Fall 2024 COS597R:
Deep Dive into Large Language Models

Danqi Chen, Sanjeev Arora

Lecture 15: LLM reasoning + inference-time compute (cont'd)

https://princeton-cos597r.github.io/

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

Q. Large model vs small model + more inference compute?

Q. Can test-time computation substitute for pre-training?

[Submitted on 1 Aug 2024 (v1), last revised 14 Oct 2024 (this version, v2)]

Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference for Problem-Solving with Language Models

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, Yiming Yang

- **Sampling**: best-of-n, majority voting, weighted majority voting
- **Search**: MCTS, reward balanced search (this work)

[Submitted on 31 Jul 2024 (v1), last revised 16 Sep 2024 (this version, v2)]

Large Language Monkeys: Scaling Inference Compute with Repeated Sampling

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, Azalia Mirhoseini

- **Sampling**

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

Q2. This paper only considers the trade-off between pre-training and inference, and they did the analysis using a base model. What do think of the impact of post-training in this pipeline? What are some general ideas of post-training for improving the (mathematical) reasoning of LLMs?

"how one should trade off inference-time and pre-training compute"

"We conduct our analysis using the PaLM 2-S* (Codey) base model"

"Capability-specific fine-tuning is necessary to induce revision and verification capabilities into the base model on MATH"

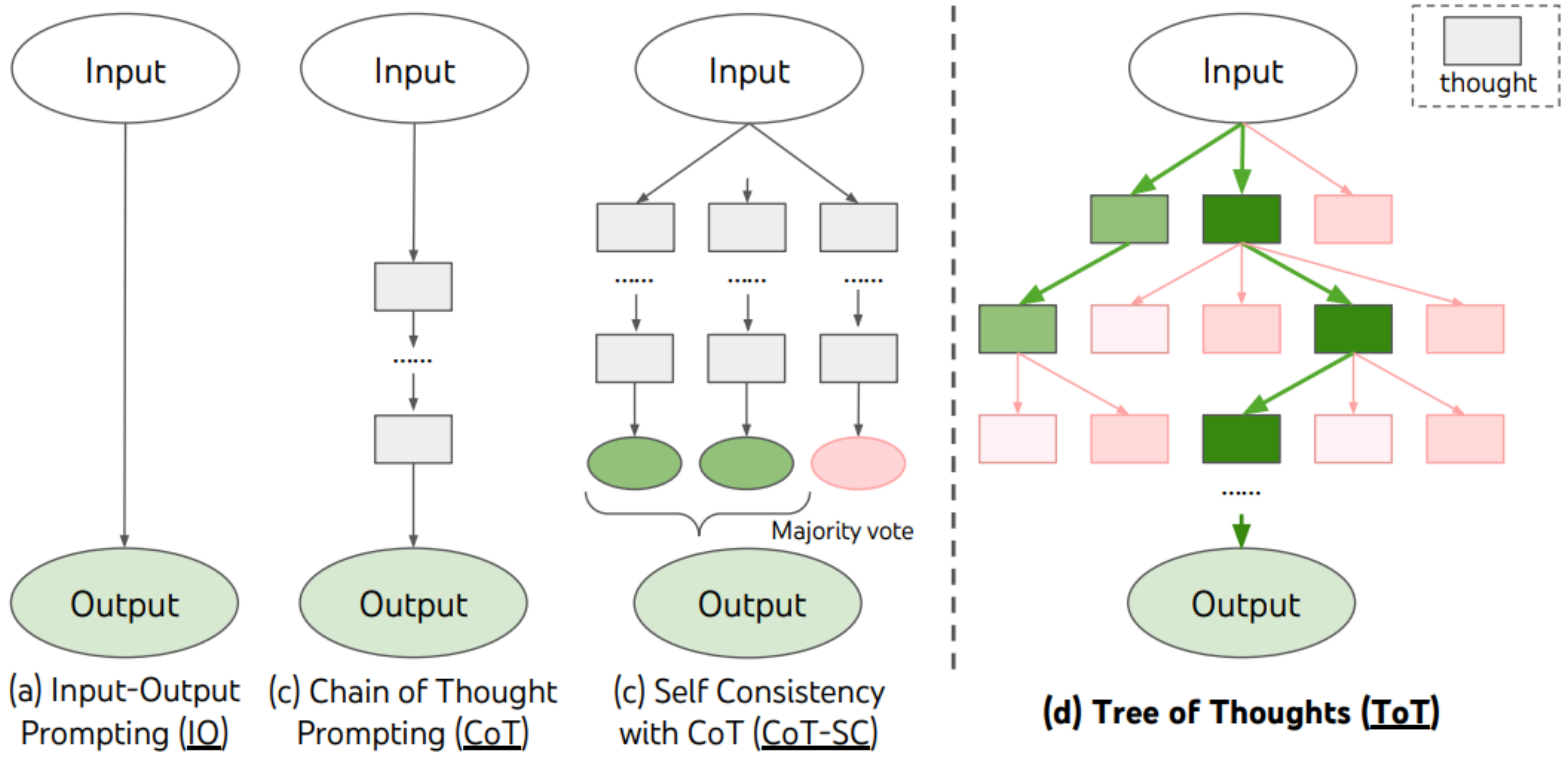
This lecture

Image: Jim Fan

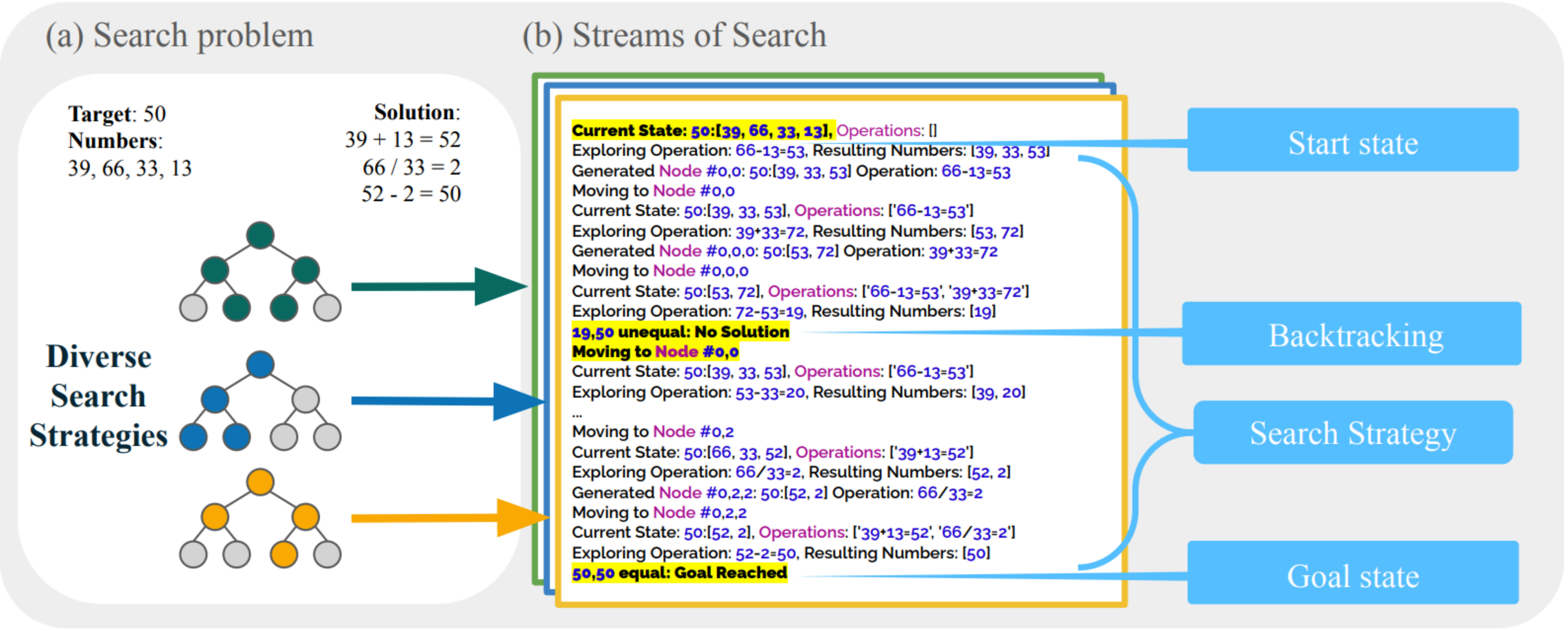
- Different test-time strategies
  - Which strategy works better in what scenario?
  - More discussion on ORMs vs PRMs

- (Brief) How to train LLMs for better reasoning (= post-training)?

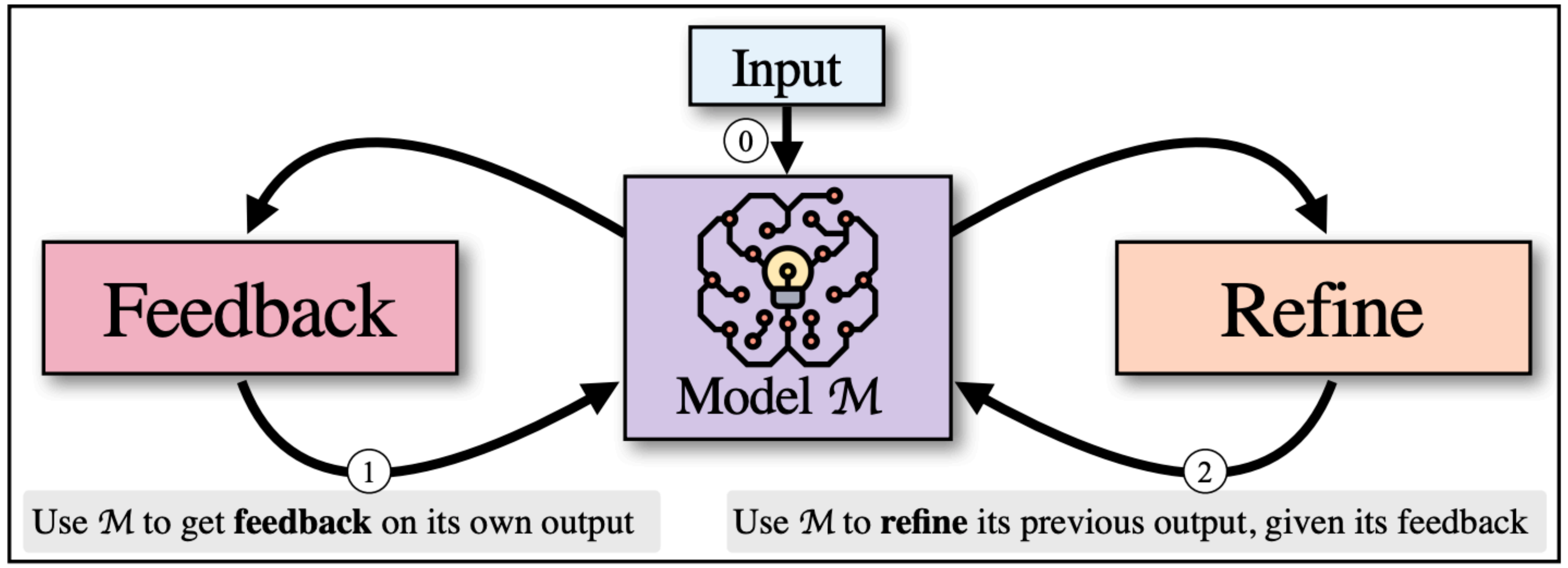
Lots of inference methods



Tree of thoughts (Yao et al., 2023)



Stream of search (Gandhi et al., 2024)



Self-refine (Madaan et al., 2023)

