Google DeepMind

2024-8-7

Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters

Charlie Snell[♦, 1], **Jaehoon Lee**[2], **Kelvin Xu**[♣, 2] **and Aviral Kumar**[♣, 2]

[♣]Equal advising, [1]UC Berkeley, [2]Google DeepMind, [♦]Work done during an internship at Google DeepMind

The benefits of “thinking” for longer in LLMs

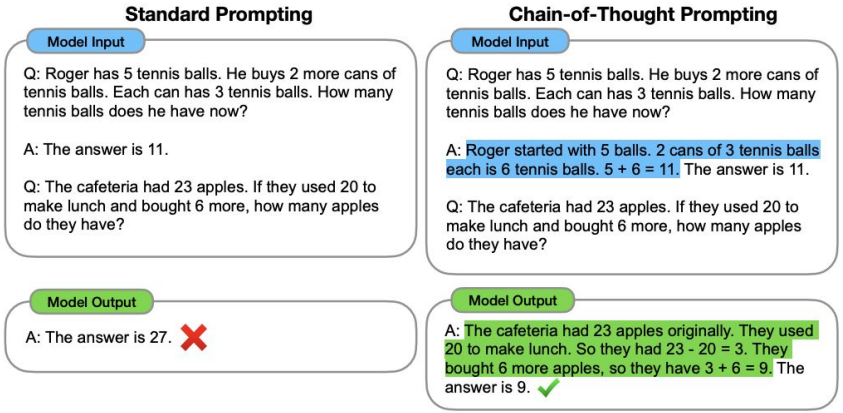


Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

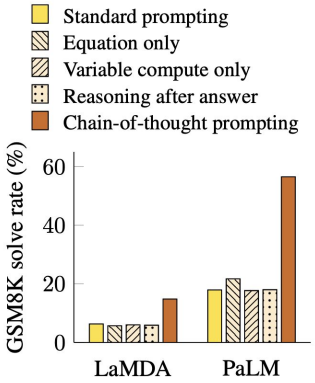
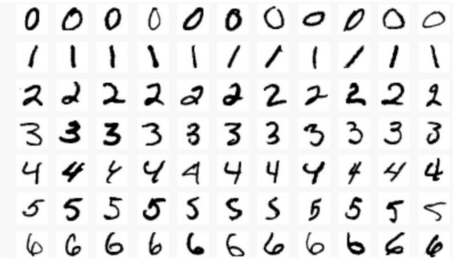


Figure 5: Ablation study for different variations of prompting using LaMDA 137B and PaLM 540B. Results for other datasets are given in Appendix Table 6 and Table 7.

[1] Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. Jason Wei et al 2023

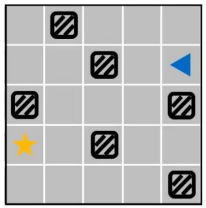
Scaling Test-Time

There are problems that benefit from being able to think for longer.

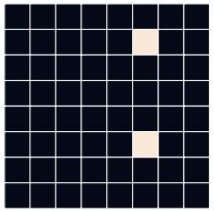


Question: What is the capital of Vietnam?
Answer: Hanoi

Sometime, we do the task subconsciously without thinking.



