# 树状分层反馈生成架构：实现大模型边思考边回答的探索

魏翔

**摘 要：**

为促使大语言模型在生成过程中实时自我修正，实现真正的“边思考边回答”，本文提出一种树状分层反馈生成架构（Tree-structured Hierarchical Feedback Generation, THFG）。该架构通过树状分层生成、轻量满意度评估、反馈信号上推、渐进保守策略、随机激活自审、极细切分与相邻配对审阅六个核心模块，在尽量不提升计算开销的前提下实现多层树结构、点间反馈与模型预防性修正。实验基于 Qwen2.5-14B 模型进行微调验证，结果表明 THFG 在 GSM8K、HellaSwag 数学推理任务上取得显著提升。本文旨在为下一代大模型推理架构的设计提供一种可能的路径。

**关键词：**大语言模型；思维链；树状推理；反馈机制；自我修正

## 一、引言

自 2022 年思维链（Chain-of-Thought, CoT）[1]被提出以来，大语言模型的推理能力经历了飞跃式提升。通过显式思考，模型在数学、逻辑、规划等任务上的表现显著改善。此后的几年间，树状搜索[3]、图推理[4]、自洽性采样[2]、并行展开[5]等方法层层叠加，推理拓扑从“链”演进到“树”，再到“图”，越来越接近人类思考的复杂结构。

但有一个根本性的问题始终悬而未决：当前的推理架构，无论是 CoT 还是 Tree-of-Thoughts，本质上都是“先思考、后回答”，模型先把推理过程完整走完，再输出答案。思考阶段和生成阶段是严格串行的两个步骤，中间的界限不可逾越。

这种二段式架构在面对长文本生成任务时暴露出三个深层缺陷：第一，“过度思考”。模型在思考阶段需要预先推演整个回答的框架，但长文本的走向往往难以在事前完整预测。为了覆盖所有可能性，模型倾向于“思考得太多”，生成篇幅远长于甚至是脱离实际需要。第二，“思考－回答脱节”。一旦进入回答阶段，思考的结果便已凝固。如果在回答过程中发现某个方向不可行，模型没有机制回到思考阶段进行调整，它只能继续输出，导致功能降级。第三，“盲区一致”。模型的思考与回答共享同一组参数，这意味着思考阶段忽略的盲区，在回答阶段同样会被忽略。没有外部视角的介入，模型无法发现自己“不知道自己不知道什么”。

本文试图回答的核心问题是：能否设计一种架构，让模型在生成过程中持续进行轻量级的自我审视，在发现偏差时将信号沿生成树向上传播，在未展开的节点上提前修正，从而实现真正的边思考边回答。

## 二、推理拓扑的演进

自 2022 年思维链把隐性推理转化为显性文本，让模型在生成时显式看到中间推理步骤开始。推理拓扑的演进呈现出从简单线性走向复杂图状、从单向生成转向动态修正的清晰脉络。早期思维链及自洽性采样依赖严格线性或多链投票，缺乏探索与修正能力；随后思维树与思维图将拓扑升至二维，支持多路径搜索却受限于“先思后答”的二段式架构；近年的并行与分层工作提升了生成效率与交错控制，但修正多局限于影响未来或层级间的规划适应。直至 2025—2026 年，反馈修正机制成为焦点，LPSR、RemeDi[9]等探索了快照回退与局部重写，而 Streaming-VR[15]、SPOC[8]等则开启了流式验证与预防性修正的新方向。尽管如此，现有方案仍未形成 THFG 所设想的“分层树+点间反馈+预防性修正”三合一完整闭环，跨层级反馈向上传播与未展开节点的预防性干预仍是核心短板。