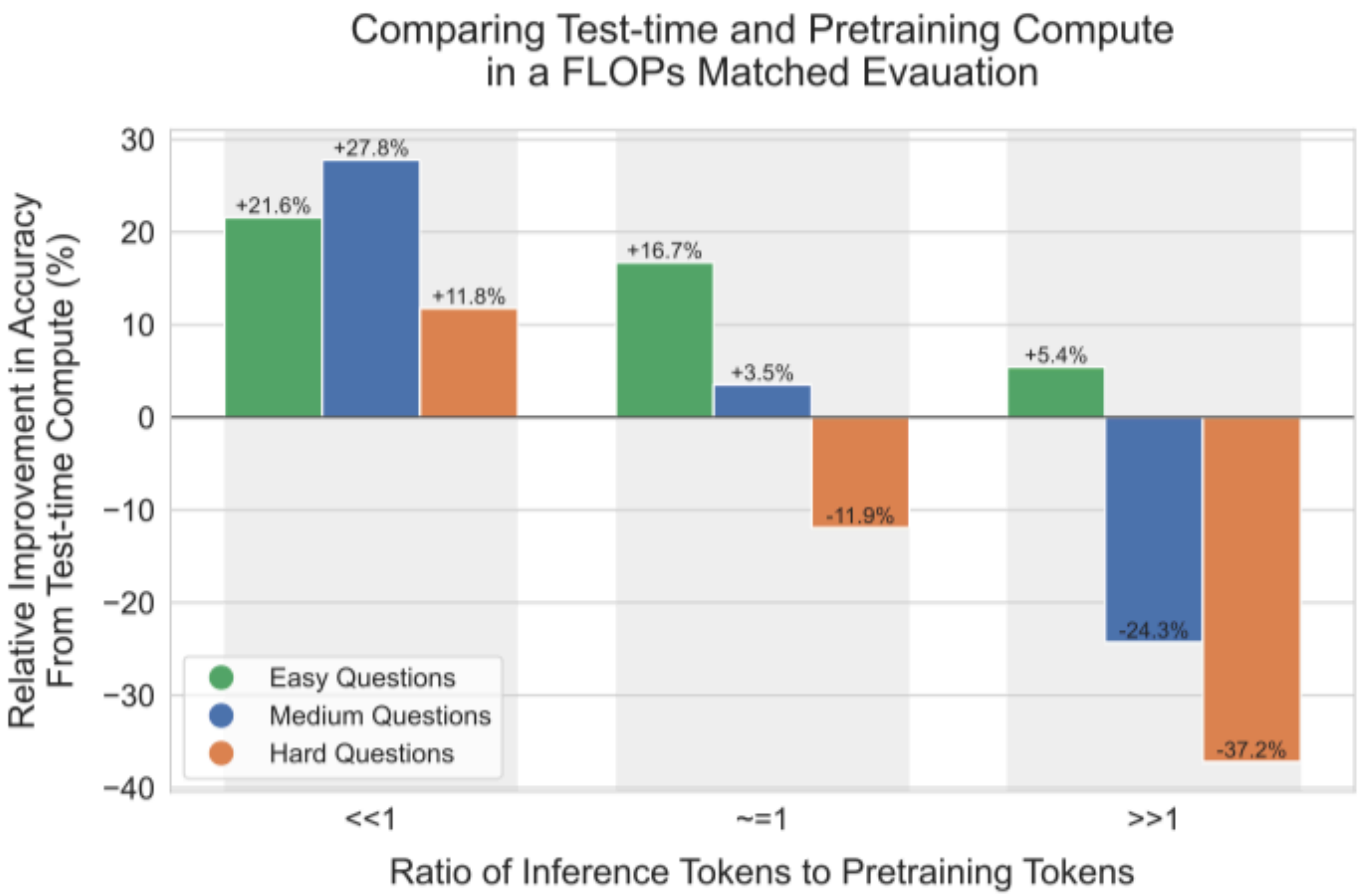
Two views of scaling test-time compute

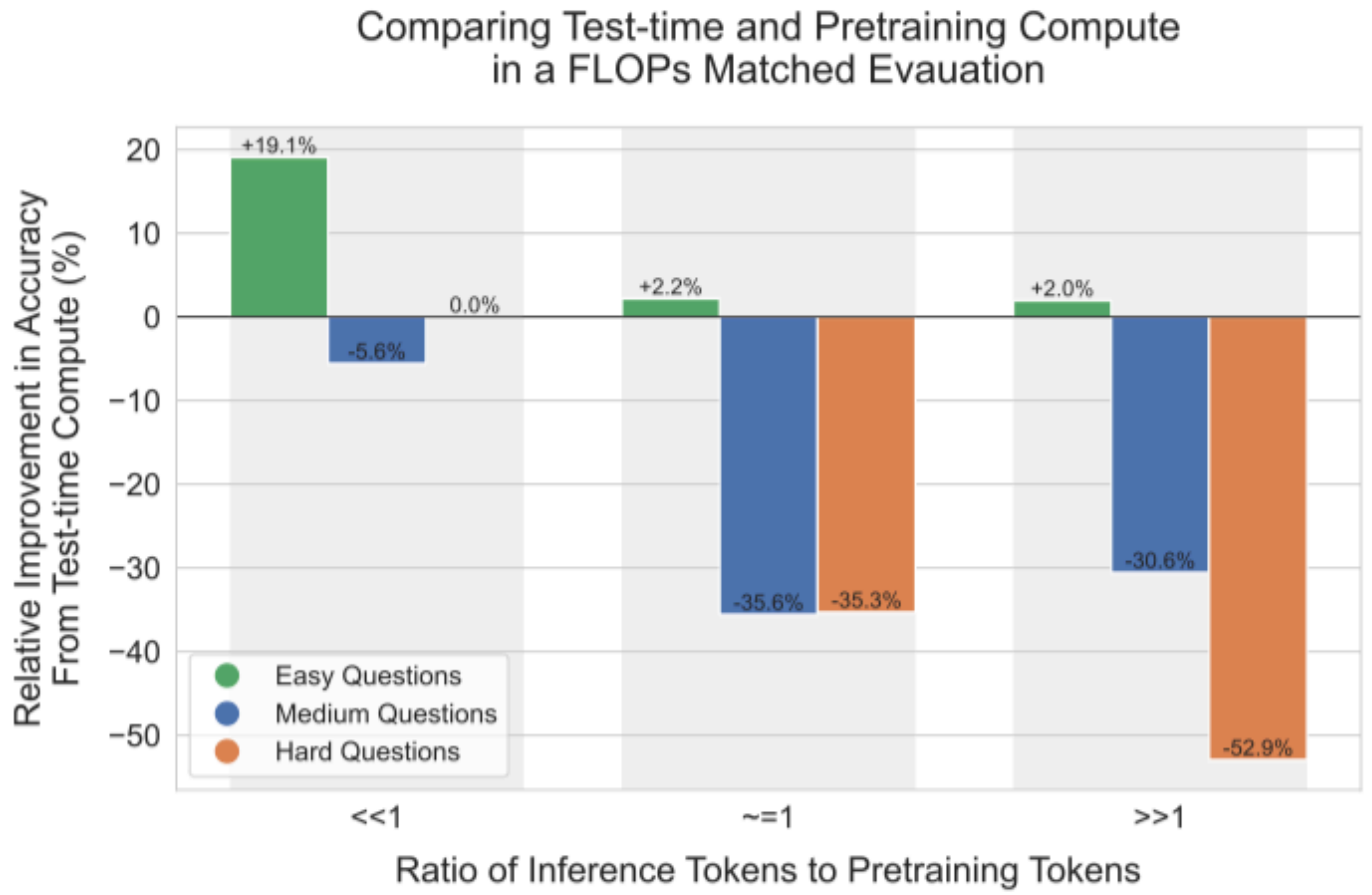
- **Input level**: augment the prompt with additional tokens (repeatedly)

  "Refining the proposal distribution"

- **Output level**: sample multiple candidates and perform surgery on these candidates

  "Searching against a (PRM) verifier"





Comparison: a 14x larger model with greedy decoding

Search against a verifier

Search against a PRM verifier

- They use a PRM (process reward model) instead of an ORM (outcome reward model) verifier

ORM $(P \times S \to \mathbb{R})$

$$\mathcal{L}_{ORM} = y_s \log r_s + (1 - y_s) \log(1 - r_s)$$

PRM $(P \times S \to \mathbb{R}^+)$

$$\mathcal{L}_{PRM} = \sum_{i=1}^{K} y_{s_i} \log r_{s_i} + (1 - y_{s_i}) \log(1 - r_{s_i})$$

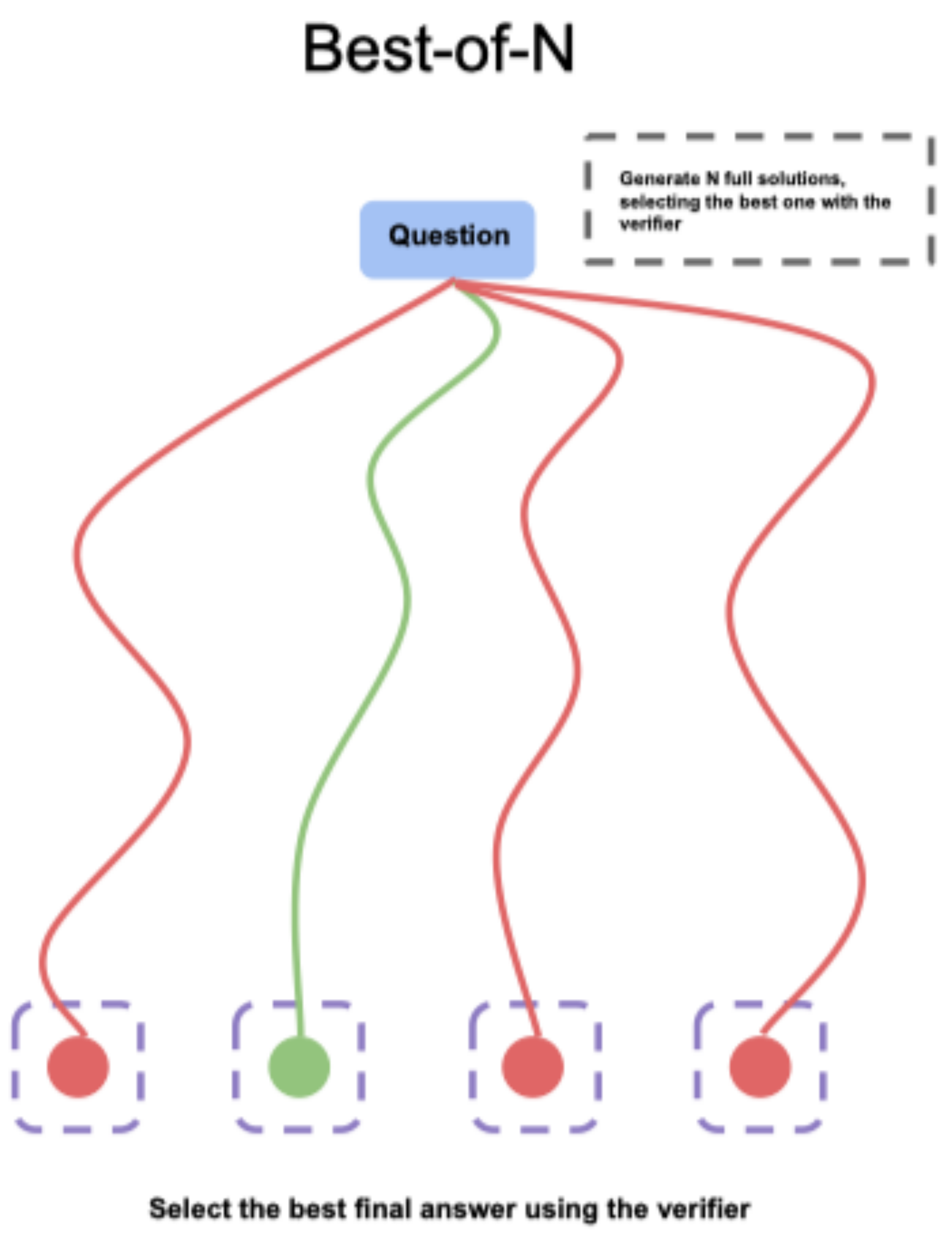
- They use **automated methods** for collecting process supervision instead of PRM800K
    - Distribution shift between GPT-4 and Palm-2 outputs?

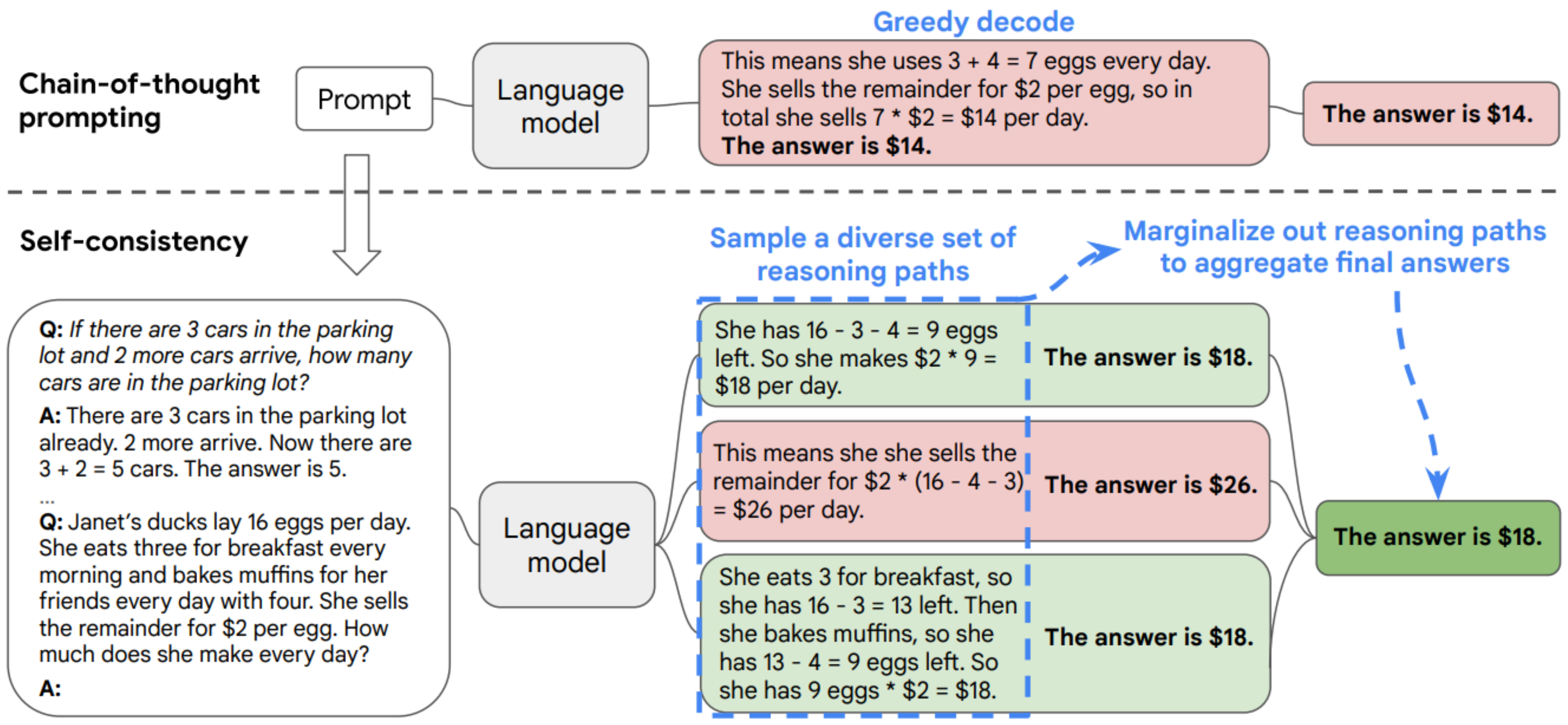
- PRM can be used for multiple strategies, but ORM can be only used for best-of-n (still PRM works better!)