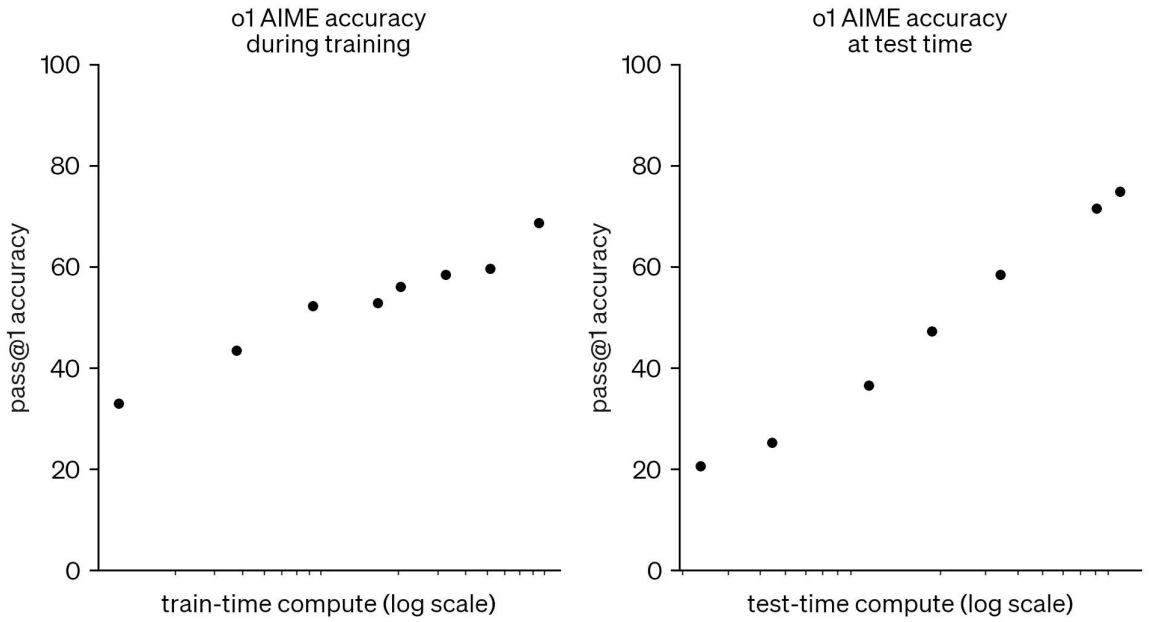
"Go to the star!"



Question: Let $k, l > 0$ be parameters. The parabola $y = kx^2 - 2kx + l$ intersects the line $y = 4$ at two points A and B. These points are distance 6 apart. What is the sum of the squares of the distances from A and B to the origin?
Answer: Step 1: Set up the intersection points…**Step 2:** Find the roots (coordinates of points A and B)...**Step 3:** Use the fact that the distance between A and B is 6…**Step 4:** Find the sum of the squares of the distances from A and B to the origin…

Sometime, we need to think/plan.

Scaling Test-Time



[1] Learning to Reason with LLMs. OpenAI's blog post (Sep 2024).

Common Approaches

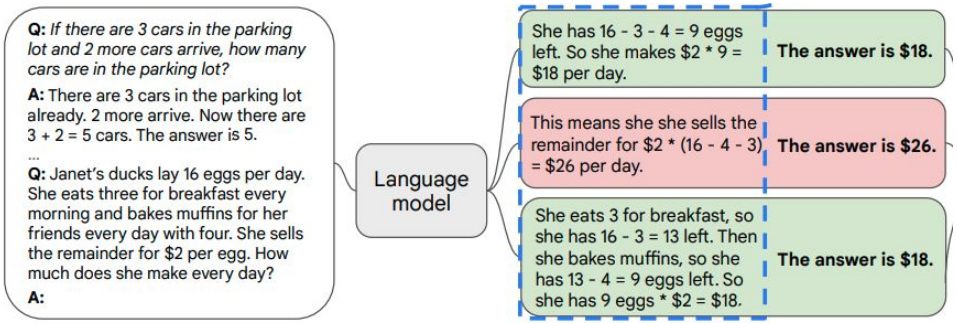
1. Scaling Test-Time Compute via **Verifiers**
2. **Refining** the Proposal Distribution

