大型语言模型中的思维树：有意识的问题解决

姚舜宇
普林斯顿大学

于典
谷歌DeepMind

杰弗里·赵
谷歌DeepMind

伊扎克·沙夫兰
谷歌DeepMind

托马斯·L·格里菲斯
普林斯顿大学

曹媛
谷歌DeepMind

卡尔蒂克·纳拉西姆汉
普林斯顿大学

摘要

语言模型越来越多地被用于各种任务的一般问题解决，但在推理过程中仍然局限于令牌级别、从左到右的决策过程。 这意味着它们在需要探索、战略前瞻或初始决策起关键作用的任务中可能会有所不足。 为了克服这些挑战，我们引入了一种新的语言模型推理框架“思维树”（ToT），它泛化了流行的“思维链”方法，使得语言模型能够对连贯的文本单元（“思维”）进行探索，这些单元作为解决问题的中间步骤。 ToT允许语言模型通过考虑多个不同的推理路径和自我评估选择来进行有意识的决策，以决定下一步的行动，同时在必要时向前或向后跟踪以进行全局选择。 我们的实验表明，ToT显著提高了语言模型在三个需要复杂规划或搜索的新任务中的问题解决能力：24点游戏、创意写作和迷你填字游戏。例如，在24点游戏中，仅使用思维链提示的GPT-4只解决了4%的任务，而我们的方法成功率达到了74%。 包含所有提示的代码存储库：https://github.com/princeton-nlp/tree-of-thought-llm.

1 引言

最初设计用于生成文本的语言模型（LMs）的扩展版本，如GPT [25，26，1，23]和PaLM [5]，已经显示出越来越能够执行越来越广泛的任务，这些任务需要数学、符号、常识和知识推理的能力。 令人惊讶的是，在所有这些进展的背后，仍然是生成文本的原始自回归机制，它逐个进行标记级别的决策，并以从左到右的方式进行。 这样一个简单的机制是否足以构建一个通用问题解决器的LM？ 如果不是，当前范式将面临哪些问题，应该有什么替代机制？

人类认知的文献提供了一些线索来回答这些问题。 关于“双过程”模型的研究表明，人们在决策中有两种模式-一种是快速、自动、无意识的模式（“系统1”），另一种是缓慢、有意识的模式（“系统2”）[30, 31, 16, 15]。 这两种模式以前已与机器学习中使用的各种数学模型相关联。 例如，关于人类和其他动物的强化学习的研究已经探讨了它们进行联想“无模型”学习或更深思熟虑的“有模型”规划的情况[7]。 LM的简单联想标记级别选择也类似于“系统1”，因此可能受益于更深思熟虑的“系统2”规划过程，该过程（1）维护和探索当前的多种选择

第37届神经信息处理系统会议（NeurIPS 2023）

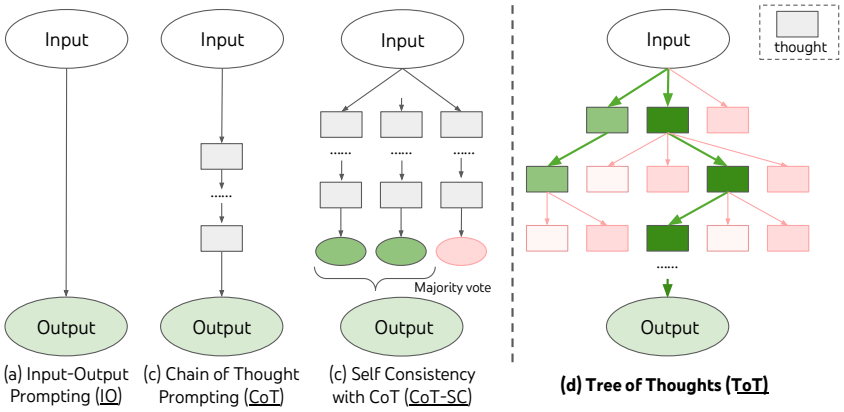

图1：示意图，说明了使用LLM进行问题解决的各种方法。每个矩形框代表一个思维，它是一个连贯的语言序列，用作问题解决的中间步骤。 在图2、4、6中可以看到思维是如何生成、评估和搜索的具体示例。

而不仅仅是选择一个，并且（2）评估其当前状态并积极地向前或向后看以做出更全局的决策。

为了设计这样一个规划过程，我们回到人工智能（和认知科学）的起源，从Newell、Shaw和Simon在20世纪50年代开始探索的规划过程中汲取灵感[21,22]。 Newell和他的同事将问题解决[21]描述为对组合问题空间的搜索，表示为一棵树。 因此，我们提出了思维树（ToT）框架，用于语言模型的一般问题解决。 如图1所示，虽然现有的方法（详见下文）对于问题解决采样连续的语言序列，但ToT积极地维护着一棵思维树，其中每个思维都是一个连贯的语言序列，作为问题解决的中间步骤（表1）。 这样一个高级语义单元允许语言模型通过有意识的推理过程自我评估不同中间思维在解决问题方面的进展（图2,4,6）。 通过LM自我评估和思考来实现搜索启发式是新颖的，因为以前的搜索启发式要么是编程的，要么是学习的。 最后，我们将这种基于语言的生成和评估多样思维的能力与搜索算法相结合，例如广度优先搜索（BFS）或深度优先搜索（DFS），这些算法允许对思维树进行系统性的探索，具有前瞻性和回溯。

根据实证，我们提出了三个新问题，即使使用最先进的语言模型GPT-4 [23]，也挑战了现有的语言模型推理方法：24点游戏、创意写作和纵横填字（表1）。 这些任务需要演绎、数学、常识、词汇推理能力，以及一种整合系统化规划或搜索的方法。 我们展示了思维树在这三个任务上取得了优秀的结果，因为它具备足够的通用性和灵活性，可以支持不同层次的思维、不同的思维生成和评估方式，以及适应不同问题性质的搜索算法。 我们还通过系统的剔除分析了这些选择对模型性能的影响，并讨论了未来的发展方向，以更好地训练和使用语言模型。

2 背景

我们首先对一些使用大型语言模型进行问题解决的现有方法进行形式化，这些方法启发了我们的方法，并进行了比较。 我们使用 p_θ来表示具有参数 θ的预训练语言模型，使用小写字母x、y、z、s、$\cdots$来表示语言序列，即 $x = (x[1], \cdots, x[n])$，其中每个 $x[i]$是一个标记，因此 $p_\theta(x) = \prod^n{}_{i=1} p_\theta(x[i]|x[1...i])$。 我们使用大写字母 S、$\cdots$来表示一组语言序列。