- I believe they fine-tuned the same base model as the verifier

#1: Best-of-n weighted

- Best-of-n: sample n full solutions and use RM to pick the best one
- Majority vote: get n final answers, and pick the one with the highest vote (no RM used)



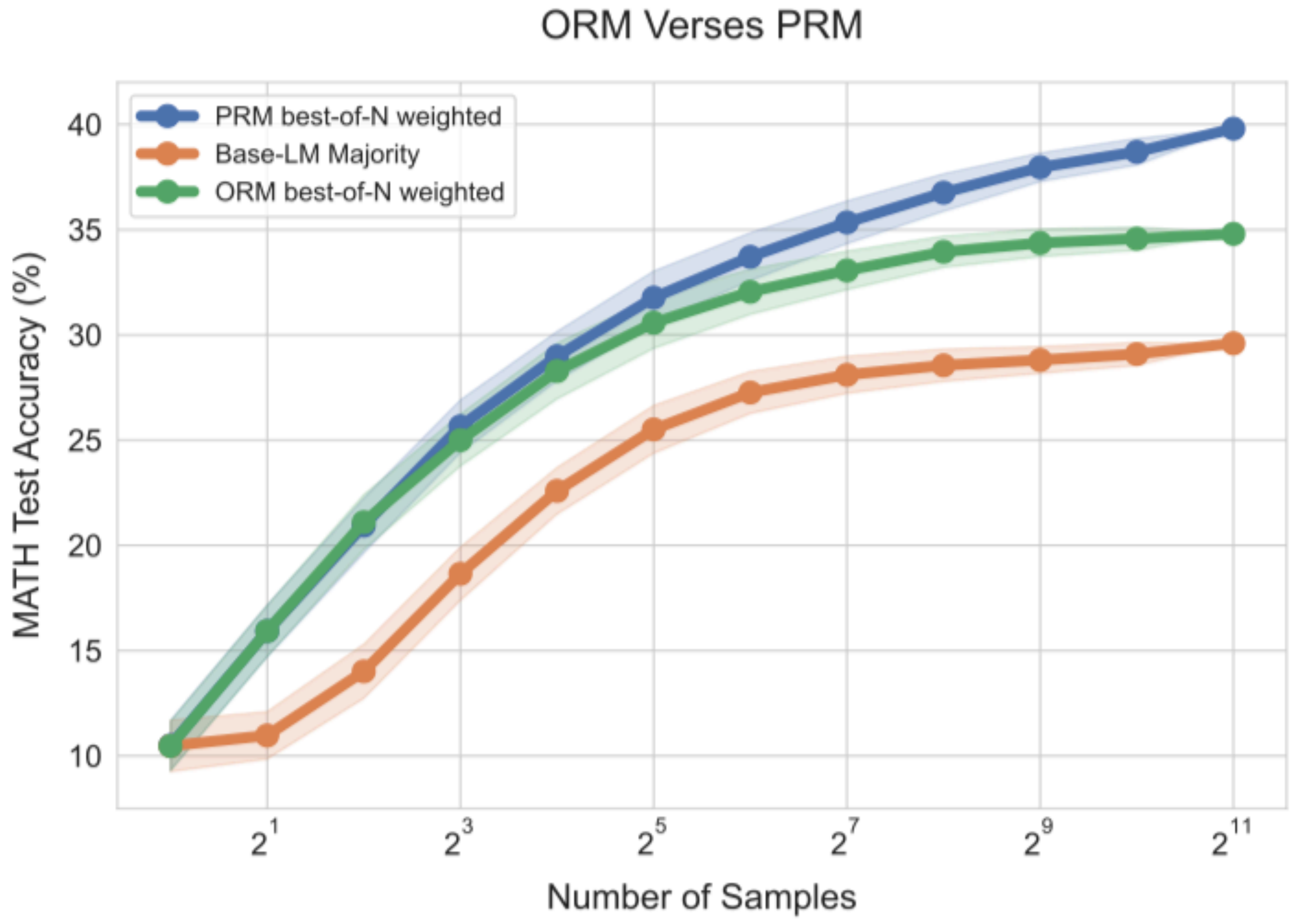


Also called self-consistency (Wang et al., 2022)

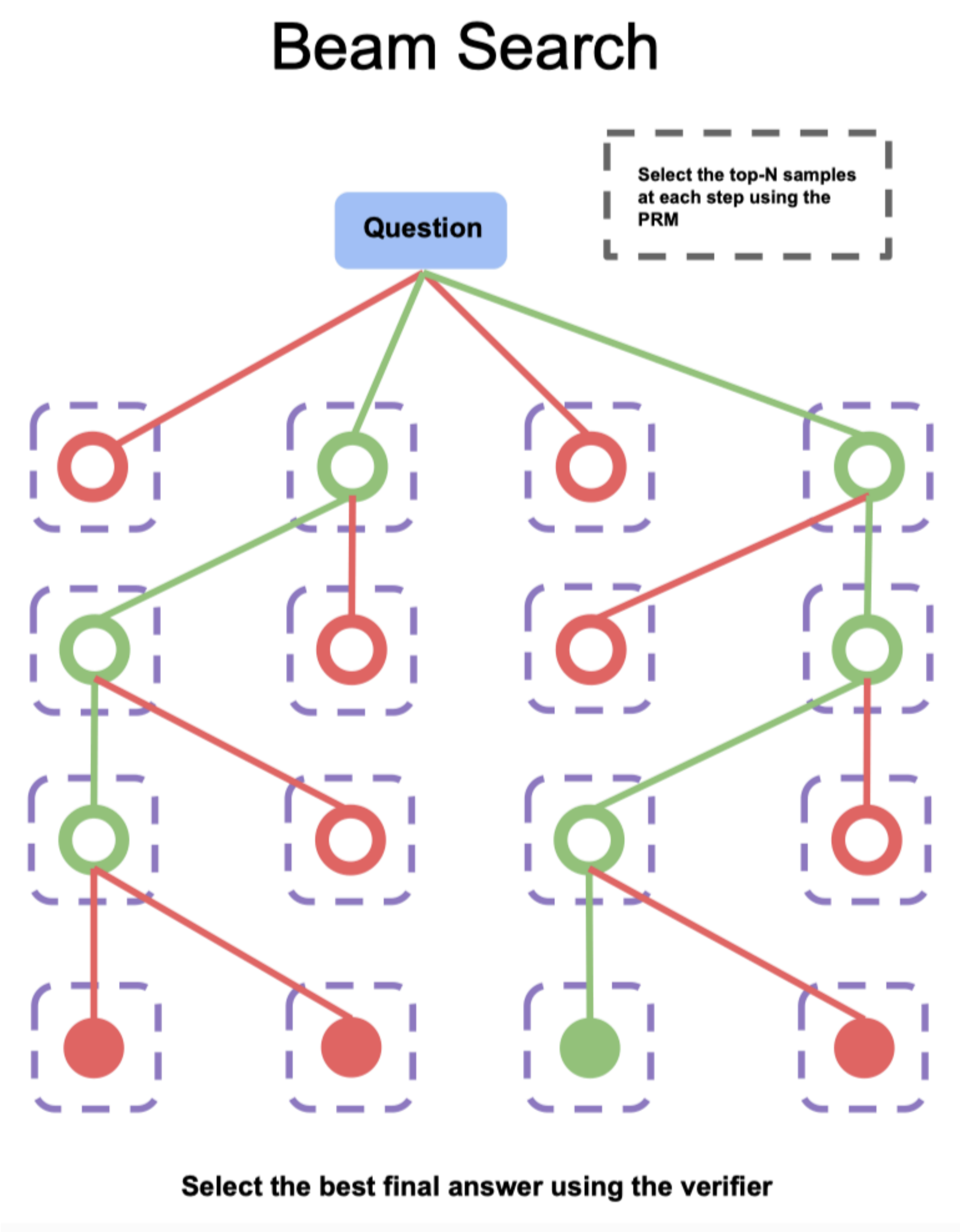
- Best-of-n weighted: get n final answer, each answer has a weight assigned by RM, aggregate and weights and pick the one with highest sum

#1: Best-of-n weighted

- How to get an **aggregated score** from PRM when ranking full answers?
    - They used PRM's prediction at the last step as the full-answer score
    - Prior work used product or minimum

