Scaling over Scaling: Exploring Test-Time Scaling Pareto in Large Reasoning Models

Jian Wang, Boyan Zhu, Chak Tou Leong, Yongqi Li[*] Wenjie Li
Department of Computing, The Hong Kong Polytechnic University
jian51.wang@polyu.edu.hk {boyan.zhu,chak-tou.leong}@connect.polyu.hk
liyongqi0@gmail.com cswjli@comp.polyu.edu.hk

Abstract

Large reasoning models (LRMs) have exhibited the capacity of enhancing reasoning performance via internal test-time scaling. Building upon this, a promising direction is to further scale test-time compute to unlock even greater reasoning capabilities. However, as we push these scaling boundaries, systematically understanding the practical limits and achieving optimal resource allocation becomes a critical challenge. In this paper, we investigate the *scaling Pareto* of test-time scaling and introduce the Test-Time Scaling Performance Model (TTSPM). We theoretically analyze two fundamental paradigms for such extended scaling, parallel scaling and sequential scaling, from a probabilistic modeling perspective. Our primary contribution is the derivation of the saturation point on the scaling budget for both strategies, identifying thresholds beyond which additional computation yields diminishing returns. Remarkably, despite their distinct mechanisms, both paradigms converge to a unified mathematical structure in their upper bounds. We empirically validate our theoretical findings on challenging reasoning benchmarks, including AIME, MATH-500, and GPQA, demonstrating the practical utility of these bounds for test-time resource allocation. We hope that this work provides insights into the cost-benefit trade-offs of test-time scaling, guiding the development of more resource-efficient inference strategies for large reasoning models.

1 Introduction

Large language models (LLMs) have demonstrated impressive capabilities across a wide range of reasoning tasks. Their performance can be further enhanced by increasing compute at inference time (or test time), a technique commonly referred to as *test-time scaling* (TTS) [32]. Recent advancements have enabled LLMs to generate longer and more coherent Chain-of-Thought (CoT) reasoning traces, as seen in models like OpenAI's o1 [15], o3 [16], and DeepSeek-R1 [4]. This approach, known as *internal scaling*, extends the model's deliberation within a single forward pass, significantly advancing LLMs into a new class of large reasoning models (LRMs).

Building upon the foundation of internally scaled reasoning, we naturally pose the next question: *Can these already powerful LRMs be further enhanced by allocating even more compute during inference?* This inquiry motivates our exploration of test-time scaling over internally scaled LRMs, a concept we refer to as **scaling over scaling**. However, as we consider increasing test-time compute, such as by generating additional solution candidates or performing more rounds of iterative refinement, it becomes essential to ask: Is such scaling always beneficial? In other words, *does a point of diminishing returns exist, beyond which additional computational effort yields negligible performance gains?* Identifying and characterizing this **scaling Pareto** is critical for preventing resource over-

[*] Corresponding author.

Preprint. Under review.

allocation, especially in most TTS strategies where the absence of supervised feedback complicates efficient compute usage.

To address the above questions, we introduce the **T**est-**T**ime **S**caling **P**erformance **M**odel (**TTSPM**), a theoretical probabilistic model that reimagines test-time scaling performance through the lens of probabilistic modeling (as detailed in Section 3). TTSPM begins with two fundamental TTS paradigms: 1) parallel scaling [26], where multiple reasoning paths and solutions are generated independently; and 2) sequential scaling [24], where a solution is iteratively refined round by round. Their performance gains with increasing compute often exhibit a characteristic saturation curve. We develop a general probabilistic model that captures this behavior, positing that the probability of success with the scaling budget N approaches a maximum achievable performance F_{max} as N grows. Based on this probabilistic model, we analyze the marginal performance gain per additional generation, which is referred to as a scaling unit. We then formally define the *scaling Pareto* as the point where this marginal gain is no longer higher than a predefined small threshold ϵ as increasing scaling units. Using this definition, we derive a unified saturation point for the scaling budget, which signifies the critical point beyond which further scaling is deemed inefficient.

We empirically validate the practical utility of our theoretical bounds on challenging mathematical reasoning and PhD-level science question answering benchmarks, including AIME [11], MATH-500 [9], and GPQA [20]. Our experiments (see Section 4) demonstrate that the derived bounds effectively predict the onset of the scaling Pareto, aligning with observed performance saturation. These findings offer valuable insights. First, LRMs can indeed be further improved via simple, verifier-free test-time scaling. Second, this improvement is not limitless and exhibits a predictable saturation point. Our TTSPM provides a principled method to estimate this point, enabling more efficient resource allocation by avoiding unnecessary token consumption.

Overall, the contributions of this paper are as follows: 1) We propose TTSPM, a theoretical probabilistic model that effectively characterizes test-time scaling Pareto in large reasoning models. 2) With TTSPM, we derive a general upper bound for scaling units. It is applicable to both parallel and sequential scaling strategies, revealing a consistent mathematical structure underlying their saturation behavior. 3) We outline how this theoretical bound can be empirically validated and discuss its practical implications for optimizing test-time compute, thereby guiding the development of more resource-efficient inference scaling strategies for large reasoning models.

2 Related Work

The quest to enhance the reasoning abilities of large language models (LLMs) beyond standard pre-training and fine-tuning has led to significant interest in techniques that leverage increased computation during inference, commonly referred to as test-time scaling (TTS) [32, 28]. These methods aim to improve performance on complex tasks by allocating more resources for "thinking" or exploration before generating final answers, without modifying the model's parameters. Test-time scaling strategies can be broadly categorized based on *what* aspect of the generation process is scaled and *how* the scaling is implemented [32].

Parallel Test-Time Scaling. Parallel scaling methods explore multiple reasoning paths or solutions concurrently. Self-Consistency (SC) [26] is a prominent example, where multiple reasoning paths are sampled independently using Chain-of-Thought (CoT), and the final answer is determined by majority voting over the outcomes. This approach improves robustness against occasional reasoning errors in single paths. Best-of-N (BoN) sampling [2] also generates multiple candidate outputs and selects the best one based on a scoring mechanism, often using the model's own likelihood scores or an external verifier [9]. Tree of Thoughts (ToT) [30] explores reasoning as a tree search, where multiple reasoning paths are explored in parallel at each step. While parallel methods effectively broaden the search space, they often involve redundant computations as paths are generated independently without collaboration. Majority voting in SC can also be misled if multiple paths converge to the same incorrect answer. Recent work has also explored adaptive parallel reasoning [18], but explicit theoretical mechanisms remain less studied.

