ChatCoT and ToT

韩子坚

华中师范大学计算机学院

2024 年 12 月 7 日

Content

Tool

利用外部工具的方法可以大致分为以下两类：

1. **模型参数微调**
   这一类方法（如 Gao et al., 2023；Parisi et al., 2022；Qiao et al., 2023）通过训练模型参数来支持外部工具的集成与使用。这通常需要收集或生成工具增强示例以用于模型参数微调（如 Schick et al., 2023；Patil et al., 2023；Hao et al., 2023）。

Tool

利用外部工具的方法可以大致分为以下两类：

1. **模型参数微调**
   这一类方法（如 Gao et al., 2023；Parisi et al., 2022；Qiao et al., 2023）通过训练模型参数来支持外部工具的集成与使用。这通常需要收集或生成工具增强示例以用于模型参数微调（如 Schick et al., 2023；Patil et al., 2023；Hao et al., 2023）。
2. **基于提示的工具指导**
   第二类方法（如 Gao et al., 2022；Yao et al., 2022；Zhang et al., 2023）主要依靠设计提示（prompts）来引导大型语言模型（LLMs）使用外部工具。这些方法重点在于设计合适的提示或工具操作策略，以便在需要时选择和使用工具（如 Liang et al., 2023；Shen et al., 2023；Yao et al., 2022）。

文章采用了第二种方法。

链式推理（Chain-of-Thought, CoT）

链式推理（CoT）提示策略（Wei et al., 2022; Kojima et al., 2022）旨在通过引导大型语言模型（LLMs）生成中间推理步骤来增强其推理能力。这种方法通过两种主要机制实现：

1. **特殊指令**：例如"Let us think step by step"，直接引导模型输出分步骤的推理过程。
2. **上下文示例**：提供详细的、包含中间推理步骤的示例，帮助模型从类似问题中学习逐步推理的方法。

通过上述策略，LLMs 能够依次进行每个推理步骤，直到得出最终答案，显著提升整体性能。

基于 CoT 的改进研究

近年来，基于 CoT 的方法被进一步优化，以改进其推理性能，主要包括以下几个方向：

- **问题分解**：将复杂问题拆解为多个子问题，逐步解决后组合答案（Zhou et al., 2022; Dua et al., 2022）。
- **示例选择**：选择更恰当、更具代表性的上下文示例以指导推理（Ye et al., 2022; Shi et al., 2023）。
- **结果后处理**：在生成答案后通过后处理减少错误或调整输出（Wang et al., 2022; Madaan et al., 2023; Zheng et al., 2023）。
- **推理格式变更**：尝试修改推理格式或流程以提升模型理解和输出的准确性（Yao et al., 2023; Wu et al., 2023）。

现存问题与改进方向

尽管链式推理方法取得了显著进展，但其生成过程通常为**一次性完成（one-pass generation）**，这意味着模型从问题到答案的推理流程是自上而下的连续生成。若在中间步骤需要使用外部工具（如计算器、检索系统等），可能会中断推理的流畅性，影响生成效果。

为了解决这一问题，作者提出了一种**将链式推理与工具操作无缝结合的统一方法**。利用大型语言模型在多轮对话中的强大能力，设计了基于多轮对话的链式推理策略，能够在不同步骤灵活调用工具的同时，保持推理过程的连贯性，从而更好地解决复杂问题。

工具操作 (Tool Manipulation)

大型语言模型（LLMs）在处理算术计算时比较困难，而这些问题可以通过使用特定的外部工具（如计算器）解决。这些外部工具被表示为 $\{T_1, \ldots, T_n\}$。为操作工具，现有方法主要依赖编写详细的 prompt，描述如何让 LLM 使用这些工具。具体流程为：

1. 编写提示以指导 LLM 选择有用的工具；
2. 生成工具的参数输入；
3. 调用工具的 API 获取结果。

工具操作 (Tool Manipulation)

