
Parason: Revealing Subtask- and Trial Parallelism in LLM Reasoning

Anonymous Author(s)

Affiliation

Address

email

Abstract

Scaling test-time reasoning has substantially improved the problem-solving ability of large language models (LLMs), but standard autoregressive decoding still executes long reasoning traces sequentially, creating severe latency for difficult tasks (up to days and weeks). Parallel reasoning offers a natural remedy. However, prior systems primarily focus on *Subtask Parallelism*, where the model learns to decompose a high-level task into smaller chunks that can be solved independently. This approach overlooks another pervasive form of parallelism: *Trial Parallelism*, where multiple speculative attempts explore, verify, and aggregate competing hypotheses in parallel. In this paper, we introduce **Parason**, which reveals and learns both forms of parallelism in LLM reasoning. Our analysis identifies Trial Parallelism as the majority of parallelizable reasoning computation (65.5% in DeepSeek-V4’s reasoning steps in HLE), and it becomes increasingly dominant on hard problems. Guided by this taxonomy, Parason converts sequential reasoning traces into structured parallel trajectories with a context-free grammar, then trains models with Parallelism-Aware Group Relative Policy Optimization (PA-GRPO), whose reward jointly balances accuracy, latency, and the two parallelism ratios. At inference time, Parason executes the learned parallel structure through tool calls, translating theoretical savings to real-world wall-clock acceleration. Experiments on mathematical reasoning benchmarks including AIME24 and AIME25 show that Parason achieves **an average acceleration about $1.7\times$** while maintaining competitive accuracy.

1 Introduction

Recent advances in large language model (LLM) reasoning have made a simple recipe increasingly effective: spend more test-time compute, and obtain better answers. Chain-of-Thought prompting [Wei et al., 2022] and reasoning-oriented systems such as OpenAI o1 [OpenAI et al., 2024] and DeepSeek-R1 [DeepSeek-AI et al., 2025] show that long, deliberate reasoning traces can substantially improve performance on mathematical and logical tasks. Yet this recipe has an immediate systems bottleneck. Standard autoregressive decoding serializes every token and every intermediate thought, so stronger reasoning often translates directly into longer waiting time. On difficult problems, reasoning traces can grow to hundreds of thousands or even millions of tokens [Hubert et al., 2025, Luong et al., 2025], making the usual “think longer” strategy impractical for interactive agents, coding assistants, and other latency-sensitive applications. For example, Google’s AlphaProof [DeepMind, 2024] requires up to *three days* to solve challenging problems from the International Mathematical Olympiad (IMO) 2024, highlighting the extreme latency inherent in current sequential reasoning systems.

A natural response is to make reasoning parallel: let multiple reasoning branches proceed concurrently instead of decoding every thought in sequence. Existing methods explore this direction

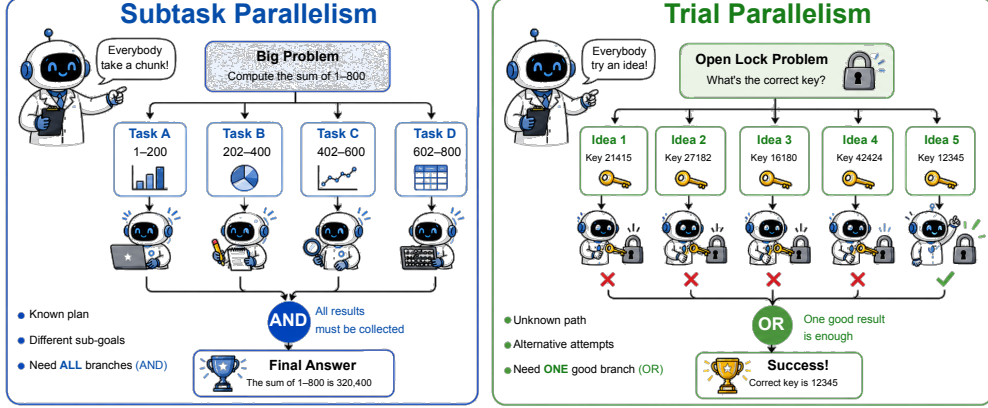


Figure 1: **Overview of two types of parallelism.** The figure contrasts Subtask Parallelism, where all decomposed branches are needed for the final answer, with Trial Parallelism, where multiple uncertain paths are explored as alternative (OR) branches. Despite this OR relation, the outputs of all Trial branches are concatenated into the subsequent context, making every branch’s content available for final synthesis. While previous work mainly focuses on Subtask Parallelism, our analysis shows that Trial Parallelism accounts for the majority of parallelizable reasoning on HLE for every model we study, and exceeds 58% for DeepSeek-R1 and DeepSeek-V4 across both datasets.

through independent sampling, such as self-consistency and Best-of-N [Wang et al., 2022, Hu et al., 2026], or adaptive branching frameworks [Yang et al., 2025b, Jin et al., 2025, Zheng et al., 2025, Lian et al., 2025, Wang et al., 2025]. However, methods such as Multiverse [Yang et al., 2025b], ThreadWeaver [Lian et al., 2025], and APR [Pan et al., 2025] mainly exploit *subtask-style* parallelism: splitting a high-level task into smaller chunks and merging their results. Trial-style exploration, which tries uncertain paths and incorporates diverse trajectories, is less discussed.

In this work, we argue that the key missing piece is a semantic taxonomy of parallel reasoning. As illustrated in Figure 1, we identify two complementary forms of parallelism. **Subtask Parallelism** applies when a problem can be decomposed into independent steps: each branch solves a distinct sub-goal, and the final answer combines all branch results. This is the form emphasized by most prior adaptive parallel systems. **Trial Parallelism** applies when the model is uncertain about which path will work: multiple branches test competing hypotheses, and the final trajectory keeps the useful results. These modes have different execution semantics. Subtask branches are usually all necessary, while trial branches are speculative and most useful when the reasoning path is uncertain. Our empirical analysis shows that this distinction is practical, not merely conceptual. Trial Parallelism accounts for the majority of analyzed parallelizable steps on Humanity’s Last Exam for every model we study, including 73.8% and 65.5% for DeepSeek-R1 and DeepSeek-V4, respectively. On OpenMath, it remains the majority for most models and reaches 68.5% and 58.1% for the same two open-source models. This suggests that hard reasoning is not only decomposition, but also trying and refining uncertain solution paths. Systems that focus solely on subtask decomposition therefore miss much of the computation in hard cases.

Guided by this observation, we introduce **Parason**, a training-and-inference codesign framework for revealing and exploiting both Subtask Parallelism and Trial Parallelism. Parason converts sequential reasoning traces into structured parallel trajectories without changing their final semantics, and uses a context-free grammar to mark parallel regions, individual branches, and summaries in an engine-parseable format. We then train the model with Parallelism-Aware Group Relative Policy Optimization (PA-GRPO), which augments outcome-based reinforcement learning with rewards for lower latency along the longest token path and balanced use of the two parallelism types. At inference time, the generated structure is executed through tool calls, allowing Parason to dispatch independent workers with subtasks and competing trial branches. Across mathematical reasoning benchmarks including AIME24 and AIME25, Parason preserves competitive accuracy while reducing effective reasoning latency. Our contributions are summarized as follows:

- **A taxonomy of parallel reasoning.** We distinguish **Subtask Parallelism** and **Trial Parallelism**, identifying Trial Parallelism as the missing component. Based on this taxonomy,

Sequential Reasoning is Naturally Parallelizable – A Case of 24 Puzzle Solving

Prompt: Calculate 24 using 6 9 9 10

<think> We are given the numbers 6, 9, 9, 10 and we need to make 24 using basic arithmetic operations (+, -, *, /) and possibly parentheses. We need to find an expression that equals 24. Let's try different combinations: We have two 9's, one 6, and one 10. Goal: 24. Some common approaches: $24 = 4*6$, $24=30-6$, $24=2.4*10$, etc.

Since we have a 6, maybe we can get 4 from 9,9,10 to multiply by 6? $4*6=24$. How to get 4 from 9,9,10? $10 - 9 = 1$, then $9 - 1 = 8$, not 4. Or $(9+9)/10? =18/10=1.8$, not 4. Maybe $24 = 30 - 6$. So if we can make 30 from 9,9,10 and then subtract 6? But we have to use all numbers. So maybe $(9+9+10) = 28$, then $28 - 4?$ But we have 6. Alternatively, $(9*10)/?$ Not sure. Or $24 = (10-6)*6$, but we only have one 6. Maybe using fractions: $24 = 9/(something)$ etc. Let's try other combinations systematically: We can think of possible expressions: One idea: $10 * (9 - 6) = 10*3=30$, then $30 - 9 = 21$, not 24. $10 * (9 - 6) - 9 = 30-9=21$. $9 * (10 - 6) = 9*4=36$, then $36 - 9 = 27$, not 24. $(9+9)*(10-6)? =18*4=72$ too big. $(9+9-10)*6 = (8)*6=48$, too big.

Maybe divide: $(9*10)/(something) = 90/something$. $90/4 = 22.5$, not 24. Or $(9*10*9)/something$. $810/33.75 = 24?$ Not likely. Maybe using subtraction and division: $24 = (10*9)/(something) - something$. Let's try: $9*10 = 90$, $90/6=15$, then $15+9=24$! That works: $9*10/6+9 = 90/6+9=15+9=24$. But we have to use two 9's and one 10 and one 6. In that expression we used one 9, one 10, one 6, and then added the other 9. So it's $9*10/6+9$. But check order of operations: multiplication and division first: $9*10=90$, $90/6=15$, $15+9=24$. Yes.

So the expression is: $9 \times 10 \div 6 + 9 = 24$. we found a solution. But let's check if there are other ways: maybe $(9 - 6) \times (10 - \sqrt{9})$ but we cannot use square root unless specified. So basic operations only. Another: $(10 - 9/9)*?$ $(10-1)=9$, $9?$ Not 24. $(9+9+10-6)? =22$, not 24. $(9*9-10)/6?$ $(81-10)/6=71/6\approx11.83$, no.

So the one we found seems simplest. Thus, the answer is: $9 \times 10 \div 6 + 9 = 24$. </think>

One solution is: $9 \times 10 \div 6 + 9 = 24$

Figure 2: **Visualization of Trial and Subtask Parallelism.** The 24 Puzzle trace generated by DeepSeek-R1 [DeepSeek-AI et al., 2025] highlights **Subtask Parallelism in blue** and **Trial Parallelism in green**. Subtask branches implement divide-and-conquer execution, while Trial branches explore and verify competing hypotheses; *Trial steps occupy the majority of reasoning tokens*.