输入-输出（IO）提示是将问题输入x转化为输出y的最常见方法，其中$y \sim p_\theta(y|p\mathbf{rompt\ IO}(x))$，$p\mathbf{rompt\ IO}(x)$将输入$x$与任务说明和/或少量示例包装在一起。 为简单起见，我们用p_θ表示$p\mathbf{rompt}$ $(\mathtt{output}|\mathtt{input}) = p_\theta(\mathtt{output}|p\mathtt{rompt}(\mathtt{input}))$，因此IO提示可以表示为$y \sim p\mathrm{IO}_\theta(y|x)$。

思维链（**CoT**）提示[38]被提出来解决输入x到输出y的映射非平凡的情况（例如，当x是数学问题而y是最终数值答案时）。
关键思想是引入一系列思维z_1，...，z_n来连接x和y，其中每个z_i是一个连贯的语言序列，作为解决问题的有意义的中间步骤（例如，z_i可以是数学问答的中间方程）。为了用CoT解决问题，每个思维
$z_i \sim p_\theta^{CoT}(z_i \mid x, z_{1\cdots i-1})$按顺序抽样，然后输出$y \sim p_\theta^{CoT}(y \mid x, z_{1\cdots n})$。 在实践中，$[z_{1\cdots n}, y] \sim p_\theta^{CoT}(z_{1\cdots n}, y \mid x)$被连续的语言序列抽样，思维的分解（例如每个 z_i是一个短语、一个句子还是一个段落）是模糊的。

与**CoT**（**CoT-SC**）的自洽性[36]是一种集成方法，它抽样 k**i.i.d.**思维链：$[z_1^{(i)}{}_{\cdots n}, y^{(i)}] \sim p_\theta^{CoT}(z_{1\cdots n}, y \mid x)\ (i = 1 \cdots k)$，然后返回最频繁的输出：$\arg\max_y \#\{i \mid y^{(i)} = y\}$。 **CoT-SC**改进了**CoT**，因为对于相同的问题通常存在不同的思维过程（例如证明相同定理的不同方法），通过探索更丰富的思维集合，输出决策可以更加忠实。 然而，在每个链中没有对不同思维步骤的局部探索，而且“最频繁”启发式仅适用于输出空间有限的情况（例如多项选择题）。

3 大型语言模型中的思维树：有意识的问题解决

> 一个真正的问题解决过程涉及到反复使用可用信息来启动探索，这样依次揭示更多信息，直到最终发现解决方案的方法。—— N*e*w*ell*等人[21]

关于人类问题解决的研究表明，人们通过搜索组合问题空间来解决问题 - 一棵树，其中节点表示部分解决方案，分支对应修改它们的操作[21,22]。 选择哪个分支是由启发式决定的，它有助于导航问题空间并引导问题解决者走向解决方案。 这种观点突出了使用语言模型解决一般问题的现有方法的两个关键缺点：1）在思维过程中，它们不会探索不同的延续 - 树的分支。2）在全局范围内，它们不会结合任何类型的规划、前瞻或回溯来帮助评估这些不同的选项 - 这种启发式引导搜索似乎是人类问题解决的特征。

为了解决这些缺点，我们引入了思维树（ToT），这是一种允许语言模型在思维上探索多个推理路径的范式（图$_1$(c)）。 ToT将任何问题都视为在树上进行搜索，其中每个节点都是一个状态$s = [x, z_{1\cdots i}]$，表示具有输入和迄今为止的思维序列的部分解决方案。ToT的具体实例涉及回答四个问题：$_1$. 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. 如何启发式地评估状态；4. 使用什么搜索算法。

1. 思维分解。虽然**CoT**在没有明确分解的情况下连贯地采样思维，但**ToT**利用问题属性来设计和分解中间思维步骤。 如表**1**所示，根据不同的问题，一个思维可以是几个单词（填字游戏），一行方程式（24点游戏）或一整段写作计划（创意写作）。总的来说，一个思维应该足够“小”，以便语言模型可以生成有前景且多样化的样本（例如，生成整本书通常太“大”而不连贯），但又足够“大”，以便语言模型可以评估其解决问题的前景（例如，生成一个标记通常太“小”而无法评估）。

2. 思维生成器 $G(p_\theta, s, k)$**.**给定一个树状态 $s = [x, z_{1\cdots i}]$，我们考虑两种策略来生成下一个思维步骤的候选：

(a) 从CoT提示（创意写作，图4）中随机抽取独立同分布的思维：$z^{(j)} \sim p_\theta^{CoT}(z_{i+1} \mid s) = p_\theta^{CoT}$当思维空间丰富时（例如，每个思维是一个段落），这种方法效果更好，独立同分布的样本能够带来多样性；(b) 使用“提议提示”（24点游戏，图2；纵横字谜，图6）逐个提出思维：$[z^{(1)}, \cdots, z(k)] \sim p_\theta^{propose}(z_{i+1}^{(1\cdots k)} \mid s)$。当思维空间更受限制时，这种方法效果更好（例如，每个思维只是一个词或一行），因此在相同的上下文中提出不同的思维可以避免重复。

3. 状态评估器 $V(p_\theta, S)$。给定不同状态的前沿，状态评估器评估它们在解决问题方面的进展，作为搜索算法确定保持探索和以哪种顺序探索的启发式。 虽然启发式是解决搜索问题的标准方法，但它们通常是编程的（例如，**D**eep**B**l**ue** [3]）或

学习的（例如，AlphaGo [29]）。我们提出了第三种选择，即使用语言模型有意识地推理状态。在适用的情况下，这种有意识的启发式方法可以比编程规则更灵活，比学习模型更节约样本。与思维生成器类似，我们考虑独立或一起评估状态的两种策略：(a)独立地对每个状态进行价值评估：$V(p_\theta, S)(s) \sim p_\theta^{value}(v|s)\ \forall\ s \in S$，其中一个值提示关于状态$s$的原因来生成一个标量值 v（例如1-10）或一个分类（例如确定/可能/不可能），可以启发性地转化为一个值。这种评估推理的基础可以因问题和思维步骤而异。在这项工作中，我们通过少量的前瞻模拟（例如快速确认5、5、14可以通过5 + 5 + 14达到24，或者“hot l”可以通过在“ ”中填充“e”来表示“inn”）以及常识（例如1 2 3太小无法达到24，或者没有单词可以以“tzxc”开头）来探索评估。前者可能促进“好”状态，而后者可以帮助消除“坏”状态。这种评估不需要完美，只需要在决策过程中大致有帮助。