作者延续了这一思路，主要使用三种工具：

1. **计算器（Calculator）**：给定数学表达式，计算器可以根据算术规则（如合并同类项、化简分数）计算表达式的值或进行化简。
2. **方程求解器（Equation Solver）**：给定方程组和未知变量，方程求解器可以利用相关算法计算未知变量的值。
3. **检索器（Retriever）**：给定查询，检索器旨在从候选集中提取最相关的信息（如文档）。根据检索语料的类型，可以通过专门的模型（如密集检索模型）实现。

实现方法

- **计算器与方程求解器**：使用 Python 符号计算库 SymPy（Meurer et al., 2017）中的不同函数实现 Calculator 和 Equation Solver 工具。
- **检索器**：采用 SimCSE（Gao et al., 2021）句子嵌入模型来计算文本语义相似性，利用该方法检索出语义上最相似的前 k 个示例，并将这些示例的问题描述 Q 和解决方案 S 拼接起来，作为输入提示。同时将期望的回答一并输入到大型语言模型中。

当输入的表达式或方程格式不正确（ill-formed）或无法求解时，以上工具都会返回错误结果。

现有的两种工具增强方法的缺点

这两种工具增强方法各自存在一些潜在的问题:

依赖大型语言模型 (LLM) 预先安排工具使用计划并在随后执行的方法（Zhou et al., 2022; Jiang et al., 2023b）无法在生成计划后与工具交互，即使发现明显的错误也无法修正。

现有的两种工具增强方法的缺点

这两种工具增强方法各自存在一些潜在的问题:

依赖大型语言模型 (LLM) 预先安排工具使用计划并在随后执行的方法（Zhou et al., 2022; Jiang et al., 2023b）无法在生成计划后与工具交互，即使发现明显的错误也无法修正。

这种方法将工具的使用视为一个独立于推理过程的步骤，LLM 首先要规划好所有需要使用的工具以及它们的调用顺序，然后才开始执行计划。这种方式的缺点在于，LLM 在规划阶段无法预知推理过程中可能出现的各种情况，因此计划很容易出现错误。并且，由于计划和执行是分离的，LLM 在执行过程中即使发现计划有误也无法进行调整，只能继续执行错误的计划，最终导致推理结果出错。

现有的两种工具增强方法的缺点

需要针对特定任务设计形式化动作的方法（Dua et al., 2022; Khattab et al., 2022; Jiang et al., 2023a）必须频繁地在 LLM 推理和执行动作之间切换，这会损害 CoT 推理过程的连续性。

现有的两种工具增强方法的缺点

需要针对特定任务设计形式化动作的方法（Dua et al., 2022; Khattab et al., 2022; Jiang et al., 2023a）必须频繁地在 LLM 推理和执行动作之间切换，这会损害 CoT 推理过程的连续性。

这种方法将工具的使用融入到推理过程中，LLM 在推理的每一步都需要决定是否使用工具，以及使用哪个工具。这种方式的缺点在于，LLM 需要频繁地在推理和工具使用之间切换，这会打断推理的思路，降低推理的效率。并且，由于每个工具都需要设计特定的动作，这种方法的可扩展性较差，难以应用于新的工具或新的任务。

现有的两种工具增强方法的缺点

需要针对特定任务设计形式化动作的方法（Dua et al., 2022; Khattab et al., 2022; Jiang et al., 2023a）必须频繁地在 LLM 推理和执行动作之间切换，这会损害 CoT 推理过程的连续性。

这种方法将工具的使用融入到推理过程中，LLM 在推理的每一步都需要决定是否使用工具，以及使用哪个工具。这种方式的缺点在于，LLM 需要频繁地在推理和工具使用之间切换，这会打断推理的思路，降低推理的效率。并且，由于每个工具都需要设计特定的动作，这种方法的可扩展性较差，难以应用于新的工具或新的任务。

总的来说，这两种方法都试图将外部工具与 LLM 结合起来以解决复杂推理任务，但在如何协调工具使用和推理过程方面存在不足。前者缺乏灵活性，后者缺乏效率。

ChatCoT 框架的工作原理

ChatCoT 框架的工作原理：

1. **在对话的早期阶段提供背景知识**。例如工具描述、相关任务示例和聊天中分解思维链的演示。例如"[*T*] can help you [*Y*]", 其中 [*T*] 是工具名称，[*Y*] 显示其详细功能。然后把这些描述作为输入提示。