- we introduce Parason, which converts sequential reasoning data into grammar-constrained parallel trajectories that can be parsed and executed by inference engines.
- **Parallelism-aware reinforcement learning.** We propose PA-GRPO, a multi-objective RL objective that jointly optimizes answer accuracy, latency along the longest token path, and the model's use of the two parallelism modes.
 - **Empirical speedups with competitive accuracy.** Experiments across challenging math benchmarks demonstrate that Parason improves the latency–accuracy Pareto frontier, achieving an average acceleration about $1.7\times$ while maintaining competitive accuracy. Under latency-constrained settings, Parason matches the performance of an 8k-token latency budget using only 25% of the budget.
 - **Seamless inference engine support.** We define a CFG that marks parallel regions as tool calls and integrate Parason into SGLang [Zheng et al., 2023]. This makes parallel reasoning directly executable in a real inference engine, while prior work either reports only theoretical speedups that are hard to translate into actual latency reduction, or introduces complicated designs that require deep modifications to modern inference engines.

2 Related Work

Test-time scaling for LLM reasoning. Scaling inference-time computation has become a central recipe for improving LLM reasoning. CoT prompting [Wei et al., 2022] elicits intermediate derivations, while recent reasoning models such as OpenAI o1 [OpenAI et al., 2024] and DeepSeek-R1 [DeepSeek-AI et al., 2025] further improve performance by producing longer and more deliberate reasoning traces. This sequential scaling is powerful but expensive: autoregressive decoding generates every token one after another, so longer reasoning directly increases latency. Parallel reasoning is an attractive direction for preserving test-time compute while shortening the longest token path.

Independent parallel sampling and majority voting. A simple form of parallel reasoning is to sample multiple complete solutions and aggregate their answers. Self-consistency [Wang et al., 2022], verifier-guided Best-of-N sampling [Cobbe et al., 2021], and confidence-based variants such as DeepConf [Fu et al., 2025] improve robustness by exploring several candidate trajectories. These

Model	Dataset: OpenMath		Dataset: HLE	
	Subtask (%)	Trial (%)	Subtask (%)	Trial (%)
Open Source Models				
DeepSeek-R1 [DeepSeek-AI et al., 2025]	31.5	68.5 ^{+37.00}	26.2	73.8 ^{+47.60}
Qwen3-30B-A3B [Yang et al., 2025a]	43.6	56.3 ^{+12.70}	38.8	61.2 ^{+22.40}
MiniMax 2.7 [MiniMax AI, 2025]	47.7	52.3 ^{+4.60}	38.2	61.8 ^{+23.60}
Kimi-2.6 [Moonshot AI, 2025]	43.0	57.0 ^{+14.00}	42.1	57.9 ^{+15.80}
DeepSeek-V4 [DeepSeek-AI, 2026]	41.9	58.1 ^{+16.20}	34.5	65.5 ^{+31.00}
Commercial Models				
Gemini-2.5-Pro [Google, 2025]	62.9	37.1 ^{-25.80}	49.4	50.6 ^{+1.20}
Claude-Opus-4.5 [Anthropic, 2025b]	44.9	55.1 ^{+10.20}	29.9	70.1 ^{+40.20}
Gemini-3-Pro [Google DeepMind, 2025]	57.4	42.6 ^{-14.80}	46.9	53.1 ^{+6.20}
GPT-5.5 [OpenAI, 2026]	34.4	65.6 ^{+31.20}	23.9	76.1 ^{+52.20}

Table 1: **Trial Parallelism dominates on harder reasoning datasets.** We report Subtask and Trial ratios in reasoning traces from OpenMathReasoning [Moshkov et al., 2025] and Humanity’s Last Exam (HLE) [Phan et al., 2025]. Ratios are computed on the first 100 problems of each dataset, and annotation details are provided in Appendix A. For closed-source models, the original thinking trace is hidden, thus we counted summary statistics instead.

methods implicitly use trial-style computation, since different samples may try different solution paths. However, these samples are independent full traces. They do not share intermediate work, expose branch structure inside a trace, or tell an inference engine where parallel execution should begin and end. As a result, they improve accuracy at the cost of redundant computation with little control over latency.

Structured and adaptive parallel reasoning. Another line of work introduces explicit structures for parallel reasoning. Tree-of-Thoughts [Yao et al., 2023], Graph-of-Thoughts [Besta et al., 2023], Skeleton-of-Thought [Ning et al., 2023], and agentic decomposition methods split reasoning into trees, graphs, outlines, or sub-agents. More recent adaptive systems, including PASTA [Jin et al., 2025], Multiverse [Yang et al., 2025b], Parallel-R1 [Zheng et al., 2025], APR [Pan et al., 2025], ThreadWeaver [Lian et al., 2025], and PaCoRe [Hu et al., 2026], train or prompt models to create parallel branches and merge their outputs. PaCoRe further scales test-time compute with multiple rounds of coordinated parallel exploration and message passing. Figure 2 visualizes the same broader opportunity: *LLM reasoning can be parallelized*.

Yet their main mechanism is still *subtask-centric*: the model decomposes a high-level task into smaller chunks, runs the chunks in parallel, and merges the results. This is effective when the problem admits a clean divide-and-conquer structure, but it misses a common behavior in difficult reasoning: trying uncertain paths, rejecting failed attempts, and keeping the path that works. Our analysis shows that this missing *Trial Parallelism is not rare*; it accounts for most parallelizable reasoning steps in hard cases. Parason therefore separates **Subtask Parallelism** from **Trial Parallelism** and trains the model to exploit both forms explicitly.

3 Methodology

Parason turns long sequential reasoning traces into executable parallel programs, which is critical for long-horizon tasks such as Google’s AlphaProof [DeepMind, 2024], OpenAI’s IMO reasoning efforts [Luong et al., 2025], and recent agentic coding workflows such as Claude Code [Anthropic, 2025a] and OpenAI Codex [OpenAI, 2025]. The framework has three parts. First, we define two semantic forms of parallel reasoning: Subtask Parallelism decomposes a problem into independent required parts, while Trial Parallelism explores competing uncertain attempts and incorporates the diverse exploration into the final reasoning path. Second, we define a context-free grammar (CFG) that gives Parason a formal syntax, helps data generation, processing, and rollouts avoid syntax errors, and makes later system integration simple. Third, we propose Parallelism-Aware GRPO (PA-GRPO), a training algorithm that teaches the model to use parallelism effectively while preserving final-answer correctness.

Parallel Reasoning in Parason – A Case of 24 Puzzle Solving

```

Prompt: Calculate 24 using 6, 9, 9, 10.

<think> We are given the numbers 6, 9, 9, 10 and need to make 24 using basic arithmetic operations (+, -, *, *, /).
Goal: 24.

<Parallel>
<Outlines>
<Subtask> 1. Try to use 6 as a multiplier and derive 4 from 9, 9, 10. </Subtask>
<Subtask> 2. Try division with 9*10 and use the remaining 6 and 9. </Subtask>
<Subtask> 3. Verify whether there are simpler alternatives. </Subtask>

<Thread> 1.
<Parallel>
<Outlines>
<Trial> 1. Try use 9, 9, 10 to
make 4.  $10-9=1$ , then  $9-1=8$ , not 4.
This route does not find 4.
</Trial>
<Trial> 2. Try  $10*(9-6)-9$ . This
gives 21, not 24. This route does
not find 24. </Trial>
<Trial> 3. Try  $9*(10-6)-9$ . This
gives 27, not 24. </Trial>
<Trial> 4. Try  $(9+9)*(10-6)$ . This
gives 72, not 24. </Trial>
</Outlines>
</Parallel>
</Thread>

<Thread> 2.
<Parallel>
<Outlines>
<Trial> 1. Try  $9*10/6+9$ .  $90/6=15$ ,
 $15+9=24$ . This works. </Trial>
</Outlines>
</Parallel>
</Thread>

<Thread> 3.
<Parallel>
<Outlines>
<Trial> 1. Try  $(9-6)*(10-\text{sqrt}(9))$ .
Square root is not allowed.
</Trial>
<Trial> 2. Try  $9+9+10-6$ . Result
is 22, not 24. </Trial>
<Trial> 3. Try  $(9*9-10)/6$ . Result
is 11.83, not 24. </Trial>
</Outlines>
</Parallel>
</Thread>

</Outlines>
</Parallel>
Thus, the answer is  $9 \times 10 \div 6 + 9 = 24$ . </think>

```

Figure 3: **Solving the 24 Puzzle in Parason format.** This example is adapted from a real 24 Puzzle reasoning trace. The **blue** region gives the Subtask plan, splitting the problem into three independent subtasks. The **green** regions show Trial Parallelism, where each subtask explores candidate routes in parallel. The correct solution appears in the first Trial branch of the second subtask.

3.1 Parallelism in Reasoning

A reasoning trace is not uniformly sequential, as shown by the visualization of Trial and Subtask Parallelism in Figure 2. Some steps derive independent facts that will all be used later; other steps try uncertain ideas, explore multiple paths, and aggregate their findings. Following the overview and examples in Figures 1, 2, and 3, Parason treats these behaviors as two semantic forms of parallelism with different merge rules.

Subtask Parallelism is an *AND-branch* form of parallelism. A problem is split into independent sub-goals $g_1, \dots, g_k$, each branch computes a necessary result r_i , and the main trajectory aggregates all results $\{r_i\}_{i=1}^k$ to continue the derivation. These branches are parallel because they do not depend on one another, but all of them are mandatory: dropping one branch would remove the information required for generating the final answer. For example, a geometry solution may compute two distances independently before combining them in a final formula. This is the dominant form assumed by prior adaptive parallel systems [Yang et al., 2025b, Lian et al., 2025, Pan et al., 2025, Hu et al., 2026].

Trial Parallelism is an *OR-branch* form of parallelism. Given an uncertain state, the model launches competing attempts $a_1, \dots, a_k$ that test different hypotheses, heuristics, or solution routes. The merge step concatenates the branches to incorporate the exploratory trajectories into the reasoning history, rather than selecting a single branch. Thus, Trial Parallelism is parallel search inside a single reasoning trace, rather than decomposition into mandatory parts. This mirrors how strong reasoning models often behave on hard problems: they try a path, detect a dead end, and revise. Our empirical analysis in Table 1 shows that Trial Parallelism accounts for more than 50% of parallelizable steps for every model on HLE [Phan et al., 2025] and for most models on OpenMath, with higher Trial shares on HLE across all listed models.

3.2 Parallel Trajectory Format

To make these two modes executable, Parason represents reasoning traces with a grammar-constrained format. We extend ordinary `<think>` traces with parallel tags: `<Parallel>` marks a parallel region,

Context-Free Grammar (CFG) for Parason

Let $G = (V, \Sigma, R, S)$ be the context-free grammar for structured parallel reasoning.