(b)跨州投票：$V(p_\theta, S)(s) = \mathbb{1}[s = s^*]$，其中一个“好”的州 $s^* \sim p_\theta^{vote}\ (s^*|S)$是根据有意识地比较 S中不同的州进行投票淘汰的。当直接评估问题成功的难度较大时（例如，段落连贯性），自然而然地会选择比较不同的部分解决方案并投票选择最有希望的一个。这在精神上类似于“逐步”自洽策略，即将“要探索的状态”视为多项选择问题，并使用语言模型样本进行投票。

对于这两种策略，我们可以多次提示语言模型以汇总价值或投票结果，以便在时间/资源/成本上进行权衡，以获得更可靠/稳健的启发式方法。

算法 1 ToT-BFS($x, p_\theta, G, k, V, T, b$)要求：输入 x，语言模型 p_θ，思维生成器 $G()$和大小限制 k，状态评估器 $V()$，步数限制 T，广度限制 b。

$S_0 \leftarrow \{x\}$

对于 $t = 1, \cdots, T$**执行**

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

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

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

结束循环

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

算法2 ToT-DFS($s, t, p_\theta, G, k, V, T, v_{th}$)要求：当前状态 s，步骤 t，语言模型 p_θ，思维生成器 $G()$和大小限制 k，状态评估器$V()$，步骤限制 T，阈值 v_{th}如果$t > T$，则记录输出 $G(p_\theta, s, 1)$结束如果

对于 $s' \in G(p_\theta, s, k)$ 执行 ▷排序的候选项

  如果 $V(p_\theta, \{s'\})(s) > v_{thres}$则 ▷剪枝

    DFS($s', t + 1$)

  结束如果

结束循环

4. 搜索算法。最后，在**ToT**框架内，可以根据树结构插入和使用不同的搜索算法。我们探索了两种相对简单的搜索算法，并将更高级的算法（例如A* [11]，MCTS [2]）留给未来的工作：

(a) 广度优先搜索（**BFS**）（算法1）每一步都维护一个最有希望的状态集合 b。这用于24点游戏和创意写作，其中树的深度有限（$T \leq 3$），初始思考步骤可以评估和修剪为一个小集合（$b \leq 5$）。

(b) 深度优先搜索（**DFS**）（算法2）首先探索最有希望的状态，直到达到最终输出（$t > T$），或者状态评估器认为从当前状态无法解决问题$V(p_\theta, \{s\}) (s) \leq v_{th}$，其中$v_{th}$是一个值阈值。在后一种情况下，从$s$开始的子树被修剪以换取探索和利用的平衡。在这两种情况下，**DFS**会回溯到 s的父状态以继续探索。

从概念上讲，ToT作为一种使用LM进行一般问题解决的方法具有几个优点：(1)通用性。IO、CoT、CoT-SC和自我完善可以看作是ToT的特殊情况（即有限深度和广度的树；图1）。(2)模块化。基本LM以及思维分解、生成、评估和搜索过程都可以独立变化。(3)适应性。可以适应不同的问题属性、LM能力和资源限制。(4)便利性。无需额外的训练，只需一个预训练的LM即可。下一节将展示这些概念上的优势如何在不同问题中体现出强大的实证性能。

4 实验

我们提出了三个任务，即使从最先进的语言模型GPT-4 [23]中进行采样，使用标准的IO提示或思维链（CoT）提示，这些任务仍然很难。我们展示了如何

	24的游戏	创意写作	5x5填字游戏
输入	4个数字（4 9 10 13）	4个随机句子	10个线索（h1. 呈现;..）
输出	一个等式得到24 (13-9)*(10-4)=24	4段文字 以4个句子结尾	5x5字母：SHOWN; WIRRA; AVAIL; ...
思考	3个中间等式 (13-9=4 (剩下4,4,10); 10-4=6 (剩下4,6); 4*6=24)	一个简短的写作计划 (1. 介绍一本连接的书...)	填入线索的词语: (h1. 呈现; v5. naled; ...)
#ToT步骤	3	1	5-10 (变量)

表1：任务概述。 输入，输出，思考示例以蓝色显示。

在思维树中进行有意识的搜索（ToT）可以产生更好的结果，更重要的是，以更有趣和有前途的方式使用语言模型来解决需要搜索或规划的问题。
除非另有说明，我们使用Chat Completion模式的GPT-4[1]进行实验
采样温度为0.7。

4.1 24的游戏

24点游戏是一个数学推理挑战，目标是使用4个数字和基本的
算术运算（+-*/)来得到24。 例如，给定输入“4 9 10 13”，一个解决方案输出
可以是“(10 - 4) * (13 - 9) = 24”。

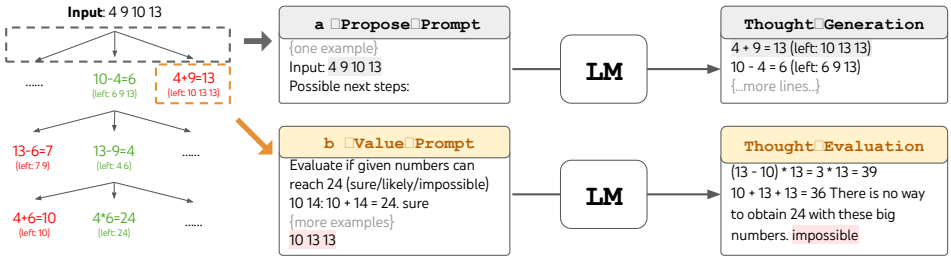

图2：ToT在24点游戏中的应用。 LM被提示进行思维生成和评估。

任务设置。我们从4nums.com上获取数据，该网站有1,362个游戏，按照人类解题时间从简单到困难排序，并使用索引为901-1,000的相对困难的游戏子集进行测试。
对于每个任务，如果输出是一个等于24的有效方程，并且每个输入数字仅使用一次，我们将其视为成功。 我们将100个游戏的成功率作为指标进行报告。

基准。我们使用5个上下文示例的标准输入-输出（IO）提示。对于链式-思维（CoT）提示，我们使用3个中间方程式增强每个输入-输出对，每个对剩下的两个数字进行操作。 例如，给定输入“4 9 10 13”，思维可能是“13 - 9 = 4（剩下：4 4 10）；10 - 4 = 6（剩下：4 6）；4 * 6 = 24（剩下：24）”。 对于每个游戏，我们进行100次IO和CoT提示的采样，以获得平均性能。 我们还考虑了CoT自一致性基准，该基准从100个CoT样本中选择多数输出，并在最多10次迭代中对IO样本进行迭代优化。 在每次迭代中，如果输出不正确，LM会根据所有先前的历史记录进行调整，以“反思错误并生成精炼的答案”。 请注意，它使用关于方程正确性的真实反馈信号。

