversity and relevance, and relatively worse on coherence and immersion.

SoT-R – Definition and Framework

- **Prompting Router**
 - Ask the LLM if the desired answer is in a list of independent points

Prompt 4. LLM Prompting as the Router

[User:] Question: *{question}*

How would you like to answer the question?

A. Organize the answer as a list of points or perspectives (in the format of 1., 2., 3., etc.), and the points or perspectives can be answered independently without referring to the contents of the previous points.

B. Organize the answer as a list of points or perspectives (in the format of 1., 2., 3., etc.), and the contents of later points or perspectives cannot be answered independently without referring to the contents of the previous ones.

C. Do not organize the answer as a list of points or perspectives.

Just say A, B, or C. Do not explain. Do not provide an answer to the question.

[Assistant:]

- **Trained Router**
 - **Annotate** the LIMA training set: a label of 1 or 0.
 - **Fine-tune** a RoBERTa model using the labeled data.
 - Ask the RoBERTa to **classify** if the SoT is suitable for the desired answer

SoT-R – Evaluation

- SoT-R obtains **lower speed-ups** than SoT
- SoT-R significantly **improves the answer quality** on questions where SoT is not suitable.
- The two types of SoT-R perform similarly to a human router.

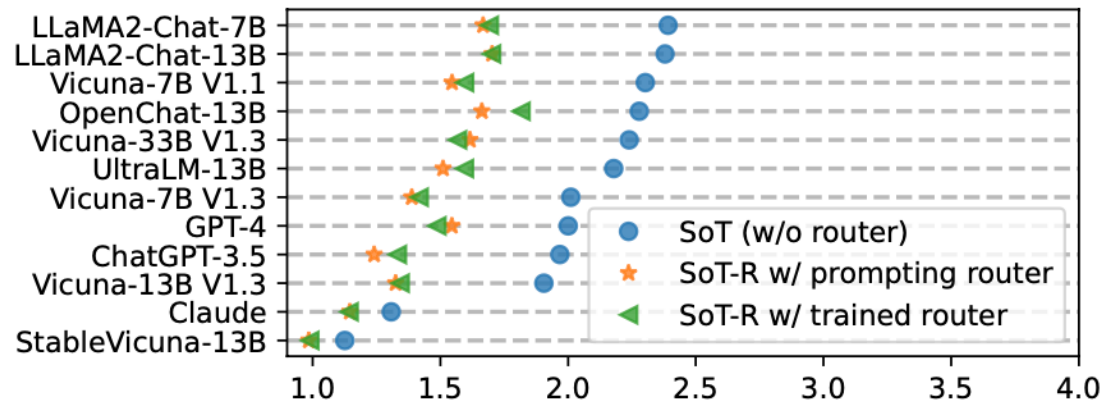


Figure 7: Speed-ups of SoT and SoT-R on different models across all question categories of the Vicuna-80 dataset.

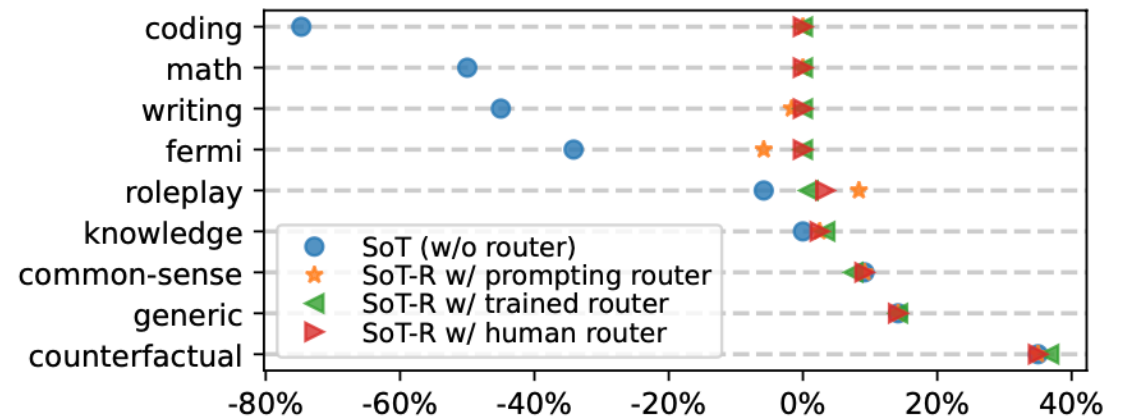


Figure 8: Net win rates of SoT and SoT-R on different question categories of the Vicuna-80 dataset (evaluated with the FastChat metrics).

Conclusion

- Efficient LLM methods at model and system levels
 - SoT is a **data-level** technique.
- Prompting methods for LLMs
 - SoT is the first attempt at exploiting the **power of prompting to improve efficiency**.
- Answer quality evaluation
 - The answer quality evaluation is far from perfect due to the limited prompt set, the potential bias of GPT-4 judges, and the inherent difficulty of evaluating LLM generations.
- Efficiency and overhead of SoT in different scenarios
 - **higher costs** due to the increased number of API calls and tokens.
 - **computation overhead**
- Eliciting or improving LLMs' ability
 - Graph-of-Thoughts

Paper 3

Topologies of Reasoning: Demystifying Chains, Trees, and Graphs of Thoughts Presenters

Ali Zafar Sadiq (mzw2cu)
Jeffrey Chen (fyy2ws)

Topologies of Reasoning: Demystifying Chains, Trees, and Graphs of Thoughts

Presenter
Ali Zafar Sadiq (mzw2cu)

Evolution of reasoning topologies

Thoughts and Reasoning Topologies

What is a Thought ?

- In CoT, a thought refers to a **statement within a paragraph** that contains a **part of the reasoning process** aimed at **solving the input task**.
- In ToT, in some tasks, such as Game of 24, a thought means **an intermediate or a final solution** to the **initial question**.
- In GoT, a thought contains a **solution of the input task (or of its subtask)**.

Therefore, Paper proposes thought to be

"Semantic unit of task resolution, i.e., a step in the process of solving a given task"

What is a Reasoning Topology?

Authors models thoughts as nodes; edges between nodes correspond to dependencies between these thoughts and a topology can be defined as $G=(V,E)$,

Taxonomy of Reasoning Scheme

Blueprint & taxonomy of a structure-enhanced reasoning scheme

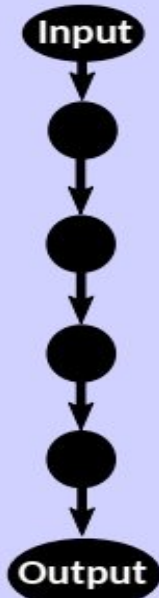
Topology

1 Class

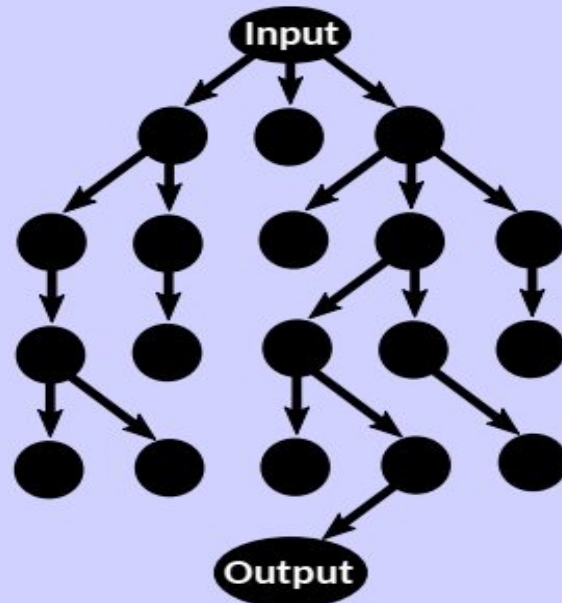
1 Topology Class

What is the connection structure of reasoning steps?

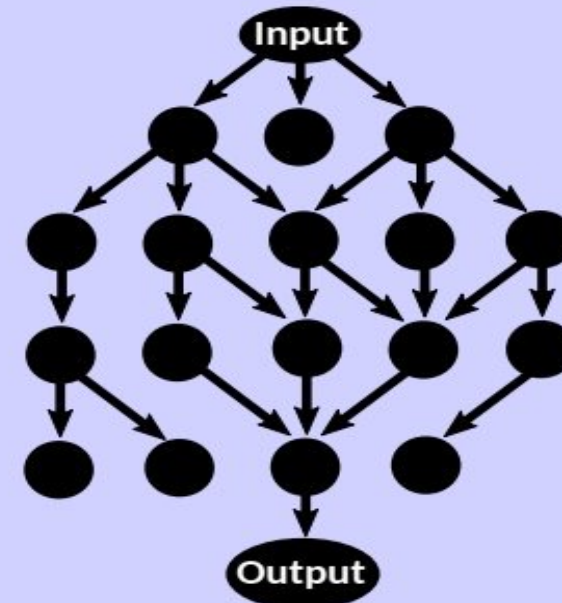
Chain



Tree



Graph



Taxonomy of Reasoning Scheme

Blueprint & taxonomy of a structure-enhanced reasoning scheme

Topology

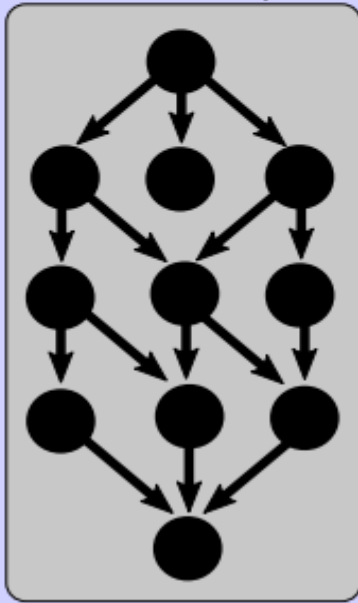
- 1 Class
- 2 Scope

2 Topology Scope

Can the topology extend beyond a single prompt?

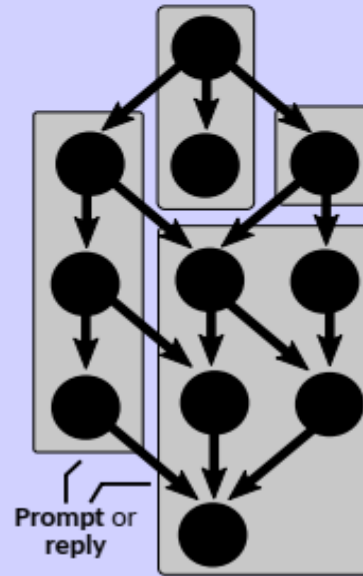
Single-prompt

The structure is contained within a single prompt/reply

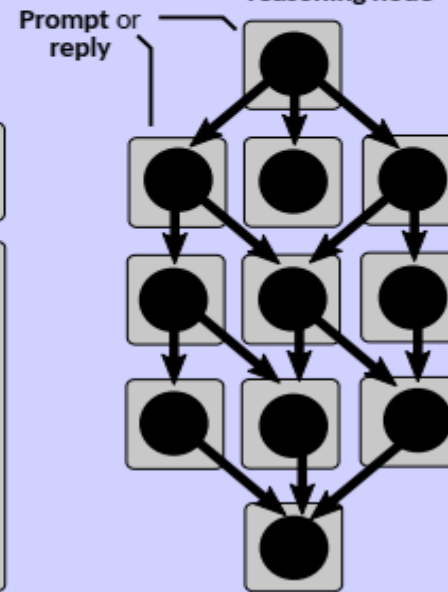


Multi-prompt

One prompt/reply can contain multiple reasoning nodes



One prompt/reply can contain a single reasoning node



Taxonomy of Reasoning Scheme

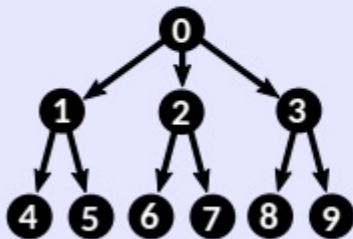
Blueprint & taxonomy of a structure-enhanced reasoning scheme

Topology

- 1 Class
- 2 Scope
- 3 Representation

3 Topology Representation

How is the topology structure represented?