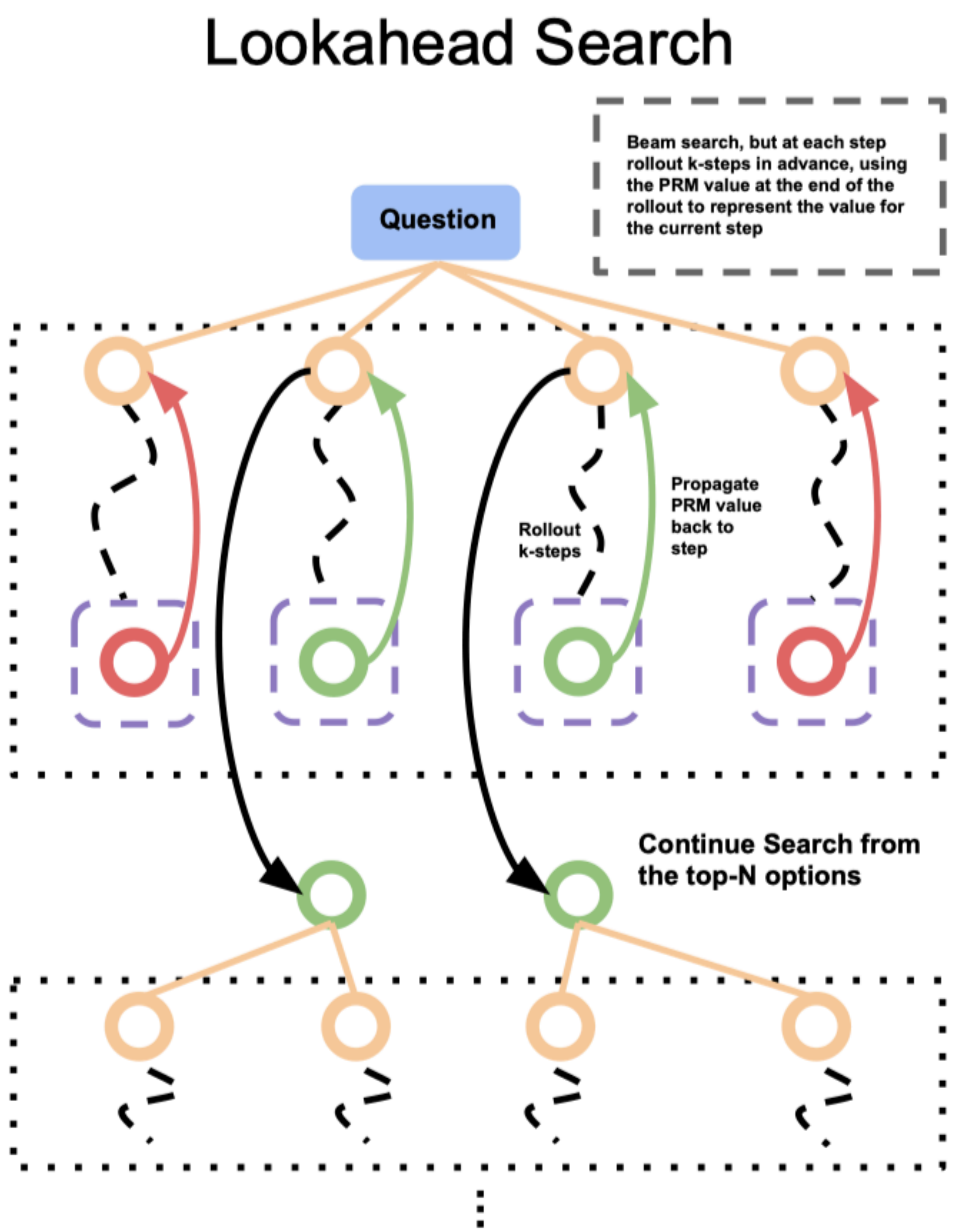


#2: Beam search



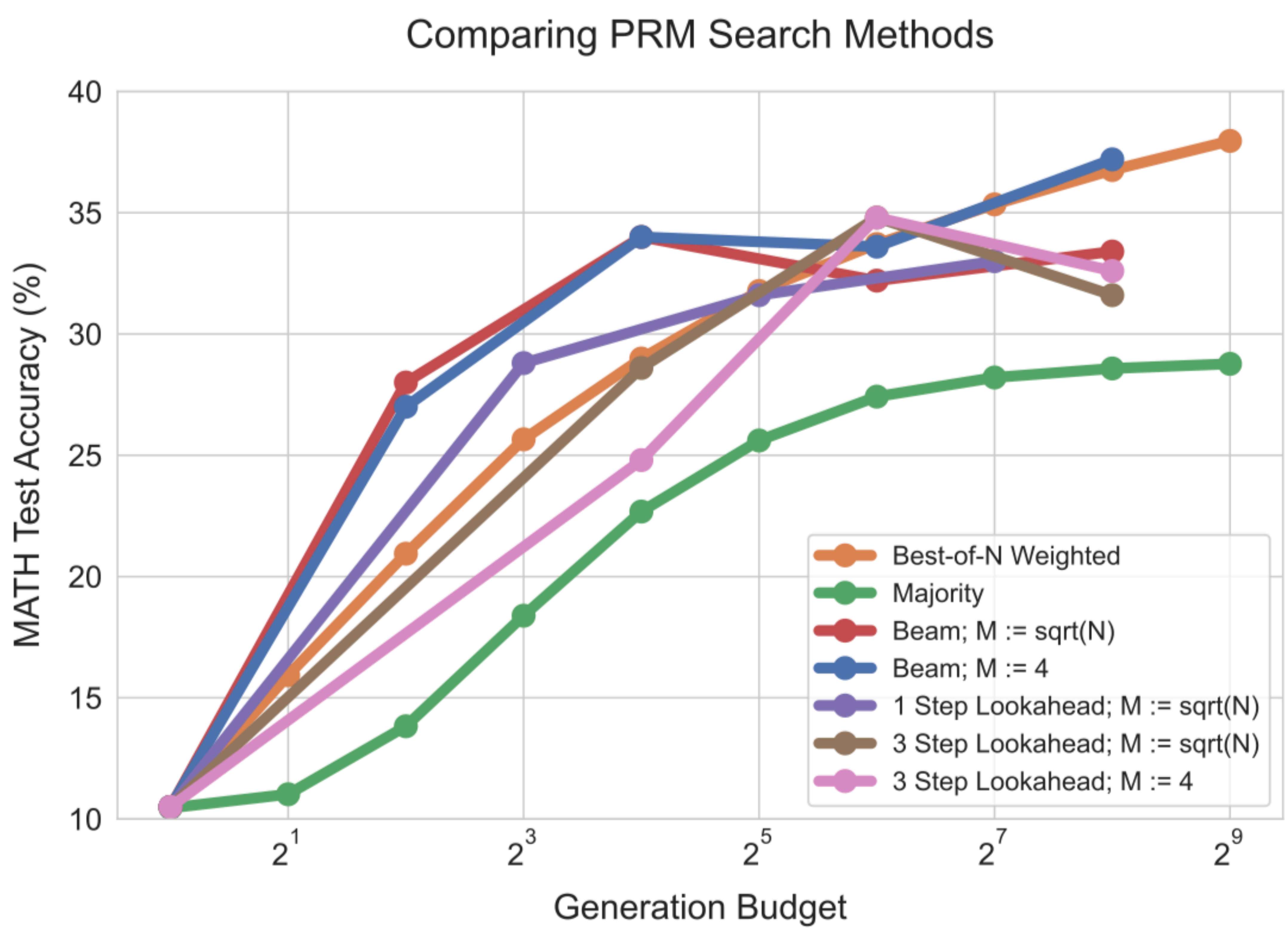
- N: beam size
- M: sample M steps from each node

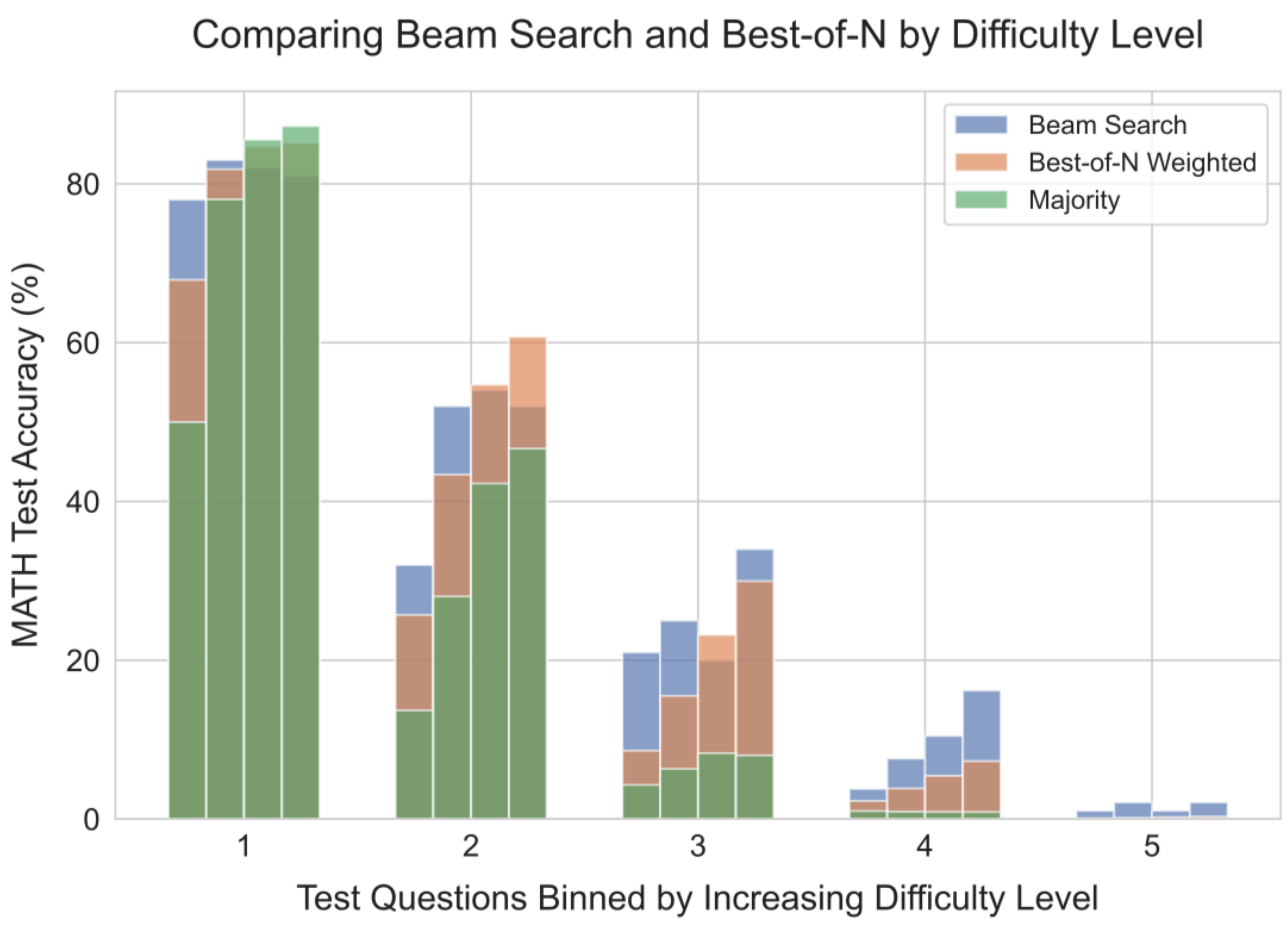
#3: Lookahead search



- N: beam size
- M: sample M steps from each node
- k: rolling out up to k steps, and use PRM's score
- Representative of MCTS-style methods

Results

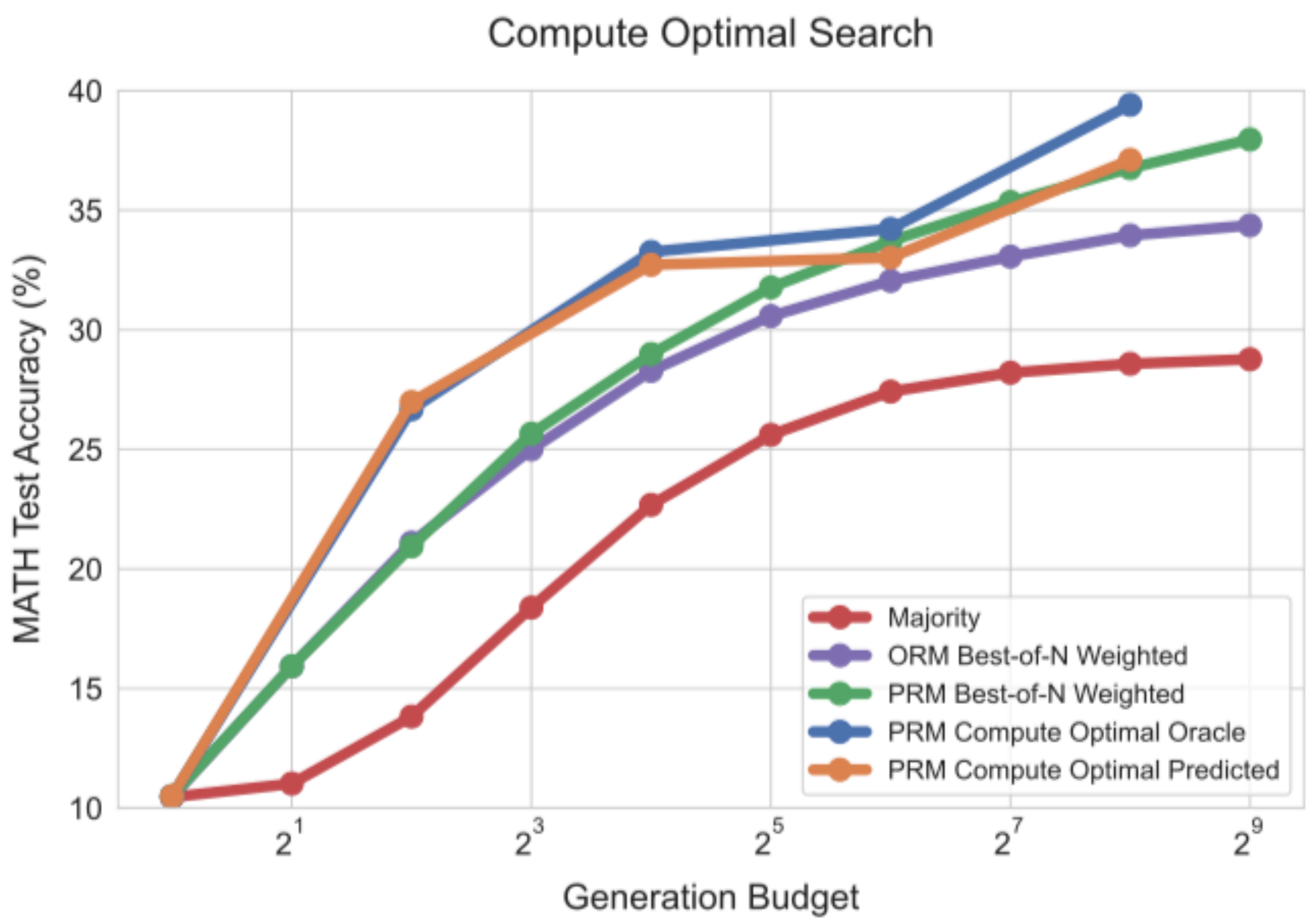




- Beam search is best with less budget
- Best-of-n weighted is the best with large budget

- Beam search is better for harder questions
- No meaningful progress for hardest questions

Results: Adaptive compute-optimal strategy



- The optimal test-time strategy should depend on question difficulty!
- Can outperform PRM best-of-N up to 4x less test-time compute
- Note: estimating difficulty of prompts also incurs test-time compute but omitted in this study ("a crucial avenue for future work")

Aside: how to collect PRM data automatically?