ToT设置。将24点游戏转化为**ToT**，将思维分解为3个步骤，每个步骤都是一个中间方程。如图2(a)所示，在每个树节点上，我们提取剩余的数字，并提示LM提出一些可能的下一步。 尽管只有一个包含4个输入数字的示例，但相同的“提出提示”用于所有3个思考步骤。在**ToT**中，我们执行广度优先搜索（BFS），在每一步中保留最佳 $b = 5$候选项。 为了在**ToT**中进行有意识的BFS，如图2(b)所示，我们提示LM根据达到24点的可能性对每个思考候选项进行评估，评估结果为“确定/可能/不可能”。 目标是促进可以在少数前瞻试验中得出结论的正确部分解，并根据“太大/太小”的常识消除不可能的部分解，保留其余的“可能”。 我们对每个思考值进行 3次采样。

[1]实验在2023年5月5日至16日之间进行。

方法	成功
IO提示	7.3%
CoT提示	4.0%
CoT-SC (k=100)	9.0%
我们的ToT (b=1)	45%
我们的ToT (b=5)	**74%**
IO + 优化 (k=10)	27%
IO (100个样本中的最佳)	33%
CoT (100个样本中的最佳)	49%

表2：24点游戏结果。

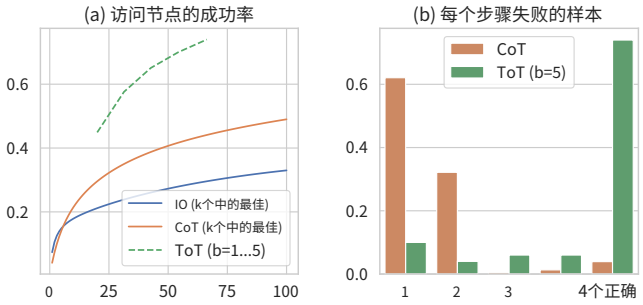

图3：24点游戏的(a) 规模分析和(b) 错误分析。

结果。如表2所示，IO、CoT和CoT-SC提示方法在任务中表现不佳，成功率仅为7.3%、4.0%和9.0%。 相比之下，ToT在 $b=1$的情况下已经达到了45%的成功率，而 $b=5$则达到了74%。 我们还考虑了IO/CoT的oracle设置，通过计算使用最佳 k样本的成功率 $(1 \leq k \leq 100)$。 为了将IO/CoT(k个中的最佳)与ToT进行比较，我们考虑计算ToT中每个任务访问的树节点数，跨 $b=1 \cdots 5$进行映射，并将5个成功率在图3(a)中表示，将IO/CoT (k个中的 k)视为在赌博机中访问 k个节点。
毫不奇怪，思维树比IO更具规模，100个思维树样本中最好的样本成功率达到了 49%，但仍远远不及在思维树中探索更多节点（$b>1$）的情况。

错误分析。图3(b)显示了CoT和ToT样本在哪个步骤中失败了任务，即在思维树中的思维（对于CoT）或所有 b思维（对于ToT）无效或无法达到24。 值得注意的是，在生成第一步或者等效地说，生成前三个单词（例如“$4+9$”）后，约60%的CoT样本已经失败了任务。这凸显了直接从左到右解码的问题。

4.2 创意写作

接下来，我们设计了一个创意写作任务，输入是4个随机句子，输出应该是一个有4个段落的连贯段落，每个段落以相应的输入句子结尾。 这样的任务是开放性的和探索性的，挑战创造性思维和高层次规划能力。

任务设置。我们从randomwordgenerator.com随机抽取句子来形成100个输入，并且每个输入约束都没有真实的参考文本。 我们发现GPT-4大多数时候可以遵循输入约束，因此我们专注于通过两种方式评估段落的连贯性：使用GPT-4的零-shot提示来提供一个1-10的标量分数，或者使用人工判断来比较不同方法生成的输出对。 对于前者，我们抽取5个分数并对每个任务输出取平均值，我们发现这5个分数通常是一致的，平均标准差约为 0.56。对于后者，我们在盲目研究中使用作者的一个子集来比较CoT与ToT生成的段落对的连贯性，其中段落的顺序在100个输入中是随机翻转的。

基准。考虑到任务的创造性质，IO和CoT提示都是零-shot的。 虽然前者提示LM直接根据输入约束生成连贯的段落，而后者提示LM先制定一个简要计划，然后编写段落，即计划作为中间思考步骤。 我们为每个任务生成10个IO和CoT样本。 我们还考虑在每个任务的随机IO样本上进行迭代-改进（$k \leq 5$）方法，其中LM根据输入约束和最后生成的段落来决定段落是否已经“完全连贯”，如果没有则生成一个改进的段落。

ToT设置。我们构建了一个深度为2的ToT（仅有1个中间思考步骤）- 语言模型首先生成$k=5$个计划并投票选出最佳计划（图4），然后类似地生成$k=5$个基于最佳计划的段落并投票选出最佳段落。这里的广度限制$b=1$，每个步骤只保留一个选择。 我们使用一个简单的零-shot投票提示（“分析下面的选择，然后得出哪个对指令最有希望”）来采样两个步骤的5个投票。

结果。图5(a)显示了100个任务中GPT-4的平均得分，其中ToT（7.56）的生成的段落比IO（6.19）和CoT（6.93）平均更连贯。 虽然这样的自动度量可能会有噪音，但图5(b)通过显示人们在100个段落对中更喜欢ToT而不是CoT的结果来确认这一发现，有41个段落对中人们更喜欢ToT，只有21个段落对中人们更喜欢CoT（其他38个段落对被认为“同样连贯”）。 最后，迭代细化在这个自然语言任务上更加有效，其中

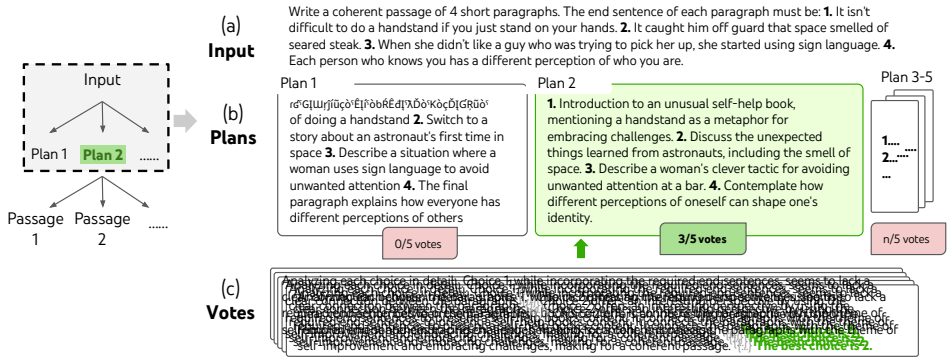

图4：在随机选择的创意写作任务中进行有意识搜索的一步。给定输入，语言模型样本生成5个不同的计划，然后投票5次来决定哪个计划最好。 多数选择用相同的样本-投票过程写出输出段落。