ChatCoT 框架的工作原理

ChatCoT 框架的工作原理：

1. **在对话的早期阶段提供背景知识**。例如工具描述、相关任务示例和聊天中分解思维链的演示。例如"[*T*] can help you [*Y*]", 其中 [*T*] 是工具名称，[*Y*] 显示其详细功能。然后把这些描述作为输入提示。
2. **将思维链推理过程分解为多轮对话**。从训练集中随机抽取五个问题，并手动标注这些问题的完整多轮对话作为示例。然后将所有示例的对话逐轮输入到对话的 prompt 中，作为**上下文**来引导 LLM 遵循该格式进行推理。

ChatCoT 框架的工作原理

ChatCoT 框架的工作原理：

1. **在对话的早期阶段提供背景知识**。例如工具描述、相关任务示例和聊天中分解思维链的演示。例如"[*T*] can help you [*Y*]", 其中 [*T*] 是工具名称，[*Y*] 显示其详细功能。然后把这些描述作为输入提示。
2. **将思维链推理过程分解为多轮对话**。从训练集中随机抽取五个问题，并手动标注这些问题的完整多轮对话作为示例。然后将所有示例的对话逐轮输入到对话的 prompt 中，作为**上下文**来引导 LLM 遵循该格式进行推理。
3. **使用工具增强推理步骤，以逐步执行推理，直到获得最终答案**。在每个迭代中，大型语言模型执行推理，选择适当的工具，并执行所选工具以获得当前步骤的中间结果。

ChatCoT 框架的工作原理

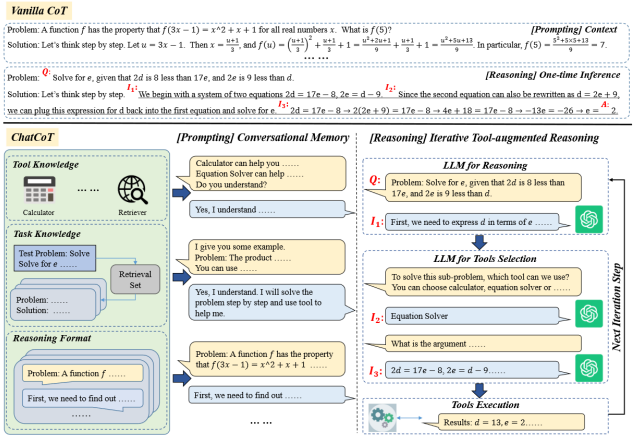


图 1: ChatCoT 架构示意图

举例

Tool Knowledge

The two-turn utterances are:

User:
"You can use tool to help you solve the problem and I give you the instruction of tools usage. T_1 can help you Y_1 $\cdots$ Do you understand?"

LLM:
"Yes, I understand. I will use tool to help me solve the problem."

举例

Retrieval-Augmented Task Knowledge

The two-turn utterances are:

User:
"I give you some examples. Problem: Q_1 Solution: S_1 $\cdots$ You can use the knowledge and theory in these problems. Do you understand?"

LLM:
"Yes, I understand. I will solve the problem step by step and use tool to help me."

举例

Multi-turn Reasoning Format

The multi-turn utterances are based on the following pattern:

User:
"Problem: Q'_1 Let's think step by step and use knowledge in similar problems to solve this problem."

LLM:
"I_1" ... "I_n"

LLMs 推理（LLM for Reasoning）

LLMs 借助上下文中的示例直接用自然语言进行推理，直到需要工具的功能来推动解决任务。

LLMs 工具选择（LLM for Tools Selection）

在完成推理之后，利用 LLM 来选择合适的工具（如计算器）以提供所需的功能。输入给 LLM 的提示是：

”解决这个子问题，我们可以使用哪个工具？”

模型根据该提示判断是否需要使用工具：

- 如果需要使用工具，模型会选择一个合适的工具，并进一步生成工具的输入参数，例如数学表达式。
- 如果不需要工具，模型直接回答 “Do not use tool”，并继续进行推理。