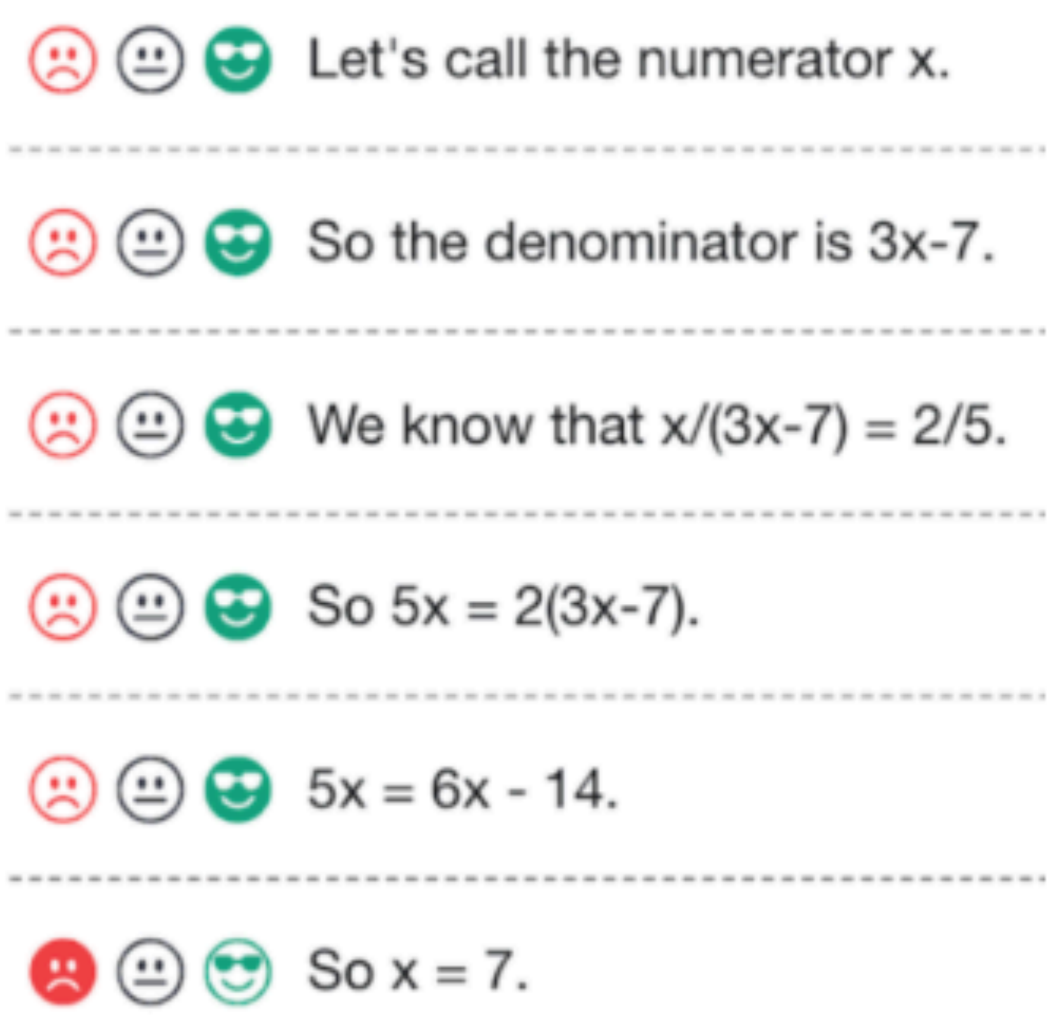
Human annotations

[Submitted on 31 May 2023]

Let's Verify Step by Step

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, Karl Cobbe

The denominator of a fraction is 7 less than 3 times the numerator.
If the fraction is equivalent to 2/5, what is the numerator of the fraction?



PRM800K: Dataset contains 800K step-level labels provided by human raters across 75K solutions to 12K problems (MATH 8K training set + 4K test questions).

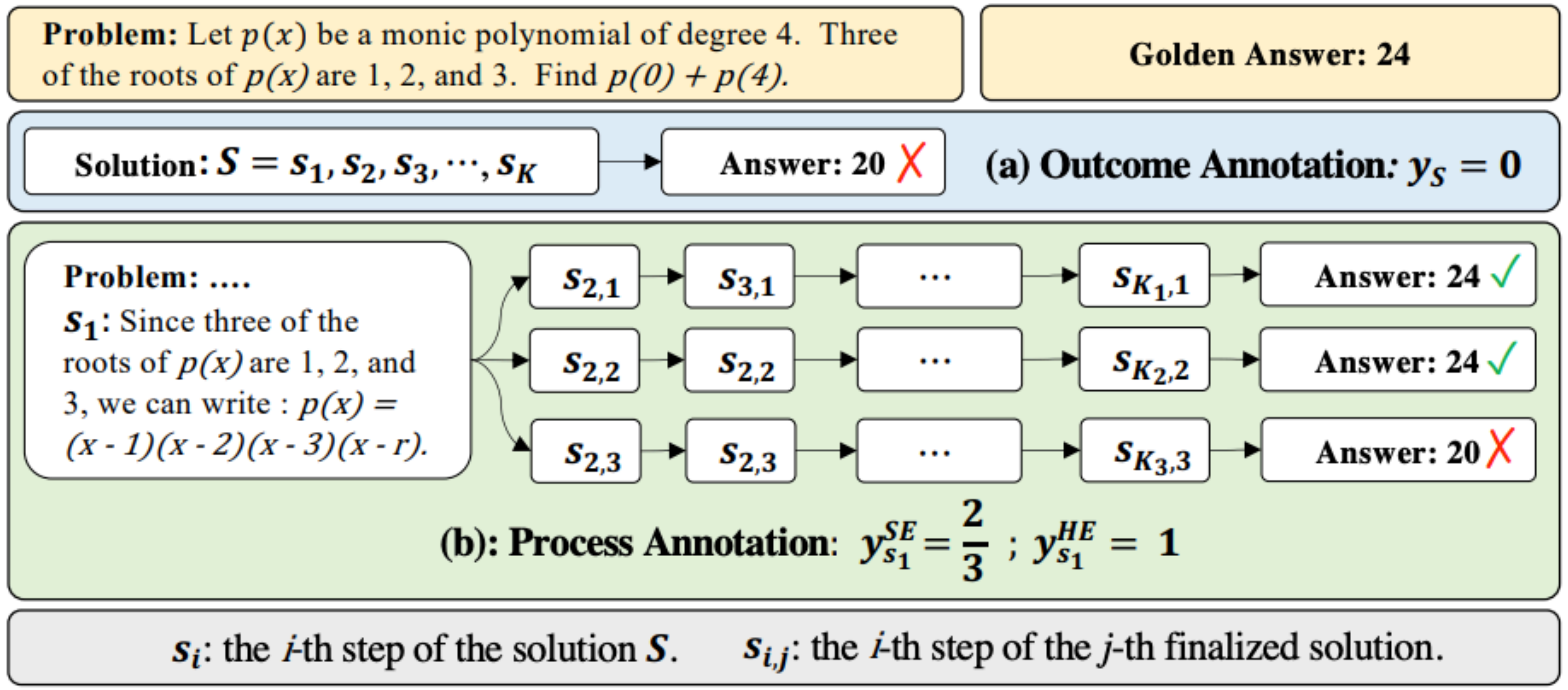
- "We deliberately choose to supervised only up to **first incorrect step**"
- Use **active learning** to decide which steps to annotate "convincing wrong-answer solution"
- Only evaluates using best-of-n sampling

Process supervision without human labels

MATH-SHEPHERD: VERIFY AND REINFORCE LLMS STEP-BY-STEP WITHOUT HUMAN ANNOTATIONS

Peiyi Wang[1†] Lei Li[3] Zhihong Shao[4] R.X. Xu[2] Damai Dai[1] Yifei Li[5]
Deli Chen[2] Y. Wu[2] Zhifang Sui[1]
[1]National Key Laboratory for Multimedia Information Processing, Peking University
[2]DeepSeek-AI [3]The University of Hong Kong
[4]Tsinghua University [5]The Ohio State University
{wangpeiyi9979, nlp.lilei}@gmail.com
li.14042@osu.edu szf@pku.edu.cn

Project Page: MATH-SHEPHERD



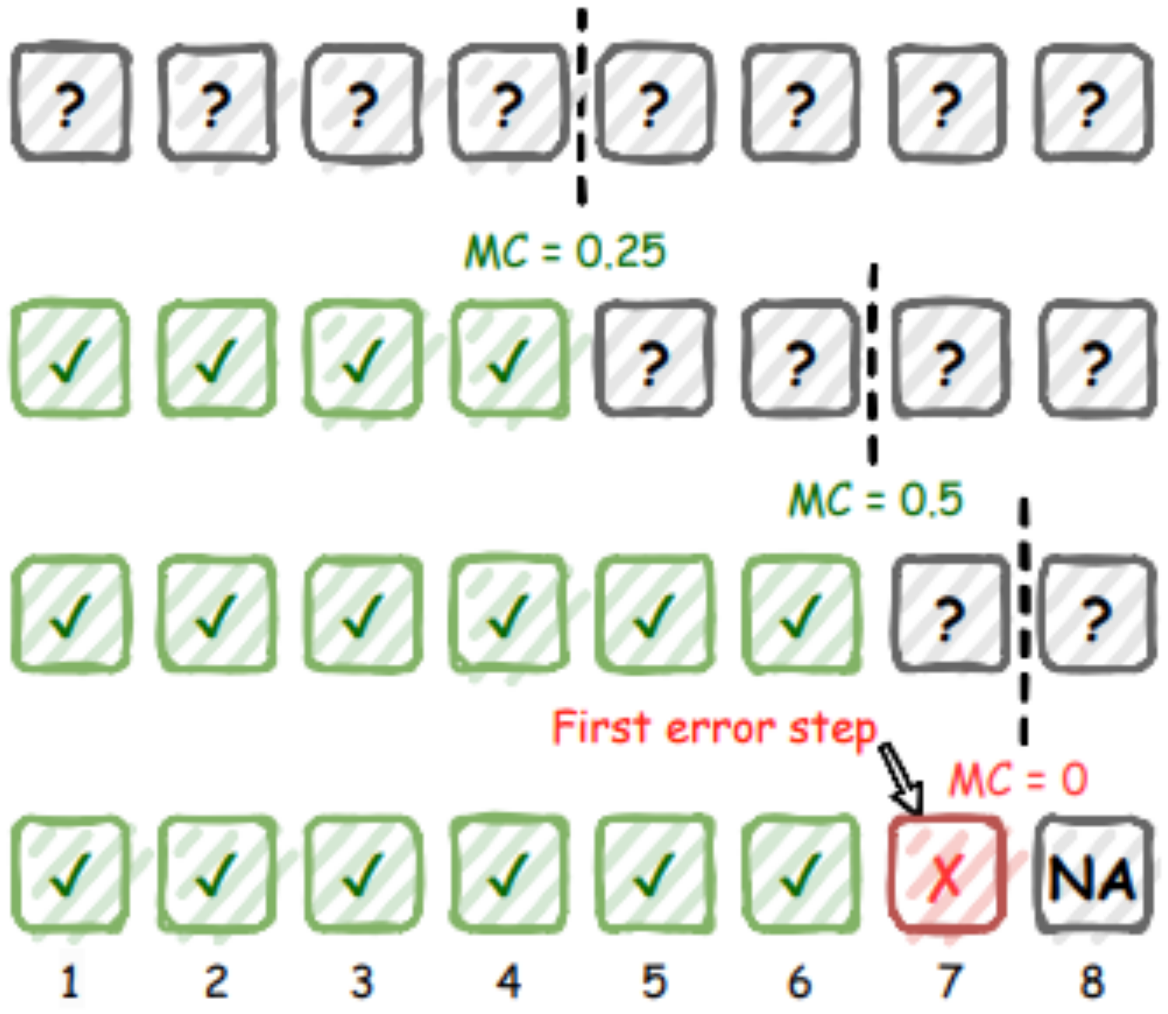
- **The quality of a reasoning step** = its potential to deduce the correct answer
- For each step, perform N rollouts, estimate how likely it will lead to the correct answer
- They did evaluations on a) best-of-n weighted; b) RL with PRM
- This is the method the Snell et al paper used!

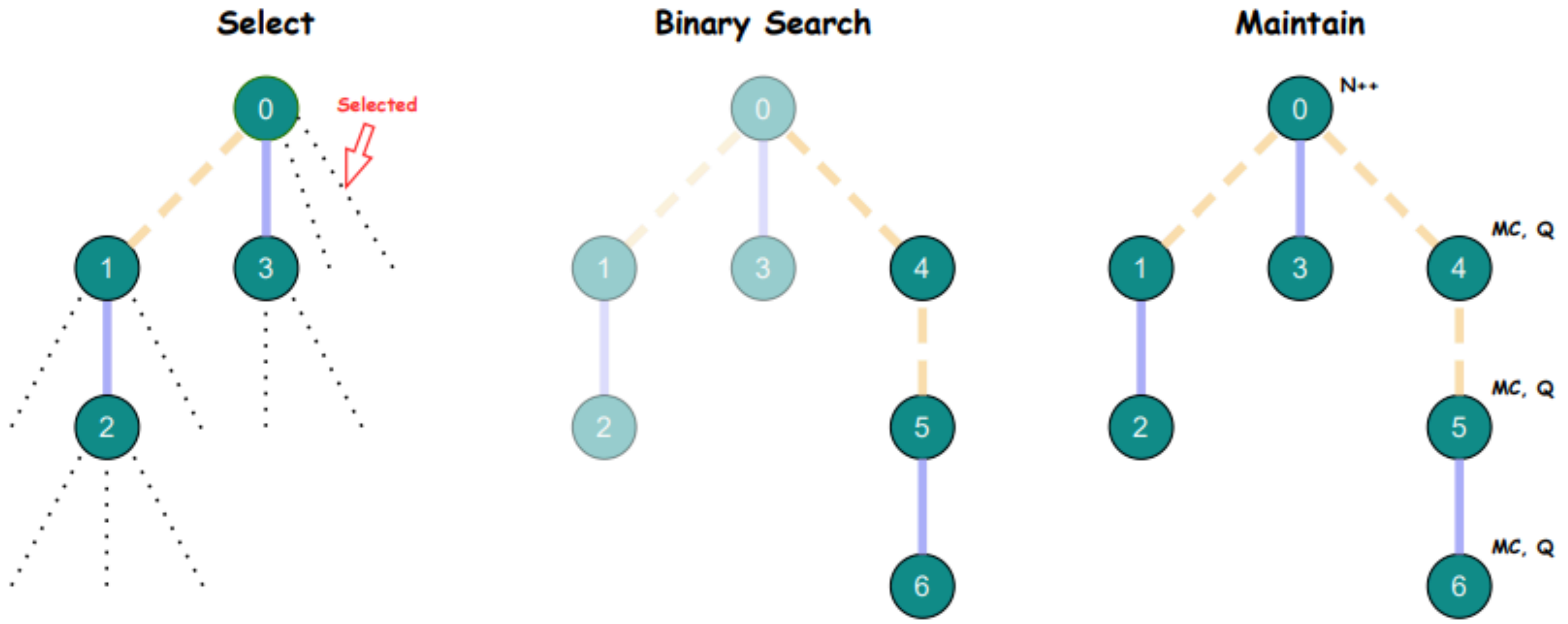
Process supervision without human labels

