Agent Reasoning Pt 2: Inference Time Scaling

Prithviraj Ammanabrolu

THE BETTER YOUR REWARD THE EASIER THIS WHOLE PROCESS IS

(A GOOD) REWARD IS ENOUGH

How to side step hard problems

Humans are messy, ignore them

LLM as Judge / Learned Rewards have errors, ignore those too

Just calculate rewards on easily verifiable things

Math	Code	Games

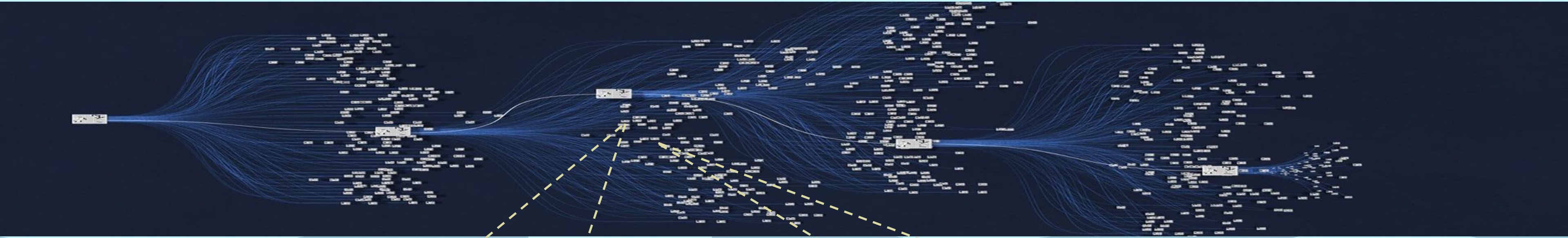
Why does verifiable reward matter?

- Reward hacking is learning loopholes
- Close the loopholes and it will learn to correctly solve the question

But Raj do you really need MDPs?
(Can’t you just do 1-step bandits?)

- Yes (No)

Step 1 Step 2 Step 3 Step 4 ...



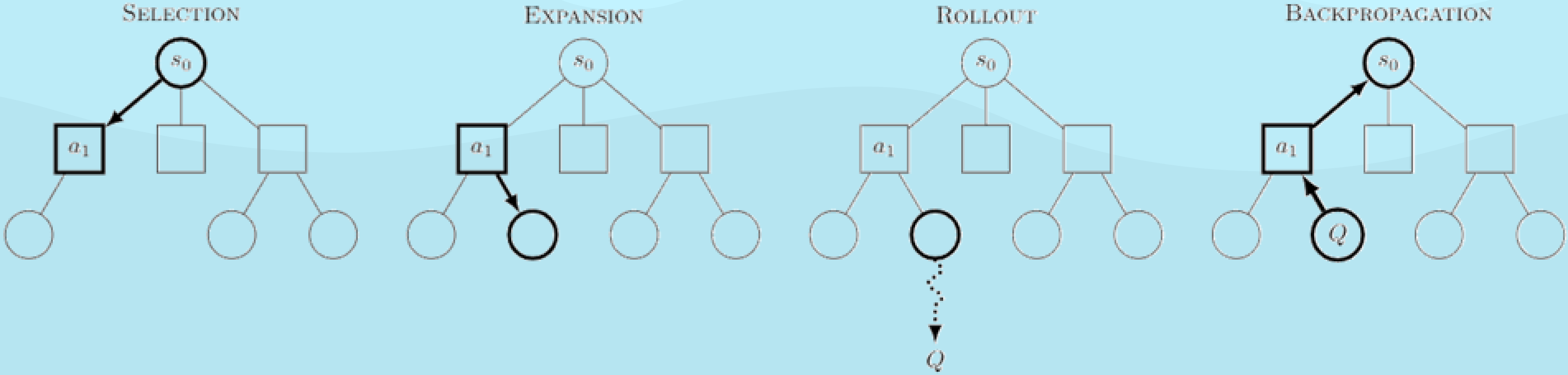
Open	mailbox	with	colleague
Go	north	in	comma
Examine	house	on	magic
The	my	above	amazing
Shout	four	below	scrolls
Carry	shoulder	until	some
Show	movie	was	bronze
Mount	bottom	over	cyclops
Cross	box	under	
Shred	Bozbar		
Adjust			