[1] SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

Common Approaches

1. Scaling Test-Time Compute via Verifiers

Proposer - LLMs
Verifier - Score models
- Solution level
- Step level



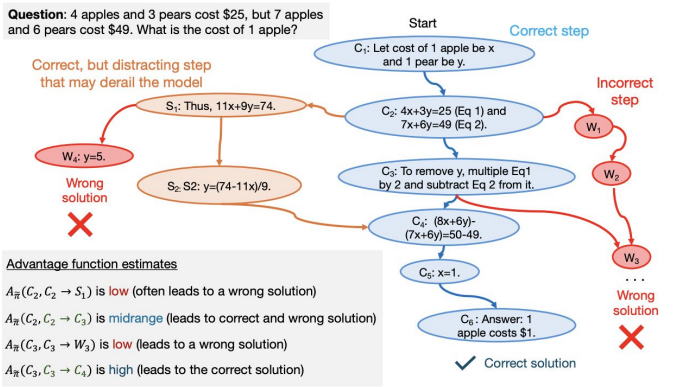


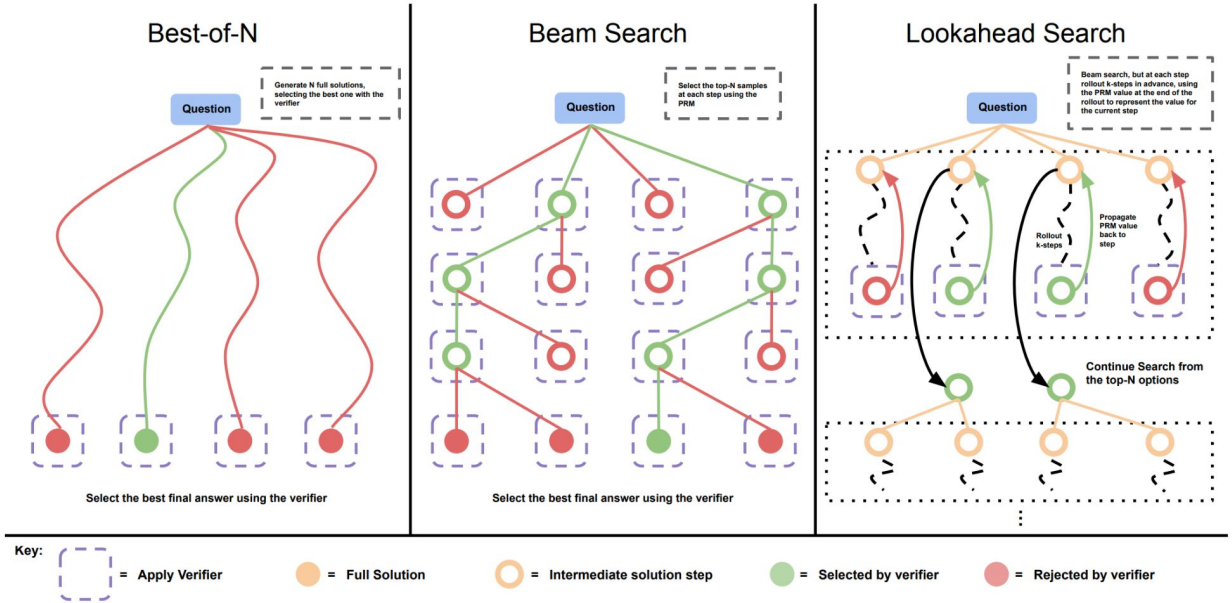
Figure 6: Illustration of advantage estimation from negative data on a didactic example in synthetic model generations. Critical steps are those with high advantage values.

[1] SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

Common Approaches

1. **Scaling Test-Time Compute via Verifiers**

Proposer, Verifier, and Search



Common Approaches

2. **Refining the Proposal Distribution**
   Self-correction

Problem. If $n \equiv 2 \pmod 7$, then find the remainder when $(n+2)(n+4)(n+6)$ is divided by 7.

***SCoRe* turn 1 solution (incorrect).** Since $n \equiv 2 \pmod 7$, we can write $n = 7k+2$ for some integer k. Substituting, we get

$$\begin{aligned}(n+2)(n+4)(n+6) &= (7k+4)(7k+6)(7k+8)\\ &\equiv 4 \cdot 6 \cdot 8 \pmod 7\\ &\equiv 192 \pmod 7\\ &\equiv 1 \pmod 7.\end{aligned}$$

Final Answer: The final answer is 1. I hope it is correct.

Self-correction instruction. There might be an error in the solution above because of lack of understanding of the question. Please correct the error, if any, and rewrite the solution.

***SCoRe* turn 2 solution (correct).** Since $n \equiv 2 \pmod 7$, we can write $n = 7k+2$ for some integer k. Substituting, we get

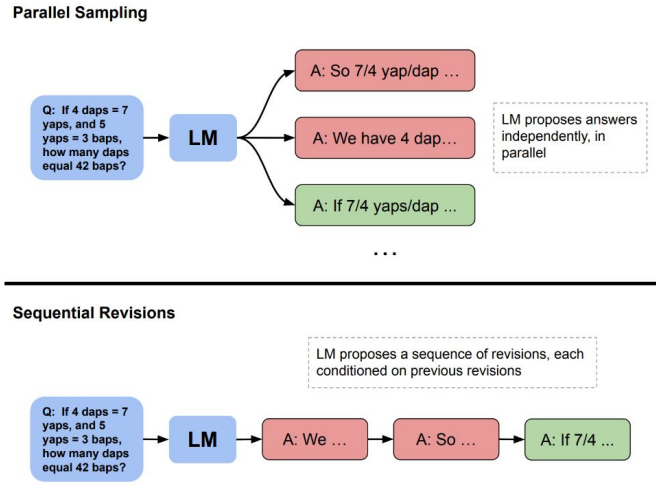
$$\begin{aligned}(n+2)(n+4)(n+6) &= (7k+4)(7k+6)(7k+8)\\ &\equiv 4 \cdot 6 \cdot 8 \pmod 7\\ &\equiv 4 \cdot 6 \cdot 1 \pmod 7\\ &\equiv 24 \pmod 7\\ &\equiv 3 \pmod 7.\end{aligned}$$