Google DeepMind 2024-05-22

Improve Mathematical Reasoning in Language Models by Automated Process Supervision

Liangchen Luo[1*], Yinxiao Liu[1*], Rosanne Liu[1], Samrat Phatale[1], Harsh Lara[1], Yunxuan Li[2], Lei Shu[1], Yun Zhu[1], Lei Meng[2], Jiao Sun[2] and Abhinav Rastogi[1]
[1]Google DeepMind, [2]Google

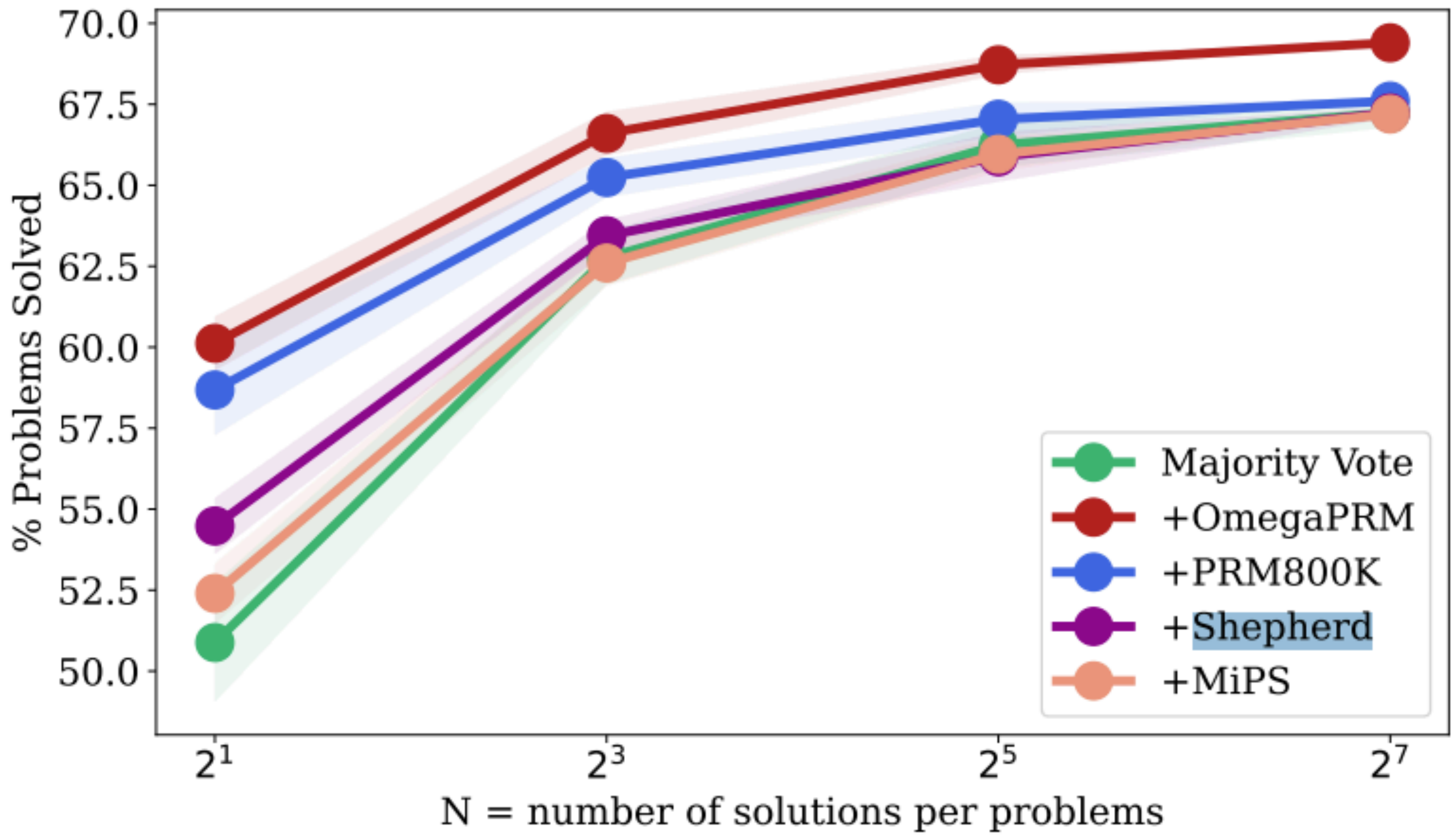




- They claim using per-step Monte Carlo estimation as in Math-Shepherd is not efficient
- They use a complex MCTS process to decide how to do roll-outs and how to collect PRM data

Process supervision without human labels

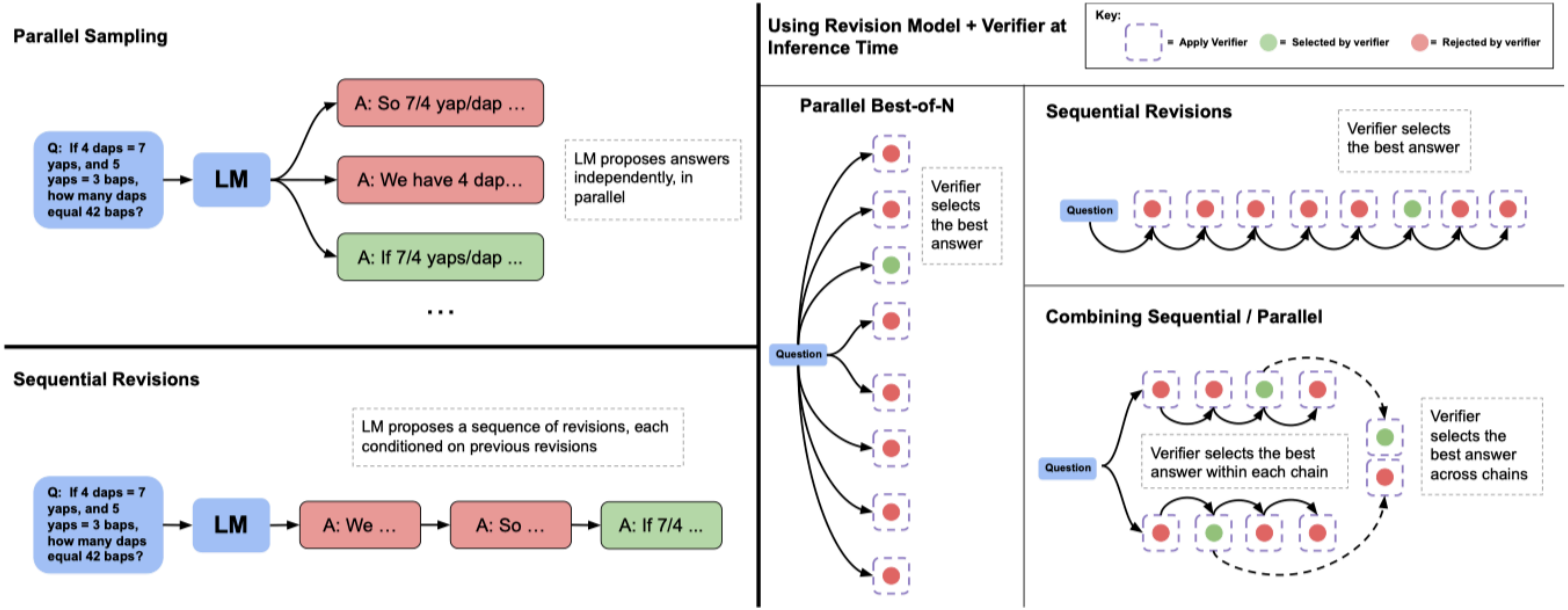
	Majority Vote	+OmegaPRM	+PRM800K	+Shepherd	+MiPS
% Solved (@128)	67.2	**69.4**	67.6	67.2	67.2



best-of-n sampling

Refining the proposal distribution

Using a revision model



- The revision model takes the previous 4 responses and propose a new revision
- Pick the best output according to a verifier

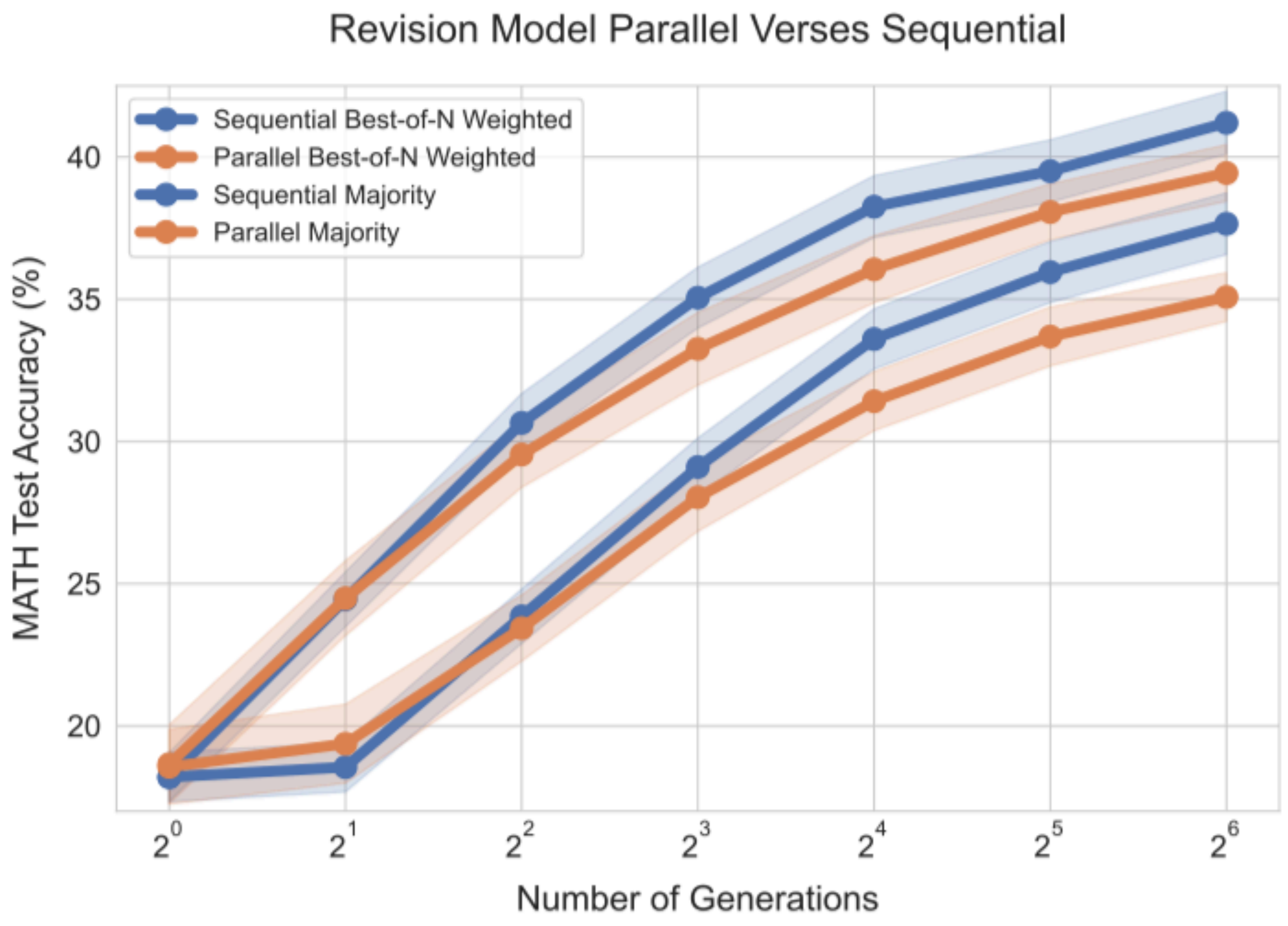
How to train the revision model?

- **Data**: "Specifically, following the recipe of [1], we pair up each correct answer with a sequence of incorrect answers from this set as context to construct multi-turn finetuning data. We include up to four incorrect answers in context, where the specific number of solutions in context is sampled randomly from a uniform distribution over categories 0 to 4. We use a character edit distance metric to prioritize selecting incorrect answers which are correlated with the final correct answer."

[1] Training revision models with synthetic data. Coming soon, 2024.