Terminology

- **Terminals (Σ):** {<think>, <Parallel>, <Outlines>, <Subtask>, <Trial>, <Thread>, their matching closing tags, Text}
- **Non-terminals (V):**
 - S : start of reasoning
 - B : reasoning body
 - P : parallel region
 - O : outlines block
 - G : list of outline entries
 - E : one Subtask or Trial entry
 - H : list of thread bodies
 - W : one Thread body
 - T : free-form reasoning text

- **Start symbol:** S
- **Rules:** R defines valid tag nesting and branch structure.

Production Rules R

$S \rightarrow \text{<think>} B \text{</think>}$
 $B \rightarrow TB \mid PB \mid \epsilon$
 $P \rightarrow \text{<Parallel>} OH \text{</Parallel>}$
 $O \rightarrow \text{<Outlines>} G \text{</Outlines>}$
 $G \rightarrow EG \mid E$
 $E \rightarrow \text{<Subtask>} T \text{</Subtask>}$
 $\quad \mid \text{<Trial>} T \text{</Trial>}$
 $H \rightarrow WH \mid W$
 $W \rightarrow \text{<Thread>} B \text{</Thread>}$
 $T \rightarrow \text{nonempty free-form text excluding reserved tags}$

Figure 4: **Context-Free Grammar for Parason.** Each parallel region contains an Outlines block with one or more correctly matched Subtask or Trial entries, followed by one or more Thread bodies. As in Figure 3, each Thread may contain free-form reasoning or a nested parallel region.

156 <Outlines> describes the purpose of the region, <Subtask> and <Trial> mark branch type,
 157 <Thread> stores branch outcome. The full CFG is shown in Figure 4. While real-world reasoning
 158 often exhibits mixed dependencies (e.g., a subtask internally spawning trial branches), our training
 159 data keeps a strict separation for simplicity. However, we observe that the model can generalize to
 160 these complex compositions during evaluation.

161 The format is designed around two requirements. First, it is *semantic*: the tags indicate whether a
 162 branch is a necessary subtask or a speculative trial branch, not merely that it can run in parallel. This
 163 matters because prior parallel-reasoning methods often mix the two forms under a single branching
 164 interface, making it hard to decide whether branches should be all merged or aggregated as exploration
 165 history. Second, it is *engine-parseable*: each branch has explicit start and stop tags, so an inference
 166 runtime can dispatch workers without modifying the model architecture. Compared with free-form
 167 summaries or prompt-only, the CFG gives the training and inference infra a shared contract.

168 3.3 Parallelism-Aware Reinforcement Learning

169 Supervised conversion teaches the model to imitate parallel traces, but it does not directly opti-
 170 mize latency or decide *when* and *what* parallelism is worth using. We therefore RL fine-tune with
 171 **Parallelism-Aware Group Relative Policy Optimization (PA-GRPO)**. The reward keeps correct-
 172 ness as the primary objective, while shaping the model toward useful parallel branches. For a sampled
 173 trajectory i , let T_i denote its token-level latency, and let R_{subtask}^i and R_{trial}^i be the ratios of tokens
 174 placed in Subtask and Trial branches. We define:

175 Definition 3.1. Parallel RL Reward

$$\begin{aligned}
 R_i = & -1 + 2 \times \mathbb{1}(\text{correct}_i) \\
 & \times \left(1 - \alpha f\left(\frac{T_i - \mu_T}{\sigma_T}\right) + \beta_{\text{subtask}} f\left(\frac{R_{\text{subtask}}^i - \mu_s}{\sigma_s}\right) \right. \\
 & \left. + \beta_{\text{trial}} f\left(\frac{R_{\text{trial}}^i - \mu_r}{\sigma_r}\right) + \min(\rho \cdot \eta(s), \rho_{\text{clip}}) \right).
 \end{aligned}$$

176

177 The symbols in Eq. 3.1 are defined as follows:

- 178 • R_i : final reward for sampled trajectory i .
- $\mathbb{1}(\text{correct}_i)$: correctness indicator; incorrect answers receive the base negative reward.
- T_i : token-level latency of trajectory i , measured by its critical-path token count.
- α : coefficient controlling the normalized latency penalty.
- R_{subtask}^i : Number of tokens placed in Subtask Parallelism branches/Total generated tokens
- $\eta(s)$: $\eta(s) = 1 - \frac{L_{\text{latency}}}{L_{\text{total}}}$, where L_{latency} and L_{total} denote the number of latency tokens and the total number of tokens, respectively.
- ρ, ρ_{clip} : Parameters for controlling acceleration reward.
- R_{trial}^i : Number of tokens placed in Trial Parallelism branches/Total generated tokens
- β_{subtask} and β_{trial} : incentives for Subtask and Trial Parallelism.
- f : shaping function applied to normalized reward signals; here we use the linear function $y = x$.
- μ_T, μ_s, μ_r and $\sigma_T, \sigma_s, \sigma_r$: means and standard deviations used to normalize latency, Subtask ratio, and Trial ratio, respectively.

179 The first term makes correctness the primary signal by assigning a negative base reward to incorrect
 180 answers and a positive base reward to correct answers. The latency term, weighted by α , penalizes
 181 trajectories with high normalized critical-path latency. The Subtask and Trial incentives, weighted by
 182 β_{subtask} and β_{trial} , encourage useful parallel structure, while the clipped acceleration reward directly
 183 favors reductions in the critical path relative to total generation length. It is important to note
 184 that β_{subtask} and β_{trial} target fundamentally different optimization goals: Trial Parallelism improves
 185 accuracy by trading total tokens for search width, while Subtask Parallelism reduces latency by
 186 compressing the longest token path. In practice, this reward lets us separately tune these two
 187 dimensions, as detailed in Section 4.2.

188 3.4 Integration into Modern Inference Engine

189 Parason integrates with modern inference engines by treating parallel branches as tool calls rather than
 190 requiring deep runtime changes. Following the CFG in Figure 4, the runtime decodes normally until
 191 a `<Parallel>` region appears, parses the `<Outlines>`, dispatches each `<Subtask>` or `<Trial>` as
 192 an independent worker, and returns the resulting `<Thread>` outputs to the main trajectory; Figure 3
 193 shows this format on the 24 Puzzle. XGrammar [Dong et al., 2024] enforces valid tags, and we
 194 implement this tool-call execution in SGLang [Zheng et al., 2023].

195 4 Experiments

196 We provide details on training data curation, training framework, and evaluation below, and will
 197 open-source our implementation to facilitate reproducibility.

198 **Training Data Curation.** We build parallel training data from the 964 annotated Qwen3-8B traces
 199 released by ThreadWeaver [Lian et al., 2025]. Gemini-3-Flash labels each parallel stage as Subtask
 200 or Trial; for Trial stages, Qwen3-8B samples extra branches and Gemini-3-Flash generates their
 201 `<Trial>` goals. For reinforcement learning (RL), we use Polaris-53k [An et al., 2025], a collection
 202 of about 53,000 complex reasoning problems.

203 **Training Framework.** We use a two-stage post-training pipeline. First, we fine-tune the model with
 204 TRL [von Werra et al., 2020] on the curated parallel trajectories. Second, we train with VeRL [Sheng
 205 et al., 2024] using Parallelism-Aware GRPO (PA-GRPO). Unless otherwise noted, RL uses learning
 206 rate $1e-6$, batch size 128, 8 rollouts, lower clip ratio 0.2, and upper clip ratio 0.28; SFT uses learning
 207 rate $1e-5$, batch size 16, 8 epochs, and a cosine learning-rate scheduler.

208 **Evaluation.** We evaluate on AIME 2024 [Mathematical Association of America, 2024], AIME
 209 2025 [Mathematical Association of America, 2025], AMC [Mathematical Association of America,
 210 2023], Math500 [Hendrycks et al., 2021], and Minerva Math [Lewkowycz et al., 2022], and compare
 211 against sequential baselines trained from the same base model and data source as well as external
 212 parallel and efficient-reasoning systems. We report final-answer accuracy and executable latency.
 213 For Parason, **model latency** is the token length of the longest generation path: sequential tokens are

Method	AIME24	AIME25	Math500	AMC	Avg	Avg. Latency #Tokens
Parallel-R1 (4B)	19.4	19.2	-	-	-	-
ShorterBetter (7B)	53.3	-	-	-	-	-
ThinkPrune (32B)	72.5	-	93.8	95.9	-	-
DYNASOR-COT (32B)	<u>78.0</u>	-	93.0	94.0	-	-
Dynamic Early Exit (32B)	70.0	-	94.8	95.0	-	-
AdaptThink (7B)	55.6	-	92.0	-	-	-
Multiverse (32B)	53.8	45.8	91.8	-	-	-
ThreadWeaver (8B)	79.9	60.5	92.3	91.4	81.0	14.8k
Parason-8B (w/vanilla SFT)	73.5	67.9	93.4	93.9	82.2	13.2k
+ PA-GRPO ($\beta_{\text{subtask}} = 0.025$)	75.1	69.7	93.8	96.6	83.8	13.8k
+ PA-GRPO ($\beta_{\text{subtask}} = 0.050$)	76.3	<u>70.0</u>	94.6	95.0	84.0	15.6k
+ PA-GRPO ($\beta_{\text{subtask}} = 0.100$)	<u>77.3</u>	68.5	94.6	96.3	84.2	14.1k
+ PA-GRPO ($\beta_{\text{trial}} = 0.025$)	75.5	69.9	<u>94.4</u>	96.3	84.0	14.1k
+ PA-GRPO ($\beta_{\text{trial}} = 0.050$)	76.5	70.6	94.0	96.9	<u>84.5</u>	14.3k
+ PA-GRPO ($\beta_{\text{trial}} = 0.100$)	78.2	68.9	94.2	97.5	84.7	14.4k
<i>Additional runs with $\alpha = 0.1$</i>						
+ PA-GRPO ($\beta_{\text{subtask}} = 0.025$)	75.7	67.6	93.6	95.9	83.2	<u>12.2k</u>
+ PA-GRPO ($\beta_{\text{subtask}} = 0.050$)	74.2	66.1	93.8	95.9	82.5	<u>12.2k</u>
+ PA-GRPO ($\beta_{\text{subtask}} = 0.100$)	75.5	65.9	92.6	97.5	82.9	13.2k
+ PA-GRPO ($\beta_{\text{trial}} = 0.025$)	76.9	66.6	94.6	<u>97.2</u>	83.8	12.1k
+ PA-GRPO ($\beta_{\text{trial}} = 0.050$)	<u>77.3</u>	67.3	93.4	96.6	83.7	12.5k
+ PA-GRPO ($\beta_{\text{trial}} = 0.100$)	78.2	69.2	93.4	95.6	84.1	13.4k

Table 2: **Trial rewards improve accuracy, while Subtask rewards reduce latency.** We compare Parason with prior systems and ablate β_{subtask} and β_{trial} . Avg and Token Latency average AIME24, AIME25, Math500, and AMC for each PA-GRPO row. Bold and underlined values mark the highest and second-highest PA-GRPO results, except Token Latency, where lower is better.