工具执行（Tools Execution）

基于 LLM 所选择的工具和生成的输入参数，使用该工具并以参数执行计算或任务，获取当前迭代结果。再由 LLM 判断结果是否符合预期，如果不符合预期可以重复执行工具。

问题：由 LLM 判断结果是否符合预期？这种验证是否缺乏说服力和可信度？

实验结果分析

1. 从表中可以看到，检索增强方法（如 ChatCoT 和 CoT w/ Retri）优于其他方法。可能是因为检索到的示例可能包含更多相关的知识和推理步骤，ChatGPT 对几何领域的知识和符号相对较为陌生。如果没有类似的示例，大型语言模型很难准确理解这些内容。

实验结果分析

1. 从表中可以看到，检索增强方法（如 ChatCoT 和 CoT w/ Retri）优于其他方法。可能是因为检索到的示例可能包含更多相关的知识和推理步骤，ChatGPT 对几何领域的知识和符号相对较为陌生。如果没有类似的示例，大型语言模型很难准确理解这些内容。
2. 可以发现，直接在推理过程中使用外部工具甚至可能产生负面影响。原因可能在于，将工具使用融入 CoT 推理过程会影响推理的连贯性。

实验结果图表

Models	Prompt Strategy	MATH							
		Algebra	CP	PC	PA	Geometry	IA	NT	Avg.
GPT-3	CoT	6.0	4.7	4.0	7.7	3.1	4.4	4.4	5.2
PaLM	CoT	9.7	8.4	4.4	19.2	7.3	3.5	6.0	8.8
LLaMA	CoT	-	-	-	-	-	-	-	10.6
Galactica	CoT	29.0	13.9	12.8	27.2	12.3	9.6	11.7	20.4
Minerva	CoT	51.3	28.0	18.0	55.0	26.8	13.7	21.2	33.6
PaLM 2	CoT	-	-	-	-	-	-	-	34.3
ChatGPT	CoT	48.1	31.4	21.1	56.6	22.3	18.3	29.1	35.1
	CoT w/ Tool	35.9	22.6	9.3	40.5	13.6	9.4	19.4	23.8
	CoT w/ Retri	52.7	32.7	18.9	58.4	29.2	**19.9**	31.7	37.7
	LP	49.6	30.2	16.3	52.3	22.5	16.9	29.8	34.0
	PHP	51.1	33.7	16.1	57.7	25.4	17.1	**35.1**	36.5
	ChatCoT	**56.1**	**34.2**	**23.8**	**59.2**	**29.9**	19.5	32.6	**39.4**

图 2: math 结果

实验结果图表

Methods	HotpotQA
CoT	38.0
CoT w/ Tool	31.4
ChatCoT w/o Feedback	53.8
ChatCoT	**59.2**

图 3: HotpotQA 结果

Prompt 设计

```
use_tool = {
    "role": "user",
    "content": "To solve this sub-problem, which tool can we use? You can choose
calculator, equation solver or do not use tool"
}
continue_reasoning = {
    "role": "user",
    "content": "Continue reasoning"
}
final_get_answer = {
    "role": "user",
    "content": "Base on the context, what is the answer?"
}
EquationSolver_UnknownVariable = {
    "role": "user",
    "content": "Give me the unknown variable of the equation system in latex style and
separated by comma"
}
EquationSolver_EquationSystem = {
    "role": "user",
    "content": "Give me the equation system in latex style and separated by comma"
}
Calculator_Equation = {
    "role": "user",
    "content": "Give me the equation to calculate in latex style and separated by
comma"
}
```