Implicit

"The first preliminary solution should be enhanced three times. Each of these three enhanced solutions should be further augmented in two attempts"

Explicit

<node 0> connects to <node 1>, <node 2>, <node 3>
<node 1> connects to <node 4>, <node 5>
<node 2> connects to <node 6>, <node 7>
<node 3> connects to <node 8>, <node 9>

Taxonomy of Reasoning Scheme

Blueprint & taxonomy of a structure-enhanced reasoning scheme

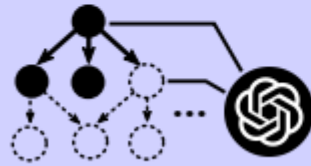
Topology

- 1 Class
- 2 Scope
- 3 Representation
- 4 Derivation

4 Topology Derivation

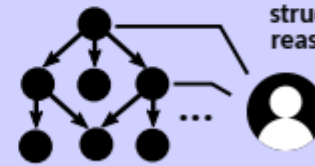
How is the topology structure derived?

Automatic, semi-automatic



The structure is constructed by the LLM on-the-fly, either fully (automatic) or partially, with certain control from the user (semi-automatic)

Manual



The user statically prescribes the structure before reasoning starts

Taxonomy of Reasoning Scheme

Blueprint & taxonomy of a structure-enhanced reasoning scheme

Topology

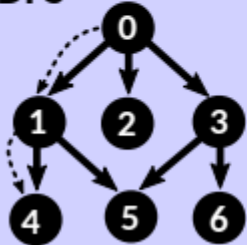
Schedule

- 5 Reasoning Schedule
- 6 Schedule Representation

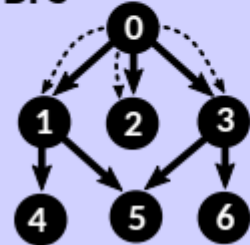
5 Schedule Class

How is the topology structure explored?

DFS



BFS



...

6 Schedule Representation

How is the schedule represented?

Textual description

"Proceed in the BFS manner"

"Proceed in the DFS manner"

In-context examples

"Traverse nodes <0>, <1>, <4>"

"Traverse nodes <0>, <1>, <2>, <3>"

...

Taxonomy of Reasoning Scheme

Blueprint & taxonomy of a structure-enhanced reasoning scheme

Topology

Schedule

Extensions

7 Harnessed Parts of the Generative AI Pipeline

7 Generative AI Pipeline

What parts of the generative AI pipeline are used, besides prompting?

Modalities?



Pre-training?

Fine-tuning?



Tools?



Retrieval?



LLM Reasoning Schemes Represented With Taxonomy

| Scheme | Topology | | | Reasoning Schedule | | | AI Pipeline | | | | Remarks | | | |
|--|-----------------------|---------------|--------------|------------------------------|-----|-----|--------------------------|-----|-----|---|---------|---|---|-----------------|
| | single-prompt | | multi-prompt | Scheme | | | Modalities | | | | | | | |
| | Class | Rp. | Dv. | Class | Rp. | Dv. | Scheme | Rp. | Dv. | P | F | R | T | |
| Chain-of-Thought (CoT) 195 | chain | I (text) | SA | - | - | - | - | - | - | × | × | × | × | text |
| Zero-shot-CoT 112 | chain | I (text) | SA | - | - | - | - | - | - | × | × | × | × | text |
| SelfAsk 152 | chain | I (text) | SA | - | - | - | - | - | - | × | × | ▣ | × | text |
| Plan-and-Solve Prompting 188 | chain | I (text) | SA | - | - | - | - | - | - | × | × | × | × | text |
| Program of Thoughts (PoT) 41 | chain | I (text,code) | SA | - | - | - | - | - | - | × | × | × | ▣ | text,code,table |
| Selection-Inference (SI) 51 | - | - | - | chain | E | M | linear | I | M | × | ▣ | × | × | text |
| Chain-of-symbol (CoS) 89 | chain | I (text) | SA | chain | E | SA | linear | I | M | × | × | × | × | text |
| Least-to-Most Prompting 233 | - | - | - | chain | E | SA | linear | I | M | × | × | × | × | text |
| Decomposed Prompting 105 | - | - | - | chain | E | SA | linear | I | M | × | × | ▣ | ▣ | text |
| LogiCoT 231 | chain | I (text) | SA | tree | E | SA | linear | I | M | × | × | × | × | text |
| SELF-REFINE 140 | - | - | - | chain | E | SA | linear | I | M | × | × | × | × | text |
| Reflexion 168 | - | - | - | chain | E | SA | linear | I | M | × | × | × | × | text |
| Reasoning Graph Verifier (RGV) 35 | chain | I (text) | SA | graph | E | SA | linear | I | M | × | × | × | × | text |
| Plan, Verify and Switch (PVS) 131 | chain | I (text,code) | SA | chain | E | SA | linear | I | M | × | × | × | ▣ | text,code |
| Chameleon 136 | - | - | - | chain | E | SA | linear | I | M | × | × | ▣ | ▣ | text,code |
| ChatCoT 45 | chain | I (text) | SA | chain | E | SA | linear | I | M | × | × | ▣ | ▣ | text |
| Tree-of-Thought (ToT) 133 | tree | I (text) | M | tree | E | SA | arbitrary | E | M | × | × | × | × | text |
| Tree of Thoughts (ToT) 213 | tree | I (text) | M | tree | E | SA | arbitrary | E | M | × | × | × | × | text |
| Thought Decomposition 205 | tree | I (text) | M | tree | E | SA | beam [†] | E | SA | × | × | × | ▣ | text,code |
| Self-Consistency with CoT 190 | chain | I (text) | M | tree (▣) [†] | E | SA | - | - | - | × | × | × | × | text |
| Creswell and Shanahan 50 | tree | I (text) | M | tree | E | SA | beam | E | A | × | ▣ | × | × | text |
| Dynamic Least-to-Most Prompting 58 | tree | I (text) | M | tree | E | A | bottom up | E | A | × | × | ▣ | × | text,code |
| Algorithm of Thoughts (AoT) 166 | tree | I (text) | M | - | - | - | DFS, (BFS) | I | M | × | × | × | × | text |
| Tree of Uncertain Thought (TouT) 145 | tree | I (text) | M | tree | E | SA | BFS, DFS | E | M | × | × | × | × | text |
| Tree-of-Mixed-Thought 91 | tree | I (text) | M | tree | E | SA | DFS | E | A | × | × | × | ▣ | scene graphs |
| Tree of Clarifications (ToC) 106 | tree (▣) [†] | I (text) | M | tree | E | SA | BFS | E | A | × | × | ▣ | × | text |
| Tree Prompting 170 | - | - | - | tree | E | A | top-down | E | A | × | × | × | × | text |
| Skeleton-of-Thought (SoT) 148 | tree (▣) [†] | I (text) | M | tree (▣) [†] | E | A | parallel | E | A | × | × | × | × | text |

LLM Reasoning Schemes Represented With Taxonomy

| Scheme | Topology | | | | | | Reasoning Schedule | | | | AI Pipeline | | | | Remarks | |
|---|------------------|-----------|-----|--------------|-----|------|--------------------|-----|-----|---|-------------|----|---|------------|------------|--------------------------------------|
| | single-prompt | | | multi-prompt | | | Reasoning Schedule | | | | AI Pipeline | | | | | |
| | Class | Rp. | Dv. | Class | Rp. | Dv. | Scheme | Rp. | Dv. | P | F | R | T | Modalities | | |
| Branch-Solve-Merge (BSM) 162 | tree (depth one) | I (text) | M | graph (🗃️)† | E | SA | BFS | E | M | | × | × | × | × | text | † double tree (🗃️) |
| Thought Propagation (TP) 218 | graph (🗃️) | arbitrary | M | graph (🗃️)† | E | SA | BFS | E | M | | × | × | × | × | text | † double tree (🗃️) |
| Socratic Questioning 154 | tree (depth one) | I (text) | M | graph (🗃️)† | E | SA | DFS | E | M | | × | × | × | × | multi | † double tree |
| Graph of Thoughts (GoT) 10 | graph (🗃️) | arbitrary | M | graph | E | M | arbitrary | E | M | | × | × | × | × | text | |
| Graph of Thought (GoT) 119 | 🔍 | 🔍 | 🔍 | graph | E | (S)A | DFS | E | 🔍 | | × | × | × | × | text | |
| Graph-of-Thought (GoT) 215 | graph | I (text) | M | chain | E | M | linear | E | M | | × | 🗃️ | × | × | text,image | |
| ControlLLM 132 | graph | E (json) | M | graph | E | M | DFS | E | M | | × | × | † | × | 🗃️ | text,image,video,audio † can be used |
| Cumulative Reasoning 224 | graph (DAG) | I (text) | M | graph (DAG) | E | SA | arbitrary | E | M | | × | × | × | × | text | |
| Everything of Thoughts (XoT) 57 | graph | 🔍 | L | chain | E | M | linear | E | M | | 🗃️ | × | × | × | text | |
| ResPrompt 99 | graph | I (text) | M | - | - | - | - | - | - | | × | × | × | × | text | |
| Hypergraph-of-Thought (HoT) 212 | hypergraph | 🔍 | M | - | - | - | - | - | - | | × | 🗃️ | × | × | text,image | |
| BatchPrompt 124 | batch | E (text) | M | chain | E | M | linear | E | M | | × | × | × | × | text | |
| Memory Injections 163 | - | - | - | - | - | - | - | - | - | | × | × | × | × | text | |

Chain of Thought Works

Math Reasoning

- **Multi-Step Reasoning :**
 - Chain-of-Thought (A **single-prompt scheme**, which uses **few-shot examples**, to guide LLM)
 - Program of Thoughts (PoT) (**use code to obtain a stepby-step generated**, functional Python program)
 - SelfAsk (**expands each step** in chain and **pose a follow-up question** which is then **answered next**)

User Prompt

Q: Alexis is applying for a new job and bought a new set of business clothes to wear to the interview. She went to a department store with a budget of \$200 and spent \$30 on a button-up shirt, \$46 on suit pants, \$38 on a suit coat, \$11 on socks, and \$18 on a belt. She also purchased a pair of shoes, but lost the receipt for them. She has \$16 left from her budget. How much did Alexis pay for the shoes?