Sequential Test-Time Scaling. Sequential scaling methods enhance reasoning by generating intermediate steps or thoughts in a serial manner. The seminal work on Chain-of-Thought (CoT) prompting [27] demonstrated that prompting LLMs to produce step-by-step reasoning traces significantly improves performance on arithmetic, commonsense, and symbolic reasoning tasks. Building upon this, various techniques have explored iterative refinement and self-correction. Self-Refine [12] enables models to iteratively refine their outputs based on self-generated feedback. Similarly, methods like [8] employ stepwise natural language self-critique to enhance reasoning. Other approaches focus on extending the length or depth of sequential thought. For example, [24] proposed Multi-round Thinking, scaling the number of thinking rounds at test time. [29] suggested breaking length limits in long-context reasoning. While effective in guiding the model, purely sequential approaches can suffer from high latency, potential context length overflow, and the risk of error propagation, where an early mistake derails the entire reasoning process [13, 31].

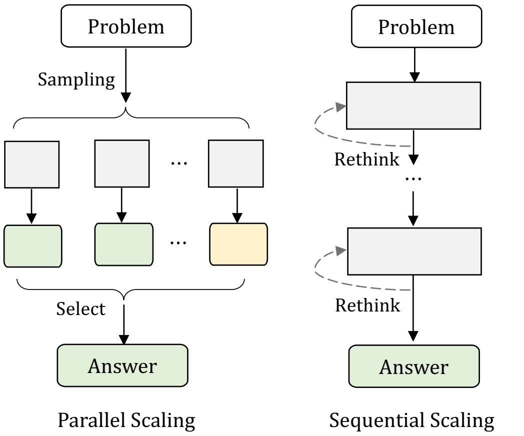


Figure 1: Conceptual illustration of parallel scaling vs. sequential scaling strategy.

Hybrid and Other Scaling Strategies. Recognizing the limitations of purely sequential or parallel methods, hybrid approaches attempt to combine their strengths. Some methods integrate verification steps within the generation process [9, 3, 22]. Reinforcement learning has also been employed to control the reasoning process or optimize test-time compute allocation [1, 19, 4]. For example, [7] proposed aligning reasoning with logic units. [6] investigated multi-agent collaborative reasoning. Recent work has also explored adaptive or hybrid approaches that try to combine the benefits of both parallel and sequential methods or dynamically allocate compute [18, 23, 10, 19, 17, 14]. They primarily investigated the role of verifiers or reinforcement learning to guide the scaling process [1, 22, 21, 3].

However, a fundamental question that remains underexplored is the inherent limit of performance gain from these test-time scaling strategies, particularly for verifier-free approaches. While it is intuitive that more computation should lead to better performance, the relationship is unlikely to be linear indefinitely. Understanding when these scaling efforts hit a point of diminishing returns—a *scaling Pareto*—is critical for efficient resource utilization. Our work attempts to address this gap by theoretically modeling this trend and deriving explicit upper bounds (saturation points) on the computational budget for both parallel and sequential scaling, beyond which further scaling offers minimal performance gains. This provides a principled basis for optimizing test-time compute allocation, complementing existing empirical explorations of TTS efficiency.

3 Test-Time Scaling Performance Model

Building on the premise that large reasoning models (LRMs) can be further improved by scaling test-time computation, but that such improvements are not limitless, we aim to build a theoretical **T**est-**T**ime **S**caling **P**erformance **M**odel (**TTSPM**) to quantify reasoning performance gain varying by scaling budget. TTSPM captures the saturation behavior common to various test-time scaling strategies, despite their differing underlying mechanisms. From this model, we derive a general upper bound (saturation point), identifying the scaling budget beyond which further scaling efforts yield diminishing returns below a practical threshold.

3.1 Probabilistic Modeling of Test-Time Scaling

Despite test-time scaling strategies differing in their fine-grained dynamics, a common macro-level phenomenon is widely observed: as more generations per problem are invested at test time, the performance tends to improve but eventually saturates, approaching a practical maximum. To capture this overarching saturation behavior in a general manner, we start from two fundamental scaling paradigms: *parallel scaling* and *sequential scaling*.

Parallel Scaling: Sampling to Selection. Given an input problem q and a large reasoning model $\mathcal{M}$, parallel scaling consists of two crucial steps (as shown in Figure 1): generating N candidate answers $\{a_i\}_{i=1}^N$ (i.e., sampling) and then applying a majority voting [26] mechanism or taking a verifier [25] to select the best answer based on the scores of these candidates (i.e., selection strategy, denoted as $\mathcal{V}$). Each answer a_i is either correct ($c_i = 1$) or incorrect ($c_i = 0$). The probability of any single candidate answer a_i being deemed correct is given by:

$$p_{\text{sample}} = P(c_i = 1 | q, \mathcal{M}, \mathcal{V}). \tag{1}$$

Let K be the number of correct answers among these N candidates. Since each answer is independently assessed with probability p_{sample} of being correct, K follows a Binomial distribution. We are interested in the event that at least one answer is correct, i.e., $K \geq 1$. The probability is given by:

$$P(K \geq 1) = 1 - P(K = 0), \tag{2}$$

where $P(K = 0)$ is the probability that all N answers are incorrect. Due to the independence of parallel sampling, we have:

$$P(K = 0) = (P(c_i = 0))^N = (1 - p_{\text{sample}})^N. \tag{3}$$

Thus, the probability of finding at least one correct answer is:

$$P(K \geq 1) = 1 - (1 - p_{\text{sample}})^N. \tag{4}$$

Sequential Scaling: Round-by-round Rethinking. Sequential scaling is concluded as asking the model to iteratively rethink its previous answer round by round (see Figure 1). We model this as a discrete-time Markov process with two states: the current answer (denoted as s_0) and the correct answer (denoted as s_1). State s_1 is an absorbing state, meaning once a correct answer is reached, the process stops and remains correct. Given the model $\mathcal{M}$ and the input problem q, let p_{rethink} be the probability of transitioning from an any state s_0 to the correct state s_1 in a single round:

$$p_{\text{rethink}} = P(s_t = s_1 | s_{t-1} = s_0, q, \mathcal{M}). \tag{5}$$

This probability p_{rethink} quantifies the model's intrinsic capability to self-correct or improve its answer within a single iteration. Consequently, the probability of remaining in state s_0 (i.e., failing to correct the answer in one round) is $1 - p_{\text{rethink}}$. Let K be the random variable representing the number of rethinking rounds required to first reach the correct state S_1. For the process to first reach S_1 at round k (where $k \geq 1$), it must have remained in S_0 for the preceding $k - 1$ rounds and then transitioned to S_1 at the k-th round. The probability of this sequence of events is $P(K = k) = (1 - p_{\text{rethink}})^{k-1} p_{\text{rethink}}$. This indicates that K follows a geometric distribution with success parameter p_{rethink}.