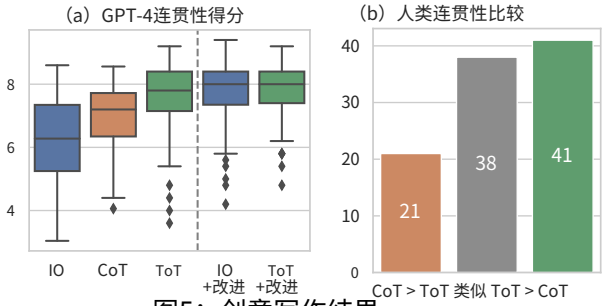

图5：创意写作结果。

方法	成功率（%） 字母	单词	游戏
IO	38.7	14	0
CoT	40.6	15.6	1
ToT（我们的）	**78**	**60**	**20**
+最佳状态	82.4	67.5	35
-修剪	65.4	41.5	5
-回溯	54.6	20	5

表3：迷你填字游戏结果。

它将IO连贯性得分从6.19提高到7.67，将ToT连贯性得分从7.56提高到7.91。我们相信它可以被视为ToT框架中思维生成的第三种方法，其中新的思维可以从改进旧的思维中产生，而不是独立或顺序生成。

4.3 迷你填字游戏

在24点游戏和创意写作中，思维树相对较浅——最多需要3个思考步骤才能得出最终结果。在这里，我们探索 5×5的迷你填字游戏，这是一个更难的搜索问题，涉及到自然语言。 再次强调，目标不仅仅是解决任务，因为更一般的填字游戏可以通过专门的NLP流水线[34]轻松解决，该流水线利用大规模检索而不是语言模型。相反，我们的目标是探索语言模型作为一个通用问题解决器的极限，它通过有意识的推理来探索自己的思维并引导自己的探索。

任务设置。我们从GooBix网站上获取数据，其中包含156个 5×5的迷你填字游戏。 由于观察到相邻的游戏包含相似的线索，我们使用了索引为 $1, 6, \cdots, 91$的20个游戏。 96 用于测试和游戏 $136, 141, 146, 151, 156$以进行提示。 对于每个任务，输入描述了5个水平线索和5个垂直线索，输出应该是一个解决纵横字谜的 $5 \times 5 = 25$个字母的棋盘。 对于评估，我们考虑三个成功的级别：正确字母的比例（每个游戏25个），单词（每个游戏10个）和游戏。

基准。 我们在IO提示中提供了5个示例输入输出对，在CoT提示中还包括按照h1..5然后v1..5的顺序的中间单词。我们对每个提示运行10个样本并平均结果。

ToT设置。我们利用深度优先搜索（算法2），不断探索最有希望的下一个单词线索，直到状态不再有希望，然后回溯到父状态以探索替代思路。 为了使搜索可行，后续的思路受到限制，不能改变任何填充的单词或字母，这样**ToT**最多有10个中间步骤。 对于思路生成，在每个状态下，我们将所有现有的思路（例如，图6(a)中的“h2.motor; h1.tasks”）转化为剩余线索的字母约束（例如，“v1.To heap: tm ;...”），并提示提案提示 5次，以提出下一个单词的填充位置和内容的候选项。

重要的是，我们还提示语言模型给出不同思维的置信水平，并进行聚合

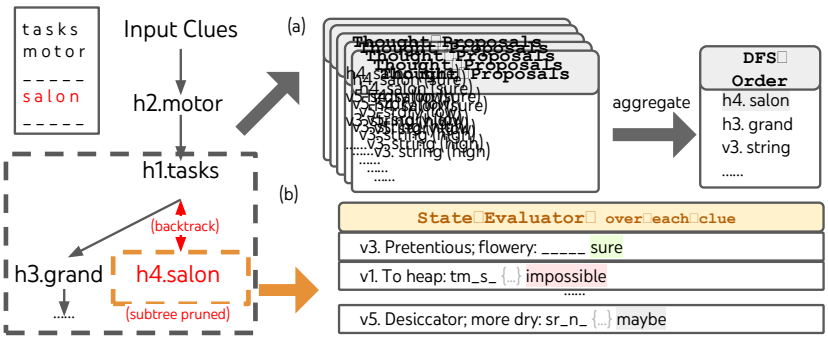

图6：在Mini Crosswords中，(a) 思维是如何被提出和在优先队列中聚合的，用于深度优先搜索(DFS)，以及 (b) 如何根据填充每个剩余单词提示的可能性来评估状态，并且如果语言模型认为任何剩余提示不可能填充，则进行剪枝。然后DFS回溯到父状态并探索下一个有希望的提示。

通过这些横向提案，获得一个排序的下一个要探索的思维列表 (图6(a)). 对于状态评估，我们同样将每个状态转化为剩余提示的字母约束，然后根据约束评估每个提示是否可能填充。 如果任何剩余提示被认为是“不可能”填充的（例如“v1. To heap: tm s ”），则状态的子树探索被剪枝，DFS回溯到其父状态以探索下一个有希望的思维。 我们将DFS搜索步骤限制为100步，并将最深度探索的状态（如果有多个，则为第一个探索的状态）呈现为最终输出。

结果。如表3所示，IO和CoT提示方法在单词级别的成功率低于16%时表现不佳，而ToT显著改善了所有指标，实现了单词级别的成功率为60%，并解决了20个游戏中的4个。 这样的改进并不令人惊讶，因为IO和CoT缺乏尝试不同线索、更改决策或回溯的机制。

Oracle和消融研究。 当从每个任务的**oracle**最佳DFS状态（而不是启发式确定的最佳状态）输出时，ToT的性能甚至更高，并且实际上解决了
7/20个游戏（表3，“+最佳状态”），表明我们简单的输出启发式方法可以很容易地改进。有趣的是，有时当填字游戏实际上被解决时，状态评估器可能仍然认为某些单词是“不可能的”并进行剪枝 - 可能是因为设计为5×5的填字游戏中有一些GPT-4无法识别的罕见或过时的单词。 鉴于状态评估作为剪枝启发式的不完美性，我们还探索了剪枝的消融，并发现性能通常更差（表3，“-剪枝”）。 然而，它实际上可以找到4/20个游戏的正确解决方案（尽管只通过启发式输出1个），其中3个是ToT+剪枝在100步内无法解决的游戏。 因此，在这种情况下，更好的DFS剪枝启发式对于问题解决至关重要。 最后，我们通过运行一个消融实验来确认回溯的重要性，该实验在最多20步内保持填充最有希望的线索，允许覆盖。 这类似于具有广度限制为$b = 1$的“贪婪”BFS搜索，并且在单词级别的成功率仅为20%时表现不佳（表3，“-回溯”）。