A: Let S be the amount Alexis paid for the shoes.

She spent $S + 30 + 46 + 38 + 11 + 18 = S + 143$.

She used all but \$16 of her budget, so $S + 143 = 200 - 16 = 184$.

Thus, Alexis paid $S = 184 - 143 = 41$ for the shoes.

So the answer is 41.

Q: [more in-context examples...]

Q: Tobias is buying a new pair of shoes that costs \$95. He has been saving up his money each month for the past three months. He gets a \$5 allowance a month. He also mows lawns and shovels driveways. He charges \$15 to mow a lawn and \$7 to shovel. After buying the shoes, he has \$15 in change. If he mows 4 lawns, how many driveways did he shovel?

LLM Answer

Let D be the number of driveways Tobias shoveled.

He earned \$5 allowance per month for three months, so he has $\$5 * 3 = \15 from his allowance.

He mowed 4 lawns, each earning him \$15, so he made $\$15 * 4 = \60 from mowing lawns.

He also earned \$7 for each driveway shoveled, and he shoveled D driveways, so he made $\$7 * D$ from shoveling driveways.

In total, he has \$15 (allowance) + \$60 (mowing lawns) + \$7D (shoveling driveways) before buying the shoes.

After buying the \$95 shoes, he has \$15 in change.

Therefore: $\$15 + \$60 + \$7D - \$95 = \$15$

So, Tobias shoveled 5 driveways.

Chain of Thought Works

- **Zero-Shot Reasoning Instructions :** Elicit the same multi-step reasoning chains, but without the use of handtuned, problem-specific in-context examples
 - **Zeroshot-CoT--** “Let’s think step by step”
 - **Zeroshot-PoT--** “Let’s write a Python program step by step and return the result. Firstly we need to define the variables.”.

Creative Writing

User Prompt

Write a coherent passage of 4 short paragraphs. The end sentence of each paragraph must be:

1. It isn't difficult to do a handstand if you just stand on your hands.
2. It caught him off guard that space smelled of seared steak.
3. When she didn't like a guy who was trying to pick her up, she started using sign language.
4. Each person who knows you has a different perception of who you are.

Let's think step by step.

Overview of Chain of Thought Works

Creative Writing

User Prompt

Write a coherent passage of 4 short paragraphs. The end sentence of each paragraph must be:

1. It isn't difficult to do a handstand if you just stand on your hands.
2. It caught him off guard that space smelled of seared steak.
3. When she didn't like a guy who was trying to pick her up, she started using sign language.
4. Each person who knows you has a different perception of who you are.

Let's think step by step.

LLM Answer

Mastering a handstand may seem daunting at first, but the key lies in simplicity. It isn't difficult to do a handstand if you just stand on your hands. Breaking down the complex maneuver into a straightforward action can make the seemingly impossible, achievable. Sometimes, the most challenging tasks become manageable when approached with a clear and uncomplicated perspective.



In the vast expanse of outer space, surprises await even the most seasoned explorers. It caught him off guard that space smelled of seared steak. The unexpected aroma challenged preconceptions, reminding us that the universe is full of mysteries that extend beyond the visual and auditory senses. The interplay of senses in unfamiliar environments can reshape our understanding of the cosmos.



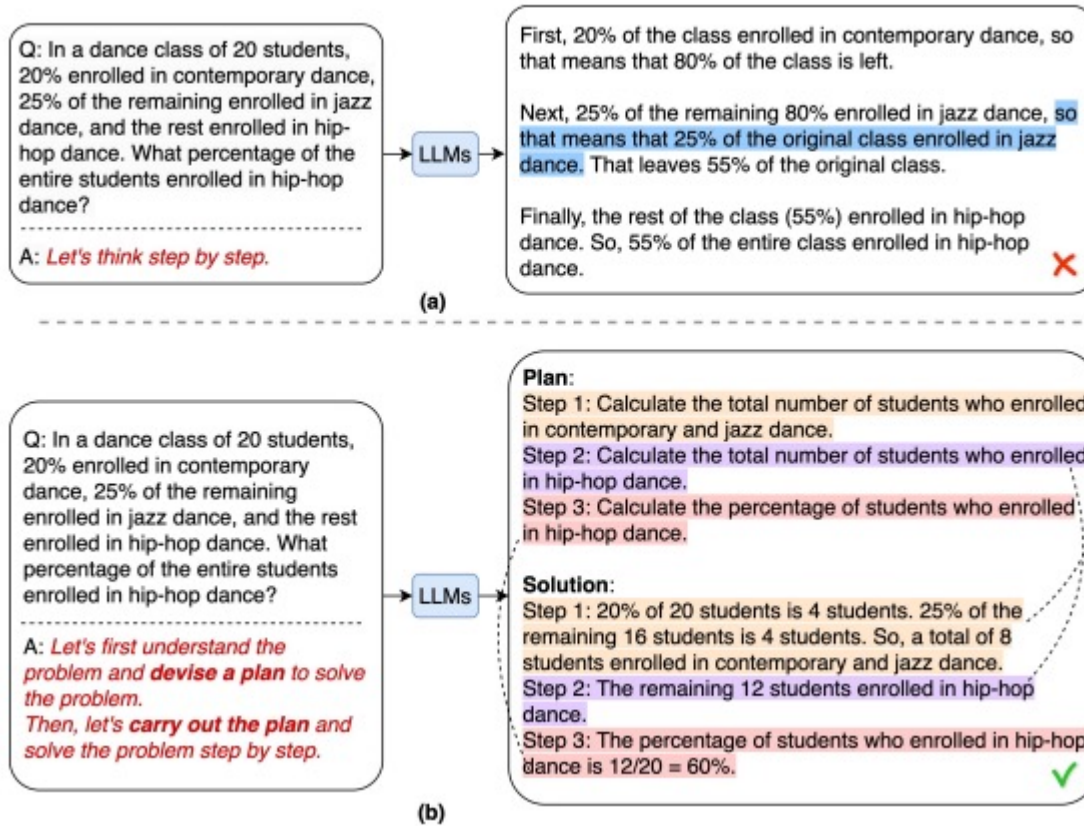
Navigating social interactions often demands creative strategies, especially when faced with unwanted advances. When she didn't like a guy who was trying to pick her up, she started using sign language. Non-verbal communication became her shield, a tool to ward off unwelcome attention without resorting to confrontation. In these moments, the power of expression transcends spoken words, allowing for a graceful escape from uncomfortable situations.



Perceptions of identity are as diverse as the individuals who form them. Each person who knows you has a different perception of who you are. It's a kaleidoscope of perspectives, shaped by unique experiences and interactions. Understanding that our multifaceted nature is reflected in the eyes of those around us can foster empathy and open the door to a richer tapestry of human connections.

Overview of Chain of Thought Works

- **Planning & Task Decomposition:**
 - Plan-and-Solve (PS) Prompting
First divides the complex task into a chain of sub-tasks and then executes these step-by-step.



Overview of Chain of Thought Works

- **Planning & Task Decomposition:**
 - Least-to-Most Prompting
Where decomposition of complex tasks or questions is conducted in the first node and the subtasks/sub-questions are solved in the subsequent nodes.

Stage 1: Decompose Question into Subquestions

Q: It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?

Language Model

A: To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"

Stage 2: Sequentially Solve Subquestions

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Subquestion 1 — Q: How long does each trip take?

Language Model

A: It takes Amy 4 minutes to climb and 1 minute to slide down. $4 + 1 = 5$. So each trip takes 5 minutes.

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Append model answer to Subquestion 1 — Q: How long does each trip take?

A: It takes Amy 4 minutes to climb and 1 minute to slide down. $4 + 1 = 5$. So each trip takes 5 minutes.

Subquestion 2 — Q: How many times can she slide before it closes?

Language Model

A: The water slide closes in 15 minutes. Each trip takes 5 minutes. So Amy can slide $15 \div 5 = 3$ times before it closes.

Overview of Chain of Thought Works

- **Planning & Task Decomposition:**

- Decomposed Prompting
A modular framework for a detailed decomposition of complex tasks.

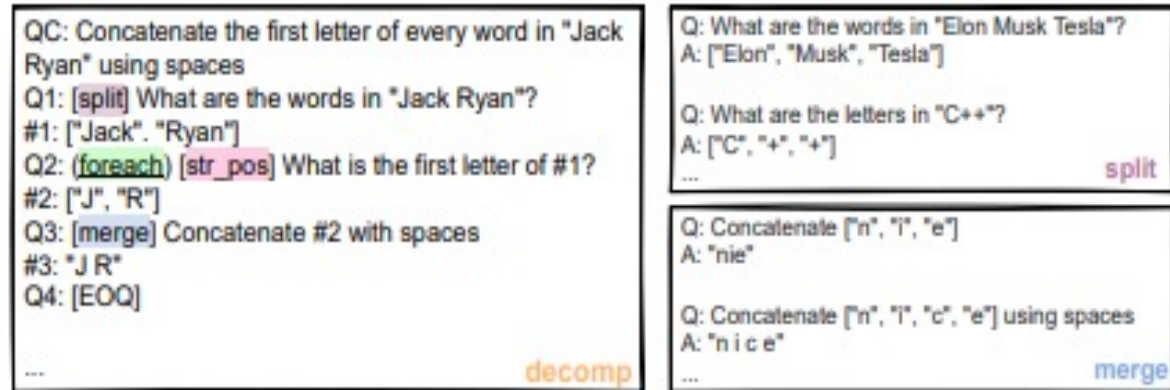


Figure 2: Prompts for the decomposer and the `split` and `merge` sub-tasks used by the decomposer. The decomposer specifies the sequence of questions and corresponding sub-tasks (within square braces). The sub-task prompts can be written independent of the complex task examples and can even capture generalizations, e.g., letters in word (`split`) and no delimiter (`merge`).

LLM is prompted by demonstrations comprised of sequential question-operation, which form “sub-questions”

Overview of Chain of Thought Works

- **Task Preprocessing:**

- **Selection-Inference (SI) :**

Selection-Inference (SI) is designed to tackle multi-step logical reasoning problems where all essential information is already present within the input context.

Logical Reasoning

User Prompt

If something is quiet and cold then it is smart.
 All young, cold things are kind.
 If something is kind then it is nice.
 All smart, blue things are kind.
 If something is cold then it is quiet.
 If Anne is smart and Anne is kind then Anne is cold.
 If something is nice and blue then it is young.
 If something is nice and young then it is cold.

Anne is young.
 Gary is nice.
 Charlie is quiet.
 Anne is blue.

[...]

Dave is smart.
 Anne is quiet.
 Dave is blue.
 Gary is cold.
 Anne is nice.

Does it imply that the statement "Dave is not quiet" is True?

LLM Answer

Selection

All smart, blue things are kind.
 We know that Dave is smart and Dave is blue.