We are primarily interested in the cumulative probability of achieving a correct answer within a maximum of N rethinking rounds, denoted $P(K \leq N)$. This is the sum of probabilities of first reaching S_1 in any round from 1 to N:

$$P(K \leq N) = \sum_{k=1}^{N} P(K = k) = \sum_{k=1}^{N} (1 - p_{\text{rethink}})^{k-1} p_{\text{rethink}}. \tag{6}$$

This sum represents the first N terms of a geometric series. The first term is p_{rethink} (for $k = 1$) and the common ratio is $(1 - p_{\text{rethink}})$. The sum evaluates to:

$$\begin{aligned} P(K \leq N) &= p_{\text{rethink}} \frac{1 - (1 - p_{\text{rethink}})^N}{1 - (1 - p_{\text{rethink}})} \\ &= p_{\text{rethink}} \frac{1 - (1 - p_{\text{rethink}})^N}{p_{\text{rethink}}} \\ &= 1 - (1 - p_{\text{rethink}})^N. \end{aligned} \tag{7}$$

This equation provides the probability of attaining a correct answer via sequential rethinking within N rounds. This formulation is structurally analogous to the success probability in parallel scaling, $1 - (1 - p_{\text{sample}})^N$, differing primarily in the interpretation of the base success probability.

3.2 Performance Function to Scaling Pareto

Performance Function. Without loss of generality, we let p_x denote the probability of attaining a correct answer, either by parallel scaling (p_{sample}) or by sequential scaling (p_{rethink}). We define N as the *scaling budget*, which refers to the number of generations per problem, with each generation denoted as a scaling unit. We use $F(N)$ to represent the model performance (e.g., Hit@k or Pass@k), given by:

$$F(N) = F_{\text{max}} \cdot (1 - (1 - p_x)^N), \tag{8}$$

where F_{max} ($0 < F_{\text{max}} \leq 1$) represents the theoretical maximum performance achievable by the model $\mathcal{M}$ on the specific task, even with an unbounded scaling budget. It is determined by the inherent capabilities and potential limitations of the model and training for the task at hand.

Marginal Performance Gain. To precisely identify the onset of diminishing returns, we introduce the marginal performance gain, $\Delta F(N)$, which is the additional performance improvement obtained by increasing the scaling budget from N to $N + 1$ units. This is given by:

$$\begin{aligned}
\Delta F(N) &= F(N+1) - F(N) \\
&= F_{\text{max}} \cdot (1 - (1 - p_x)^{N+1}) - F_{\text{max}} \cdot (1 - (1 - p_x)^N) \\
&= F_{\text{max}} \cdot [(1 - p_x)^N - (1 - p_x)^{N+1}] \\
&= F_{\text{max}} \cdot (1 - p_x)^N [1 - (1 - p_x)] \\
&= F_{\text{max}} \cdot p_x \cdot (1 - p_x)^N.
\end{aligned} \tag{9}$$

Eq. (9) shows that $\Delta F(N)$ decreases exponentially as N increases (since $0 < 1 - p_x < 1$). This exponential decay signifies that each successive scaling unit contributes progressively less to the overall performance improvement than its predecessor.

Derivation of Scaling Pareto. We define the *scaling Pareto* as the operational region where this marginal performance gain, $\Delta F(N)$, becomes practically insignificant. Specifically, we posit that further scaling is inefficient if $\Delta F(N)$ falls below a predefined small positive threshold ϵ. This threshold ϵ represents the minimum performance improvement deemed worthwhile for the cost of an additional scaling unit. Our objective is to identify an upper bound for the scaling budget, N_{upper}, such that for any $N \geq N_{\text{upper}}$, the marginal gain $\Delta F(N)$ is consistently less than the chosen threshold ϵ. This condition is formally expressed as:

$$F_{\text{max}} \cdot p_x \cdot (1 - p_x)^N < \epsilon. \tag{10}$$

To solve for N, we rearrange the terms:

$$(1 - p_x)^N < \frac{\epsilon}{F_{\text{max}} \cdot p_x}. \tag{11}$$

For this inequality to have a meaningful solution for $N \geq 1$ and for the subsequent logarithmic operations to be well-defined, we require $0 < \frac{\epsilon}{F_{\text{max}} \cdot p_x} < 1$. This condition implies that $\epsilon < F_{\text{max}} \cdot p_x$, meaning the chosen threshold for negligible gain must be smaller than the initial marginal gain provided by the first scaling unit (i.e., $\Delta F(0) = F_{\text{max}} \cdot p_x$). If this condition is not met (i.e., $\epsilon \geq F_{\max} \cdot p_x$), it signifies that even the very first scaling unit does not provide a sufficient performance boost, and thus N_{upper} could be considered to be 0 or 1.

Assuming $\epsilon < F_{\max} \cdot p_x$, we take the natural logarithm of both sides of Eq. (11):

$$N \ln(1 - p_x) < \ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right). \tag{12}$$

Given that $0 < p_x < 1$, it follows that $0 < 1 - p_x < 1$, which makes $\ln(1 - p_x)$ a negative value. Therefore, when dividing by $\ln(1 - p_x)$, the direction of the inequality sign is reversed:

$$N > \frac{\ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right)}{\ln(1 - p_x)}. \tag{13}$$

According to the right-hand side of Eq. (13), the smallest integer N that satisfies this inequality represents the point at which the marginal gain consistently drops below ϵ. We define this as the unified upper bound (or saturation point), N^*, which is given by:

$$N^* = \left\lceil \frac{\ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right)}{\ln(1 - p_x)} \right\rceil. \tag{14}$$

This N^* signifies the scaling budget beyond which scaling additional units is expected to yield performance improvements below the threshold ϵ, effectively marking the scaling Pareto.

Insights into the Scaling Pareto. The core finding of this paper is that both parallel and sequential test-time scaling strategies, despite their operational differences, exhibit a performance saturation phenomenon—the *scaling Pareto*—that can be characterized by a unified mathematical structure, as induced in Eq. (14). This suggests a fundamental principle at play: as more computational units (N) are invested at test time, the probability of encountering an unsolved problem that can be successfully addressed by the next unit diminishes. The exponential decay term $(1 - p_x)^N$ in our marginal gain formula (Eq. (9)) captures this essence. The derived saturation point, N^*, provides a quantitative tool to identify when the expected marginal gain drops below a practical threshold ϵ, signaling that further scaling is unlikely to be cost-effective.