表 1 推理拓扑演进历程

| 时间 | 工作 | 推理拓扑 | 核心机制 | 边生成边修正能力 |
|---|---|---|---|---|
| 2022.01 | CoT [1] | 链 | 显式写出中间推理 | ✗ |
| 2023.03 | Self-Consistency [2] | 多链采样 | 多数投票聚合 | ✗ |
| 2023.05 | ToT [3] | 树 | BFS/DFS 搜索+剪枝 | ✗ |
| 2023.07 | SoT [5] | 一层骨架+并行叶子 | 分阶段并行展开 | ✗ |
| 2024.08 | GoT [4] | 有向无环图 | 多路径汇聚 | ✗ |
| 2025.06 | SPOC [8] | 链+交错验证 | 单轮自发性自纠正 | ✓（流式修正） |
| 2025.12 | FlyThinker [7] | 双模型并行流 | 推理－生成同步 | 半✓（只能影响未来） |
| 2025.12 | RemeDi [9] | 扩散迭代 | 再掩码重写 | ✓（token 级局部修正） |
| 2025.12 | Streaming-VR [15] | 流式验证 | 实时验证+修正 | ✓（流式/预防性修正） |
| 2026.02 | IADT [6] | 交替分析起草 | 交错阶段 | ✗ |
| 2026.02 | STAR-LDM [14] | 扩散→自回归 | 先扩散思考再自回归 | ✗ |
| 2026.03 | WriteHERE [17] | 递归任务分解 | 异构规划+动态集成 | 半✓（规划自适应） |

一个清晰的演进规律浮现出来：推理拓扑从链到树到图，越来越接近人类的多路径探索；修正能力始终是短板，现有方案要么不能修正，要么只能“回到过去重做”或做 token 级局部修正；缺少一个关键机制，反馈在层级之间向上传播，在未展开节点上预防性修正。

## 三、横向对比

将前沿方案放在同一张平面上比较，可以清晰地显示出一个架构空白：没有任何一个现有方案——包括最新的 WriteHERE[17]和 SPOC[8]——同时具备“多层树结构+点间反馈+预防性修正”。而这个组合，恰恰是本文设想的核心。

表 2 与现有方案的横向对比

| 维度 | CoT/ToT/GoT | SoT | FlyThinker | RemeDi[9] | SPOC[8] | THFG |
|---|---|---|---|---|---|---|
| 推理拓扑 | 链/树/图 | 一层+叶子 | 双流并行 | 扩散迭代 | 链+交错验证 | 多层递归树 |

| 思考与生成时序 | 串行 | 阶段并行 | 完全并行 | 迭代 | 串行+验证 | 分层交替 |
|---|---|---|---|---|---|---|
| 点间反馈 | 无/仅回溯 | 无 | 仅前向 | token 级 | 交错纠正 | 上推+预防性修正 |
| 修正方向 | —/剪枝 | — | — | 重写当前 | 纠正当前 | 修正未来 |
| 满意度评估 | 无/评估函数 | 无 | 无 | 置信度 | 自发检测 | 文本信号+随机审阅 |