Dave is kind.

If something is kind then it is nice.
 We know that Dave is kind.

Dave is nice.

If something is nice and blue then it is young.
 We know that Dave is nice and Dave is blue.

Dave is young.

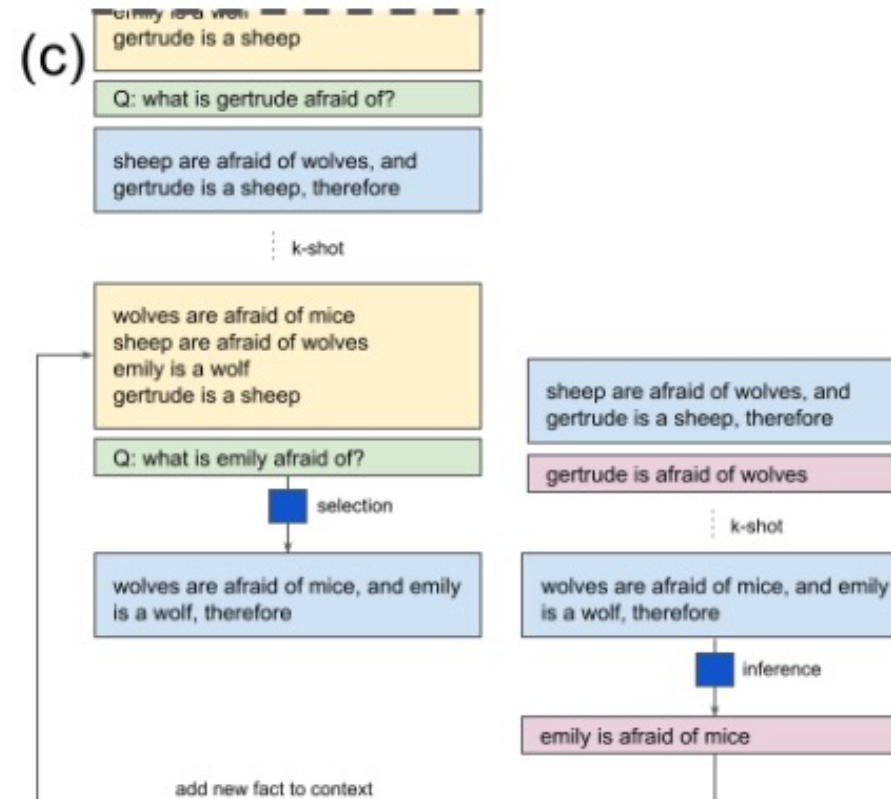
If something is nice and young then it is cold.
 We know that Dave is nice and Dave is young.

Dave is cold.

If something is cold then it is quiet.
 We know that Dave is cold.

Dave is quiet.

Inference



Overview of Chain of Thought Works

- **Iterative Refinement:**
Verification enables the reasoning frameworks to iteratively refine the generated context and intermediate results.
- **Tool Utilization:**
To better integrate multiple execution methods, more effective schemes opt to devise a plan that specifies tools for handling each sub-task, before executing the reasoning chain. Examples include AutoGPT , Toolformer , Chameleon , ChatCot , PVS and others .

Topologies of Reasoning: Demystifying Chains, Trees, and Graphs of Thoughts

Presenters

Jeffrey Chen (fyy2ws)

Reasoning With Trees

Tree Topology Variants

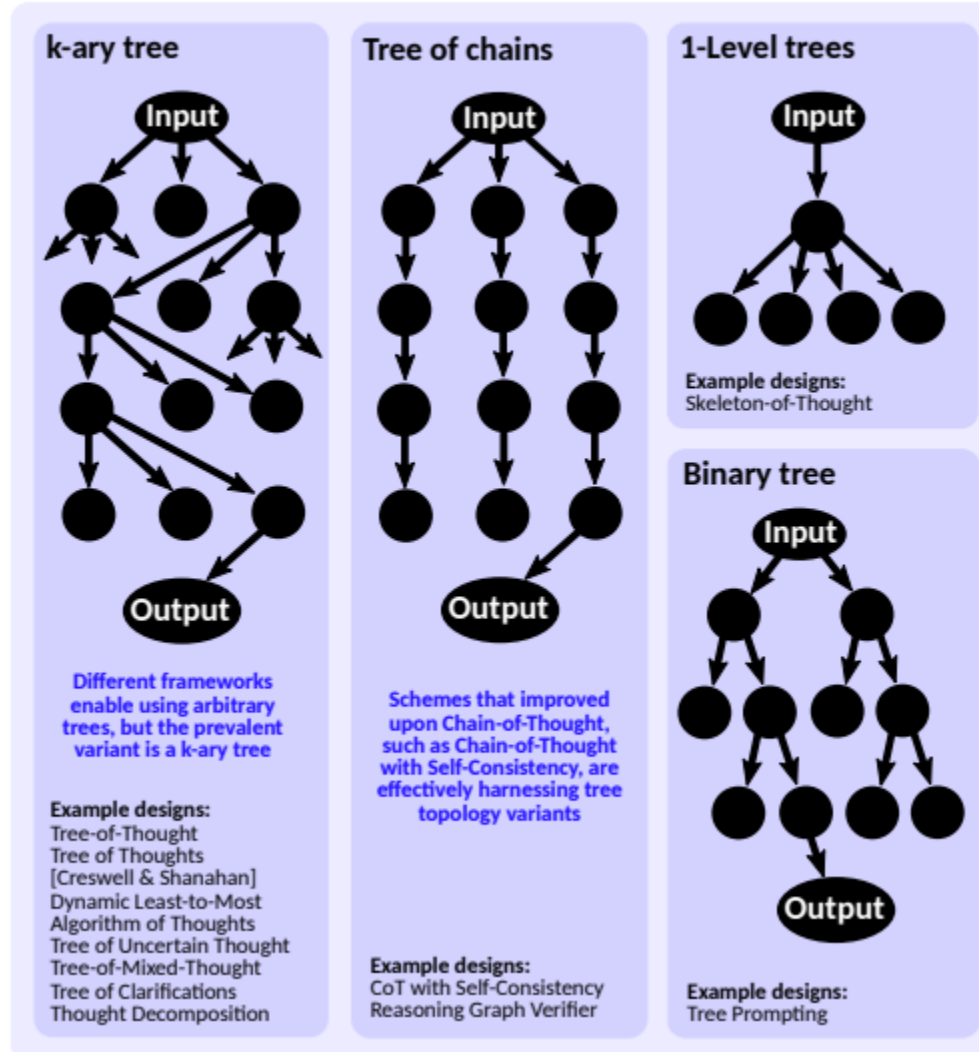


Figure : Variants of tree prompting topologies

Motivation for Reasoning With Trees

- Exploration
 - Generate multiple thoughts from a given thought
 - Sampling
 - Task decomposition
- Voting
 - Automatic selection of best outcome of generated outputs

K-Ary Tree Example

Game of 24

User Prompt

Use numbers and basic arithmetic operations (+ - * /) to obtain 24.
[in-context examples]

User:

14 8 8 2

Assistant:

LLM Answer

Trying a promising first operation:

1. $8 + 2: (14, 10, 8)$

- $14 + 10: (24, 8)$ 32, 16, 192, 3

- $14 - 10: (8, 4)$ 12, 8, 32, 2

- $14 * 10: (140, 8)$ 148, 132, 1120, fractional

- $14 / 10: (8, 1.4)$ fractional

- $14 + 8: (22, 10)$ 32, 12, 220, fractional

- $14 - 8: (10, 6)$ 16, 4, 60, fractional

- $14 * 8: (112, 10)$ 122, 102, 1120, fractional

- $14 / 8: (17.5, 1)$ fractional

- $10 + 8: (18, 14)$ 32, 4, 252, fractional

- $10 - 8: (14, 2)$ 16, 12, 28, 7

- $10 * 8: (80, 14)$ 94, 66, big, fractional

- **$10 / 8: (1.25, 1)$ fractional**

Trying another promising first operation:

2. $8 / 2: (14, 8, 4)$

- **$14 + 8: (22, 4)$ 26, 18, 88, fractional**

- **$14 - 8: (6, 4)$ 10, 2, $24 = 6 * 4 \rightarrow$ found it!**

Backtracking the solution:

Step 1:

$8 / 2 = 4$

Step 2:

$14 - 8 = 6$

Step 3:

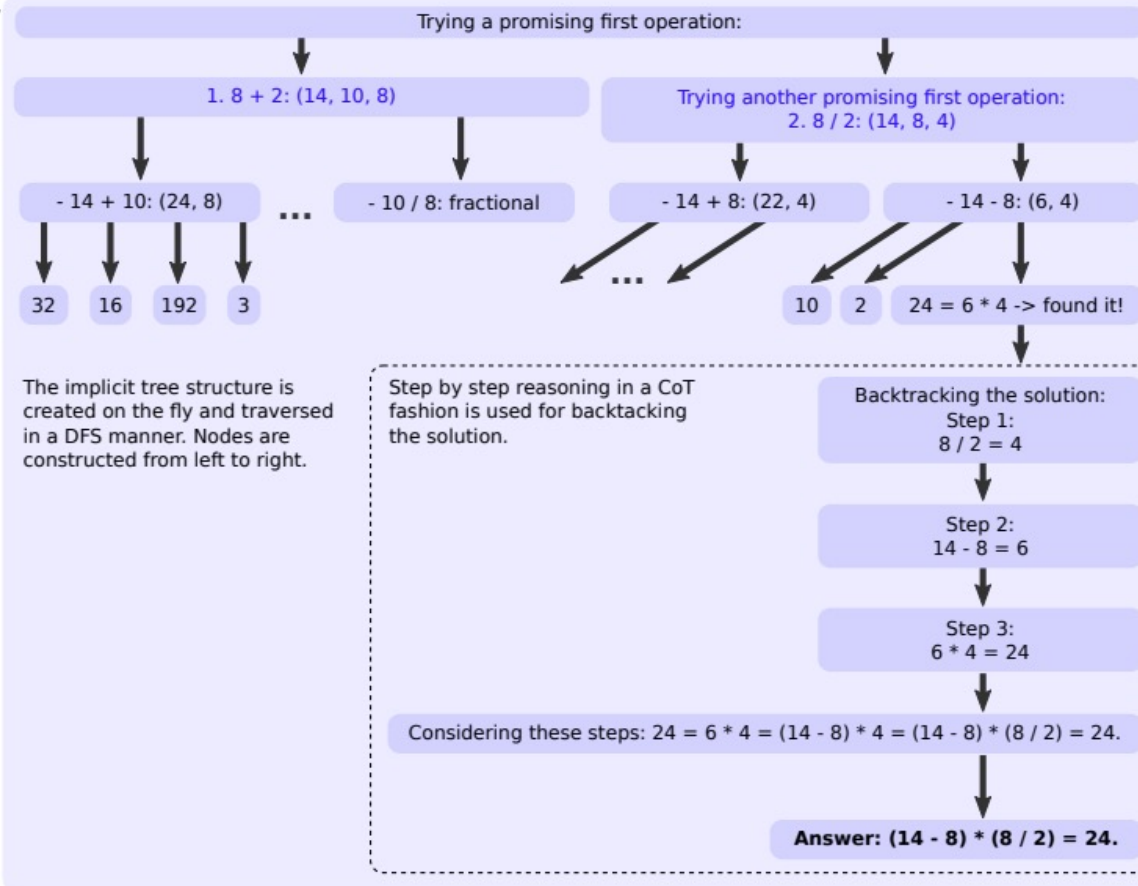
$6 * 4 = 24$

Considering these steps: $24 = 6 * 4 = (14 - 8) * 4$

$= (14 - 8) * (8 / 2) = 24.$

Answer: $(14 - 8) * (8 / 2) = 24.$

LLM Answer: Implicit Tree Structure



k-ary tree

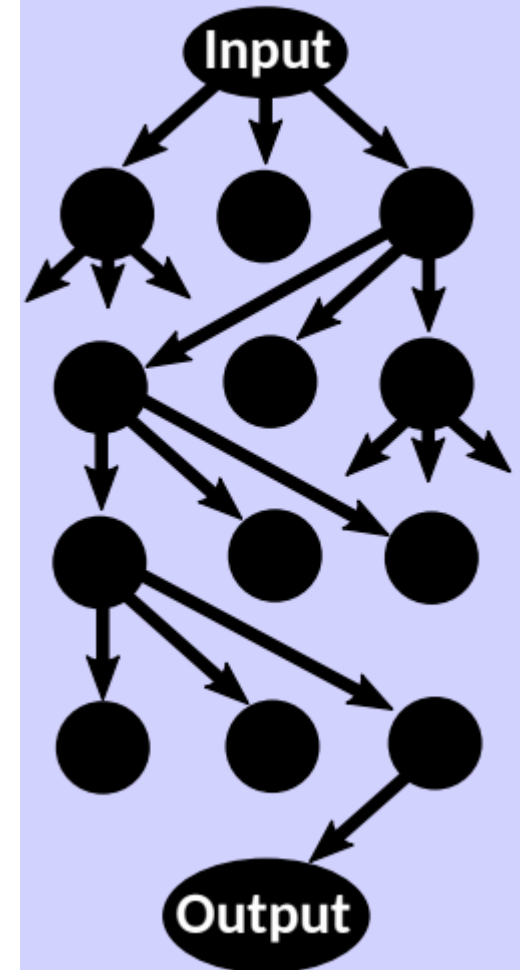


Fig. 10: GAME OF 24. An example implicit single-prompt tree topology, encoded with text. It demonstrates a Game of 24 DFS in-context example from AoT [166]. The left view shows the user prompt and the single textual answer from the LLM. The right view shows the implicit tree structure that is explored during the generation of the LLM answer. We mark text corresponding to implicit nodes as **bold**.

K-Ary Tree Example

Creative Writing

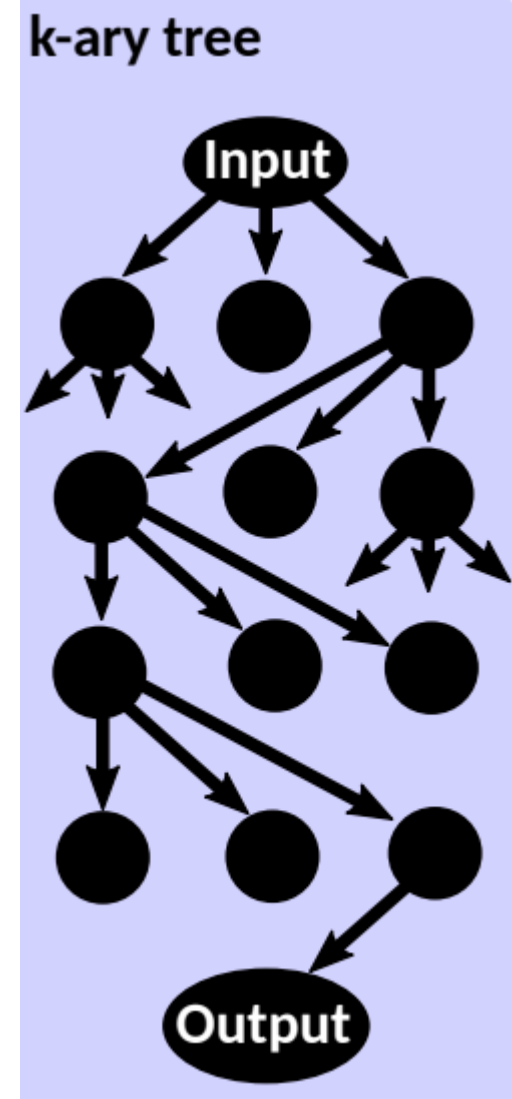
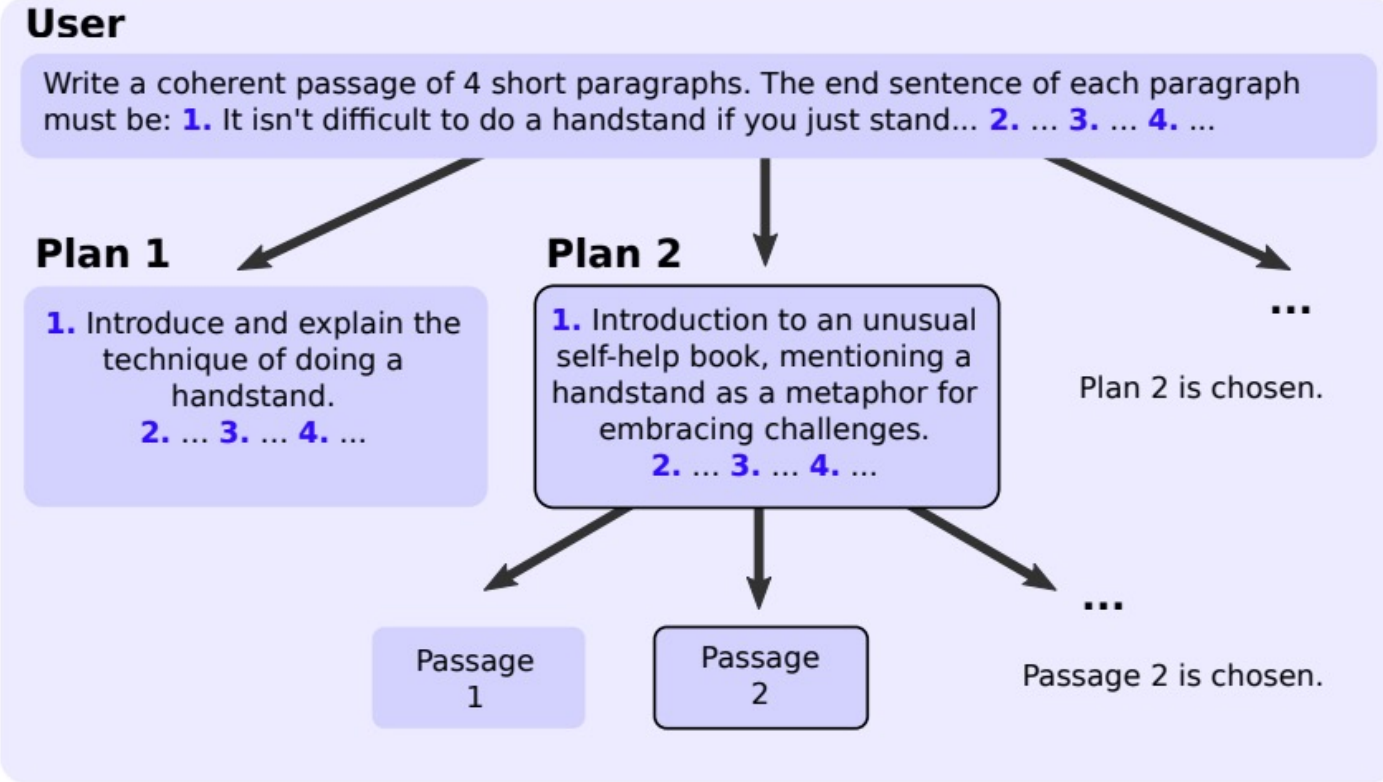


Fig. 11: CREATIVE WRITING. An example explicit multi-prompt tree topology, encoded with text, from the Tree of Thoughts (ToT) scheme [213] for creative writing. Given the task of writing a coherent passage of four paragraphs ending in given sentences, first multiple plans (nodes) are generated and then ranked. In a next step, the best plan is used to generate multiple possible passages as outputs. Finally, the best ranked passage is the output of the ToT reasoning.