3.3 Practical Considerations and Parameter Estimation

Applying the scaling budget bound in practice requires estimating the model parameters of the maximum performance $F_{\max}$, effective success probability p_x (either p_{sample} or p_{rethink}), and setting a suitable gain threshold ϵ. We observe that: 1) $F_{\max}$ is challenging to determine precisely. It can be empirically estimated by running the LRM with a very large N on a representative validation set and observing the performance asymptote. Alternatively, it could be set based on known human performance levels or theoretical limits for a given task. The choice of $F_{\max}$ will influence N^*; a higher $F_{\max}$ (if achievable) might justify a larger N^*. 2) p_x is specific to the scaling strategy and the model-task combination. For parallel scaling (p_{sample}), it can be estimated from the initial rate of performance increase with N or by analyzing the success rate of individual samples on a validation set. For sequential scaling (p_{rethink}), it reflects the probability of successful correction/improvement per step, which can also be estimated from validation data. These parameters might not be constant across all problem instances, which is a simplification in our current model. 3) ϵ is a user-defined hyperparameter reflecting the acceptable trade-off between performance gain and computational cost. A smaller ϵ will lead to a larger N^*, indicating a willingness to invest more compute for smaller gains. Its choice depends on the application context, available resources, and latency requirements.

4 Experiments and Analysis

4.1 Experimental Setup

Benchmarks and Backbone Models. We experiment on multiple representative reasoning benchmarks. **AIME 2024** [11] and **AIME 2025**[2]: Each contains 30 pre-Olympiad level problems from the American Invitational Mathematics Examination, designed to test advanced mathematical reasoning. **MATH-500** [5]: A challenging subset of the MATH dataset with 500 high school competition problems across algebra, geometry, and so on. **GPQA** [20]: A science-domain question answering benchmark involving PhD-level science questions. We employ `DeepSeek-R1-Distill-Qwen-1.5B` [4] and `DeepSeek-R1-Distill-Qwen-7B` [4] as our large reasoning models for investigation.

Test-Time Scaling Setup. We implement parallel scaling based on Self-Consistency [26], which generates N candidate answers and applies a majority voting to select the final answer. We implement sequential scaling by asking the model to rethink its previous answer and obtain the final answer round by round, similar to Multi-round Thinking [24]. To assess the test-time scaling performance of different methods, we vary the number of generations per problem (i.e., N) from 1 to 32. For all models, we use a sampling temperature of 0.6 and a top-p of 0.95, following the suggested parameter

[2] `https://huggingface.co/datasets/math-ai/aime25`.

Table 1: Comparison of different test-time scaling methods on several representative reasoning benchmarks. “R1-1.5B” and “R1-7B” represent `DeepSeek-R1-Distill-Qwen-1.5B` and `DeepSeek-R1-Distill-Qwen-7B`, respectively. “Seq.” and “Par.” are short for *sequential* scaling and *parallel* scaling, respectively.

Model	Method	AIME 2024		AIME 2025		MATH-500		GPQA	
		Acc.	Hit@N	Acc.	Hit@N	Acc.	Hit@N	Acc.	Hit@N
R1-1.5B	Vanilla	30.0	30.0	23.3	23.3	74.2	74.2	21.3	21.3
	Seq. ($N = 4$)	36.7	43.3	23.3	30.0	78.4	86.6	34.8	44.4
	Seq. ($N = 8$)	26.7	46.7	26.7	33.3	76.9	88.2	35.4	51.0
	Seq. ($N = 32$)	30.0	70.0	26.7	43.3	80.6	90.6	35.4	72.2
	Par. ($N = 4$)	30.0	56.7	30.0	33.3	79.2	86.4	39.4	71.7
	Par. ($N = 8$)	46.7	60.0	33.3	40.0	81.6	89.2	**39.9**	85.4
	Par. ($N = 32$)	**53.3**	**80.0**	**40.0**	**56.7**	**83.6**	**91.8**	39.4	**96.5**
R1-7B	Vanilla	56.7	56.7	40.0	40.0	82.0	82.0	41.9	41.9
	Seq. ($N = 4$)	63.3	70.0	53.3	60.0	85.8	88.2	41.4	55.6
	Seq. ($N = 8$)	63.3	76.7	40.0	60.0	85.0	88.2	44.9	55.6
	Seq. ($N = 32$)	73.3	76.7	43.3	60.0	86.4	89.6	46.5	58.6
	Par. ($N = 4$)	73.3	76.7	53.3	60.0	85.8	88.6	55.1	75.8
	Par. ($N = 8$)	**76.7**	80.0	**60.0**	60.0	85.4	90.4	54.5	85.5
	Par. ($N = 32$)	**76.7**	**86.7**	53.3	**70.0**	**86.8**	**92.2**	**56.6**	**90.4**

settings [4]. We set the maximum length to 32,000, allowing the model to perform sufficient reasoning. All experiments are conducted in one NVIDIA A6000 server with 8 GPUs. We report the metrics of Accuracy (Acc.) and Hit@N. Accuracy measures the percentage of problems solved correctly, while Hit@N evaluates the proportion of problems for which at least one correct solution is found among N generated outputs.

4.2 Main Results of Test-Time Scaling

As shown in Table 1, both sequential and parallel test-time scaling strategies significantly improve the performance of large reasoning models across all benchmarks. First, we observe that test-time scaling consistently enhances model performance regardless of model size. For the smaller 1.5B model, parallel scaling with $N = 32$ improves accuracy by up to 23.3 percentage points on AIME 2024 (from 30.0% to 53.3%) and percentage points on GPQA (from 21.3% to 39.4%). Similarly, the larger 7B model shows substantial gains, with improvements of up to 20.0 percentage points on AIME 2024 (from 56.7% to 76.7%) and 14.7 percentage points on GPQA (from 41.9% to 56.6%). Second, parallel scaling consistently outperforms sequential scaling across all benchmarks and model sizes. This performance gap is particularly pronounced for the 1.5B model on AIME 2024, where parallel scaling with $N = 32$ achieves 53.3% accuracy compared to 30.0% with sequential scaling at the same budget. The superior performance of parallel scaling can be attributed to its ability to explore a more diverse solution space through independent sampling, whereas sequential scaling may suffer from error propagation or local optima in the refinement process.

4.3 Analysis of the Scaling Pareto

Figure 2 shows the scaling curves across different benchmarks, model sizes, and scaling strategies, exhibiting a consistent pattern of initial rapid improvement followed by a Pareto, confirming our theoretical model. We conclude that several important insights emerge from these curves: 1) **The scaling Pareto is a universal phenomenon observed across all experimental settings**. For both the 1.5B and 7B models, performance improvements become increasingly marginal as the number of generations increases, eventually reaching a point where additional scaling yields negligible gains. This empirical observation strongly validates our theoretical derivation of the scaling Pareto. 2) **Through test-time scaling, a smaller model can approach and sometimes even surpass the performance of a larger model without scaling**. On MATH-500, the 1.5B model with parallel scaling at $N = 32$ achieves 83.6% accuracy, exceeding the 7B model’s vanilla performance of 82.0%.