Model	AIME24 Accuracy (%)			
	$B = 2,048$	$B = 8,192$	$B = 16,384$	$B = 24,576$
SFT only	16.8	41.8	66.8	70.2
PA-GRPO ($\beta_{\text{subtask}} = 0.025$)	16.7 _{-0.10}	45.2 _{+3.40}	68.0 _{+1.20}	73.2 _{+3.00}
PA-GRPO ($\beta_{\text{subtask}} = 0.050$)	26.8 _{+10.00}	60.0 _{+18.20}	72.2 _{+5.40}	74.8 _{+4.60}
PA-GRPO ($\beta_{\text{subtask}} = 0.100$)	34.7 _{+17.90}	60.3 _{+18.50}	74.4 _{+7.60}	75.7 _{+5.50}

Table 3: **Accuracy under token-latency budgets.** Each column reports AIME24 accuracy for a fixed thinking budget B on the longest token path. Subscripts show absolute accuracy-point gains relative to SFT only under the same budget; red indicates a decrease. Subtask-aware PA-GRPO provides the largest gains in low-budget regimes: at $B = 2,048$, $\beta_{\text{subtask}} = 0.100$ reaches 34.7% accuracy, compared with 16.8% for SFT only.

counted normally, while each <Parallel> block contributes the maximum branch length rather than the sum of all branches. This metric captures the wall-clock benefit available to an inference engine that runs branches concurrently.

4.1 Main Results

Parason improves accuracy over prior parallel reasoning systems. Table 2 compares Parason with existing parallel and efficient-reasoning methods. Parason gives the best reported AIME25 and AMC results among the listed systems, reaches the best Parason four-benchmark average accuracy of 84.7%, and stays close to ThreadWeaver on AIME24. Across the original settings and the additional $\alpha = 0.1$ runs, PA-GRPO reaches 70.6% on AIME25, 97.5% on AMC, and 94.6% on Math500. These gains show that Parason improves both accuracy and average benchmark coverage over prior parallel work, even though it uses an 8B model rather than several 32B baselines.

Method	Trigger Ratio	Token Latency	Acceleration Ratio	Avg. Math Acc.
ThreadWeaver (8B)	83.7	13.3k	1.18x	77.6
Multiverse (32B)	65.6	10.3k	1.16x	63.8
Parason-8B (w/ SFT only)	69.3	14.1k	1.27x	78.3
<i>Original runs with $\alpha = 0.000$</i>				
+ PA-GRPO ($\beta_{\text{trial}} = 0.025, \alpha = 0.000$)	77.3	15.1k	1.46x	79.9
+ PA-GRPO ($\beta_{\text{trial}} = 0.050, \alpha = 0.000$)	84.0	15.5k	1.61x	<u>80.4</u>
+ PA-GRPO ($\beta_{\text{trial}} = 0.100, \alpha = 0.000$)	98.8	15.6k	1.71x	80.4
+ PA-GRPO ($\beta_{\text{subtask}} = 0.025, \alpha = 0.000$)	70.1	14.9k	1.48x	79.5
+ PA-GRPO ($\beta_{\text{subtask}} = 0.050, \alpha = 0.000$)	79.3	16.6k	1.75x	80.3
+ PA-GRPO ($\beta_{\text{subtask}} = 0.100, \alpha = 0.000$)	89.6	15.4k	1.73x	80.1
<i>Additional runs with $\alpha = 0.100$</i>				
+ PA-GRPO ($\beta_{\text{trial}} = 0.025, \alpha = 0.100$)	62.3	13.2k	1.46x	79.4
+ PA-GRPO ($\beta_{\text{trial}} = 0.050, \alpha = 0.100$)	79.7	13.6k	1.50x	79.3
+ PA-GRPO ($\beta_{\text{trial}} = 0.100, \alpha = 0.100$)	95.3	14.5k	1.72x	80.3
+ PA-GRPO ($\beta_{\text{subtask}} = 0.025, \alpha = 0.100$)	53.7	<u>13.4k</u>	1.34x	79.0
+ PA-GRPO ($\beta_{\text{subtask}} = 0.050, \alpha = 0.100$)	81.3	13.2k	1.60x	78.0
+ PA-GRPO ($\beta_{\text{subtask}} = 0.100, \alpha = 0.100$)	<u>96.5</u>	14.3k	<u>1.74x</u>	78.0

Table 4: **PA-GRPO produces usable parallelism across latency-penalty settings.** For PA-GRPO, Trigger Ratio and Avg. Math Acc. are averaged over AIME24, AIME25, and Math500. **Acceleration ratio** is total tokens / longest-path tokens, and **trigger ratio** is the fraction of samples containing at least one `<Parallel>` block. Bold and underlined values mark the best and second-best PA-GRPO results among available measurements.

225 **Parason gives a better token-latency-accuracy trade-off under constraints.** Table 3 shows that
226 Parason is most useful when the model has a small thinking budget on the longest token path. At
227 2048 tokens, the best Subtask-aware PA-GRPO model reaches 34.7% AIME24 accuracy, compared
228 with 16.8% for SFT only. At 8192 tokens, it reaches 60.3%, compared with 41.8% for SFT only
229 at the same budget. Thus, Parason does not simply spend more tokens; it moves useful work into
230 parallel branches and keeps the longest path short.

231 4.2 Insights and Findings

232 **Trial rewards give the clearest accuracy gains, while the latency penalty shortens the critical**
233 **path.** Table 2 compares PA-GRPO with ThreadWeaver, which obtains 81.0% average accuracy with
234 a four-benchmark token latency of 14.8k. The original $\beta_{\text{trial}} = 0.100$ setting gives the best average
235 accuracy (84.7%), while $\beta_{\text{trial}} = 0.050$ gives the best AIME25 result (70.6%). The best AIME24
236 result (78.2%) is shared by the original and additional $\beta_{\text{trial}} = 0.100$ settings, and the best AMC result
237 (97.5%) is shared by the original $\beta_{\text{trial}} = 0.100$ model and the additional $\alpha = 0.1, \beta_{\text{subtask}} = 0.100$
238 model. Among the additional runs, $\beta_{\text{trial}} = 0.025$ gives the lowest four-benchmark token latency at
239 12.1k. The additional sweep spans 12.1–13.4k, below ThreadWeaver’s 14.8k average latency.

240 **PA-GRPO turns more parallel structure into executable acceleration.** Table 4 shows that the
241 learned parallel regions are usable by the runtime. SFT alone already triggers parallel execution
242 on 69.3% of samples and yields $1.27\times$ token-level acceleration. Under the original $\alpha = 0.000$
243 setting, the Trial sweep raises the trigger ratio from 77.3% to 98.8% and the acceleration ratio
244 from $1.46\times$ to $1.71\times$, while maintaining 79.9–80.4% average math accuracy. Subtask incentives
245 provide another route to high acceleration, with $\beta_{\text{subtask}} = 0.050$ reaching the best acceleration ratio
246 of $1.75\times$ and 80.3% average accuracy. In the additional $\alpha = 0.100$ runs, acceleration rises with
247 the parallelism coefficient: $\beta_{\text{subtask}} = 0.100$ reaches $1.74\times$ and $\beta_{\text{trial}} = 0.100$ reaches $1.72\times$. The
248 lowest three-benchmark token latency is 13.2k, reached by $\beta_{\text{trial}} = 0.025$ and $\beta_{\text{subtask}} = 0.050$ after
249 rounding.

250 **Harder questions create more parallelizable work, not less acceleration.** Table 5 breaks down
251 the step-200 AIME24 run by difficulty. Generated tokens grow sharply with difficulty, from 21.3k
252 on easy problems to 36.8k on medium problems and 50.3k on hard problems. The longest token
253 path also grows, but much more slowly: 12.5k, 21.2k, and 29.0k tokens, respectively. As a result,
254 Parason keeps a stable 1.70 – $1.74\times$ token-level acceleration ratio across all three difficulty groups.

Difficulty	#Problems	Token-Level Metrics				Wall-Clock Metrics	
		Generated #Tokens	Latency #Tokens	Acceleration Ratio	Saved #Tokens	Parallel (s)	Sequential (s)
Easy	13	21.3k	12.5k	$1.70\times$	8.8k	188.3	304.2
Medium	11	36.8k	21.2k	$1.74\times$	15.6k	336.2	464.7
Hard	6	50.3k	29.0k	$1.73\times$	21.3k	487.3	716.3

Table 5: **Harder questions save more tokens, while parallel execution reduces wall-clock latency across all difficulty levels.** We break down AIME24 thinking tokens and measured wall-clock time by difficulty. Latency #Tokens denotes the length of the longest token path, Acceleration Ratio is generated tokens divided by latency tokens, and Saved #Tokens is generated tokens minus latency tokens. The experiment is executed on A800 GPUs.

This translates into measured wall-clock speedups of $1.62\times$ on easy, $1.38\times$ on medium, and $1.47\times$ on hard problems. The absolute number of saved tokens increases from 8.8k on easy problems to 21.3k on hard problems, showing that harder questions expose more branch-level work that can be moved off the critical path. This supports the central motivation of Parason: difficult reasoning still requires long and diverse computation, but much of that computation does not need to remain serial.

5 Conclusion

In this work, we introduced **Parason**, an algorithm-system co-design framework that brings parallel processing to sequential LLM reasoning. Parason distinguishes deterministic **Subtask Parallelism** from speculative **Trial Parallelism**, and provides a stable pipeline for data curation, CFG-based structure, PA-GRPO training, and inference-engine execution. This design enables efficient parallel reasoning and improves the responsiveness of interactive agents, coding assistants, and scientific problem-solving tools. Our findings show that Trial Parallelism is a key mechanism for complex mathematical reasoning, although prior work has mainly focused on subtask decomposition. Across benchmarks, Parason reduces **token latency by about $1.7\times$** while **maintaining comparable accuracy**. We hope Parason raises the community’s awareness of parallel reasoning and its two parallelism schemes, and inspires future work on more efficient and scalable reasoning systems. We include limitations and the impact statement in Appendix A, and will open-source our implementation to support reproducibility.