√

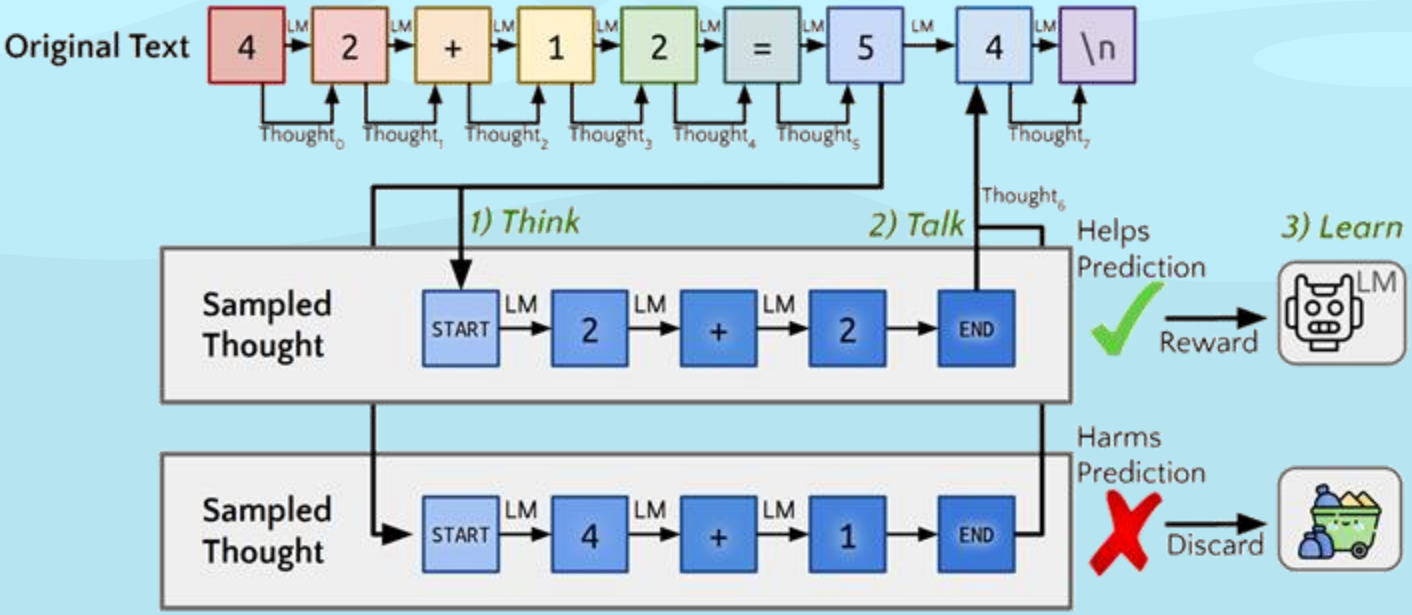
Open	mailbox	a	colleague
Go	north	in	show
Examine	house	on	magical
The	me	above	man
It	four	below	scrolls
Carry	shoulder	until	some
Show	movie	from	bronze
Mount	was	quite	cyclops
Cross	box	under	
Shred	Bozbar		
Adjust			