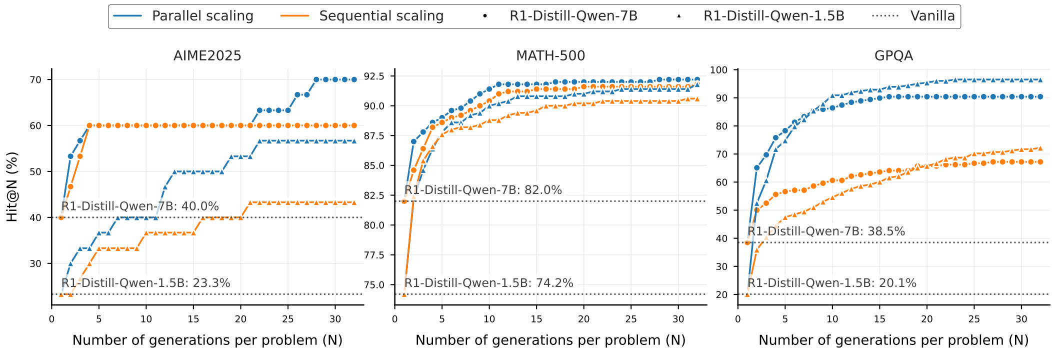


Figure 2: Scaling curves across three benchmarks (including AIME 2025, MATH-500, and GPQA), illustrating how Hit@N varies with the number of generations per problem (N) under different scaling strategies (parallel vs. sequential) and model sizes.

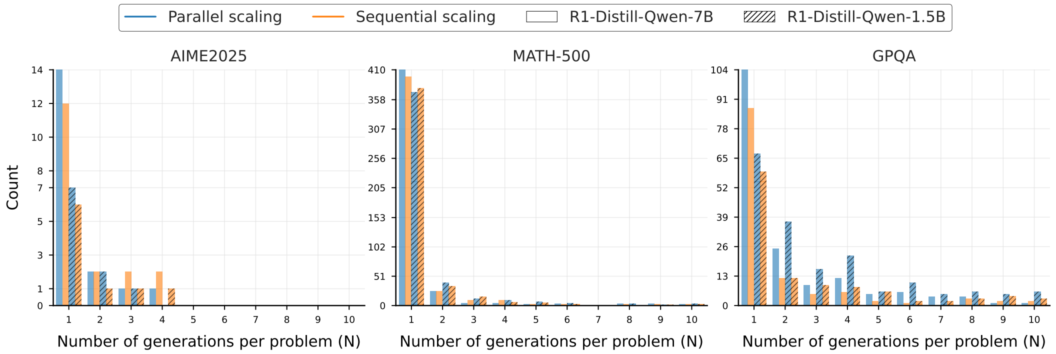


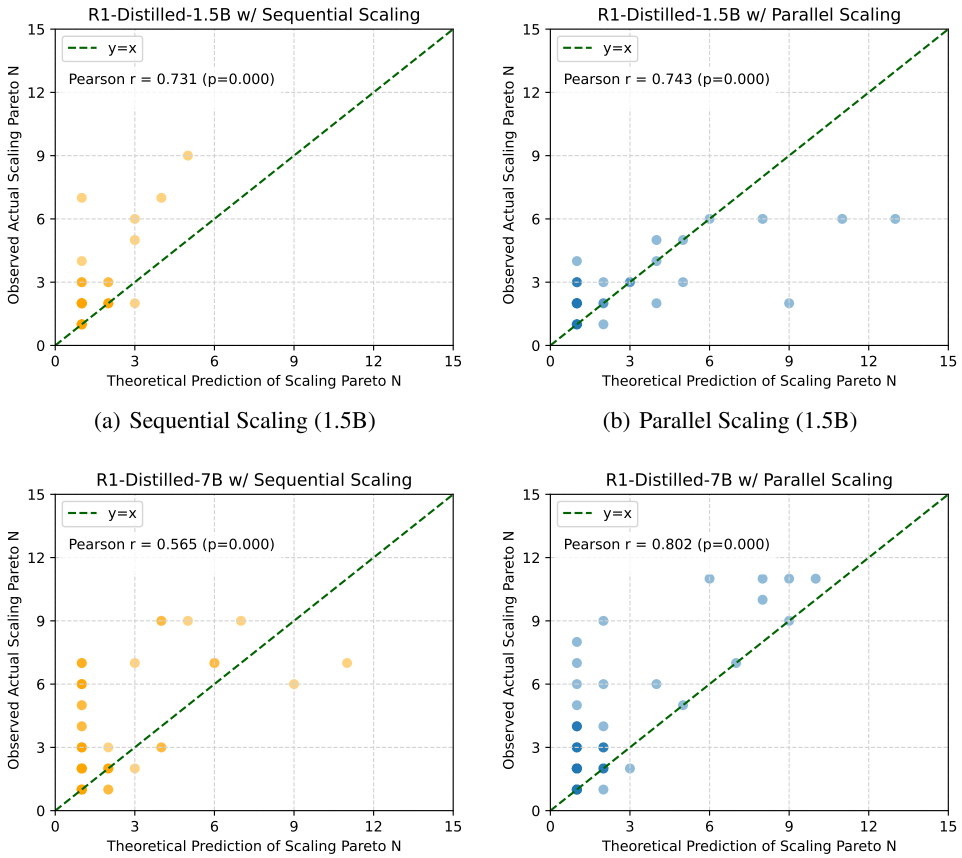
Figure 3: Statistics of the number of generations per problem (N) required to reach saturation points, comparing different scaling strategies (parallel vs. sequential) and model sizes.

This demonstrates that test-time scaling can be a cost-effective alternative to model size scaling, offering a practical approach to enhance reasoning capabilities without the computational burden of training larger models. 3) **The distribution discrepancy of different benchmark results in distinct scaling characteristics**. GPQA shows the steepest initial improvement, particularly for parallel scaling, suggesting that this benchmark benefits most from diverse solution exploration. In contrast, AIME2025 exhibits more gradual improvements, indicating that the problems in this benchmark may require more focused refinement rather than broad exploration. Further accuracy scaling curves presented in Appendix C demonstrate similar insights.