References

- Chenxin An, Zihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL <https://hkunlp.github.io/blog/2025/Polaris>.
- Anthropic. Claude code: Deep coding at terminal velocity. <https://www.anthropic.com/claude-code>, 2025a.
- Anthropic. Introducing claude opus 4.5. <https://www.anthropic.com/news/claude-opus-4-5>, 2025b.
- Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Nuelle, Niklas Hoffra, et al. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*, 2023.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL <https://arxiv.org/abs/2110.14168>.
- Google DeepMind. Ai solves imo problems at silver medal level. <https://deepmind.google/blog/ai-solves-imo-problems-at-silver-medal-level/>, 2024. Accessed: 2024-07-25.

291 DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence,
 292 2026. URL [https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/blob/main/](https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/blob/main/DeepSeek_V4.pdf)
 293 [DeepSeek_V4.pdf](https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/blob/main/DeepSeek_V4.pdf).

294 DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu,
 295 Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu,
 296 Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao
 297 Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan,
 298 Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao,
 299 Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding,
 300 Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang
 301 Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong,
 302 Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao,
 303 Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang,
 304 Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang,
 305 Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L.
 306 Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang,
 307 Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng
 308 Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng
 309 Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan
 310 Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang,
 311 Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen,
 312 Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li,
 313 Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang,
 314 Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan,
 315 Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yudian Wang, Yue Gong, Yuheng Zou, Yujia
 316 He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong
 317 Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha,
 318 Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang,
 319 Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li,
 320 Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen
 321 Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.
 322 URL <https://arxiv.org/abs/2501.12948>.

323 Yixin Dong, Charlie F Ruan, Yaxing Cai, Ruihang Lai, Ziyi Xu, Yilong Zhao, and Tianqi Chen.
 324 Xgrammar: Flexible and efficient structured generation engine for large language models. *arXiv*
 325 *preprint arXiv:2411.15100*, 2024. URL <https://arxiv.org/abs/2411.15100>.

326 Yichao Fu, Xuwei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. *arXiv*
 327 *preprint arXiv:2508.15260*, 2025.

328 Google. Gemini 2.5: Our most intelligent ai model, March 2025. URL [https://blog.google/](https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-thinking)
 329 [technology/google-deepmind/gemini-model-thinking-updates-march-2025/](https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-thinking)
 330 [#gemini-2-5-thinking](https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-thinking).

331 Google DeepMind. Introducing gemini 3 pro. [https://deepmind.google/technologies/](https://deepmind.google/technologies/gemini/)
 332 [gemini/](https://deepmind.google/technologies/gemini/), 2025.

333 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song,
 334 and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL
 335 <https://arxiv.org/abs/2103.03874>.

336 Jingcheng Hu, Yinmin Zhang, Shijie Shang, Xiaobo Yang, Yue Peng, Zhewei Huang, Hebin Zhou,
 337 Xin Wu, Jie Cheng, Fanqi Wan, Xiangwen Kong, Chengyuan Yao, Kaiwen Yan, Ailin Huang,
 338 Hongyu Zhou, Qi Han, Zheng Ge, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Pacore:
 339 Learning to scale test-time compute with parallel coordinated reasoning, 2026. URL [https://](https://arxiv.org/abs/2601.05593)
 340 arxiv.org/abs/2601.05593.

341 Thomas Hubert, Rishi Mehta, Laurent Sartran, Miklós Z. Horváth, Goran Žužić, Eric Wieser,
 342 Aja Huang, Julian Schrittwieser, Yannick Schroecker, Hussain Masoom, Ottavia Bertolli, Tom
 343 Zahavy, Amol Mandhane, Jessica Yung, Iuliya Beloshapka, Borja Ibarz, Vivek Veeriah, Lei Yu,

344 Oliver Nash, Paul Lezeau, Salvatore Mercuri, Calle Sönne, Bhavik Mehta, Alex Davies, Daniel
345 Zheng, Fabian Pedregosa, Yin Li, Ingrid von Glehn, Mark Rowland, Samuel Albanie, Ameya
346 Velingker, Simon Schmitt, Edward Lockhart, Edward Hughes, Henryk Michalewski, Nicolas
347 Sonnerat, Demis Hassabis, Pushmeet Kohli, and David Silver. Olympiad-level formal mathematical
348 reasoning with reinforcement learning. *Nature*, November 2025. ISSN 1476-4687. doi: 10.1038/
349 s41586-025-09833-y. URL <http://dx.doi.org/10.1038/s41586-025-09833-y>.

350 Tian Jin, Ellie Y Cheng, Zack Ankner, Nikunj Saunshi, Blake M Elias, Amir Yazdanbakhsh, Jonathan
351 Ragan-Kelley, Suvinay Subramanian, and Michael Carbin. Learning to keep a promise: Scal-
352 ing language model decoding parallelism with learned asynchronous decoding. *arXiv preprint*
353 *arXiv:2502.11517*, 2025.

354 Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay
355 Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam
356 Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language
357 models, 2022. URL <https://arxiv.org/abs/2206.14858>.

358 Long Lian, Sida Wang, Felix Juefei-Xu, Tsu-Jui Fu, Xiuyu Li, Adam Yala, Trevor Darrell, Alane
359 Suhr, Yuandong Tian, and Xi Victoria Lin. Threadweaver: Adaptive threading for efficient parallel
360 reasoning in language models. *arXiv preprint arXiv:2501.00000*, 2025.

361 Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu
362 Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Clara Huiyi Hu, Henryk
363 Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu H. Trinh, Quoc V.
364 Le, and Junehyuk Jung. Towards robust mathematical reasoning, 2025. URL <https://arxiv.org/abs/2511.01846>.

366 Mathematical Association of America. American Mathematics Competitions 2023, 2023. URL
367 <https://maa.org/student-programs/amc/>. Accessed: 2026-05-07.

368 Mathematical Association of America. American Invitational Mathematics Examination 2024, 2024.
369 URL [https://artofproblemsolving.com/wiki/index.php/American_Invitational_](https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination)
370 *Mathematics_Examination*. Accessed: 2025-05-14.

371 Mathematical Association of America. American Invitational Mathematics Examination 2025, 2025.
372 URL [https://artofproblemsolving.com/wiki/index.php/American_Invitational_](https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination)
373 *Mathematics_Examination*. Accessed: 2025-05-14.

374 MiniMax AI. Minimax m2: Open weight large language model. [https://www.minimax.io/](https://www.minimax.io/news/minimax-m2)
375 *news/minimax-m2*, 2025.

376 Moonshot AI. Kimi k2: Open agentic intelligence. <https://moonshotai.github.io/Kimi-K2/>,
377 2025.

378 Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schif-
379 ferer, Wei Du, and Igor Gitman. Aimo-2 winning solution: Building state-of-the-art mathematical
380 reasoning models with openmathreasoning dataset, 2025. URL [https://arxiv.org/abs/2504.](https://arxiv.org/abs/2504.16891)
381 16891.

382 Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. Skeleton-of-
383 thought: Large language models can do parallel decoding. *Proceedings ENLSP-III*, 2023.

384 OpenAI. Codex: Ai coding partner from openai. <https://openai.com/codex>, 2025.

385 OpenAI. Gpt-5.5. <https://openai.com/>, 2026.

386 OpenAI, Lama Ahmad, Amanda Askell, Pamela Mishkin, Thomas O’Keefe, et al. Openai o1 system
387 card. *arXiv preprint arXiv:2412.16720*, 2024. URL <https://arxiv.org/abs/2412.16720>.

388 Jiayi Pan, Xiuyu Li, Long Lian, Charlie Snell, Yifei Zhou, Adam Yala, Trevor Darrell, Kurt Keutzer,
389 and Alane Suhr. Learning adaptive parallel reasoning with language models. *arXiv preprint*
390 *arXiv:2504.15466*, 2025.