Trees of Chains Example

Math Reasoning

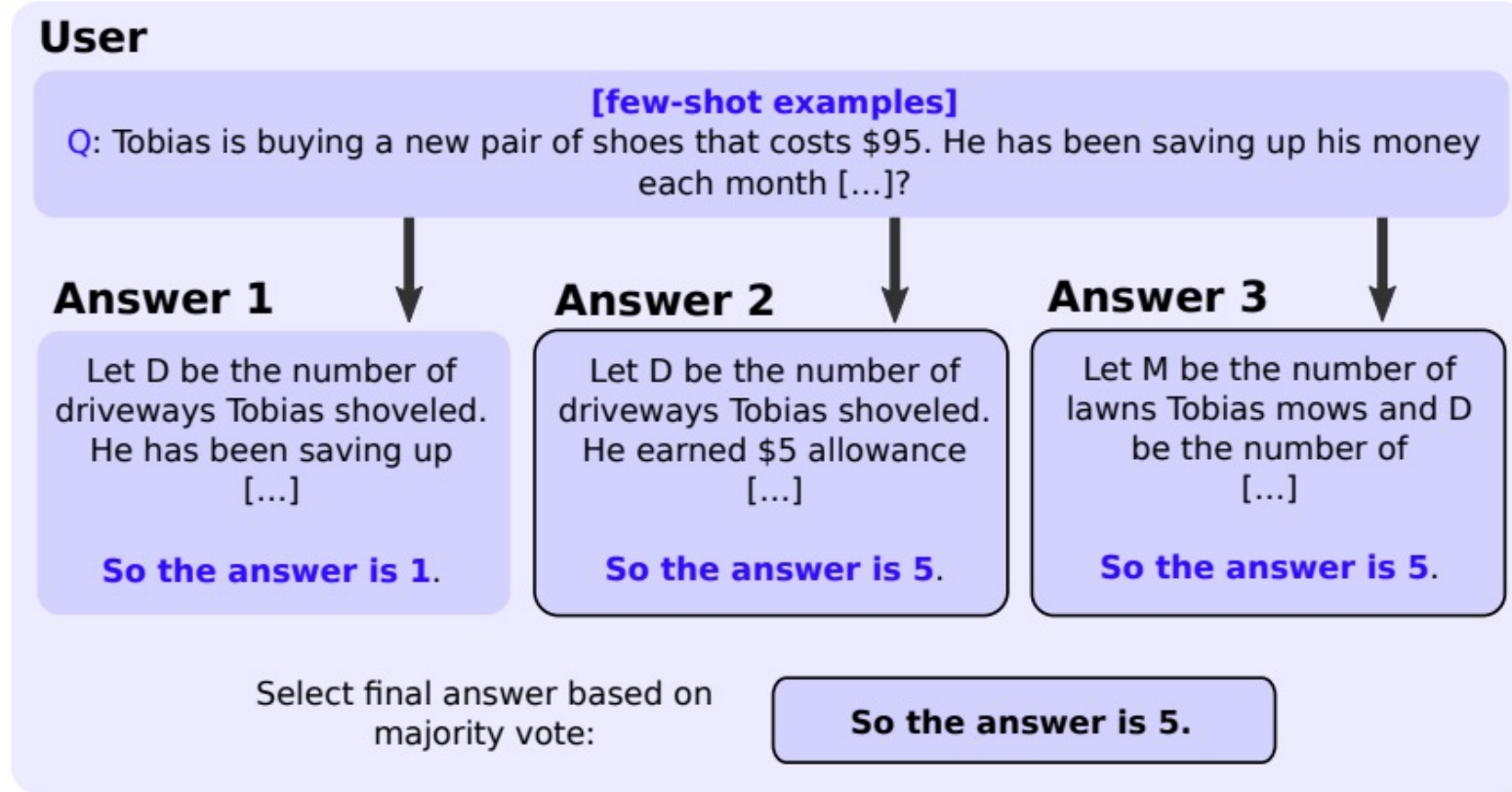
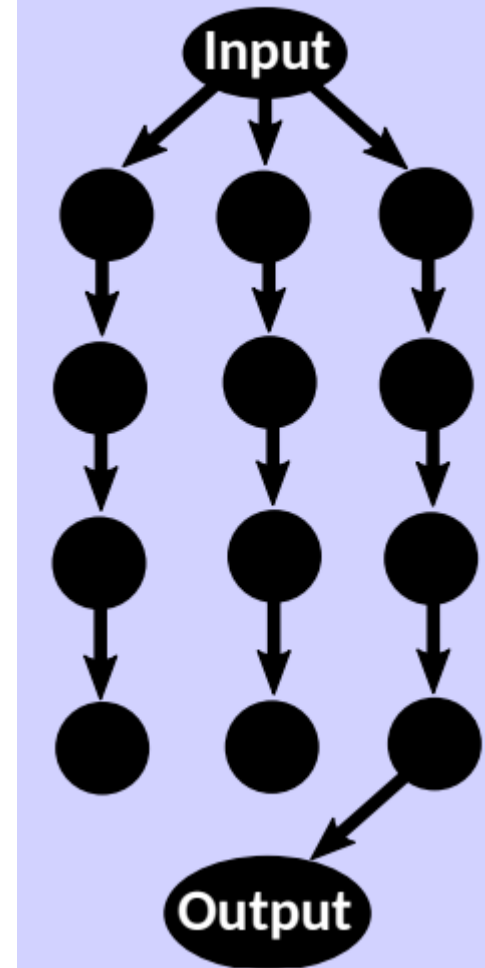


Fig. 12: MATH REASONING. An example using explicit multi-prompt tree topology, encoded with text. Given a math reasoning task, CoT-SC [190] is used to generate multiple answers and pick a final one based on majority vote. Each of the generated answers contains multiple CoT reasoning steps, depicted here in a single node.

Tree of chains



Single-Level Tree Example

Question Answering

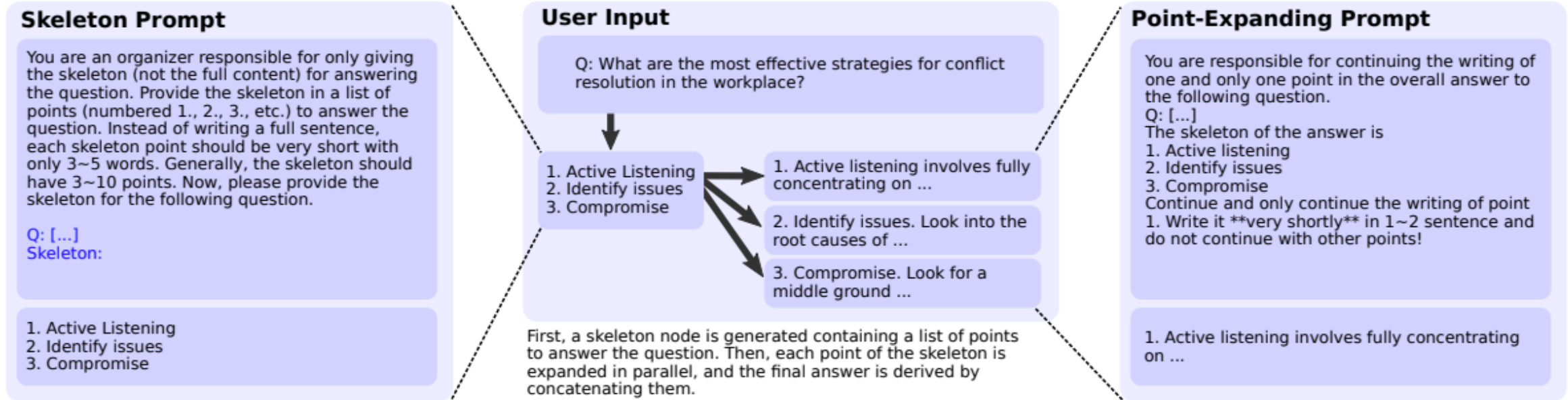
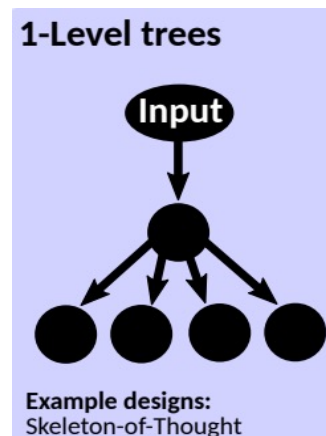


Fig. 13: An example explicit multi-prompt tree topology, encoded with text. It demonstrates the automatically derived tree topology of Skeleton-of-Thought (SoT) [148] where the individual points are expanded in parallel.



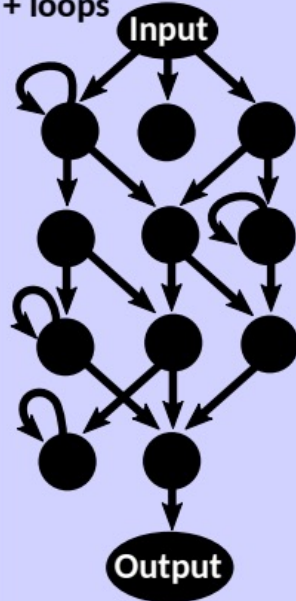
Tree Performance

- Increasing branching factor
 - Higher diversity of outcomes
 - Beneficial for accuracy
 - Increases computational cost
- Optimal branching factor is hard to find
 - Problem dependent
- More complicated problems can benefit more from decomposition into subproblems

Reasoning With Graphs

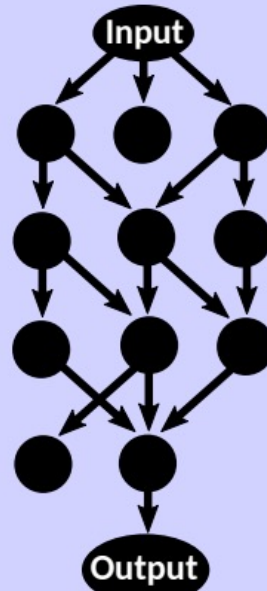
Graph Topology Variants

Directed graphs
+ loops



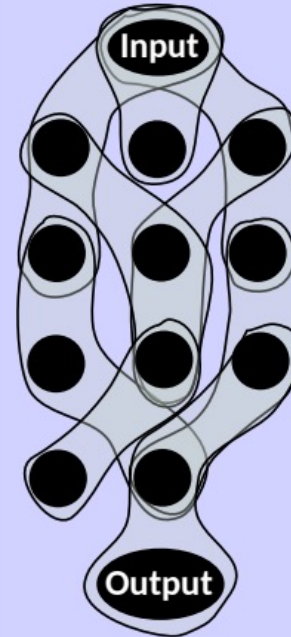
Example designs:
Graph of Thoughts

Directed graphs



Example designs:
Graph of Thoughts
Graph-of-Thought
ControlLLM
Cumulative Reasoning
Everything of Thoughts
ResPrompt

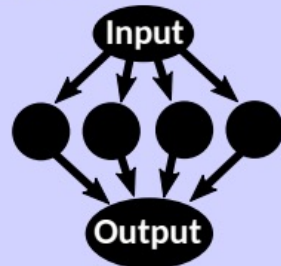
Hypergraphs



The only currently existing
Hypergraph-based scheme,
the Hypergraph-of-Thought,
uses a limited variant of
hypergraphs, in which
hyperedges span only
triples of vertices

Example design:
Hypergraph-of-Thought

Branch-Merge



Motivation for Reasoning With Graphs

- Aggregation
 - Being able to combine multiple thoughts into one
 - Synergy
 - Produce outcome better than individual parts
 - Effective composition of outcomes of tasks
- Exploration
- Flexible
 - Arbitrary

Cumulative Reasoning Example

Game of 24

Propose Prompt

Suppose you are one of the greatest AI scientists, logicians and mathematicians. You are very good at basic arithmetic operations. Use numbers and basic arithmetic operations (+ - * /) to obtain 24 with input numbers. **In each step, You are only allowed to randomly choose arbitrary TWO of the input numbers to obtain a new number using arbitrary one basic arithmetic operation (AVOID duplicating with forbidden steps). Your calculation process must be correct.** [in-context examples]

Input: 14, 8, 8, 2
Forbidden Steps : [...]
Next Step:

14 - 2 = 12
Remaining Numbers:
12, 8, 8

Verifier Prompts

Validate Step

Suppose you are one of the greatest AI scientists, logicians and mathematicians. You are very good at basic arithmetic operations. Use numbers and basic arithmetic operations (+ - * /) to obtain 24 with input numbers. **Evaluate if given intermediate step is correct and only use two existing numbers.** [in-context examples]

Input: 14, 8, 8, 2
Intermediate step: 14 - 2 = 12

The intermediate step is valid.
Judgment:
Valid

Verify Step

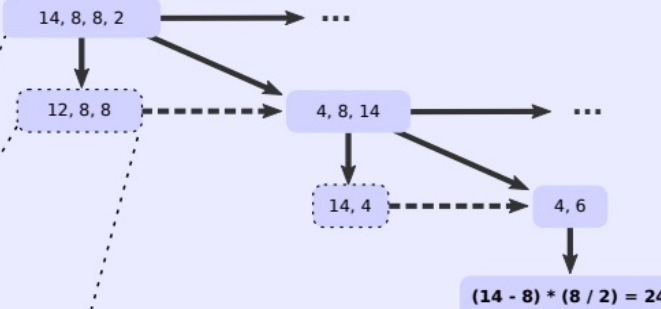
Suppose you are one of the greatest AI scientists, logicians and mathematicians. You are very good at basic arithmetic operations. Use numbers and basic arithmetic operations (+ - * /) to obtain 24 with input numbers. **Evaluate if given numbers can reach 24 (sure/likely/impossible)** [in-context examples]

Input: 12, 8, 8
Draft:

(12 - 8) * 8 = 32
12 + 8 + 8 = 28
8 / 8 + 12 = 13
I cannot obtain 24 now, and numbers are not within a reasonable range
Output:
impossible

(12, 8, 8) is added to forbidden steps.