## 四、架构设想：树状分层反馈生成

THFG 目前处于概念设计阶段，部分模块有明确的工程路径，部分需要进一步研究。下文描述的是架构蓝图。THFG 的整体架构如图 1 所示。

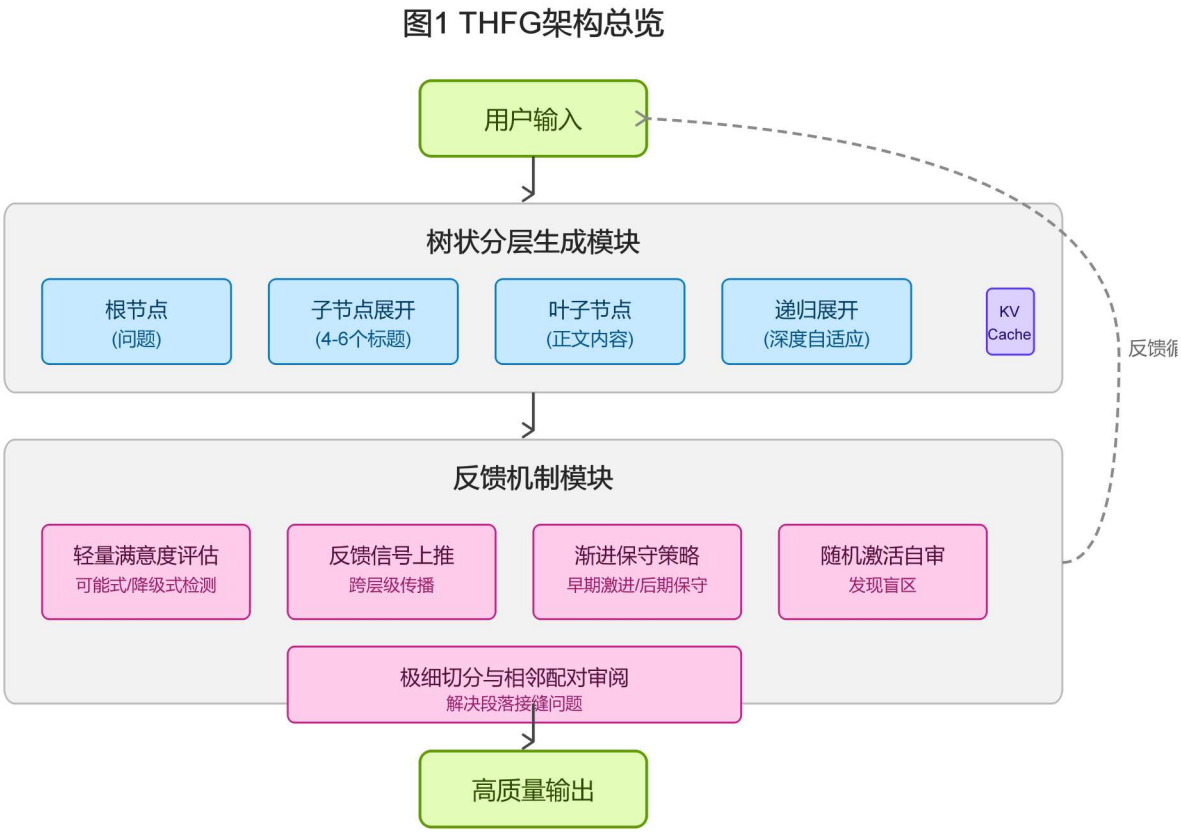


图 1 THFG 架构总览

### 4.1 概述结构

THFG 采用树状分层生成。根节点为用户的问题，第一轮生成不直接输出答案，而是生成 4-6 个子节点标题。子节点展开时，每个子节点独立展开为更下一层的节点或正文内容，展开过程使用父节点的 KV-Cache 快照作为初始状态。叶子节点的展开可以继续递归，递归深度由模型根据内容复杂度自行判断。模型自主评价节点展开过程中自然流露的不确定信号，同时随机激活自审，根据评估结果，反馈上