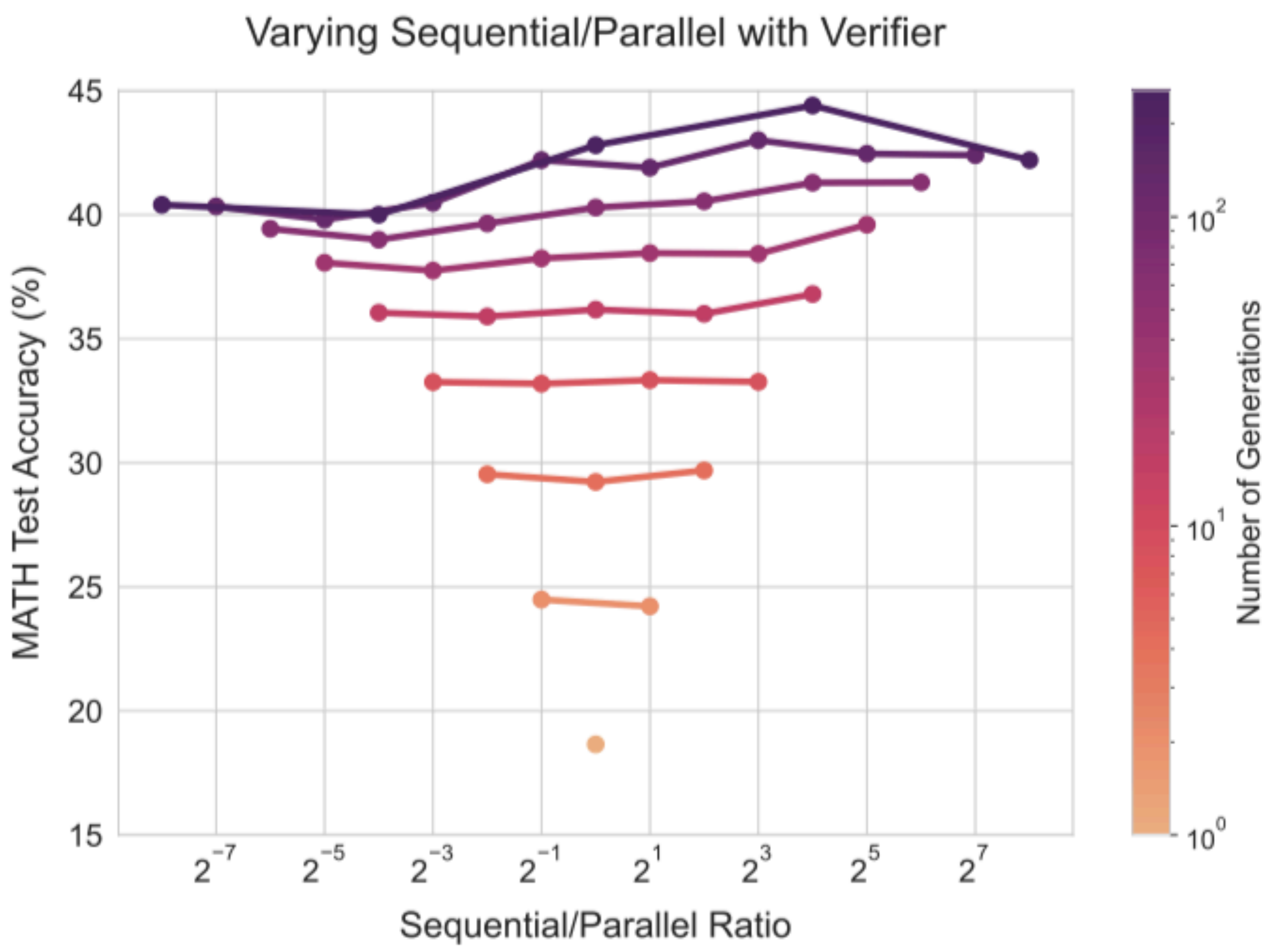
- Q: Do they have a separate revision model like the verifier model, or it is just the main model (= post-training)?

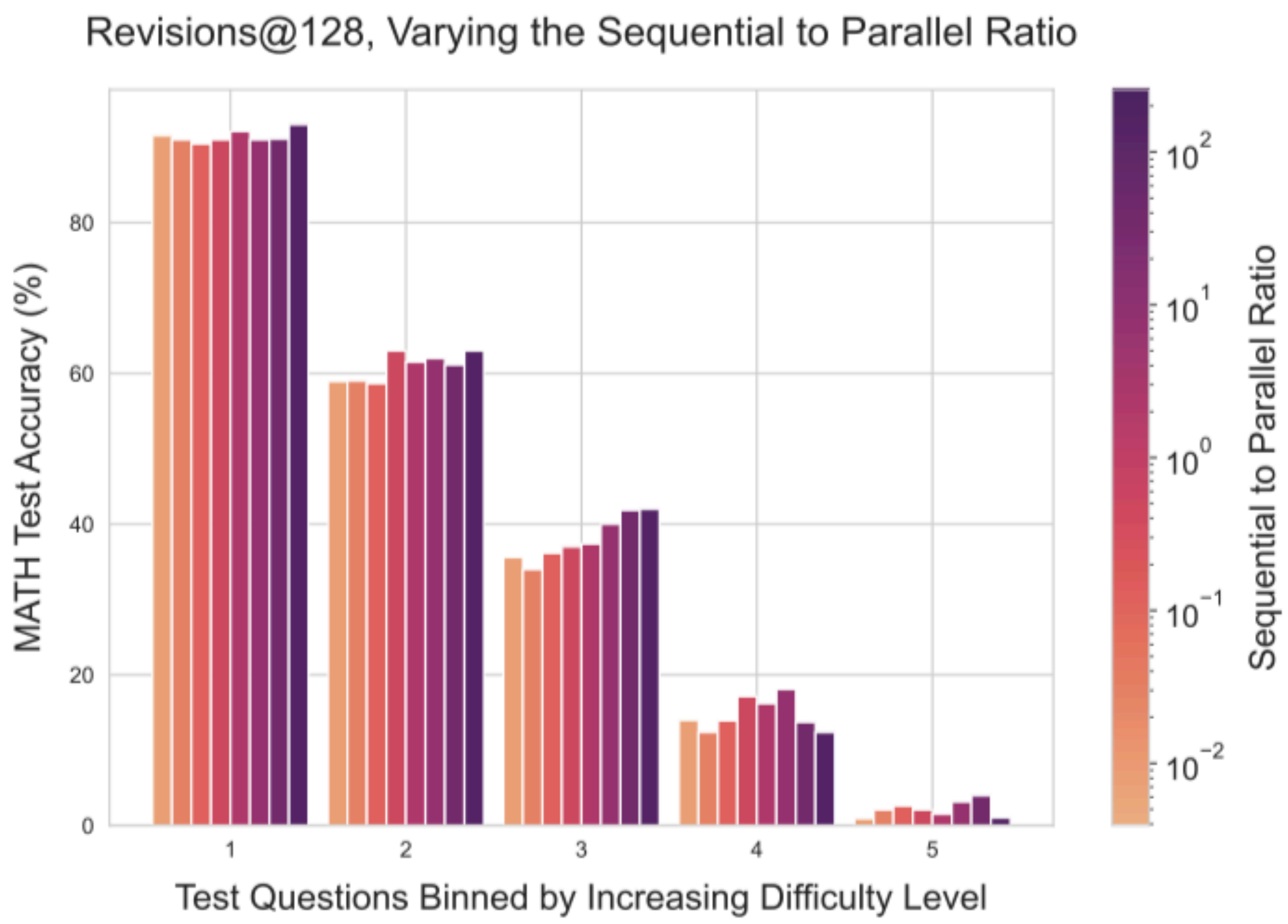
Results: parallel vs sequential



- Using N sequential revisions is always better than N parallel samplings (when controlling # of generations)
- Q: What about FLOPs?

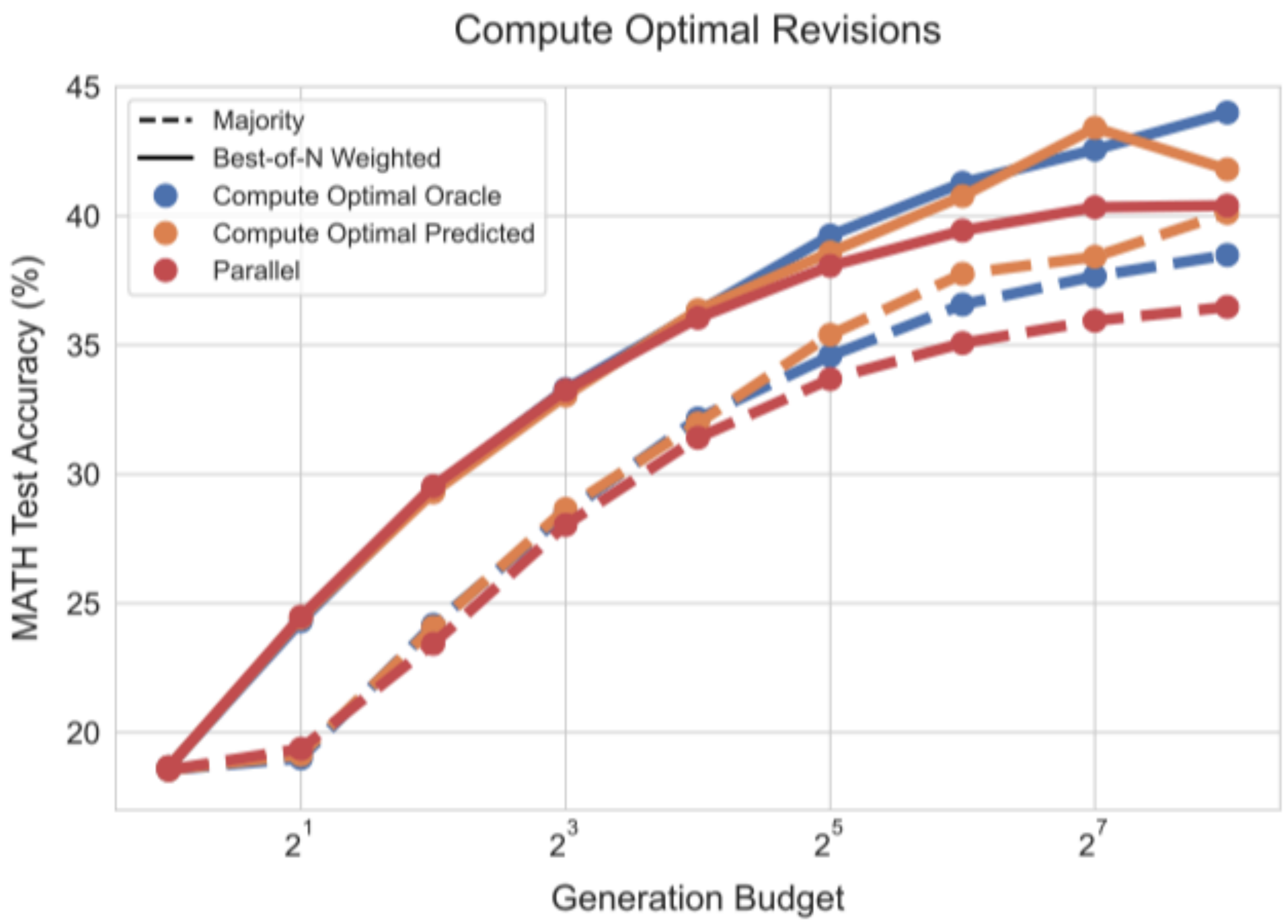
Results: sequential-to-parallel ratio





- Difficult questions need more parallel computation

Results: Adaptive compute-optimal strategy

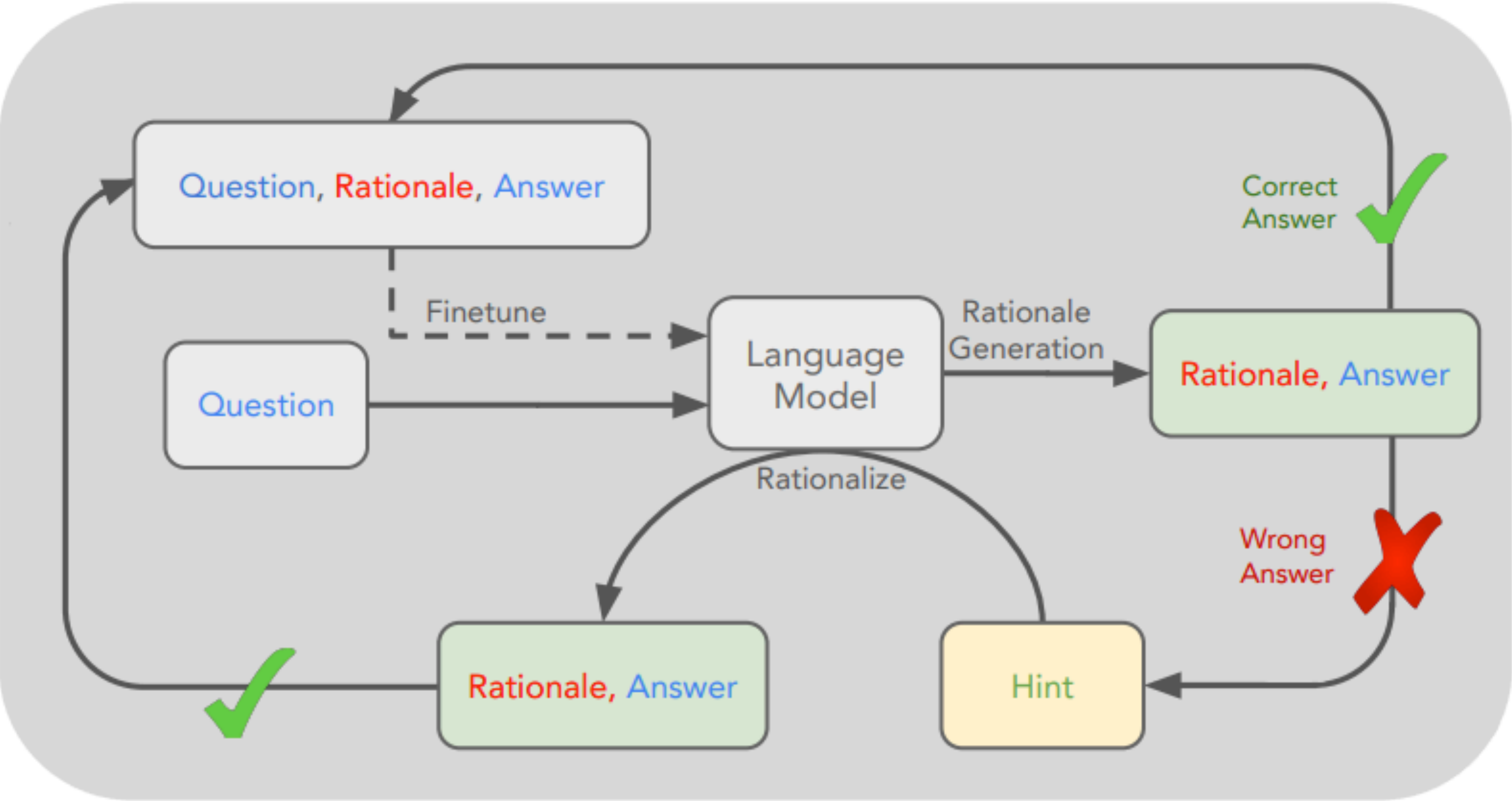


- The optimal test-time strategy should depend on question difficulty!
- Can outperform best-of-N up to 4x less test-time compute

How to (post-)train LLMs for better reasoning?