Figure 3 outlines the statistics of the number of generations per problem required to reach saturation points. We observe that the distribution of optimal scaling budgets varies significantly across benchmarks and scaling strategies. For AIME 2025, both models show a strong concentration at $N = 1$, indicating that many problems either can be solved in the first attempt or remain challenging regardless of additional scaling. In contrast, GPQA exhibits a more dispersed distribution, particularly for the 1.5B model with sequential scaling, suggesting that problems in this benchmark benefit from varying degrees of computational investment.

4.4 Verification of the Scaling Pareto

To validate our theoretical model, TTSPM, we compare the predicted scaling Pareto with the empirically observed optimal scaling budget for each problem in the MATH-500 dataset. To achieve a theoretical prediction of the scaling Pareto of N, we should first estimate the value of the probability p_x, so that we can employ Eq. (14) to calculate the saturation point of the scaled units. We randomly split the MATH-500 dataset into a "validation set" and a "test set" in a proportion of 8:2. We then utilize the "validation set" to estimate the unknown entry $\frac{\epsilon}{F_{\max}}$. With this, we can calculate the predicted scaling Pareto on the test set. Further details are presented in Appendix A.



(a) Sequential Scaling (1.5B)

(b) Parallel Scaling (1.5B)

(c) Sequential Scaling (7B)

(d) Parallel Scaling (7B)

Figure 4: Correlation between the theoretically predicted and empirically observed scaling saturation points (i.e., the number of generations per problem N) across different scaling strategies (sequential vs. parallel) and model sizes (1.5B vs. 7B).

Figure 4 presents the correlation between the theoretically predicted and the practically observed scaling saturation points in the MATH-500 dataset. The strong positive correlations observed in all different models and scaling strategies provide compelling evidence for the predictive power of our theoretical model. Parallel scaling shows particularly strong correlations, with Pearson's correlation coefficient r = 0.743 for the 1.5B model and r = 0.802 for the 7B model. These correlations indicate that our theoretical model can effectively predict the point at which additional scaling becomes inefficient, allowing us to make informed decisions about computational resource allocation. The stronger correlation observed for parallel scaling suggests that our model is particularly well-suited for predicting the scaling behavior of independent sampling approaches. Interestingly, we observe that the 7B model with parallel scaling (Figure 4(d)) shows the highest correlation, indicating that larger, more stable models may be more predictable in their scaling behavior. This finding has important implications for resource allocation in large-scale deployments, where accurate prediction of scaling benefits can lead to significant computational savings.

4.5 Discussion on Limitations

Our theoretical model, while providing valuable insights, relies on several simplifying assumptions: 1) **Independence of Scaling Units.** The derivation $1 - (1 - p_x)^N$ assumes that the "success events" associated with each scaling unit are conditionally independent given the current state. While this is a common modeling assumption, complex dependencies might exist, especially in sequential scaling, where the outcome of one step heavily influences the next. 2) **Definition of "Success" and p_x.** The precise definition and empirical estimation of an "effective success probability" p_x can be nuanced.

It encapsulates not just the model's ability to generate a correct step/sample but also any implicit selection or aggregation mechanism (e.g., majority voting in self-consistency, or the criteria for a successful rethink). 3) **Verifier-Free Assumption.** Our primary focus is on scaling strategies that do not rely on external verifiers. The dynamics might change if a strong verifier is available to prune or guide the search, potentially altering the p_x values or even the form of the performance curve.

5 Conclusion and Future Work

In this paper, we investigate the *scaling Pareto* phenomenon in test-time computation for large reasoning models, developing a unified theoretical test-time scaling performance model (TTSPM) that characterizes the point at which additional computational units yield diminishing returns. Our key contribution is the derivation of a general upper bound (saturation point) with identical mathematical structure for both parallel and sequential scaling strategies, suggesting a fundamental principle governs test-time scaling limits regardless of mechanism. Experiments on challenging reasoning benchmarks validate our theoretical bounds, enabling practitioners to make informed decisions about cost-benefit trade-offs in test-time scaling. In the future, we aim to enhance our method in several key aspects. Currently, our TTSPM is task- and model-specific, requiring careful instantiation and parameter estimation for each new setting. To address this limitation, we plan to adapt TTSPM to support rapid parameter estimation across diverse scenarios. Additionally, we intend to explore hybrid scaling strategies, develop dynamic parameter estimation techniques, and extend the core principles of our approach to a broader range of reasoning tasks.

References

[1] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

[2] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[3] Jiefeng Chen, Jie Ren, Xinyun Chen, Chengrun Yang, Ruoxi Sun, and Sercan Ö Arık. Sets: Leveraging self-verification and self-correction for improved test-time scaling. *arXiv preprint arXiv:2501.19306*, 2025.

[4] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[5] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[6] Can Jin, Hongwu Peng, Qixin Zhang, Yujin Tang, Dimitris N Metaxas, and Tong Che. Two heads are better than one: Test-time scaling of multi-agent collaborative reasoning. *arXiv preprint arXiv:2504.09772*, 2025.

[7] Cheryl Li, Tianyuan Xu, and Yiwen Guo. Reasoning-as-logic-units: Scaling test-time reasoning in large language models through logic unit alignment. *arXiv preprint arXiv:2502.07803*, 2025.

[8] Yansi Li, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Qiuzhi Liu, Rui Wang, Zhuosheng Zhang, Zhaopeng Tu, Haitao Mi, et al. Dancing with critiques: Enhancing llm reasoning with stepwise natural language self-critique. *arXiv preprint arXiv:2503.17363*, 2025.

[9] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Qin Liu, Wenxuan Zhou, Nan Xu, James Y Huang, Fei Wang, Sheng Zhang, Hoifung Poon, and Muhao Chen. Metascale: Test-time scaling with evolving meta-thoughts. *arXiv preprint arXiv:2503.13447*, 2025.

[11] MAA. American invitational mathematics examination - aime. `https://huggingface.co/datasets/AI-MO/aimo-validation-aime`, February 2024.

[12] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594, 2023.

[13] Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. Deepseek-r1 thoughtology: Let's< think> about llm reasoning. *arXiv preprint arXiv:2504.07128*, 2025.

[14] Lingrui Mei, Shenghua Liu, Yiwei Wang, Baolong Bi, Yuyao Ge, Jun Wan, Yurong Wu, and Xueqi Cheng. a1: Steep test-time scaling law via environment augmented generation. *arXiv preprint arXiv:2504.14597*, 2025.

[15] OpenAI. Introducing OpenAI o1. `https://openai.com/o1/`, September 2024.

[16] OpenAI. OpenAI o3-mini system card. `https://cdn.openai.com/o3-mini-system-card-feb10.pdf`, January 2025.