推并对未展开节点修正。树状分层生成结构如图 2 所示。

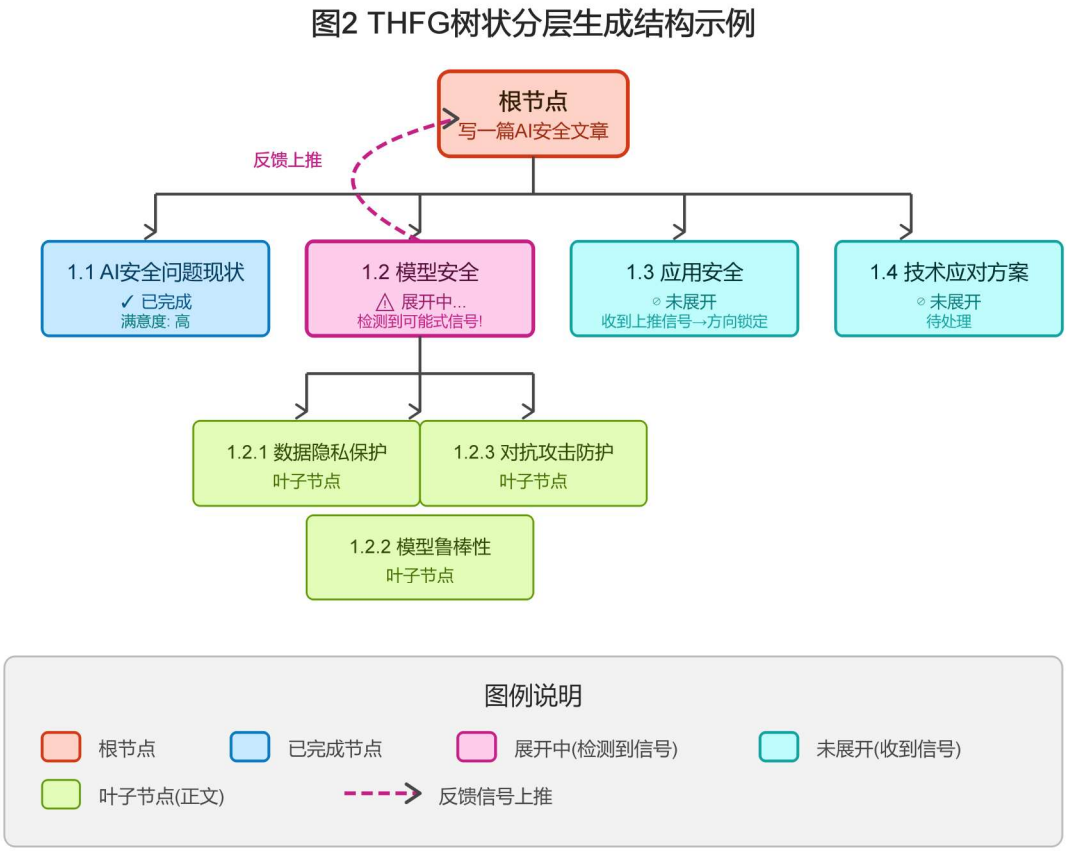


图 2 THFG 树状分层生成结构示例

### 4.2 关键技术一：轻量满意度评估

要让模型知道自己在某个节点展开得不好，需要一套评估机制。传统方案要么需要额外的评估模型，要么需要模型自己给自己打分，前者增加计算开销，后者存在“模型高估自己”的偏向。THFG 采用一种不同的策略：不主动评估质量，而是检测模型在生成过程中自然流露的不确定信号。

**信号一：可能式语句**

当模型说出用户可能想要的是、也许可以这样理解、这可能意味着、或许是、一种可能是、不确定是否等表达时，它在暴露一个事实：它拿不准当前这个节点应该往哪个方向走。已有研究[10][11]表明，LLM 在长文本生成中显式表达不确定性的现象是可检测、可分类的。THFG 不试图消除这些表达，而是把它们当作信标：出现可能式语句 → 当前节点方向模糊 → 触发反馈上推。

检测本身是字符串级别的模式匹配，在生成过程中零额外计算开销。维护一个低满意度关键词表，每命中一次扣一分，累计超过阈值即触发反馈。

**信号二：降级式语句**

当模型说出我这里采用一种简易的方法来实现、由于篇幅限制，这里简化、用近似值代替、这里就不一一列举了、简化处理时，表达了另一种性质的问题：模型知道应该怎么做，但当前条件下做不到。

这类信号的信息量比可能式更高——它不仅暴露了问题，还间接说明了问题原因，两种信号触发不同的上推策略。

检测的实时性优势：因为是基于 token 的在线检测，反馈信号可以在节点展开完成之前就发出，不需要等整个节点生成完毕。这使得 THFG 可以在生成中途就触发修正，比事后评估更及时。

### 4.3 关键技术二：反馈上推与未展开节点修正

这是 THFG 区别于所有现有方案的核心机制。每个节点生成完成后，保存其父节点的 KV-Cache 快照。满意度评估检测到信号后，根据信号类型决定上推层级。

表 3 信号上推规则

| 信号类型 | 暴露的问题 | 上推目标 | 修正内容 |
|---|---|---|---|
| 「可能式」 | 方向模糊 | 上推 1~2 级 | 锁定未展开兄弟节点的方向约束 |
| 「降级式」 | 能力不足 | 上推 1~2 级 | 调整未展开兄弟节点的深度/粒度 |
| 两者叠加 | 节点策略本身有问题 | 上推至根级 | 重新规划整棵子树 |