Final Answer: The final answer is 3. I hope it is correct.

[1] Training Language Models to self-correct

Common Approaches

1. **Scaling Test-Time Compute via Verifiers**
2. **Refining the Proposal Distribution**



[1] SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

*“Can we enable LMs to most **effectively make use of additional computation at test time** so as to improve the accuracy of their response?”*

Datasets, Models, and Metrics

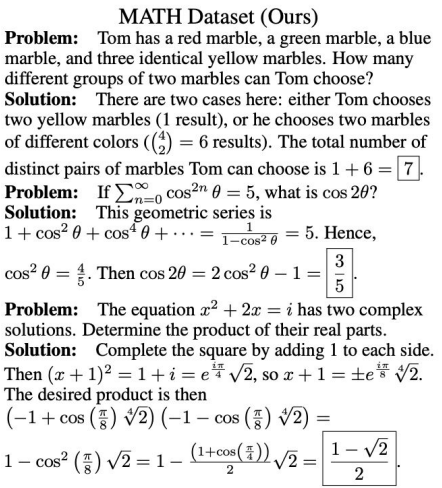
MATH benchmark [1]

high-school competition level math problems with a range of difficulty levels

Fine-tuned PaLM-2 models

Verifiers, LMs

A protocol for estimating the cost of each method.



MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?

Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.

Problem: If $\sum_{n=0}^{\infty} \cos^{2n} \theta = 5$, what is $\cos 2\theta$?

Solution: This geometric series is $1 + \cos^2 \theta + \cos^4 \theta + \cdots = \frac{1}{1-\cos^2 \theta} = 5$. Hence, $\cos^2 \theta = \frac{4}{5}$. Then $\cos 2\theta = 2\cos^2 \theta - 1 = \boxed{\frac{3}{5}}$.

Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.

Solution: Complete the square by adding 1 to each side. Then $(x+1)^2 = 1 + i = e^{\frac{i\pi}{4}}\sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}}\sqrt[4]{2}$. The desired product is then $\left(-1 + \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right)\left(-1 - \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right) = 1 - \cos^2\left(\frac{\pi}{8}\right)\sqrt{2} = 1 - \frac{\left(1+\cos\left(\frac{\pi}{4}\right)\right)}{2}\sqrt{2} = \boxed{\frac{1-\sqrt{2}}{2}}$.

[1] Measuring Mathematical Problem Solving With the MATH Dataset. Hendrycks et al. 2021

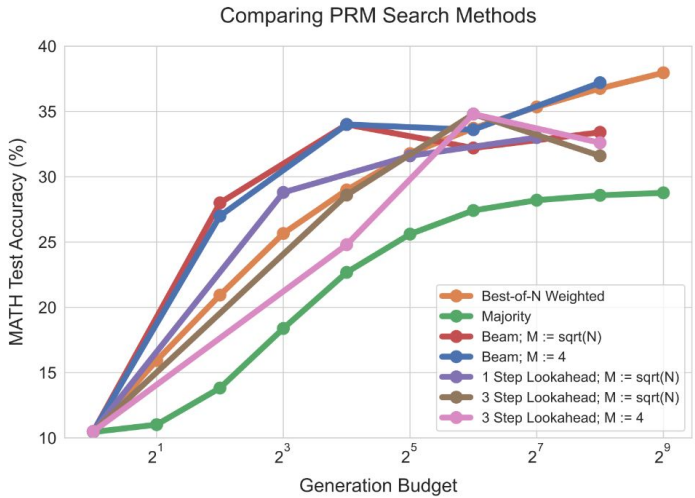
Results and Analysis - Overview

1. Search with Verifiers
    a. Comparing search algorithms
    b. Which problems does search improve?
    c. Compute-optimal” scaling trend
2. Revisions
    a. Can revision improve performance?
    b. Trading off sequential and parallel test-time compute
    c. Compute-optimal revisions
3. Exchanging Pretraining and Test-Time Compute