[17] Daniele Paliotta, Junxiong Wang, Matteo Pagliardini, Kevin Y Li, Aviv Bick, J Zico Kolter, Albert Gu, François Fleuret, and Tri Dao. Thinking slow, fast: Scaling inference compute with distilled reasoners. *arXiv preprint arXiv:2502.20339*, 2025.

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

[19] Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*, 2025.

[20] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024.

[21] Amrith Setlur, Nived Rajaraman, Sergey Levine, and Aviral Kumar. Scaling test-time compute without verification or rl is suboptimal. *arXiv preprint arXiv:2502.12118*, 2025.

[22] Wenlei Shi and Xing Jin. Heimdall: test-time scaling on the generative verification. *arXiv preprint arXiv:2504.10337*, 2025.

[23] Zhendong Tan, Xingjun Zhang, Chaoyi Hu, Yancheng Pan, and Shaoxun Wang. Adaptive rectification sampling for test-time compute scaling. *arXiv preprint arXiv:2504.01317*, 2025.

[24] Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yunjie Ji, Yiping Peng, Han Zhao, and Xiangang Li. Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. *arXiv preprint arXiv:2503.19855*, 2025.

[25] Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, 2024.

[26] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

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

[28] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[29] Yuchen Yan, Yongliang Shen, Yang Liu, Jin Jiang, Mengdi Zhang, Jian Shao, and Yueting Zhuang. Inftythink: Breaking the length limits of long-context reasoning in large language models. *arXiv preprint arXiv:2503.06692*, 2025.

[30] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822, 2023.

[31] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

[32] Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin King, Xue Liu, and Chen Ma. What, how, where, and how well? a survey on test-time scaling in large language models. *arXiv preprint arXiv:2503.24235*, 2025.

A Experimental Details

Our theoretical model for the scaling saturation point, N^*, relies on the effective probability of success for a single scaling unit, $p(x)$, which varies across problems. We estimate this probability for each problem in both validation and test sets using experimental data from up to $N = 32$ generations per problem.

- **For Parallel Scaling**: The effective probability $p(x)$ corresponds to $p_{\text{sample}}(x)$, the probability that a single candidate answer for problem x is correct. We estimate this by:

$$\hat{p}_{\text{sample}}(x) = \frac{\text{Number of correct answers for problem } x \text{ in 32 samples}}{32}$$

- **For Sequential Scaling**: The effective probability $p(x)$ corresponds to $p_{\text{rethink}}(x)$, the probability of transitioning from any state to the correct state in a single rethinking round. For a problem first solved at round $k_x \leq 32$, we use the maximum likelihood estimate for a geometric distribution:

$$\hat{p}_{\text{rethink}}(x) = \begin{cases} k_x/32 & \text{if problem } x \text{ is first solved at round } k_x \leq 32 \\ 1e-5 & \text{if problem } x \text{ is not solved within 32 rounds} \end{cases}$$

  This approach aligns with our theoretical model, where $p(x)$ represents the single-step success probability in the Markov process. For problems not solved within 32 rounds, we assign a small non-zero probability to avoid mathematical issues in subsequent logarithmic calculations.

These estimations are performed for every problem in both the validation and test sets, providing the problem-specific parameters needed for our scaling Pareto predictions.

B Probabilistic Modeling

This appendix provides supplementary mathematical details for the derivations presented in Section 3.

B.1 Detailed Derivation of Marginal Performance Gain

Recall the unified performance model from Eq. (8):

$$F(N) = F_{\max} \cdot (1 - (1 - p_x)^N) \tag{Eq. (8) revisited}$$

The marginal performance gain, $\Delta F(N)$, is defined as the difference in performance when increasing the computational budget from N to $N + 1$ units:

$$\Delta F(N) = F(N + 1) - F(N) \tag{15}$$

Substituting the performance model:

$$\begin{aligned} \Delta F(N) &= \left[F_{\max} \cdot (1 - (1 - p_x)^{N+1})\right] - \left[F_{\max} \cdot (1 - (1 - p_x)^N)\right] && (16) \\ &= F_{\max} \left[(1 - (1 - p_x)^{N+1}) - (1 - (1 - p_x)^N)\right] && (17) \\ &= F_{\max} \left[1 - (1 - p_x)^{N+1} - 1 + (1 - p_x)^N\right] && (18) \\ &= F_{\max} \left[(1 - p_x)^N - (1 - p_x)^{N+1}\right] && (19) \end{aligned}$$

We can factor out $(1 - p_x)^N$ from the terms in the bracket:

$$\begin{aligned} \Delta F(N) &= F_{\max} \cdot (1 - p_x)^N \left[1 - (1 - p_x)\right] && (20) \\ &= F_{\max} \cdot (1 - p_x)^N \left[1 - 1 + p_x\right] && (21) \\ &= F_{\max} \cdot p_x \cdot (1 - p_x)^N && (22) \end{aligned}$$

This is Eq. (9) in the main text.

B.2 Detailed Derivation of the Inequality for *N*

We start from the condition for the scaling Pareto (Eq (10)):

$$F_{\max} \cdot p_x \cdot (1 - p_x)^N < \epsilon \tag{10 revisited}$$

To solve for N, we first isolate the term $(1 - p_x)^N$. Assuming $F_{\max} > 0$ and $p_x > 0$, we can divide both sides by $F_{\max} \cdot p_x$ without changing the inequality direction (as this product is positive):

$$(1 - p_x)^N < \frac{\epsilon}{F_{\max} \cdot p_x} \tag{23}$$

This is Equation 11 in the main text.

For this inequality to be meaningful in the context of finding a positive N where performance gain diminishes, we require the right-hand side to be less than 1 (since $(1 - p_x)^N$ will be less than 1 for $N \geq 1$ and $0 < p_x < 1$). If $\frac{\epsilon}{F_{\max} \cdot p_x} \geq 1$, it means that even for $N = 0$ (or $N = 1$, depending on interpretation), the marginal gain $F_{\max} \cdot p_x$ is already less than or equal to ϵ, implying the Pareto is reached immediately. Thus, we assume $0 < \frac{\epsilon}{F_{\max} \cdot p_x} < 1$, which implies $\epsilon < F_{\max} \cdot p_x$.

Taking the natural logarithm of both sides:

$$\ln((1 - p_x)^N) < \ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right) \tag{24}$$

Using the logarithm property $\ln(a^b) = b\ln(a)$:

$$N\ln(1 - p_x) < \ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right) \tag{25}$$

Since $0 < p_x < 1$, it follows that $0 < 1 - p_x < 1$. The natural logarithm of a number between 0 and 1 is negative, so $\ln(1 - p_x) < 0$. When dividing an inequality by a negative number, the direction of the inequality sign must be reversed:

$$N > \frac{\ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right)}{\ln(1 - p_x)} \tag{26}$$

This is Eq. (13) in the main text. The unified upper bound N^* is then defined as the smallest integer satisfying this condition.