Prompt 示例

有 5 个这样的 prompt chat：

```
prompt_1_chat = [
    {
        "role": "user",
        "content": "Problem: Ramanujan and Hardy played a game where they both picked
a complex number. If the product of their numbers was $32-8i$, and Hardy picked
$5+3i$, what number did Ramanujan pick?\nLet's think step by step to solve this
problem"
    },
    {
        "role": "assistant",
        "content": "First, we need to find out how to use Hardy's number and the
product to find Ramanujan's number"
    },
    use_tool,
    {
        "role": "assistant",
        "content": "Do not use tool"
    },
    continue_reasoning,
    {
        "role": "assistant",
        "content": "Second, we can start by setting Ramanujan's number as $a+bi$ and
then use the fact that the product of their numbers is $32-8i$ to set up an equation.
We have:\n\n$(5+3i)(a+bi) = 32-8i$\n\nExpanding the left side, we get:\n\n$(5a-3b) +
(3a+5b)i = 32-8i$\n\nEquating the real and imaginary parts, we get the system of
equations:\n\n$5a-3b = 32$\n$3a+5b = -8$"
    },
```

Prompt 示例

```
    use_tool,
    {
        "role": "assistant",
        "content": "Equation solver"
    },
    EquationSolver_UnknownVariable,
    {
        "role": "assistant",
        "content": "$a$, $b$"
    },
    EquationSolver_EquationSystem,
    {
        "role": "assistant",
        "content": "$5*a-3*b = 32$, $3*a+5*b = -8$"
    },
    {
        "role": "user",
        "content": "Results: $a = 4$, $b = -4$\nContinue reasoning"
    },
    {
        "role": "assistant",
        "content": "Finally, we can conclude that Ramanujan picked the complex number
$4-4i$. Therefore, the answer is 4-4i"
    }
]
```

Begin Prompt

```
begin_prompt = {
    "user_begin": "You should solve the problem step by step and you should follow the
react in the history. In each reasoning step, you can use tool (calculator or equation
solver) to help you solve problem. Do you understand?",
    "assistant_begin": "Yes, I understand. I will follow my response in the
conversation history and solve the problem step by step.",
    "user_retrieval": "I give you some similar problems.\n{}You can use the knowledge
and thoery in these problem. Do you understand?",
    "assistant_retrieval": "Yes, I understand. I will solve the problem step by step
and use tool to help me.",
    "user_intr_tool": "You can use tool to help you solve the problem and I give you
the instruction of tools usage. Calculator can help you simplify the equation and
calculate the value of equation, and equation solver can help you solve the value of
unknown variable. Do you understand?",
    "assistant_intr_tool": "Yes, I understand. I will use tool to help me solve the
problem.",
}
```

源代码的问题

判断预测值和真实值的时候会出现本来应该判断为对，但却错误判断为错了，原因在于代码中判断正误太草率，仅仅是使用 ==，而实际上，会出现这些情况：

```
"llm_answer": "\\boxed{1.0}",
"real_answer": "1",
"score": false

"llm_answer": "\\boxed{1.0}",
"real_answer": "1",
"score": false

"llm_answer": "\\boxed{0.25}",
"real_answer": "\\frac{1}{4}",
"score": false
```

这些都应该是对的。

代码修改方案

修改为这样可以解决这种情况：

```
data['llm_answer'] = pred
data['real_answer'] = real_label

data['score'] = False
if isinstance(pred, str) and pred.startswith("\\boxed{") and pred.endswith("}"):
    pred = pred[7:-1]

    llm_value = pred
    real_value = real_label
    try:
        llm_value = float(pred)
        real_value = float(real_label)
    except ValueError:
        continue
        if (isclose(llm_value, real_value)):
            num_correct = num_correct + 1
            data['score'] = True
```

实验结果

math_algebra 一共有 1187 条数据，选择其中 100 条，分别选用 gpt-3.5-turbo 和 gpt-4o-2024-1120 进行测试，结果如下：

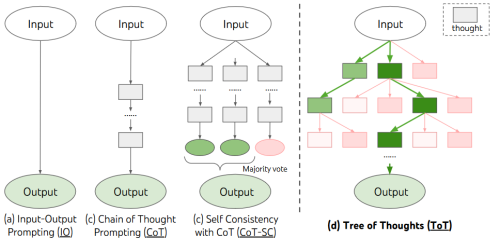
正确率：

- gpt-3.5-turbo: **58/101**（论文仓库中有一个完整的 output，正确率为 $348/1187 \approx 0.29$，但是论文中却说有 56.1%，疑似造假，但也有可能是手动把那些本应该计为正确，但代码输出为 false 的算作正确了，原始结果中有很多是这种情况的，改进后没有出现这种情况）
- gpt-4o-2024-1120: **59/101**