例如，当“模型安全”节点的展开过程中检测到多个“可能式”信号，触发信号上推，其父节点收到信号后，将“应用安全”的方向约束收紧，同时标记尚未展开的节点，提醒它们在后续展开中注意衔接。

与传统方案的关键区别：与 SPOC[8]的流式自纠正不同，SPOC 在单一线程中交错生成与验证，THFG 将这种实时修正推广到树状结构——在层级之间建立反馈通道，使得修正的影响可以覆盖整棵子树的所有未展开节点。

### 4.4 关键技术三：渐进保守策略

为控制成本，THFG 的反馈激进程度与树位置相关：越靠近根节点越容易触发上推修正，越靠近叶子越保守。早期阶段（树展开度<30%），直接上推修改主干；中期阶段（30%—70%），优先修改叶子节点；后期阶段（>70%），原则上绝不动主干，采用标记延迟处理策略。

这个策略借鉴了软件工程中的代码冻结思想：临近发布，不再做大改动。在长文本生成中，前半段的结构决定 80%的质量，优先确保前半段的方向正确；后半段允许更多本地的调整，而不是全局修正。

### 4.5 关键技术四：随机激活自审

模型最大的盲区恰恰是它意识不到自己意识不到什么。现有大模型架构中，推理时并非所有参数都参与计算。以 Mixture-of-Experts（MoE）[13]为例，每次前向传播只激活一小部分专家。因此 THFG 设想在节点展开完成后，随机激活一小部分未使用的参数，让这些参数以审阅者身份看一遍刚才生成的内容。因为激活是随机的，它可能关注到主生成路径完全忽略的角落，从而撞上模型自己没意识到的盲区。由此构成“自检+他审”双层结构。

### 4.6 关键技术五：极细切分与相邻配对审阅

THFG 建议将叶子节点的粒度控制在自然段甚至更小的范围内。但极细切分导致节点之间的接缝可能断裂，大部分文本质量问题发生在段落的交界处。THFG 引入相邻配对审阅来解决接缝问题：在完成每个子节点的生成后，将当前节点与上一个已完成的兄弟节点拼接，对拼接后的双节点组合做一次审阅。

## 五、实验验证

### 5.1 实验设置

为验证 THFG 架构的有效性，作者在 Qwen2.5-14B-Base 模型上使用按照论文生成的数据集进行微调实验。该数据集按 THFG 方法论构建包含 1024 条逻辑推理极简样本，样本本身数据较少，主要体现涵盖树状分层生成、反馈上推、渐进保守等核心机制的训练信号，微调采用 LoRA 方式进行。

### 5.2 实验结果

微调模型在六个标准评测集上的表现如表 4 所示。

表 4 微调模型评测结果

| 评测数据集 | 基准值（基础模型官方） | 微调后值 |
|---|---|---|
| C-Eval | — | 80.09 |
| ARC | 67.3(ARC-C) | 83.19 |
| GSM8K | 90.2 | 93.03 |
| BBH | 78.2 | 75.93 |
| MMLU | 79.7 | 70.27 |
| HellaSwag | — | 84.96 |

### 5.3 结果分析

THFG 微调模型在 GSM8K 上取得了显著提升。GSM8K 提升 2.83 个百分点，表明反馈上推机制在多步数学推理中发挥了正向作用，微调后 HellaSwag 接近 Qwen2.5-32B-Base 模型（85.2），ARC、C-Eval 暂无可比对象。

同时应注意，MMLU 下降 9.43 个百分点、BBH 下降 2.27 个百分点，分析认为训练样本以逻辑推理为主，且数据极少，对多领域百科知识问答任务的覆盖不足，导致通用知识评测出现性能下降。

## 六、讨论

### 6.1 计算开销分析

THFG 的计算开销预计主要体现在五个方面：快照存储（约 120MB）、随机激活审阅（开销在 5%以内）、相邻配对审阅（开销约 3%—5%）、在线信号检测（可忽略不计）、重新展开（仅在触发修正时发生）。总体估计，THFG 相较于传统自回归生成的额外开销在 15%—35%之间，换来的是文本质量和内在一致性的显著提升。