391 Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin
 392 Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra,
 393 Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, Oliver Zhang, Mantas Mazeika, Dmitry
 394 Dodonov, Tung Nguyen, Jaeho Lee, Daron Anderson, Mikhail Doroshenko, Alun Cennyth Stokes,
 395 Mobeen Mahmood, Oleksandr Pokutnyi, Oleg Iskra, Jessica P. Wang, John-Clark Levin, Mstyslav
 396 Kazakov, Fiona Feng, Steven Y. Feng, Haoran Zhao, Michael Yu, Varun Gangal, Chelsea Zou,
 397 Zihan Wang, Serguei Popov, Robert Gerbicz, Geoff Galgon, Johannes Schmitt, Will Yeadon,
 398 Yongki Lee, Scott Sauer, Alvaro Sanchez, Fabian Giska, Marc Roth, Søren Riis, Saiteja Utpala,
 399 Noah Burns, Gashaw M. Goshu, Mohinder Maheshbhai Naiya, Chidozie Agu, Zachary Giboney,
 400 Antrell Cheatam, Francesco Fournier-Facio, Sarah-Jane Crowson, Lennart Finke, Zerui Cheng,
 401 Jennifer Zampese, Ryan G. Hoerr, Mark Nandor, Hyunwoo Park, Tim Gehringer, Jiaqi Cai, Ben
 402 McCarty, Alexis C Garretson, Edwin Taylor, Damien Sileo, Qiuyu Ren, Usman Qazi, Lianghui
 403 Li, Jungbae Nam, John B. Wydallis, Pavel Arkhipov, Jack Wei Lun Shi, Aras Bacho, Chris G.
 404 Willcocks, Hangrui Cao, Sumeet Motwani, Emily de Oliveira Santos, Johannes Veith, Edward
 405 Vendrow, Doru Cojoc, Kengo Zenitani, Joshua Robinson, Longke Tang, Yuqi Li, Joshua Vendrow,
 406 Natanael Wildner Fraga, Vladyslav Kuchkin, Andrey Pupasov Maksimov, Pierre Marion, Denis
 407 Efremov, Jayson Lynch, Kaiqu Liang, Aleksandar Mikov, Andrew Gritsevskiy, Julien Guillod,
 408 Gözdenur Demir, Dakotah Martinez, Ben Pageler, Kevin Zhou, Saeed Soori, Ori Press, Henry Tang,
 409 Paolo Rissone, Sean R. Green, Lina Brüssel, Moon Twayana, Aymeric Dieuleveut, Joseph Marvin
 410 Imperial, Ameya Prabhu, Jinzhou Yang, Nick Crispino, Arun Rao, Dimitri Zvonkine, Gabriel
 411 Loiseau, Mikhail Kalinin, Marco Lukas, Ciprian Manolescu, Nate Stambaugh, Subrata Mishra, Tad
 412 Hogg, Carlo Bosio, Brian P Coppola, Julian Salazar, Jaehyeok Jin, Rafael Sayous, Stefan Ivanov,
 413 Philippe Schwaller, Shaipranesh Senthilkuma, Andres M Bran, Andres Algaba, Kelsey Van den
 414 Houte, Lynn Van Der Sypt, Brecht Verbeke, David Noever, Alexei Kopylov, Benjamin Myklebust,
 415 Bikun Li, Lisa Schut, Evgenii Zheltonozhskii, Qiaochu Yuan, Derek Lim, Richard Stanley, Tong
 416 Yang, John Maar, Julian Wykowski, Martí Oller, Anmol Sahu, Cesare Giulio Ardito, Yuzheng Hu,
 417 Ariel Ghislain Kemogne Kamdoun, Alvin Jin, Tobias Garcia Vilchis, Yuexuan Zu, Martin Lackner,
 418 James Koppel, Gongbo Sun, Daniil S. Antonenko, Steffi Chern, Bingchen Zhao, Pierrot Arsene,
 419 Joseph M Cavanagh, Daofeng Li, Jiawei Shen, Donato Crisostomi, Wenjin Zhang, Ali Dehghan,
 420 Sergey Ivanov, David Perrella, Nurdin Kaparov, Allen Zang, Iliia Sucholutsky, Arina Kharlamova,
 421 Daniil Orel, Vladislav Poritski, Shalev Ben-David, Zachary Berger, Parker Whitfill, Michael Foster,
 422 Daniel Munro, Linh Ho, Shankar Sivarajan, Dan Bar Hava, Aleksey Kuchkin, David Holmes,
 423 Alexandra Rodriguez-Romero, Frank Sommerhage, Anji Zhang, Richard Moat, Keith Schneider,
 424 Zakayo Kazibwe, Don Clarke, Dae Hyun Kim, Felipe Meneguitti Dias, Sara Fish, Veit Elser,
 425 Tobias Kreiman, Victor Efren Guadarrama Vilchis, Immo Klose, Ujjwala Ananthaswaran, Adam
 426 Zweiger, Kaivalya Rawal, Jeffery Li, Jeremy Nguyen, Nicolas Daans, Haline Heidinger, Maksim
 427 Radionov, Václav Rozhoň, Vincent Ginis, Christian Stump, Niv Cohen, Rafał Poświata, Josef
 428 Tkadlec, Alan Goldfarb, Chenguang Wang, Piotr Padlewski, Stanislaw Barzowski, Kyle Mont-
 429 gomery, Ryan Stendall, Jamie Tucker-Foltz, Jack Stade, T. Ryan Rogers, Tom Goertzen, Declan
 430 Grabb, Abhishek Shukla, Alan Givré, John Arnold Ambay, Archan Sen, Muhammad Fayeaz Aziz,
 431 Mark H Inlow, Hao He, Ling Zhang, Younesse Kaddar, Ivar Ångquist, Yanxu Chen, Harrison K
 432 Wang, Kalyan Ramakrishnan, Elliott Thornley, Antonio Terpin, Hailey Schoelkopf, Eric Zheng,
 433 Avishy Carmi, Ethan D. L. Brown, Kelin Zhu, Max Bartolo, Richard Wheeler, Martin Stehberger,
 434 Peter Bradshaw, JP Heimonen, Kaustubh Sridhar, Ido Akov, Jennifer Sandlin, Yury Makarychev,
 435 Joanna Tam, Hieu Hoang, David M. Cunningham, Vladimir Goryachev, Demosthenes Patramanis,
 436 Michael Krause, Andrew Redenti, David Aldous, Jesyin Lai, Shannon Coleman, Jiangnan Xu,
 437 Sangwon Lee, Ilias Magoulas, Sandy Zhao, Ning Tang, Michael K. Cohen, Orr Paradise, Jan Hen-
 438 drik Kirchner, Maksym Ovchinnikov, Jason O. Matos, Adithya Shenoy, Michael Wang, Yuzhou
 439 Nie, Anna Sztzyber-Betley, Paolo Faraboschi, Robin Riblet, Jonathan Crozier, Shiv Halasyamani,
 440 Shreyas Verma, Prashant Joshi, Eli Meril, Ziqiao Ma, Jérémy Andréoletti, Raghav Singhal, Jacob
 441 Platnick, Volodymyr Nevirkovets, Luke Basler, Alexander Ivanov, Seri Khoury, Nils Gustafsson,
 442 Marco Piccardo, Hamid Mostaghimi, Qijia Chen, Virendra Singh, Tran Quoc Khanh, Paul Rosu,
 443 Hannah Szlyk, Zachary Brown, Himanshu Narayan, Aline Menezes, Jonathan Roberts, William
 444 Alley, Kunyang Sun, Arkil Patel, Max Lamparth, Anka Reuel, Linwei Xin, Hanmeng Xu, Jacob
 445 Loader, Freddie Martin, Zixuan Wang, Andrea Achilleos, Thomas Preu, Tomek Korbak, Ida Bosio,
 446 Fereshteh Kazemi, Ziye Chen, Biró Bálint, Eve J. Y. Lo, Jiaqi Wang, Maria Inês S. Nunes, Jeremiah
 447 Milbauer, M Saiful Bari, Zihao Wang, Behzad Ansarinejad, Yewen Sun, Stephane Durand, Hossam
 448 Elgnainy, Guillaume Douville, Daniel Tordera, George Balabanian, Hew Wolff, Lynna Kvistad,
 449 Hsiaoyun Milliron, Ahmad Sakor, Murat Eron, Andrew Favre D. O., Shailesh Shah, Xiaoxiang

450 Zhou, Firuz Kamalov, Sherwin Abdoli, Tim Santens, Shaul Barkan, Allison Tee, Robin Zhang,
 451 Alessandro Tomasiello, G. Bruno De Luca, Shi-Zhuo Looi, Vinh-Kha Le, Noam Kolt, Jiayi Pan,
 452 Emma Rodman, Jacob Drori, Carl J Fossum, Niklas Muennighoff, Milind Jagota, Ronak Pradeep,
 453 Honglu Fan, Jonathan Eicher, Michael Chen, Kushal Thaman, William Merrill, Moritz Firsching,
 454 Carter Harris, Stefan Ciobăcă, Jason Gross, Rohan Pandey, Ilya Gusev, Adam Jones, Shashank
 455 Agnihotri, Pavel Zhelnov, Mohammadreza Mofayezi, Alexander Piperski, David K. Zhang, Kos-
 456 tiantyn Dobarskyi, Roman Leventov, Ignat Soroko, Joshua Duersch, Vage Taamazyan, Andrew Ho,
 457 Wenjie Ma, William Held, Ruicheng Xian, Armel Randy Zebaze, Mohanad Mohamed, Julian Noah
 458 Leser, Michelle X Yuan, Laila Yacar, Johannes Lengler, Katarzyna Olszewska, Claudio Di Fratta,
 459 Edson Oliveira, Joseph W. Jackson, Andy Zou, Muthu Chidambaram, Timothy Manik, Hector
 460 Haffenden, Dashiell Stander, Ali Dasouqi, Alexander Shen, Bitu Golshani, David Stap, Egor
 461 Kretov, Mikalai Uzhou, Alina Borisovna Zhidkovskaya, Nick Winter, Miguel Orbegoza Rodriguez,
 462 Robert Lauff, Dustin Wehr, Colin Tang, Zaki Hossain, Shaun Phillips, Fortuna Samuele, Fredrik
 463 Ekström, Angela Hammon, Oam Patel, Faraz Farhidi, George Medley, Forough Mohammadzadeh,
 464 Madellene Peñaflor, Haile Kassahun, Alena Friedrich, Rayner Hernandez Perez, Daniel Pyda,
 465 Taom Sakal, Omkar Dhamane, Ali Khajegili Mirabadi, Eric Hallman, Kenchi Okutsu, Mike
 466 Battaglia, Mohammad Maghsoudimehrabani, Alon Amit, Dave Hulbert, Roberto Pereira, Simon
 467 Weber, Handoko, Anton Peristyy, Stephen Malina, Mustafa Mehkary, Rami Aly, Frank Reidegeld,
 468 Anna-Katharina Dick, Cary Friday, Mukhwinder Singh, Hassan Shapourian, Wanyoung Kim, Mar-
 469 iana Costa, Hubeyb Gurdogan, Harsh Kumar, Chiara Ceconello, Chao Zhuang, Haon Park, Micah
 470 Carroll, Andrew R. Tawfeek, Stefan Steinerberger, Daattavya Aggarwal, Michael Kirchhof, Linjie
 471 Dai, Evan Kim, Johan Ferret, Jainam Shah, Yuzhou Wang, Minghao Yan, Krzysztof Burdzy, Lixin
 472 Zhang, Antonio Franca, Diana T. Pham, Kang Yong Loh, Joshua Robinson, Abram Jackson, Paolo
 473 Giordano, Philipp Petersen, Adrian Cosma, Jesus Colino, Colin White, Jacob Votava, Vladimir
 474 Vinnikov, Ethan Delaney, Petr Spelda, Vit Stritecky, Syed M. Shahid, Jean-Christophe Mourrat,
 475 Lavr Vetoshkin, Koen Sponselee, Renas Bacho, Zheng-Xin Yong, Florencia de la Rosa, Nathan
 476 Cho, Xiuyu Li, Guillaume Malod, Orion Weller, Guglielmo Albani, Leon Lang, Julien Laurendeau,
 477 Dmitry Kazakov, Fatimah Adesanya, Julien Portier, Lawrence Hollom, Victor Souza, Yuchen Anna
 478 Zhou, Julien Degorre, Yiğit Yalın, Gbenga Daniel Obikoya, Rai, Filippo Bigi, M. C. Boscá, Oleg
 479 Shumar, Kaniuar Bacho, Gabriel Recchia, Mara Popescu, Nikita Shulga, Ngefor Mildred Tanwie,
 480 Thomas C. H. Lux, Ben Rank, Colin Ni, Matthew Brooks, Alesia Yakimchyk, Huanxu, Liu,
 481 Stefano Cavalleri, Olle Häggström, Emil Verkama, Joshua Newbould, Hans Gundlach, Leonor
 482 Brito-Santana, Brian Amaro, Vivek Vajipey, Rynaa Grover, Ting Wang, Yosi Kratish, Wen-Ding
 483 Li, Sivakanth Gopi, Andrea Caciolai, Christian Schroeder de Witt, Pablo Hernández-Cámara,
 484 Emanuele Rodolà, Jules Robins, Dominic Williamson, Vincent Cheng, Brad Raynor, Hao Qi, Ben
 485 Segev, Jingxuan Fan, Sarah Martinson, Erik Y. Wang, Kaylie Hausknecht, Michael P. Brenner,
 486 Mao Mao, Christoph Demian, Peyman Kassani, Xinyu Zhang, David Avagian, Eshawn Jessica
 487 Scipio, Alon Ragoler, Justin Tan, Blake Sims, Rebeka Plecnik, Aaron Kirtland, Omer Faruk
 488 Bodur, D. P. Shinde, Yan Carlos Leyva Labrador, Zahra Adoul, Mohamed Zekry, Ali Karakoc,
 489 Tania C. B. Santos, Samir Shamseldeen, Loukmane Karim, Anna Liakhovitskaia, Nate Resman,
 490 Nicholas Farina, Juan Carlos Gonzalez, Gabe Maayan, Earth Anderson, Rodrigo De Oliveira
 491 Pena, Elizabeth Kelley, Hodjat Mariji, Rasoul Pouriamanesh, Wentao Wu, Ross Finocchio, Ismail
 492 Alarab, Joshua Cole, Danyelle Ferreira, Bryan Johnson, Mohammad Safdari, Liangti Dai, Siriphan
 493 Arthornthurasuk, Isaac C. McAlister, Alejandro José Moyano, Alexey Pronin, Jing Fan, Angel
 494 Ramirez-Trinidad, Yana Malysheva, Daphiny Pottmaier, Omid Taheri, Stanley Stepanic, Samuel
 495 Perry, Luke Askew, Raúl Adrián Huerta Rodríguez, Ali M. R. Minissi, Ricardo Lorena, Krishna-
 496 murthy Iyer, Arshad Anil Fasiludeen, Ronald Clark, Josh Ducey, Matheus Piza, Maja Somrak, Eric
 497 Vergo, Juehang Qin, Benjámín Borbás, Eric Chu, Jack Lindsey, Antoine Jallon, I. M. J. McInnis,
 498 Evan Chen, Avi Semler, Luk Gloor, Tej Shah, Marc Carauleanu, Pascal Lauer, Tran Duc Huy,
 499 Hossein Shahrtash, Emilien Duc, Lukas Lewark, Assaf Brown, Samuel Albanie, Brian Weber,
 500 Warren S. Vaz, Pierre Clavier, Yiyang Fan, Gabriel Poesia Reis e Silva, Long, Lian, Marcus
 501 Abramovitch, Xi Jiang, Sandra Mendoza, Murat Islam, Juan Gonzalez, Vasilios Mavroudis, Justin
 502 Xu, Pawan Kumar, Laxman Prasad Goswami, Daniel Bugas, Nasser Heydari, Ferenc Jeanplong,
 503 Thorben Jansen, Antonella Pinto, Archimedes Apronti, Abdallah Galal, Ng Ze-An, Ankit Singh,
 504 Tong Jiang, Joan of Arc Xavier, Kanu Priya Agarwal, Mohammed Berkani, Gang Zhang, Zhehang
 505 Du, Benedito Alves de Oliveira Junior, Dmitry Malishev, Nicolas Remy, Taylor D. Hartman, Tim
 506 Tarver, Stephen Mensah, Gautier Abou Loume, Wiktor Morak, Farzad Habibi, Sarah Hoback, Will
 507 Cai, Javier Gimenez, Roselynn Grace Montecillo, Jakub Łucki, Russell Campbell, Asankhaya
 508 Sharma, Khalida Meer, Shreen Gul, Daniel Espinosa Gonzalez, Xavier Alapont, Alex Hoover, Gun-