一共请求 API **3855** 次；一共花费 **$ 3.437**

对于 ChatCoT 来说，gpt-4o-2024-1120 比 gpt-3.5-turbo 提升不明显。

ToT: 核心思想



- ToT 将任何问题视为对一棵树的搜索。
- 每个节点表示一个状态 $s = [x, z_{1\cdots i}]$，即包含输入和当前思维序列的部分解决方案。

ToT 的具体实现包括四个问题

1. 如何将中间过程分解为思维步骤；
2. 如何从每个状态生成潜在的思维；
3. 如何通过启发式方法评估状态；
4. 应该使用什么搜索算法。

ToT 思维分解的核心思想

- 基于问题的特性设计并分解中间思维步骤。
- 根据问题类型，“思维”的粒度可以灵活变化：
  - 填字游戏：一个思维可能仅为几个单词。
  - 24 点游戏：一个思维可以是一行数学方程。
  - 创意写作：一个思维可能是整段写作计划。

对思维粒度的要求

- 必须足够小:
  - 以便 LM 能够生成多样且有希望的样本。
  - 例如：生成一本书太大而无法连贯。

对思维粒度的要求

- 必须足够小:
  - 以便 LM 能够生成多样且有希望的样本。
  - 例如：生成一本书太大而无法连贯。
- 必须足够大:
  - 以便 LM 能够有效评估其在问题求解中的前景。
  - 例如：生成一个 single token 太小而难以评估。

思维生成器的设置

- 在树状态 $s = [x, z_{1\cdots i}]$ 下，为生成下一个思维步骤的 k 个候选项。
- ToT 提供了两种生成策略。

(a) 独立同分布采样（i.i.d.）

- 通过 CoT 提示生成 k 个独立同分布的思维样本：

$$z^{(j)} \sim p_\theta^{CoT}(z_{i+1}|s) = p_\theta^{CoT}(z_{i+1}|x, z_{1\cdots i}),\ (j = 1 \cdots k).$$

(a) 独立同分布采样（i.i.d.）

- 通过 CoT 提示生成 k 个独立同分布的思维样本：

$$z^{(j)} \sim p_\theta^{CoT}(z_{i+1}|s) = p_\theta^{CoT}(z_{i+1}|x, z_{1\cdots i}),\ (j = 1 \cdots k).$$

- 特点：
  - 用于思维空间较丰富的场景（如创意写作）。
  - 提升生成样本的多样性。

(a) 独立同分布采样（i.i.d.）

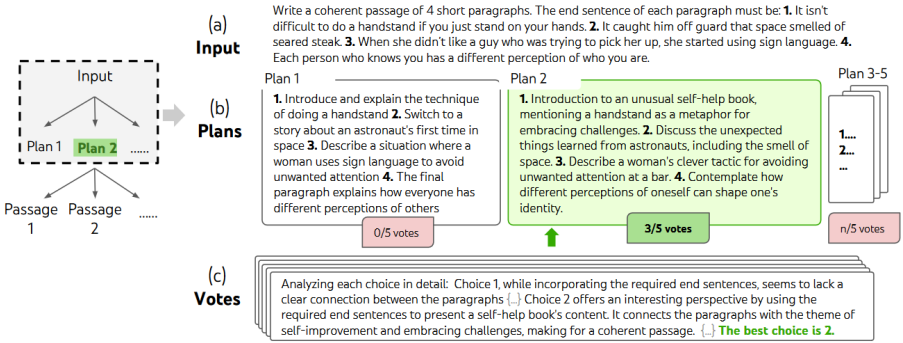


图 4: 独立同分布采样示例，创意写作

(b) 顺序提示

- 按顺序生成 k 个思维候选项：

$$[z^{(1)}, \cdots, z^{(k)}] \sim p_\theta^{propose}(z_{i+1}^{(1\cdots k)} \mid s).$$

(b) 顺序提示

- 按顺序生成 k 个思维候选项：

$$[z^{(1)}, \cdots, z^{(k)}] \sim p_\theta^{propose}(z_{i+1}^{(1\cdots k)} \mid s).$$