5 相关工作

规划和决策 智能规划和决策对于实现预定目标至关重要。. 由于它们接受了大量的世界知识和人类示例的训练，语言模型已经吸收了丰富的常识，使得它们能够根据问题设置和环境状态提出合理的计划[12, 42, 37, 13, 35, 41, 40]。 我们提出的ToT方法在每个问题解决步骤中同时考虑多个潜在可行的计划，并继续进行最有希望的计划。 思维采样和价值反馈的整合有机地结合了规划和决策机制，实现了在解决方案树内的有效搜索。 另一方面，传统的决策过程通常需要训练专门的奖励和策略模型，如强化学习中的CHAI [33]，而我们使用语言模型本身来提供决策的价值估计。 RAP [9]是一项并行工作，它处理语言模型

[2]例如，“agend”是“agendum”的过时形式，但是GPT-4认为它是“agenda”的拼写错误。 外部检索或网络交互可以增强语言模型在知识不确定性下的问题解决能力。

推理作为规划与其内部世界模型相结合，并提出了一种类似于ToT的基于MCTS的方法。然而，它的任务比我们的简单，并且其框架缺乏将不同的树搜索算法整合进来的模块化能力。

自我反思。 使用LLMs评估自己预测的可行性在问题解决中变得越来越重要。. [28, 20, 24]引入了“自我反思”机制，其中LLMs向其生成候选项提供反馈。 [4]通过注入由LLM自身基于其代码执行结果生成的反馈消息，提高了LLMs代码生成的准确性。 类似地，[17]还引入了对操作和状态的“评论”或审查步骤，决定解决计算机操作任务时要采取的下一步行动。 与我们的方法非常相关的另一项最新工作是“自我评估引导解码”[39]。 与我们的方法类似，自我评估解码也采用了从随机波束搜索解码中采样的叶子节点进行树搜索过程，然后由LLM自身使用精心准备的自我评估提示进行评估。 然而，他们的方法使用了PAL公式[8]来表示思维为代码，这使得难以处理像我们在本文中考虑的创意写作等具有挑战性的任务。 因此，我们的思维树形式更加灵活，可以处理GPT-4仅通过标准提示实现非常低准确性的具有挑战性的任务。

程序引导的LLM生成。我们的提议还与最近的进展相关，这些进展通过系统化的程序[14,44,6,43]或符号化的程序指导来组织LM的行为。 例如，Schlag等人[27]将LM嵌入到算法搜索过程中，以帮助逐步解决问题，其中搜索树通过可能提供答案的相关段落进行扩展。 然而，这种方法与我们的方法不同，树的扩展是通过抽样外部段落而不是LM自身的思考，并且没有反思或投票步骤。

另一种方法LLM+P [18]更进一步，将实际的规划过程委托给经典规划器。

经典搜索方法。最后但并非最不重要的，我们的方法可以被视为解决问题的现代版本的经典搜索方法。 例如，它可以被视为一种启发式搜索算法，类似于A* [10]，其中每个搜索节点的启发式由LM的自我评估提供。 从这个角度来看，我们的方法还与NeuroLogic A*esque解码[19]相关，它受到A*搜索的启发，但引入了对于LM来说在波束搜索或top-k采样解码中高效的前瞻启发式。 然而，这种方法受限于句子生成任务，而我们的框架则设计用于由价值反馈保护的复杂多步问题解决。

6 讨论

限制和未来方向。像ToT这样的有意识搜索可能对于GPT-4已经擅长的许多现有任务来说并不是必要的（请参见附录B.1），而这项工作只是作为一个初步步骤探索了三个相对简单的任务，挑战了GPT-4（请参见附录B.2中的一些GPT-3.5实验结果），并呼吁更好的搜索和计划能力与语言模型相结合。然而，随着我们开始将语言模型应用于更多的实际决策应用（例如编码、数据分析、机器人等），可能会出现更复杂的任务，并提供研究这些问题的新机会。 此外，像ToT这样的搜索方法需要更多的资源（例如GPT-4 API成本）来提高任务性能，但ToT的模块化灵活性允许用户自定义性能成本权衡，并且正在进行的开源努力[32]将在不久的将来大大降低这些成本。 有关成本和效率的更多细节请参见附录B.3。最后，这项工作侧重于使用现成的语言模型，并使用ToT风格的高级反事实决策（例如对下一段落的潜在选择进行思考，而不是预测下一个标记）来微调语言模型可能会提供增强语言模型问题解决能力的机会。

结论。基于搜索问题解决方案的可能路径树，可以有益地增强LM的关联“系统1”。 思维树框架为将关于问题解决的经典见解转化为当代LM的可行方法提供了一种方式。同时，LM解决了这些经典方法的一个弱点，即解决不易形式化的复杂问题，如创意写作。 我们将LM与经典AI方法的交叉视为一个令人兴奋的方向。

更广泛的影响

ToT是一个使LM更自主、更智能地做出决策和解决问题的框架。 虽然当前的任务仅限于推理和搜索问题，但未来涉及与外部环境或人类的交互的应用可能带来潜在的危险，例如促进LM的有害使用。另一方面，ToT还提高了模型决策的可解释性和人类对齐的机会，因为生成的表示是可读的、高级语言推理，而不是隐含的低级标记值。

致谢

SY和KN感谢Oracle合作研究奖和国家科学基金会2239363号资助。 本材料中表达的任何观点、发现、结论或建议-都是作者的个人观点，不一定反映国家科学基金会的观点。SY还获得普林斯顿大学的哈罗德・W・多兹奖学金的支持。

参考文献

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam,G. Sastry, A. Askell, 等。语言模型是少样本学习者。 神经信息处理系统进展，33:1877–1901，2020年。

[2] C. Browne, E. J. Powley, D. Whitehouse, S. M. M. Lucas, P. I. Cowling, P. Rohlfshagen,S. Tavener, D. P. Liebana, S. Samothrakis, 和 S. Colton. 蒙特卡洛树搜索方法综述。*IEEE*计算智能与人工智能在游戏中的交易，4:1–43，2012年。

[3] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu. 深蓝.人工智能, 134(1-2):57–83, 2002.