509 jan Chhablani, Freddie Vargus, Arunim Agarwal, Yibo Jiang, Deepakkumar Patil, David Outevsky,
 510 Kevin Joseph Scaria, Rajat Maheshwari, Abdelkader Dendane, Priti Shukla, Ashley Cartwright,
 511 Sergei Bogdanov, Niels Mündler, Sören Möller, Luca Arnaboldi, Kunvar Thaman, Muhammad Re-
 512 han Siddiqi, Prajvi Saxena, Himanshu Gupta, Tony Fruhauff, Glen Sherman, Mátyás Vincze,
 513 Siranut Usawasutsakorn, Dylan Ler, Anil Radhakrishnan, Innocent Enyekwe, Sk Md Salauddin,
 514 Jiang Muzhen, Aleksandr Maksapetyan, Vivien Rossbach, Chris Harjadi, Mohsen Bahaloohoreh,
 515 Claire Sparrow, Jasdeep Sidhu, Sam Ali, Song Bian, John Lai, Eric Singer, Justine Leon Uro,
 516 Greg Bateman, Mohamed Sayed, Ahmed Menshawy, Darling Duclosel, Dario Bezzi, Yashaswini
 517 Jain, Ashley Aaron, Murat Tiryakoglu, Sheeshram Siddh, Keith Krenek, Imad Ali Shah, Jun Jin,
 518 Scott Creighton, Denis Peskoff, Zienab EL-Wasif, Ragavendran P V, Michael Richmond, Joseph
 519 McGowan, Tejal Patwardhan, Hao-Yu Sun, Ting Sun, Nikola Zubić, Samuele Sala, Stephen Ebert,
 520 Jean Kaddour, Manuel Schottdorf, Dianzhuo Wang, Gerol Petruzella, Alex Meiburg, Tilen Medved,
 521 Ali ElSheikh, S Ashwin Hebbar, Lorenzo Vaquero, Xianjun Yang, Jason Poulos, Vilém Zouhar,
 522 Sergey Bogdanik, Mingfang Zhang, Jorge Sanz-Ros, David Anugraha, Yinwei Dai, Anh N. Nhu,
 523 Xue Wang, Ali Anil Demircali, Zhibai Jia, Yuyin Zhou, Juncheng Wu, Mike He, Nitin Chandok,
 524 Aarush Sinha, Gaoxiang Luo, Long Le, Mickaël Noyé, Michał Perelkiewicz, Ioannis Pantidis,
 525 Tianbo Qi, Soham Sachin Purohit, Letitia Parcalabescu, Thai-Hoa Nguyen, Genta Indra Winata,
 526 Edoardo M. Ponti, Hanchen Li, Kaustubh Dhole, Jongee Park, Dario Abbondanza, Yuanli Wang,
 527 Anupam Nayak, Diogo M. Caetano, Antonio A. W. L. Wong, Maria del Rio-Chanona, Daniel
 528 Kondor, Pieter Francois, Ed Chaltrey, Jakob Zsambok, Dan Hoyer, Jenny Reddish, Jakob Hauser,
 529 Francisco-Javier Rodrigo-Ginés, Suchandra Datta, Maxwell Shepherd, Thom Kamphuis, Qizheng
 530 Zhang, Hyunjun Kim, Ruiji Sun, Jianzhu Yao, Franck Dernoncourt, Satyapriya Krishna, Sina
 531 Rismanchian, Bonan Pu, Francesco Pinto, Yingheng Wang, Kumar Shridhar, Kalon J. Overholt,
 532 Glib Briia, Hieu Nguyen, David, Soler Bartomeu, Tony CY Pang, Adam Wecker, Yifan Xiong,
 533 Fanfei Li, Lukas S. Huber, Joshua Jaeger, Romano De Maddalena, Xing Han Lù, Yuhui Zhang,
 534 Claas Beger, Patrick Tser Jern Kon, Sean Li, Vivek Sanker, Ming Yin, Yihao Liang, Xinlu Zhang,
 535 Ankit Agrawal, Li S. Yifei, Zechen Zhang, Mu Cai, Yasin Sonmez, Costin Cozianu, Changhao
 536 Li, Alex Slen, Shoubin Yu, Hyun Kyu Park, Gabriele Sarti, Marcin Briński, Alessandro Stolfo,
 537 Truong An Nguyen, Mike Zhang, Yotam Perlitz, Jose Hernandez-Orallo, Runjia Li, Amin Sha-
 538 bani, Felix Juefei-Xu, Shikhar Dhingra, Orr Zohar, My Chiffon Nguyen, Alexander Pondaven,
 539 Abdurrahim Yilmaz, Xuandong Zhao, Chuanyang Jin, Muyan Jiang, Stefan Todoran, Xinyao
 540 Han, Jules Kreuer, Brian Rabern, Anna Plassart, Martino Maggetti, Luther Yap, Robert Geirhos,
 541 Jonathon Kean, Dingsu Wang, Sina Mollaei, Chenkai Sun, Yifan Yin, Shiqi Wang, Rui Li, Yaowen
 542 Chang, Anjiang Wei, Alice Bizeul, Xiaohan Wang, Alexandre Oliveira Arrais, Kushin Mukherjee,
 543 Jorge Chamorro-Padial, Jiachen Liu, Xingyu Qu, Junyi Guan, Adam Bouyamourn, Shuyu Wu,
 544 Martyna Plomecka, Junda Chen, Mengze Tang, Jiaqi Deng, Shreyas Subramanian, Haocheng Xi,
 545 Haoxuan Chen, Weizhi Zhang, YINUO Ren, Haoqin Tu, Sejong Kim, Yushun Chen, Sara Vera
 546 Marjanović, Junwoo Ha, Grzegorz Luczyna, Jeff J. Ma, Zewen Shen, Dawn Song, Cedegao E.
 547 Zhang, Zhun Wang, Gaël Gendron, Yunze Xiao, Leo Smucker, Erica Weng, Kwok Hao Lee,
 548 Zhe Ye, Stefano Ermon, Ignacio D. Lopez-Miguel, Theo Knights, Anthony Gitter, Namkyu Park,
 549 Boyi Wei, Hongzheng Chen, Kunal Pai, Ahmed Elkhanany, Han Lin, Philipp D. Siedler, Jichao
 550 Fang, Ritwik Mishra, Károly Zsolnai-Fehér, Xilin Jiang, Shadab Khan, Jun Yuan, Rishab Kumar
 551 Jain, Xi Lin, Mike Peterson, Zhe Wang, Aditya Malusare, Maosen Tang, Isha Gupta, Ivan Fosing,
 552 Timothy Kang, Barbara Dworakowska, Kazuki Matsumoto, Guangyao Zheng, Gerben Sewuster,
 553 Jorge Pretel Villanueva, Ivan Rannev, Igor Chernyavsky, Jiale Chen, Deepayan Banik, Ben Racz,
 554 Wenchao Dong, Jianxin Wang, Laila Bashmal, Duarte V. Gonçalves, Wei Hu, Kaushik Bar, Ondrej
 555 Bohdal, Atharv Singh Patlan, Shehzaad Dhuliawala, Caroline Geirhos, Julien Wist, Yuval Kansal,
 556 Bingsen Chen, Kutay Tire, Atak Talay Yücel, Brandon Christof, Veerupaksh Singla, Zijian Song,
 557 Sanxing Chen, Jiaxin Ge, Kaustubh Ponskhe, Isaac Park, Tianneng Shi, Martin Q. Ma, Joshua Mak,
 558 Sherwin Lai, Antoine Moulin, Zhuo Cheng, Zhanda Zhu, Ziyi Zhang, Vaidehi Patil, Ketan Jha,
 559 Qitong Men, Jiaxuan Wu, Tianchi Zhang, Bruno Hebling Vieira, Alham Fikri Aji, Jae-Won Chung,
 560 Mohammed Mahfoud, Ha Thi Hoang, Marc Sperzel, Wei Hao, Kristof Meding, Sihan Xu, Vassilis
 561 Kostakos, Davide Manini, Yueying Liu, Christopher Toukmaji, Jay Paek, Eunmi Yu, Arif Engin
 562 Demircali, Zhiyi Sun, Ivan Dewerpe, Hongsen Qin, Roman Pflugfelder, James Bailey, Johnathan
 563 Morris, Ville Heilala, Sybille Rosset, Zishun Yu, Peter E. Chen, Woongyeong Yeo, Eeshaan Jain,
 564 Ryan Yang, Sreekar Chigurupati, Julia Chernyavsky, Sai Prajwal Reddy, Subhashini Venugopalan,
 565 Hunar Batra, Core Francisco Park, Hieu Tran, Guilherme Maximiano, Genghan Zhang, Yizhuo
 566 Liang, Hu Shiyu, Rongwu Xu, Rui Pan, Siddharth Suresh, Ziqi Liu, Samaksh Gulati, Songyang
 567 Zhang, Peter Turchin, Christopher W. Bartlett, Christopher R. Scotese, Phuong M. Cao, Ben Wu,