- 特点：
  - 适用于思维空间较为有限的场景。
  - 例如：24 点游戏或填字游戏（每个思维为一个单词/方程）。

状态评估器的功能

- 对一组边界状态 S 进行评估，衡量其在解决问题上的进展。
- LM 通过推理提供启发，以确定需要继续探索的状态及其顺序。
- 与传统启发式方法相比：
  - 比硬编码更灵活。
  - 比学习模型高效（尤其在样本有限下）。

状态评估策略

- (a) 独立评估每个状态

$$V(p_\theta, S)(s) \sim p_\theta^{\text{value}}(v \mid s), \quad \forall s \in S.$$

状态评估策略

- (a) 独立评估每个状态

$$V(p_\theta, S)(s) \sim p_\theta^{\text{value}}(v \mid s), \quad \forall s \in S.$$

- (b) 跨状态投票

$$s^* \sim p_\theta^{\text{vote}}(s^* \mid S).$$

(a) 独立评估每个状态

- 前瞻模拟：
  - 验证状态是否有潜力。
  - 例如：数字组合是否能达到目标。

(a) 独立评估每个状态

- 前瞻模拟：
  - 验证状态是否有潜力。
  - 例如：数字组合是否能达到目标。
- 常识判断：
  - 排除明显错误的状态。
  - 例如：“tzxc”不可能构成单词。

(b) 跨状态投票

- 比较各状态，投票选择最优状态。
- 对多选问题的多次采样结果进行比较。
- 示例：文章段落的连贯性。

搜索算法

- 论文实现了 BFS 和 DFS。
- Leaving more advanced ones (e.g. A* [11], MCTS [2]) for future work。

Algorithm 1 ToT-BFS($x, p_\theta, G, k, V, T, b$)

Require: Input x, LM p_θ, thought generator $G()$ & size limit k, states evaluator $V()$, step limit T, breadth limit b.

$S_0 \leftarrow \{x\}$

for $t = 1, \cdots, T$ **do**

$S'_t \leftarrow \{[s, z] \mid s \in S_{t-1}, z_t \in \mathrm{G}(p_\theta, s, k)\}$

$V_t \leftarrow V(p_\theta, S'_t)$

$S_t \leftarrow \arg\max_{S \subset S'_t, |S|=b} \sum_{s \in S} V_t(s)$

end for

return $G(p_\theta, \arg\max_{s \in S_T} V_T(s), 1)$

BFS 搜索策略

每一步维护 b 个最佳候选项：

- 提取剩余数字并提示语言模型生成候选。
- 对每个候选项进行评估（例如：是否能达到目标）。
- 选择前 b 个部分解继续搜索。

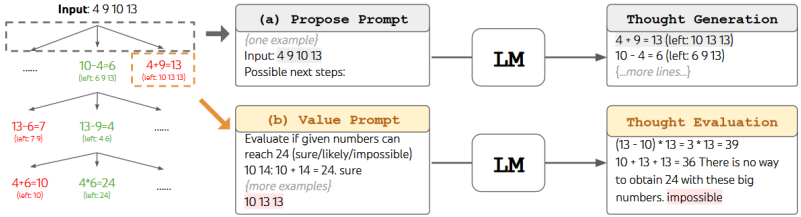


图 5: BFS 搜索示例，24 点

DFS 搜索策略

- 优先搜索当前最佳状态。
- 如果状态评估器判断当前状态不可行：
  - 剪枝该状态的子树。
  - 回溯到父节点。

Algorithm 2 ToT-DFS($s, t, p_\theta, G, k, V, T, v_{th}$)

Require: Current state s, step t, LM p_θ, thought generator $G()$ and size limit k, states evaluator $V()$, step limit T, threshold v_{th}

```
if t > T then record output G(p_θ, s, 1)
end if
for s' ∈ G(p_θ, s, k) do        ▷ sorted candidates
    if V(p_θ, {s'})(s) > v_thres then ▷ pruning
        DFS(s', t + 1)
    end if
end for
```

Thank you!