[4] X. Chen, M. Lin, N̈. Scharli, and D. Zhou. 教授大型语言模型进行自我调试, 2023.

[5] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, 等. Palm: 使用路径扩展语言建模. *arXiv* 预印本 *arXiv:2204.02311*, 2022.

[6] A. Creswell and M. Shanahan. 使用大型语言模型进行准确推理.*arXiv* 预印本 *arXiv:2208.14271*, 2022.

[7] N. D. Daw, Y. Niv, and P. Dayan. 基于不确定性的前额叶和背外侧纹状体系统之间的行为控制竞争。自然神经科学，8(12):1704–1711，2005年。

[8] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. Pal: 程序辅助的语言模型，2023年。

[9] S. Hao, Y. Gu, H. Ma, J. J. Hong, Z. Wang, D. Z. Wang, and Z. Hu. 用语言模型推理就是用世界模型进行规划。ar*Xiv*预印本 ar*Xiv:2305.14*992，*2023*年。

[10] P. E. Hart, N. J. Nilsson, and B. Raphael. 启发式确定最小成本路径的形式基础。*IEEE Transactions on Systems Science and Cybernetics*，4(2):100–107，1968年。 doi: 10.1109/TSSC.1968.300136.

[11] P. E. Hart, N. J. Nilsson, and B. Raphael. 启发式确定最小成本路径的形式基础。*I*EEE交易系统科学与控制论，4(2)：100-107，1968年。

[12] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. 语言模型作为零射击规划器：提取行动知识用于具体化代理，2022年。

[13] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch,Y. Chebotar, et al. 内心独白：通过语言规划进行具体化推理。ar*Xiv*预印本 ar*Xiv:2207.05608*，*2022*年。

[14] J. Jung, L. Qin, S. Welleck, F. Brahman, C. Bhagavatula, R. L. Bras, and Y. Choi. Maieutic提示：逻辑一致的递归解释推理。 *arXiv*预印本 *arXiv:2205.11822*，2022年。

[15] D. Kahneman. 《思考，快与慢》 . Macmillan, 2011.

[16] D. Kahneman, S. Frederick, et al. 重温代表性：直觉判断中的属性替代 .启发式与偏见：直觉判断的心理学, 49(49-81):74, 2002.

[17] G. Kim, P. Baldi, and S. McAleer. 语言模型可以解决计算机任务, 2023.

[18] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. Llm+p: 用最佳规划能力赋予大型语言模型能力, 2023.

[19] X. Lu, S. Welleck, P. West, L. Jiang, J. Kasai, D. Khashabi, R. L. Bras, L. Qin, Y. Yu,R. Zellers, N. A. Smith, and Y. Choi. 类似神经网络的解码：带有前瞻启发的约束文本生成. 在北美计算语言学协会, 2021年.

[20] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Welleck, B. P. Majumder, S. Gupta, A. Yazdanbakhsh, 和 P. Clark. 自我完善：带有自我反馈的迭代改进，2023年。

[21] A. Newell, J. C. Shaw, 和 H. A. Simon. 一般问题解决程序的报告。 在 *IFIP* 大会，第256卷，第64页。匹兹堡，宾夕法尼亚州，1959年。

[22] A. Newell, H. A. Simon, 等。人类问题解决。普林斯顿大学出版社，1972年。

[23] OpenAI. Gpt-4技术报告。 *ArXiv*，abs/2303.08774，2023年。

[24] D. Paul, M. Ismayilzada, M. Peyrard, B. Borges, A. Bosselut, R. West, 和 B. Faltings. Refiner: Reasoning feedback on intermediate representations, 2023.

[25] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, 等. 通过生成式预训练改进语言理解。*O*penAI 博客, 2018.

[26] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, 等. 语言模型是无监督的多任务学习者。*O*penAI 博客, 1(8):9, 2019.

[27] I. Schlag, S. Sukhbaatar, A. Celikyilmaz, W. tau Yih, J. Weston, J. Schmidhuber, 和 X. Li. 大型语言模型程序, 2023.

[28] N. Shinn, B. Labash, 和 A. Gopinath. Reflexion: 具有动态内存和自我反思的自主代理，2023年。

[29] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, 等。在没有人类知识的情况下掌握围棋。自然, 550 (7676):354–359, 2017年。

[30] S. A. Sloman. 两种推理系统的实证案例。心理学公告, 119(1): 3, 1996年。

[31] K. E. Stanovich.谁是理性的？ 关于推理的个体差异研究。心理学出版社, 1999年。

[32] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, 等。Llama: 开放高效的基础语言模型。 *arXiv* 预印本 *arXiv:2302.13971*, 2023年。

[33] S. Verma, J. Fu, S. Yang, and S. Levine. Chai: 一个用于任务导向对话的聊天机器人AI，通过离线强化学习进行训练。 在*2022*年北美计算语言学协会会议论文集中，人类语言技术的论文中，页码为4471-4491，*2022*年。

[34] E. Wallace, N. Tomlin, A. Xu, K. Yang, E. Pathak, M. Ginsberg, and D. Klein. 自动化的填字游戏求解。arXiv预印本 arXiv:2205.09665，2022年。

[35] L. Wang, W. Xu, Y. Lan, Z. Hu, Y. Lan, R. K.-W. Lee, and E.-P. Lim. 计划和解决提示：通过大型语言模型改进零-shot思维链推理，2023年。

[36] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, and D. Zhou. 自洽性提高了语言模型的思维链推理。arXiv预印本 arXiv:2203.11171，2022年。

[37] Z. Wang, S. Cai, A. Liu, X. Ma, and Y. Liang. 描述、解释、计划和选择：大型语言模型与开放世界多任务代理的交互式规划，2023年。

[38] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. 思维链提示引发了大型语言模型的推理。arXiv预印本 arXiv:2201.11903，2022年。

[39] Y. Xie, K. Kawaguchi, Y. Zhao, X. Zhao, M.-Y. Kan, J. He, and Q. Xie. 分解通过自我评估引导解码增强了推理能力，2023年。

[40] S. Yang, O. Nachum, Y. Du, J. Wei, P. Abbeel, and D. Schuurmans. 基于决策的基础模型：问题、方法和机会，2023年。

[41] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. ReAct：在语言模型中协同推理和行动。arXiv预印本 arXiv:2210.03629，2022年。

[42] S. Zhang, Z. Chen, Y. Shen, M. Ding, J. B. Tenenbaum, and C. Gan. 使用大型语言模型进行代码生成的规划。 在第十一届国际学习表示会议，2023年。 网址https://openreview.net/forum?id=Lr8cOOtYbfL。