User Input



Reporter Prompt

Suppose you are one of the greatest AI scientists, logicians and mathematicians. You are very good at basic arithmetic operations. Use numbers and basic arithmetic operations (+ - * /) to obtain 24 with input numbers. **You need to combine the given intermediate steps step by step into a complete expression.** [in-context examples]

Input: 14, 8, 8, 2
Intermediate steps:
8 / 2 = 4 (left 4, 8, 14)
14 - 8 = 6 (left 4, 6)
6 * 4 = 24
Draft:

Because 6 * 4 = 24, while 6 = 14 - 8. So (14 - 8) * 4 = 24.
Because (14 - 8) * 4 = 24, while 4 = 8 / 2. So (14 - 8) * (8 / 2) = 24.
Output:
(14 - 8) * (8 / 2) = 24

Starting with the user input node, one valid node is selected at random in each iteration. Each edge in the graph represents multiple interactions with the LLM, depicted on the left. First a next step is proposed. Then this step is validated and verified, judging if this can lead to the desired solution or if it is a dead end (dashed nodes). Opposed to ToT, these invalid steps are kept track of and are used in the next propose prompt as "forbidden steps", so directly influencing the next node (dashed arrows). If the validation output is "sure", the Reporter is invoked to summarise the steps and derive the final result as shown in the reporter prompt.

Directed graphs

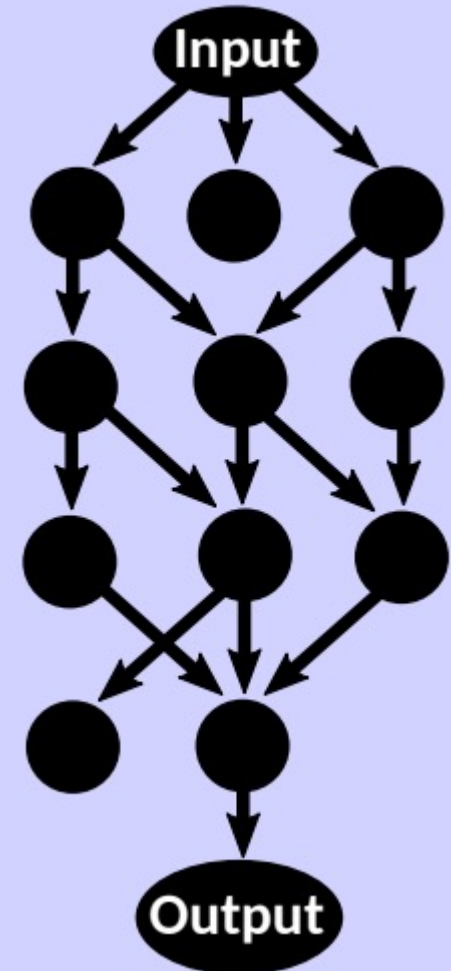


Fig. 14: GAME OF 24. An example explicit multi-prompt graph topology, encoded with text using Cumulative Reasoning [224].

ResPrompt Example

Math Reasoning

User Prompt

[in-context examples]

Q: Tobias is buying a new pair of shoes that costs \$95. He has been saving up his money each month [...].

LLM Answer

1. He has \$15 left after buying the shoes that cost \$95, which means he had $\$95 + \$15 = \$110$ in total before the purchase.
2. Next, we calculate the amount he earned from his allowance. He has been saving for three months, getting \$5 each month, [...]
3. Then we know he mowed 4 lawns and he charges \$15 per lawn, so from mowing lawns, he earned $4 * \$15 = \60 .
4. Let us sum up the money he earned from his allowance and from mowing lawns: $\$15$ (allowance) + $\$60$ (mowing) = $\$75$.
5. Now, let us subtract the total amount of money earned from allowance and mowing from his total amount of money before buying the shoes to [...].
6. As he charges \$7 to shovel a driveway, we divide the total earned from shoveling by the charge per driveway to find out the number of driveways he shoveled: $\$35 / \$7 = 5$ driveways.

LLM Answer: Implicit Tree Structure

1. He has \$15 left after buying the shoes that cost \$95, which means he had $\$95 + \$15 = \$110$ **in total before the purchase.**
2. Next, we calculate the amount he **earned from his allowance.** He has been saving for three months, getting \$5 each month, [...]
3. Then we know he mowed 4 lawns and he charges \$15 per lawn, so from mowing lawns, he earned $4 * \$15 = \60 .
4. Let us sum up the money he **earned from his allowance** and from mowing lawns: $\$15$ (allowance) + $\$60$ (mowing) = $\$75$
5. Now, let us subtract the total amount of money earned from allowance and mowing from his **total amount of money before buying the shoes** to [...].
6. As he charges \$7 to shovel a driveway, we divide the total earned from shoveling by the charge per driveway to find out the number of driveways he shoveled: $\$35 / \$7 = 5$ driveways.

Directed graphs

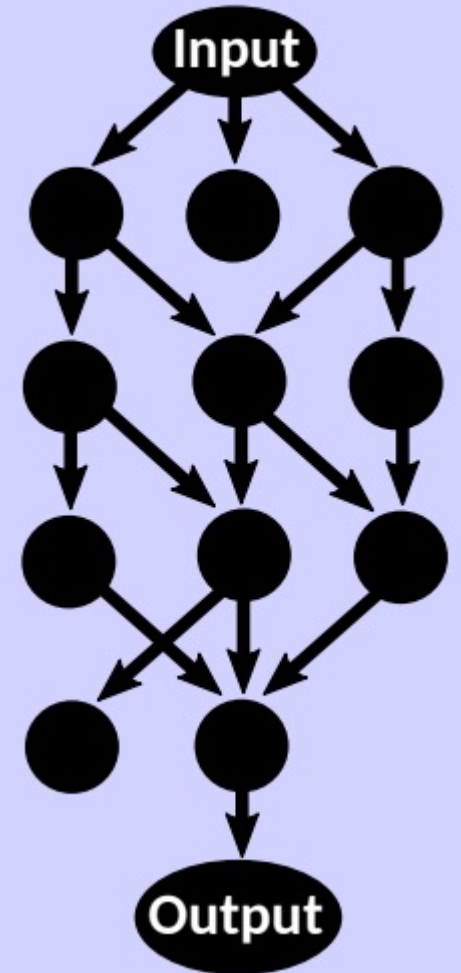
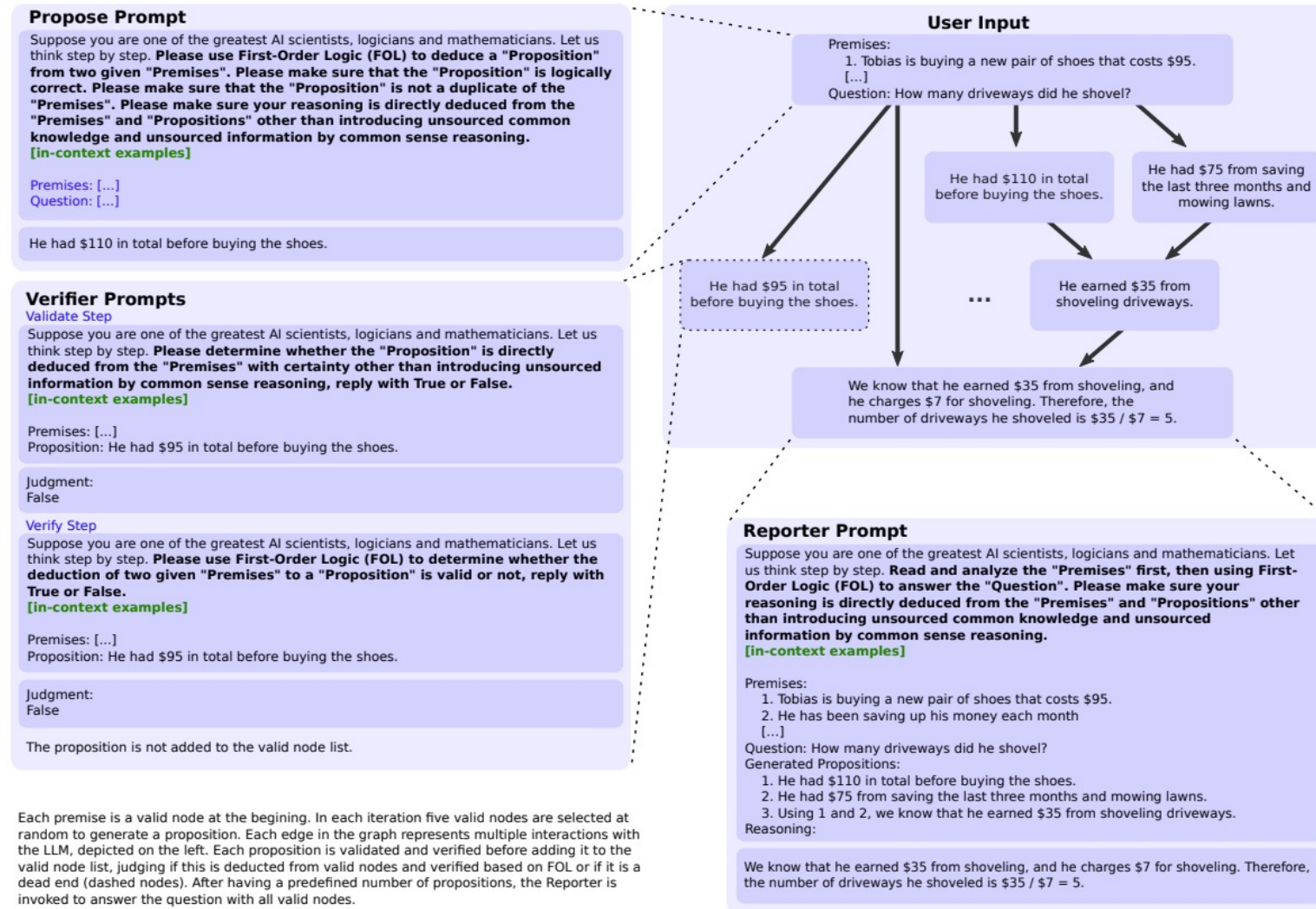


Fig. 15: MATH REASONING. An example implicit single-prompt graph topology, encoded with text. It shows an in-context example of a math question from ResPrompt [99], representing a linear sequence of six connected nodes with two implicit edges of the graph topology, marked with two different colors (red and blue), together with their corresponding nodes.