Self-Taught Reasoner (STaR)

STaR: Self-Taught Reasoner
Bootstrapping Reasoning With Reasoning

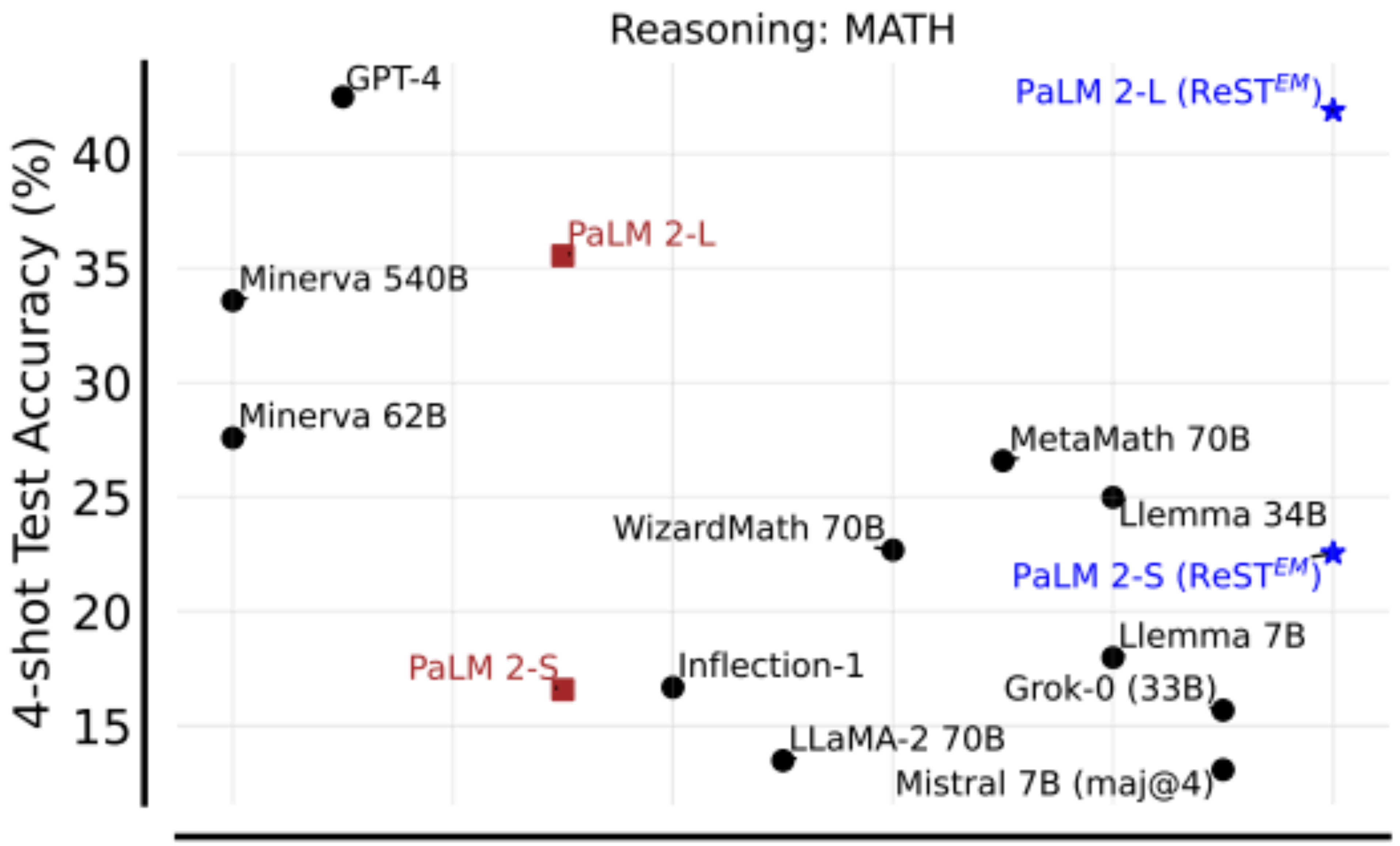


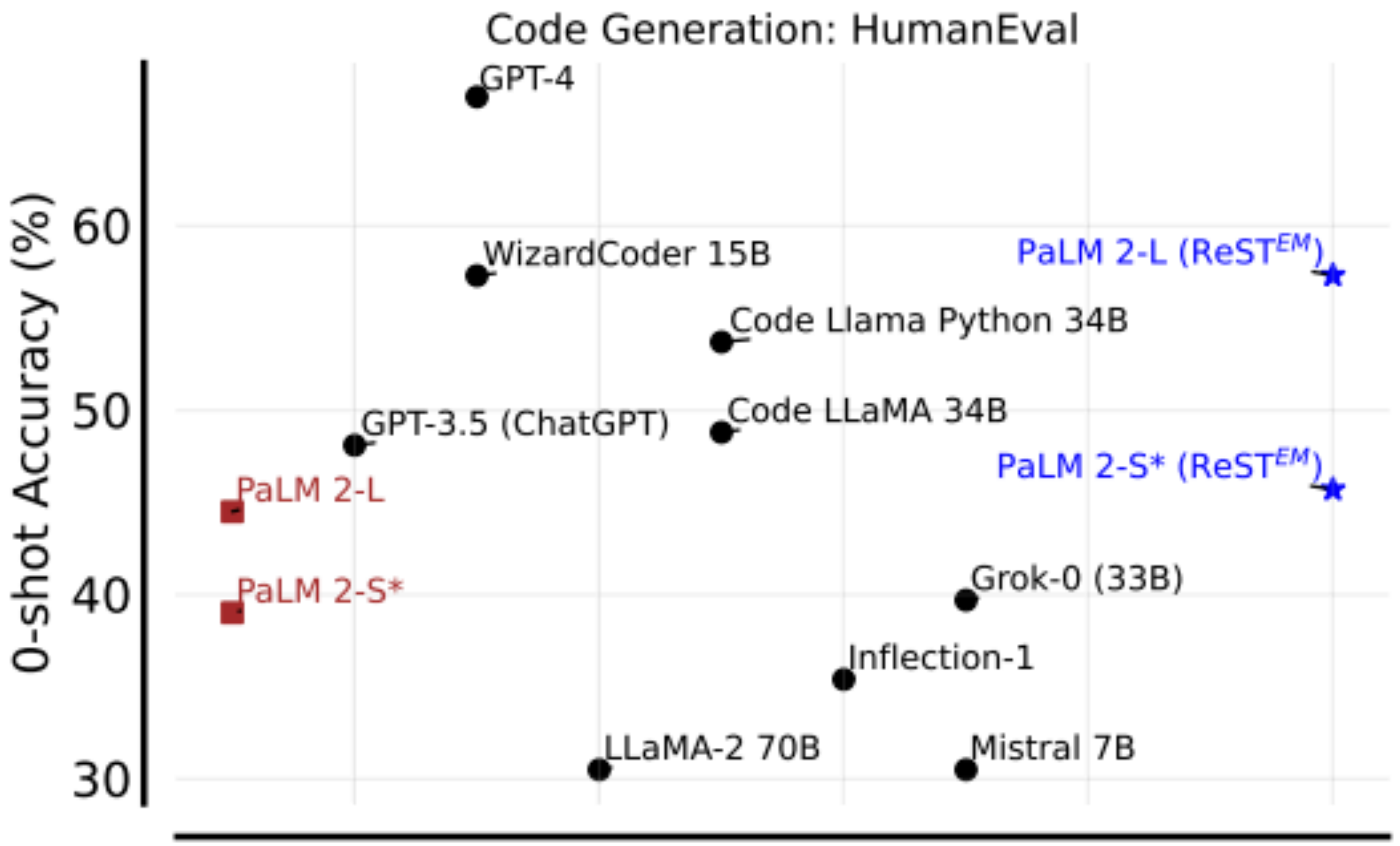
Q: What can be used to carry a small dog?
Answer Choices:
(a) swimming pool
(b) basket
(c) dog show
(d) backyard
(e) own home
A: The answer must be something that can be used to carry a small dog. Baskets are designed to hold things. Therefore, the answer is basket (b).

- Generate (rationale, answer)
- If answer is correct, add it back to the fine-tuning data

Reinforced Self-training (ReST^{EM})

1. `Generate` (`E-step`): The language model generates multiple output samples for each input context. Then, we filter these samples using a binary reward to collect the training dataset.
2. `Improve` (`M-step`): The original language model is supervised fine-tuned on the training dataset from the previous `Generate` step. The fine-tuned model is then used in the next `Generate` step.





Self-improvement and verification methods

Published as a conference paper at COLM 2024

V-STaR: Training Verifiers for Self-Taught Reasoners

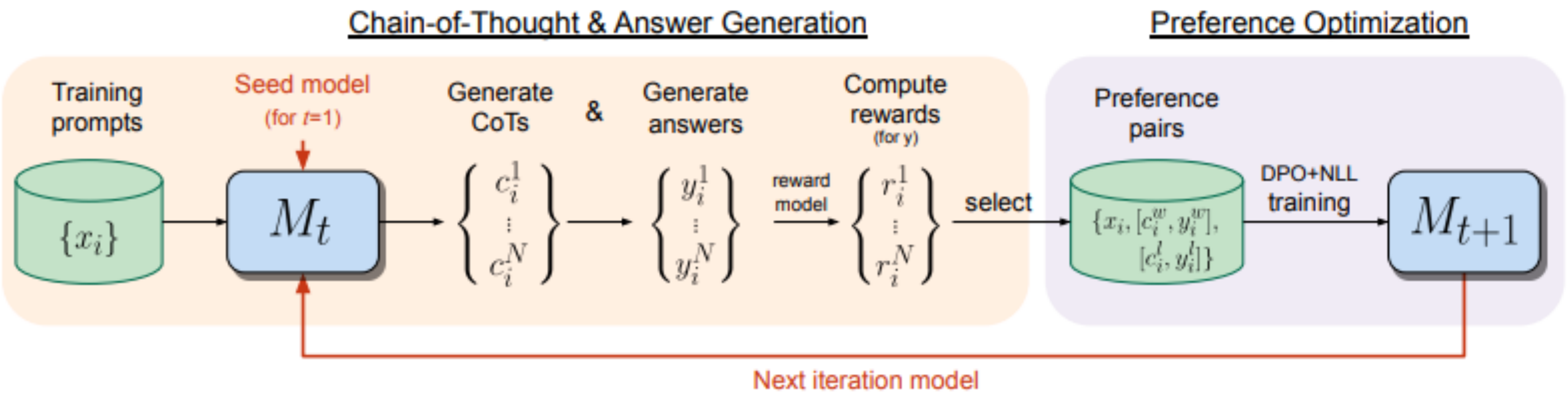
METHOD	GENERATOR DATA	VERIFIER DATA	ITERATIVE
SFT	$\mathcal{D}_{\text{SFT}}$	✗	✗
VERIFICATION	$\mathcal{D}_{\text{SFT}}$	$\mathcal{D}_{\text{SFT}} \cup$ GENERATED	✗
STAR	CORRECT GENERATED$_{t-1}$	✗	✓
RFT (STAR†[1 ITER])	$\mathcal{D}_{\text{SFT}} \cup$ CORRECT GENERATED	✗	✗
STAR†	$\mathcal{D}_{\text{SFT}} \cup$ CORRECT GENERATED$_{<t}$	✗	✓
V-STAR [1 ITER]	$\mathcal{D}_{\text{SFT}} \cup$ CORRECT GENERATED	$\mathcal{D}_{\text{SFT}} \cup$ GENERATED	✗
V-STAR	$\boldsymbol{\mathcal{D}}_{\textbf{SFT}} \cup$ **CORRECT GENERATED**$_{<t}$	$\boldsymbol{\mathcal{D}}_{\textbf{SFT}} \cup$ **GENERATED**$_{<t}$	✓

Preference optimization for reasoning

Iterative Reasoning Preference Optimization

Richard Yuanzhe Pang[1,2] **Weizhe Yuan**[1,2] **Kyunghyun Cho**[2]
He He[2] **Sainbayar Sukhbaatar**[1*] **Jason Weston**[1,2*]

[1]FAIR at Meta [2]New York University



Final thoughts

Image: Jim Fan

- How do any of these findings generalize beyond a single task?
- Post-training: any generic solutions? What data to use?