[43] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, 等. 最少到最多提示使大型语言模型能够进行复杂推理。 arXiv预印本 arXiv:2205.10625, 2022年。

[44] X. Zhu, J. Wang, L. Zhang, Y. Zhang, R. Gan, J. Zhang, 和 Y. Yang. 通过合作推理引导的语言模型解决数学问题。arXiv预印本 arXiv:2210.16257,2022年。

代码、提示、轨迹

所有代码可在https://github.com/princeton-nlp/tree-of-thought-llm找到。

所有提示可在https://github.com/princeton-nlp/tree-of-thought-llm/tree/master/src/tot/prompts找到。

轨迹可在https://github.com/princeton-nlp/tree-of-thought-llm/tree/master/logs找到。

B 附加实验结果

鉴于探索和扩展语言模型的能力边界的动机，我们在主要论文中的实验集中在与最先进的语言模型(GPT-4)和三个旨在挑战它的困难任务的设置上。在这里，我们报告了使用较弱的LLM或更容易的任务进行的附加实验，并讨论了成本和效率。

	GSM8K	StrategyQA
IO	51	73
CoT	86	82
ToT	**90**	**83**

表4：零-shot ToT和GPT-4的新任务。

	GPT-4	GPT-3.5
IO	7.3%	6%
CoT	4.0%	3%
ToT	**74%**	**19%**

表5：GPT-4与GPT-3.5的24点游戏。

	GPT-4	GPT-3.5
IO	6.19	4.47
CoT	6.93	5.16
ToT	**7.56**	**6.62**

表格 6：GPT-4 与 GPT-3.5 的创意写作对比。

B.1 对新任务（GSM8k、StrategyQA）的扩展，使用零-shot ToT

虽然更常见的自然语言处理任务对于 GPT-4 来说可能太容易了，不需要 ToT（这就是为什么我们考虑了更难的新任务），但我们相信将 ToT 应用于新任务可能是直接的。例如，我们为 GSM8K 和 StrategyQA 实现了一个简单且通用的零-shot ToT-BFS，类似于创意写作（选择 5 个问题解决策略进行投票，然后根据最佳策略选择 5 个解决方案进行投票），只需添加几行额外的代码：

```
# 定义新任务的答案格式
gsm8k_format = '“答案是 n”，其中 n 是一个数字'
strategyqa_format = '答案要么是“是”，要么是“否”'

# 定义零-shot io提示
standard_prompt = ‘使用{format}回答以下问题：{input}’

# 为零-shot cot和零-shot tot定义思维格式
cot_prompt = ‘‘‘回答以下问题：{input}

制定策略，然后写作。 你的输出应该符合以下格式：

策略：
你回答问题的策略。

回答：
你对问题的回答。 它应该以{format}结尾。
’’’

# 定义用于零-shot tot的零-shot投票
vote_prompt = ‘‘‘给出一条指令和几个选择，
决定哪个选择最有前途。
详细分析每个选择，然后在最后一行得出结论
"最佳选择是{s}，其中s是选择的整数id。
’’’
```

我们在100个随机的GSM8K测试和StrategyQA开发问题的子集上进行了评估。 如表4所示，并且如预期的那样，ToT在两个任务上都优于CoT（但只是稍微优于GPT-4+ CoT在这些任务上已经非常好，而StrategyQA的瓶颈是外部知识，而不是推理）。 考虑到计算成本，更适合尝试较小的LLMs + ToT来处理传统的NLP任务，或者使用GPT-4 + ToT来处理挑战GPT-4 + CoT推理能力的困难任务。

B.2 对新的语言模型的扩展（GPT-3.5）

为了了解ToT如何与其他LLM一起工作，我们还运行了GPT-3.5-turbo进行创意写作（表6）和24点游戏（表5）的实验。 在这两个任务上，“ToT $>$ CoT $>$IO”对于GPT-3.5仍然成立。在创意写作方面，我们发现GPT-3.5+ToT的表现优于GPT-4+IO，并且与GPT-4+CoT类似，这表明ToT在较弱的语言模型上也能很好地工作。

在24点游戏中（我们将1-shot的提示改为3-shot以使其生效），GPT-3.5+ToT的19%远远不如GPT-4+ToT的74%。 为了进一步了解生成与评估的重要性，我们进行了GPT-4生成+GPT-3.5评估（64%）和GPT-3.5生成+GPT-4评估（31%）的实验。 这表明游戏的瓶颈在于思维生成，而不同的生成/评估语言模型可能会在降低成本的同时获得不错的结果。

B.3 成本和效率

运行ToT需要比IO或CoT更多的计算。 例如，在24点游戏（下表7），使用ToT解决一个问题需要5.5k个完成标记，接近100次CoT试验（6.7k个标记）。 但是ToT的性能优于100次独立CoT试验的最佳结果。

24的游戏	生成/提示标记	每个案例的成本	成功率
IO（100次中的最佳结果）	1.8k / 1.0k	$0.13	33%
CoT（100次中的最佳结果）	6.7k / 2.2k	$0.47	49%
ToT	5.5k / 1.4k	$0.74	74%

表7：24点游戏的成本分析。

在创意写作（下表8），我们发现ToT需要大约5倍的完成标记和金钱成本，这是合理的，因为$b = 5$，并且大部分标记是生成的段落。

创意写作	生成/提示标记	每个案例的成本
IO	0.9千 / 0.4千	$0.06
CoT	0.9千 / 0.4千	$0.07
ToT	4千 / 2.9千	$0.32

表8：Game of 24的成本分析。

因此，完成Game of 24和创意写作的主要ToT实验的成本约为 $0.74 \times 100 + 0.32 \times 100 = 106$ 美元。 填字游戏的DFS实验的成本也应该在 100美元以内。 总的来说，ToT的成本和效率高度依赖于所使用的提示和搜索算法，并且可能需要比CoT生成的令牌多5-100倍。一些可行的见解：

- 我们建议在需要有意识推理的任务上使用ToT，而CoT则表现不佳。
- ToT的灵活性允许在性能和成本之间进行一些权衡，例如在BFS中更改波束大小或投票数量，few-shot vs. zero-shot提示，GPT-3.5 vs. GPT-4等。可以根据一些资源限制或性能目标配置设置。
- 有很大的提升效率的空间，例如，当找到解决方案时，BFS可以提前停止，或者在一些思考是“不可能”的情况下减小波束大小。
- 我们相信，为了使模型达到更强的智能，确实需要更多的计算量，而这不应该成为一个阻碍，因为从长远来看，（开源）语言模型将变得更便宜和更高效。 这也是一个更好地训练/微调语言模型进行思维生成和/或评估的方向。