Results and Analysis - Search with Verifiers

1. Comparing search algorithms

- With smaller generation budgets, **beam search** significantly outperforms **best-of-N**.
- As the budget is scaled up, **beam search** often underperforming the **best-of-N** baseline.
- **Lookahead-search** generally underperforms other methods at the same generation budget.
- There are signs of exploitation of the PRM's predictions



Results and Analysis - Search with Verifiers

2. Which problems does search improve?

Defining **model-specific difficulty** of a problem.

Binning the model's pass@1 rate – estimated from 2048 samples – on each question in the test set into five quantiles, each corresponding to increasing difficulty levels

- Oracle difficulty
- Model-predicted difficulty

Results and Analysis - Search with Verifiers

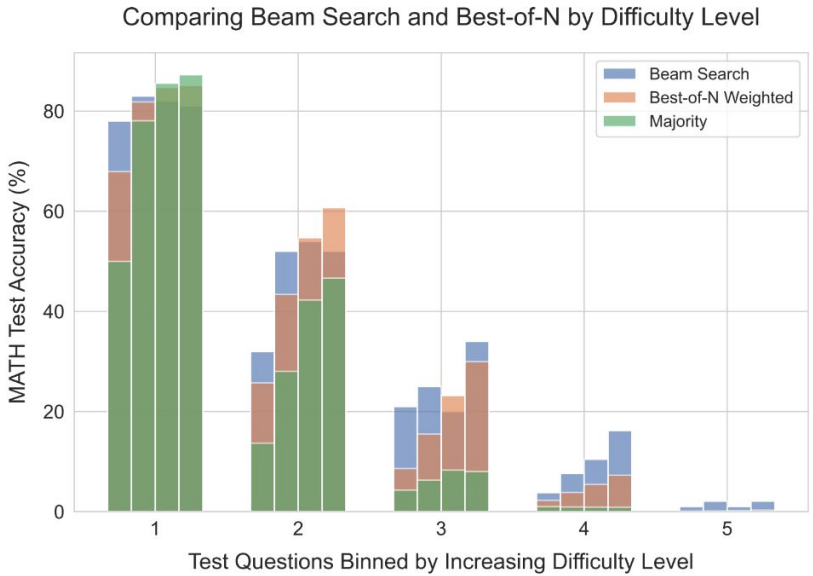
2. Which problems does search improve?

In aggregate, beam search and best-of-N perform similarly with a high generation budget

On the level 1,2 questions, beam search degrades performance as the generation budget increases

On the level 3,4 questions, beam search consistently outperforms best-of-N.

On the level 5 questions, no method makes much meaningful progress.



*The four bars in each difficulty bin correspond to increasing test-time compute budgets (4, 16, 64, and 256 generations).

Results and Analysis - Search with Verifiers

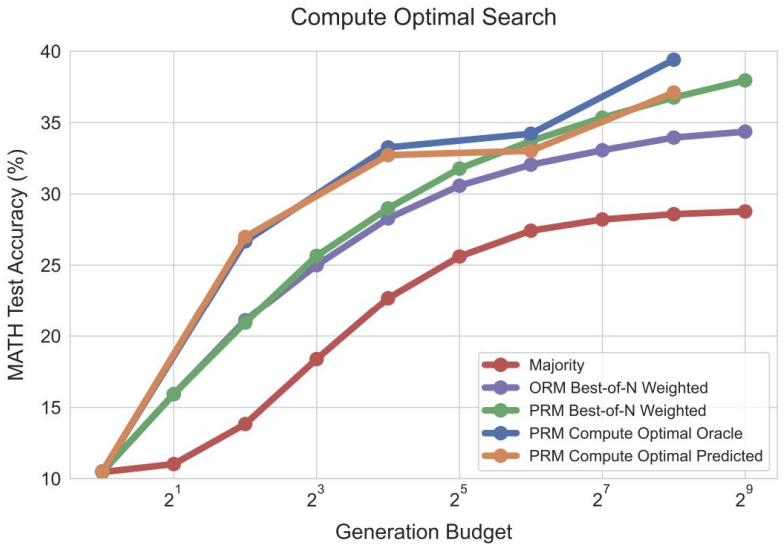
3. “Compute-optimal” scaling trend

Select the search parameters per difficulty bin

Performance gains could be obtained by adaptively allocating test-time compute during search.