Cumulative Reasoning Example

Math Reasoning



Each premise is a valid node at the beginning. In each iteration five valid nodes are selected at random to generate a proposition. Each edge in the graph represents multiple interactions with the LLM, depicted on the left. Each proposition is validated and verified before adding it to the valid node list, judging if this is deduced from valid nodes and verified based on FOL or if it is a dead end (dashed nodes). After having a predefined number of propositions, the Reporter is invoked to answer the question with all valid nodes.

Directed graphs

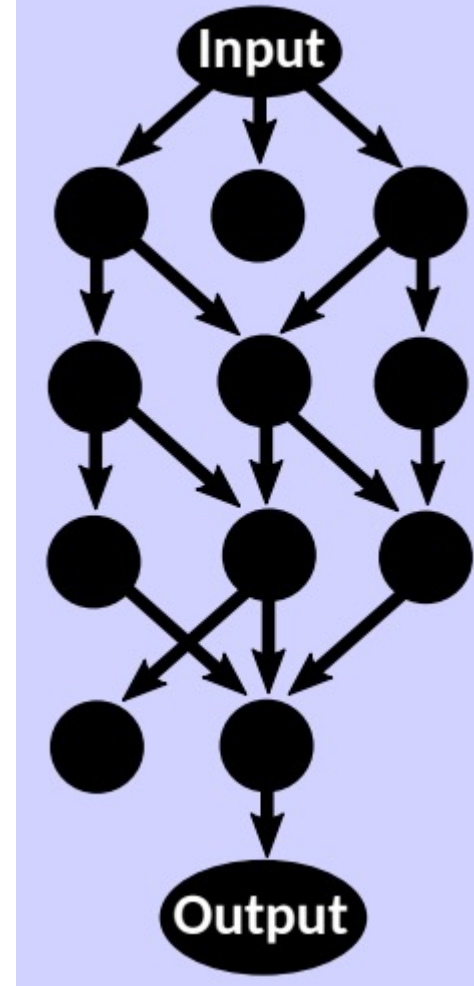


Fig. 16: MATH REASONING. An example implicit single-prompt graph topology, encoded with text, for the same question as in Figure 15, but using Cumulative Reasoning [224]. It illustrates three intermediate proposition nodes with direct connections from the first two nodes to the third one. The reasoning part indicates the final solution node, with direct connections from the third proposition and input nodes.

ControlLLM Example

JSON

User Task

Please dub the video.mp4 with a suitable background music and then generate a HTML web page to introduce this video.

LLM Task Decomposition Answer

```
[{
  "description": "Dub the video with a background music",
  "task": [ "video-editing" ],
  "id": 1,
  "args": {
    "type": "video",
    "value": "video.mp4" },
  "returns": {
    "type": "video",
    "value": "<GEN>-1" }
},
```



```
{
  "description": "Generate a HTML page to introduce the video",
  "task": [ "HTML-generation" ],
  "id": 2,
  "dep": 1,
  "args": {
    "type": "video",
    "value": "<GEN>-1" },
  "returns": {
    "type": "HTML",
    "value": "<GEN>-2" }
}
```

Directed graphs

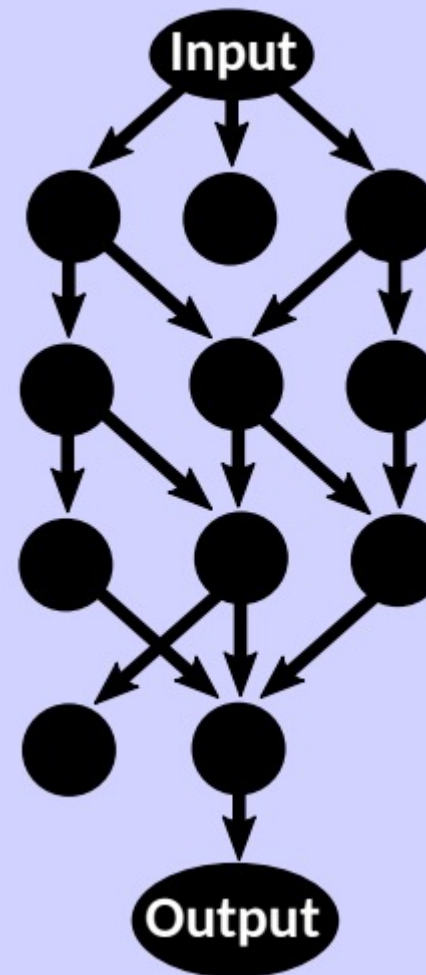


Fig. 17: JSON. An example explicit single-prompt graph topology, encoded with JSON, based on the ControlLLM scheme [132] for task decomposition. It shows two nodes describing decomposed subtasks for solving a given task. The "dep" field refers to dependent tasks, showing there is a direct edge from the first node (task 1) to the second.

Branch-Solve-Merge Example

Creative Writing

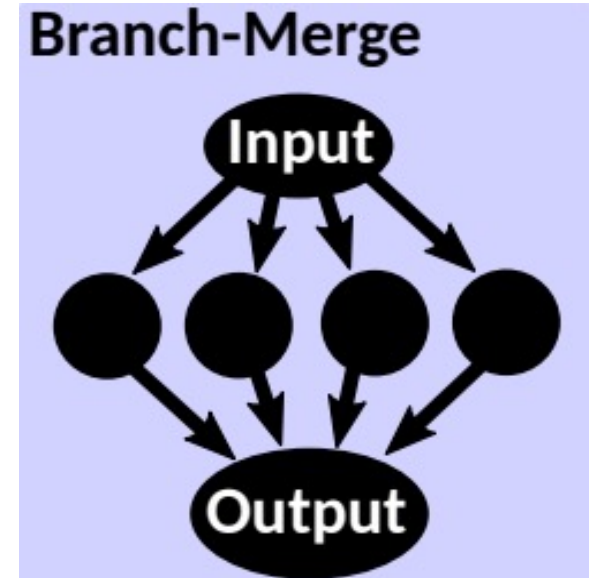
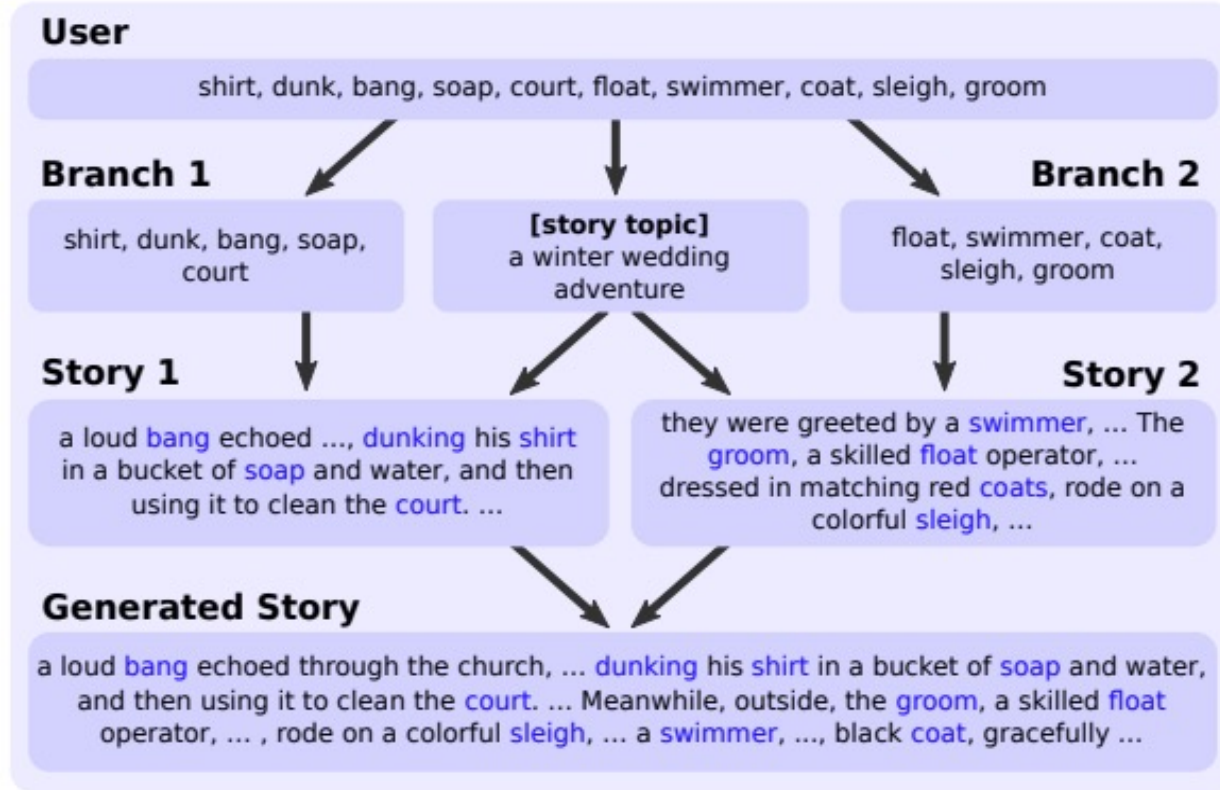


Fig. 18: CREATIVE WRITING. An example multi-prompt graph topology, encoded with text, from the Branch-Solve-Merge scheme [162] for story generation with branch, solve, and merge prompts. Given a list of concepts as input, the branch module generates three child nodes: two groups of concepts and one topic node. The solve module then creates two story nodes based on each group of concepts and the topic. Finally, these two story nodes are merged into the final solution node.

Chains vs. Trees vs. Graphs of Thoughts

- Chains
 - Explicit intermediate LLM thoughts
 - Step-by-step
 - Usually most cost effective
- Trees
 - Possibility of exploring at each step
 - More effective than chains
- Graphs
 - Most complex structure
 - Enable aggregation of various reasoning steps into one solution
 - Often see improvements in performance compared to chains and trees

Future Directions

- Exploring new topology cases
- Automatic derivation of tree/graph topologies
- Advancement in single-prompt schemes
- Investigate new scheduling approaches

Thank You

Soneya Binta Hossain (sh7hv)

Jessie Chen (hc4vb)

Ali Zafar Sadiq (mzw2cu)

Jeffrey Chen (fyy2ws)

Minjae Kwon (hbt9su)

REFERENCES

Presentation 2

<https://arxiv.org/abs/2307.15337>

Presentation 3

<https://arxiv.org/abs/2401.14295>

Presentation 1

<https://arxiv.org/abs/2310.14735>

<https://arxiv.org/pdf/2302.14045.pdf>

<https://arxiv.org/pdf/2109.01247.pdf>

<https://arxiv.org/pdf/2210.01240.pdf>

<https://arxiv.org/pdf/2204.02311.pdf>

<https://arxiv.org/abs/2302.12822>

<https://arxiv.org/pdf/2110.08387.pdf>

<https://arxiv.org/pdf/2205.10625.pdf>

<https://arxiv.org/pdf/2305.10601.pdf>

<https://arxiv.org/pdf/2308.09687.pdf>

<https://arxiv.org/pdf/2309.11495.pdf>

<https://community.openai.com/t/cheat-sheet-mastering-temperature-and-top-p-in-chatgpt-api/172683>

single-prompt zero-shot
CoT