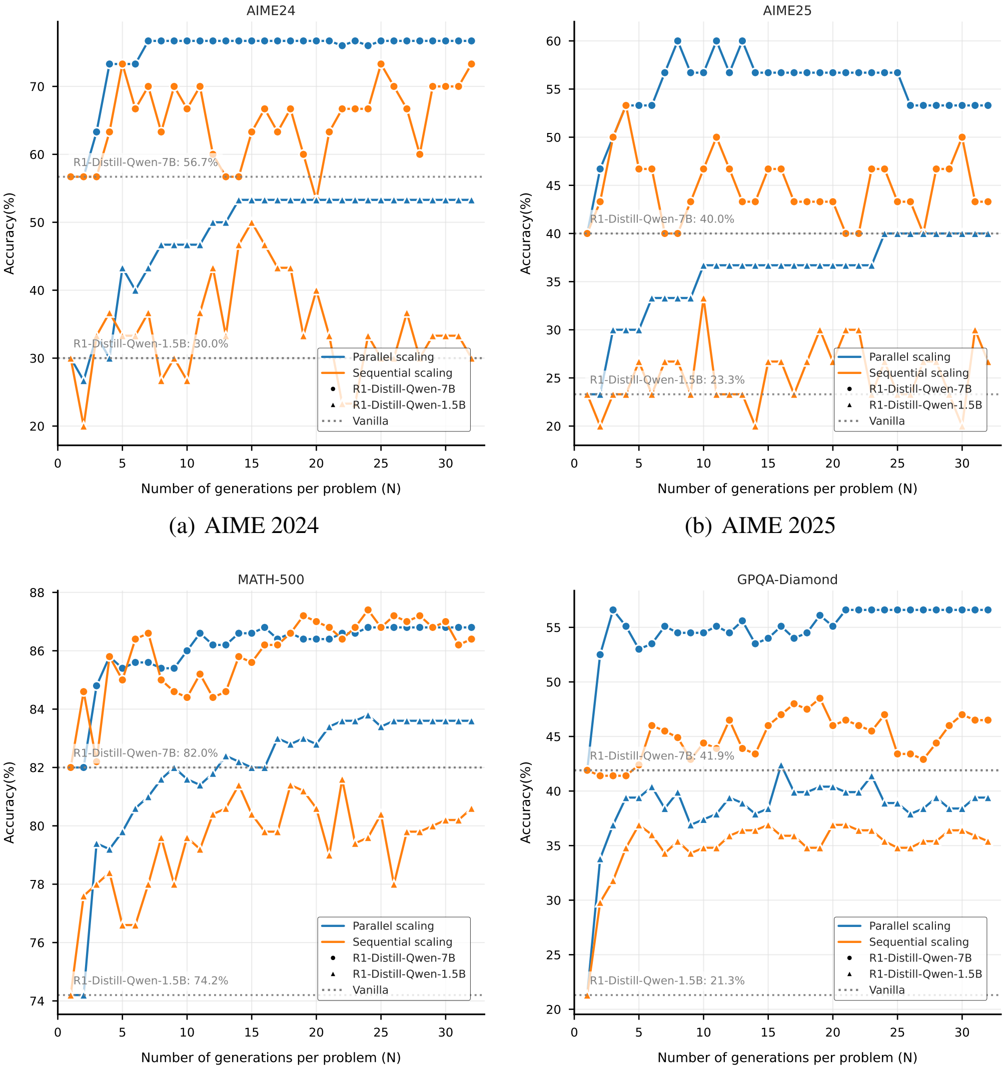
B.3 Condition for the Existence of a Non-trivial Scaling

As mentioned in Section 3.2 and elaborated above, for the derivation of N^* to yield a value $N > 0$ (or $N \geq 1$), we require the term $\frac{\epsilon}{F_{\max} \cdot p_x}$ to be less than 1. If $\frac{\epsilon}{F_{\max} \cdot p_x} \geq 1$, i.e., $\epsilon \geq F_{\max} \cdot p_x$, then the marginal gain from the very first computational unit ($N = 0$ to $N = 1$) is already less than or equal to the threshold ϵ. In such a scenario, $\ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right) \geq 0$. Since $\ln(1 - p_x)$ is negative, the fraction $\frac{\ln\left(\frac{\epsilon}{F_{\max} \cdot p_x}\right)}{\ln(1-p_x)}$ would be less than or equal to 0. The condition $N >$ non-positive value would be satisfied by any $N \geq 1$. In this case, N^* would effectively be 1 (or even 0, depending on interpretation), meaning no scaling beyond the initial state is justified if one strictly adheres to the ϵ threshold.

Therefore, the practical application of the N^* formula to find a value greater than 1 assumes that the initial potential gain $F_{\max} \cdot p_x$ is greater than the desired negligible gain threshold ϵ. This ensures that there is at least some initial phase where scaling provides a benefit exceeding ϵ.

C Additional Experimental Results

Figure 5 presents accuracy scaling curves across four benchmarks under different scaling strategies (parallel vs. sequential) and model sizes (1.5B vs. 7B). Across all settings, we observe a consistent trend: initial improvements in accuracy with increased generations per problem (N), followed by diminishing returns, forming a clear empirical scaling Pareto frontier.



(a) AIME 2024

(b) AIME 2025

(c) MATH-500

(d) GPQA

Figure 5: Accuracy scaling curves across four benchmarks (a) AIME 2024, (b) AIME 2025, (c) MATH-500, and (d) GPQA-Diamond, illustrating how accuracy varies with the number of generations per problem (N) under different scaling strategies (parallel vs. sequential) and model sizes.

In addition to the fact that the scaling Pareto phenomenon is consistent and universal across tasks, models, and strategies, the above results further lead to several key insights: 1) Test-time scaling enables smaller models to rival or outperform larger, unscaled models. For example, the 1.5B model with sufficient scaling often approaches or exceeds the performance of the 7B vanilla baseline, particularly under parallel scaling. This highlights the practical value of test-time scaling as a lightweight alternative to model scale. 2) Parallel scaling consistently outperforms sequential scaling in terms of final accuracy. While both strategies benefit from increased N, parallel scaling achieves better asymptotic performance and exhibits more stable behavior across all benchmarks. These findings not only align with our theoretical predictions but also suggest that scaling compute at test time, especially via parallel strategies, can be a powerful tool for improving performance without increasing model size.