In the low generation budget regime, compute-optimal scaling can nearly outperform best-of-N using up to 4x less test-time compute

In the higher budget regime, some of these benefits diminish.



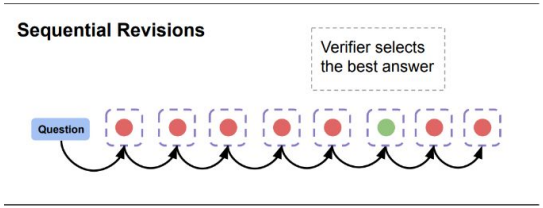
Results and Analysis - Revisions

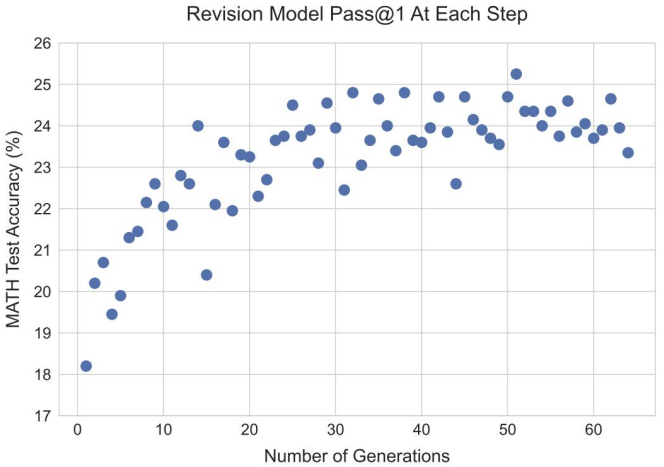
1. Revision can improve performance

- When using a naïve approach, around 38% of correct answers get converted back to incorrect ones.

=> They employ a mechanism based or verifier-based selection on sequential majority voting.

- The model's pass@1 at each revision step gradually improves.