568 Jacek Karwowski, Davide Scaramuzza, Aakaash Nattanmai, Gordon McKellips, Anish Cheraku,
569 Asim Suhail, Ethan Luo, Marvin Deng, Jason Luo, Ashley Zhang, Kavin Jindel, Jay Paek, Kasper
570 Halevy, Allen Baranov, Michael Liu, Advait Avadhanam, David Zhang, Vincent Cheng, Brad Ma,
571 Evan Fu, Liam Do, Joshua Lass, Hubert Yang, Surya Sunkari, Vishruth Bharath, Violet Ai, James
572 Leung, Rishit Agrawal, Alan Zhou, Kevin Chen, Tejas Kalpathi, Ziqi Xu, Gavin Wang, Tyler Xiao,
573 Erik Maung, Sam Lee, Ryan Yang, Roy Yue, Ben Zhao, Julia Yoon, Sunny Sun, Aryan Singh,
574 Ethan Luo, Clark Peng, Tyler Osbey, Taozhi Wang, Daryl Echeazu, Hubert Yang, Timothy Wu,
575 Spandan Patel, Vidhi Kulkarni, Vijaykaarti Sundarapandiyan, Ashley Zhang, Andrew Le, Zafir
576 Nasim, Srikar Yalam, Ritesh Kasamsetty, Soham Samal, Hubert Yang, David Sun, Nihar Shah,
577 Abhijeet Saha, Alex Zhang, Leon Nguyen, Laasya Nagumalli, Kaixin Wang, Alan Zhou, Aidan
578 Wu, Jason Luo, Anwith Telluri, Summer Yue, Alexandr Wang, and Dan Hendrycks. Humanity’s
579 last exam, 2025. URL <https://arxiv.org/abs/2501.14249>.

580 Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng,
581 Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint*
582 *arXiv: 2409.19256*, 2024.

583 Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lam-
584 bert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformers Reinforcement
585 Learning, 2020. URL <https://github.com/huggingface/trl>.

586 Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdh-
587 ery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models.
588 *arXiv preprint arXiv:2203.11171*, 2022.

589 Ziqi Wang, Boye Niu, Zipeng Gao, Zhi Zheng, Tong Xu, Linghui Meng, Zhongli Li, Jing Liu, Yilong
590 Chen, Chen Zhu, et al. A survey on parallel reasoning. *arXiv preprint arXiv:2510.12164*, 2025.

591 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V
592 Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In
593 *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022.

594 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang
595 Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*,
596 2025a.

597 Xinyu Yang, Yuwei An, Hongyi Liu, Tianqi Chen, and Beidi Chen. Multiverse: Your language
598 models secretly decide how to parallelize and merge generation. *arXiv preprint arXiv:2506.09991*,
599 2025b.

600 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan.
601 Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural*
602 *information processing systems*, 36:11809–11822, 2023.

603 Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi
604 Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, Clark Barrett, and Ying Sheng. Sglang:
605 Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*,
606 2023. URL <https://arxiv.org/abs/2312.07104>.

607 Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, Xinyu Yang, Runpeng Dai, Rui Liu,
608 Huiwen Bao, Chengsong Huang, Heng Huang, et al. Parallel-rl: Towards parallel thinking via
609 reinforcement learning. *arXiv preprint arXiv:2509.07980*, 2025.

610 A Limitations and Future Work

611 Parason’s training and evaluation mainly focus on mathematical reasoning. It remains unclear how
612 well the same taxonomy, data curation pipeline, and PA-GRPO objective transfer to other domains,
613 such as real-world agents. Second, our current experiments focus on 8B-scale models. This setting
614 provides a controlled testbed for studying parallel reasoning, but it does not fully show how Parason
615 behaves across model families and sizes. In future work, we will scale Parason to more backbones
616 and larger models to study whether the same parallelism patterns and latency gains hold at broader
617 scales.

618 B Prompts Used for Data Curation

619 We use the following prompts to identify Trial and Subtask steps in LLM reasoning trajectories.
620 For open-source models, we first split each reasoning trajectory by \n\n. The first prompt reconstructs
621 logical steps by deciding whether each segment starts a new subproblem or continues a previous one.
622 The second prompt then classifies each logical step as either a Trial step or a Subtask step.
623 For commercial models, the released reasoning summaries already mark steps with “***Step i***”.
624 We therefore apply the second prompt directly to classify each marked step as Trial or Subtask.

Reconstruction by Logical Steps

SYSTEM:

You are a helpful assistant that analyzes chain-of-thought traces from
↪ reasoning models.

The chain of thought has been split into multiple steps. For each step, decide
↪ whether it starts a new subproblem or continues the previous one.

Use the content of each step to make the decision. If a step introduces a new
↪ concept, question, or task that is distinct from previous steps, it likely
↪ starts a new subproblem. If it builds on previous steps by adding details,
↪ explanations, reflections, or calculations about the same concept, it
↪ likely continues the previous subproblem.

More specifically, steps that start with "Alternatively", "Wait", or "But" are
↪ likely to start a new subproblem, while steps that start with "Therefore",
↪ "Thus", or "Consequently" are likely to continue the previous subproblem.
Your output should be in the following format:

Step i: [New Subproblem|Continue Previous Subproblem]

If a step continues a previous subproblem, report where that subproblem
↪ starts. For example, if Step 5 continues the subproblem that started at
↪ Step 3, output:

Step 5: Continue Previous Subproblem (started at Step 3)

IMPORTANT: Output the analysis in the specified format. Use one line per step.
↪ Do not include additional explanations or text.

Example:

Step 1: New Subproblem

Step 2: Continue Previous Subproblem (started at Step 1)

Step 3: New Subproblem

Step 4: Continue Previous Subproblem (started at Step 3)

Step 5: Continue Previous Subproblem (started at Step 3)

625

Parallel Stage Identification

SYSTEM:

You are given a series of mathematical reasoning steps. Classify each step
→ into one of the following categories:

1. Trial Step: A step that introduces a new idea, approach, or line of thought
→ whose success is uncertain but worth exploring.
2. Subtask Step: A step that belongs to a known solution path and directly
→ contributes to the final solution.

Your output should be in the following format:

Step i: [Trial Step|Subtask Step] [Reason]

The number of output steps should match the number of input steps. Each step

→ has its own number and is separated by "=====

→ =====". Ignore the content under

→ "Content after </think> tag:

"

Give the reason for each classification after the step type.

626

627 C AIME 2024 Problem Indices

628 Table 6 lists the AIME 2024 validation problems used for the difficulty-based analysis in Table 5. We
629 group problems into Easy, Mid, and Hard subsets according to their difficulty labels, and report the
630 corresponding zero-based problem indices for reproducibility.

Difficulty	Problem Indices
Hard	2, 3, 13, 21, 28, 29
Mid	1, 4, 5, 10, 16, 17, 18, 20, 25, 26, 27
Easy	0, 6, 7, 8, 9, 11, 12, 14, 15, 19, 22, 23, 24

Table 6: AIME 2024 Problem Classification by Difficulty.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction clearly state the taxonomy of Subtask and Trial parallelism, the Parason framework, PA-GRPO, and the empirical speedups achieved on math benchmarks (Sections 1 and 3.1).

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss limitations regarding computational overhead (more total tokens) and the limitations for focusing mainly on math reasoning.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate “Limitations” section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren’t acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper does not include theoretical results or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide details on training data curation, RL hyperparameters, and evaluation metrics in Section 4. We also define the grammar and training objective (PA-GRPO) in Section 3.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provided data and instructions to reproduce, and will open code when less anonymous

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Training details including learning rates, batch sizes, and optimizer settings are provided in Section 4.1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Due to the high cost of RL training and sampling on AIME benchmarks, we report average@32 results which provide an estimate of stability but do not include formal error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [N/A]

Justification:

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: The Impact Statement discusses positive impacts on efficiency and responsiveness for next-generation agents.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper focuses on improving reasoning efficiency and does not release models with high risk for misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the creators of ThreadWeaver, Polaris, and benchmarks (AIME, AMC, etc.) in Section 4.1.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: No new dataset assets are released with this submission.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This paper does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This paper does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- 941 • Depending on the country in which research is conducted, IRB approval (or equivalent)
942 may be required for any human subjects research. If you obtained IRB approval, you
943 should clearly state this in the paper.
- 944 • We recognize that the procedures for this may vary significantly between institutions
945 and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the
946 guidelines for their institution.
- 947 • For initial submissions, do not include any information that would break anonymity (if
948 applicable), such as the institution conducting the review.

949 16. Declaration of LLM usage

950 Question: Does the paper describe the usage of LLMs if it is an important, original, or
951 non-standard component of the core methods in this research? Note that if the LLM is used
952 only for writing, editing, or formatting purposes and does *not* impact the core methodology,
953 scientific rigor, or originality of the research, declaration is not required.

954 Answer: [N/A]

955 Justification: The use of LLMs in this work was limited to routine coding assistance and
956 linguistic editing of the manuscript. It did not constitute an original, important, or non-
957 standard component of the research methodology.

958 Guidelines:

- 959 • The answer [N/A] means that the core method development in this research does not
960 involve LLMs as any important, original, or non-standard components.
- 961 • Please refer to our LLM policy in the NeurIPS handbook for what should or should not
962 be described.