### 6.2 局限与开放问题

THFG 的设计主要面向长文本生成任务，在数学推理、代码生成等需要严格串行依赖的任务上的适用性尚未验证。子节点独立展开可能导致的内容冗余、信号上推的衰减策略、随机激活的种子选择等问题仍需进一步研究。目前该研究仅限于初步实验，更系统的验证留待工程实现后进行。

## 七、结语

THFG 是 CoT[1]到 WriteHERE[17]、SPOC[8]这条“思考与生成逐渐融合”趋势在多层树状结构上的自然延伸。它尝试填补了“多层树结构+点间反馈+预防性修正”这一架构空白——这是任何现有方案均未同时覆盖的三合一组合，作者受限于设备，无法开展进一步研究，欢迎各位研究者验证和交流。

THFG 的六个核心模块构成了一个环环相扣的反馈闭环：满意度评估发现问题，信号上推传播问题，未展开节点修正预防问题，渐进策略控制修正成本，随机审阅发现盲区补全闭环，极细切分+配对审阅在结构层面封堵接缝漏洞。

THFG 的核心哲学可以概括为一句朴素的话：不在出口检查质量，而在过程中修正方向。这或许也是人类思考的本质——我们不是在写完之后才回头看，而是在写的过程中不断抬头，问自己：这是不是我要去的地方？不是的话，后面怎么走才能绕回来？

## 参考文献


[1] Wei J, Wang X, Schuurmans D, et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models[C]//NeurIPS, 2022.

[2] Wang X, Wei J, Schuurmans D, et al. Self-Consistency Improves Chain of Thought Reasoning in Language Models[C]//ICLR, 2023.

[3] Yao S, Yu D, Zhao J, et al. Tree of Thoughts: Deliberate Problem Solving with Large Language Models[C]//NeurIPS, 2023.

[4] Besta M, Blach N, Kubicek A, et al. Graph of Thoughts: Solving Elaborate Problems with Large Language Models[C]//AAAI, 2024.

[5] Ning X, Lin Z, Zhou Z, et al. Skeleton-of-Thought: Prompting LLMs for Efficient Parallel Generation[C]//ICLR, 2024.

[6] Wu S, Li B, Wang X, et al. Synergizing Understanding and Generation with Interleaved Analyzing-Drafting Thinking[C]//ICLR, 2026.

[7] Wang C, Zhang Y, Wang W, et al. Think-While-Generating: On-the-Fly Reasoning for Personalized Long-Form Generation[C]//ICLR, 2026.

[8] Zhao X, Xu T, Wang X, et al. SPOC: Boosting LLM Reasoning via Spontaneous Self-Correction[C]//ICML, 2025.

[9] Huang Z, Wang Y, Chen Z, et al. Don't Settle Too Early: Self-Reflective Remasking for Diffusion Language Models[C]//ICLR, 2026.

[10] Yang R, Zhang C, Zhang Z, et al. LoGU: Long-form Generation with Uncertainty Expressions[C]//ACL, 2025: 18947-18968.

[11] Liu G K M, Yona G, Caciularu A, et al. MetaFaith: Faithful Natural Language Uncertainty Expression in LLMs[C]//EMNLP, 2025.

[12] Zhang W, Shen Y, Wu L, et al. Self-Contrast: Better Reflection Through Inconsistent Solving Perspectives[C]//ACL, 2024: 3602-3622.

[13] Fedus W, Zoph B, Shazeer N. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity[J]. JMLR, 2022, 23(120): 1-39.

[14] Lovelace J, Belardi C, Zalouk S, et al. Stop-Think-AutoRegress: Language Modeling with Latent Diffusion Planning[C]//COLM, 2025.

[15] Ko J, Baek J, Hwang S J. Real-time Verification and Refinement of Language Model Text Generation[C]//arXiv:2501.07824, 2025.

[16] Xiong R, Chen Y, Khizbullin D, et al. Beyond Outlining: Heterogeneous Recursive Planning for Adaptive Long-form Writing with Language Models[C]//EMNLP, 2025.