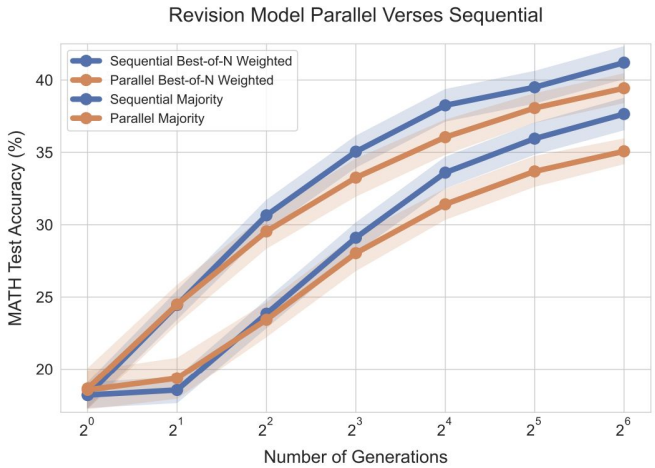




Results and Analysis - Revisions

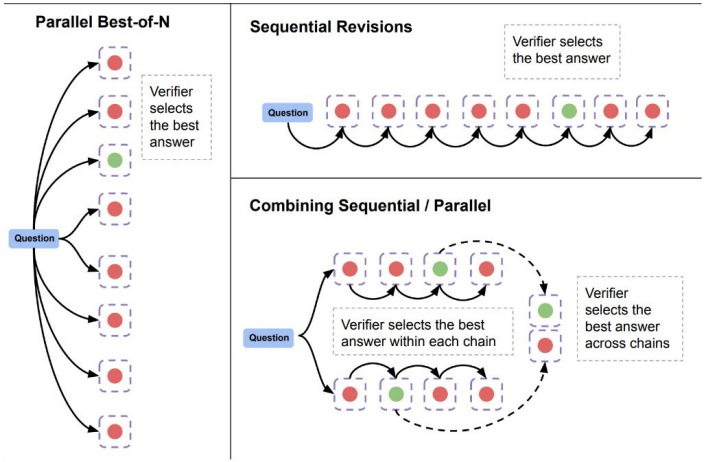
2. Sampling in sequence vs in parallel

- Sampling solutions in sequence outperforms sampling them in parallel



Results and Analysis - Revisions

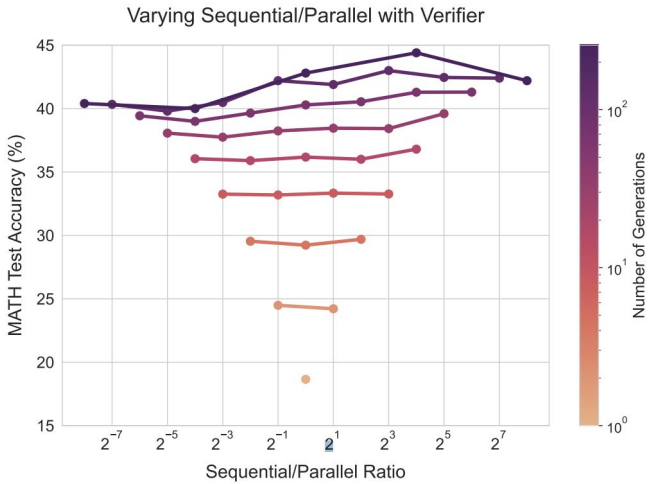
3. Trading off sequential and parallel test-time compute

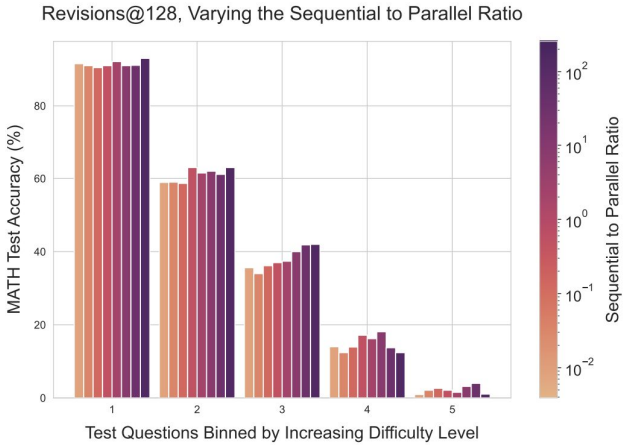


Results and Analysis - Revisions

3. Trading off sequential and parallel test-time compute

- There exists an ideal sequential to parallel ratio, that achieves the maximum accuracy.
- The ideal ratio of sequential to parallel varies depending on a given question's difficulty



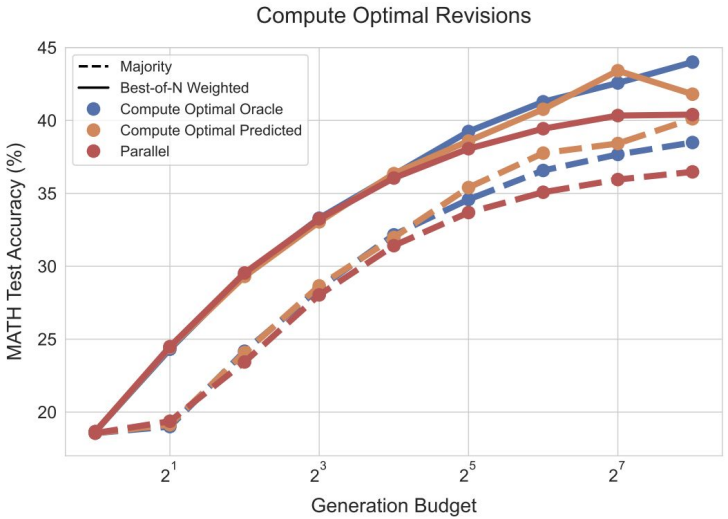


Results and Analysis - Revisions

4. Compute-optimal revisions

* Select the ideal ratio of sequential to parallel compute per difficulty bin.

- Compute-optimal scaling can outperform best-of-N using up to 4x less test-time compute (e.g. 64 samples verses 256)



Exchanging Pretraining and Test-Time Compute

Test-time and pretraining compute are not 1-to-1 “exchangeable”.

On easy and medium questions, or in settings with small inference requirement, test-time compute can easily cover up for additional pretraining.

However, on challenging questions or under higher inference requirement, pretraining is likely more effective for improving performance.

Conclusion

1. Scaling test-time compute can be achieved via Search and Revisions
2. The effectiveness of scaling test-time compute depends on the difficulty of problems
3. In some scenarios, scaling test-time compute is more effective than scaling training-time compute