X

MCTS – Monte Carlo Tree Search

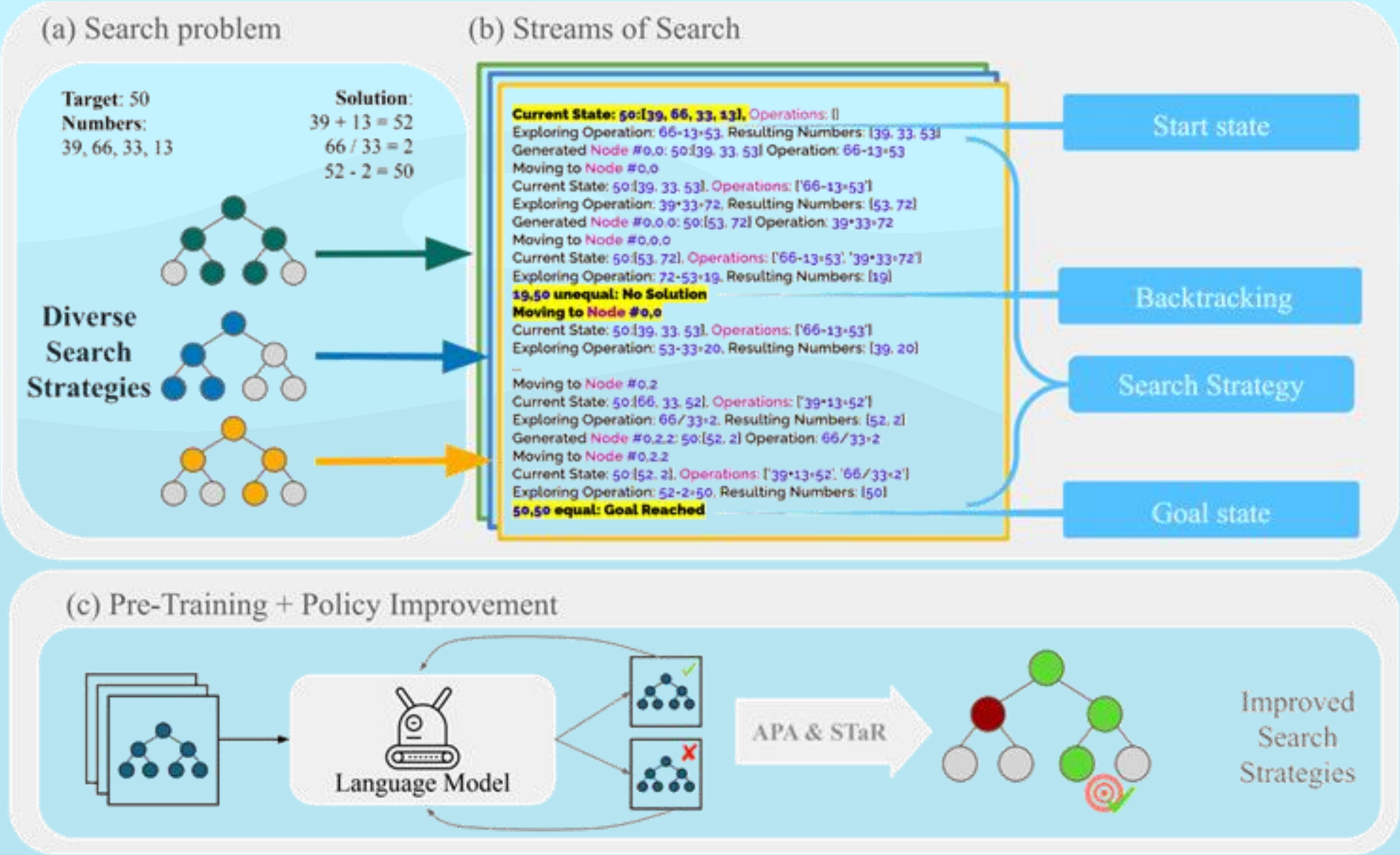


Quiet-STaR



Zelikman et al. 2024. Quiet-STaR: Language Models Can Teach Themselves to Think Before Speaking.

Stream of Search



Gandhi et al. 2024. Stream of Search (SoS): Learning to Search in Language.

Intuitions on why this works

Say you have a tree, every step on the tree is a choice of which action to do

Traditional MCTS usually chooses this by framing it as a bandit problem

Fixed equations e.g. UCT – bad inductive bias

Just learn when to backtrack

Key Assumptions Made

You need to see at least some +ve rewards

The better your base model the more chance there is of this

Constantly backtracking and rethinking works

Let's do more of that (where did that behavior come from?)

RL process will reward trajectories that do this

Tadaaa reasoning!

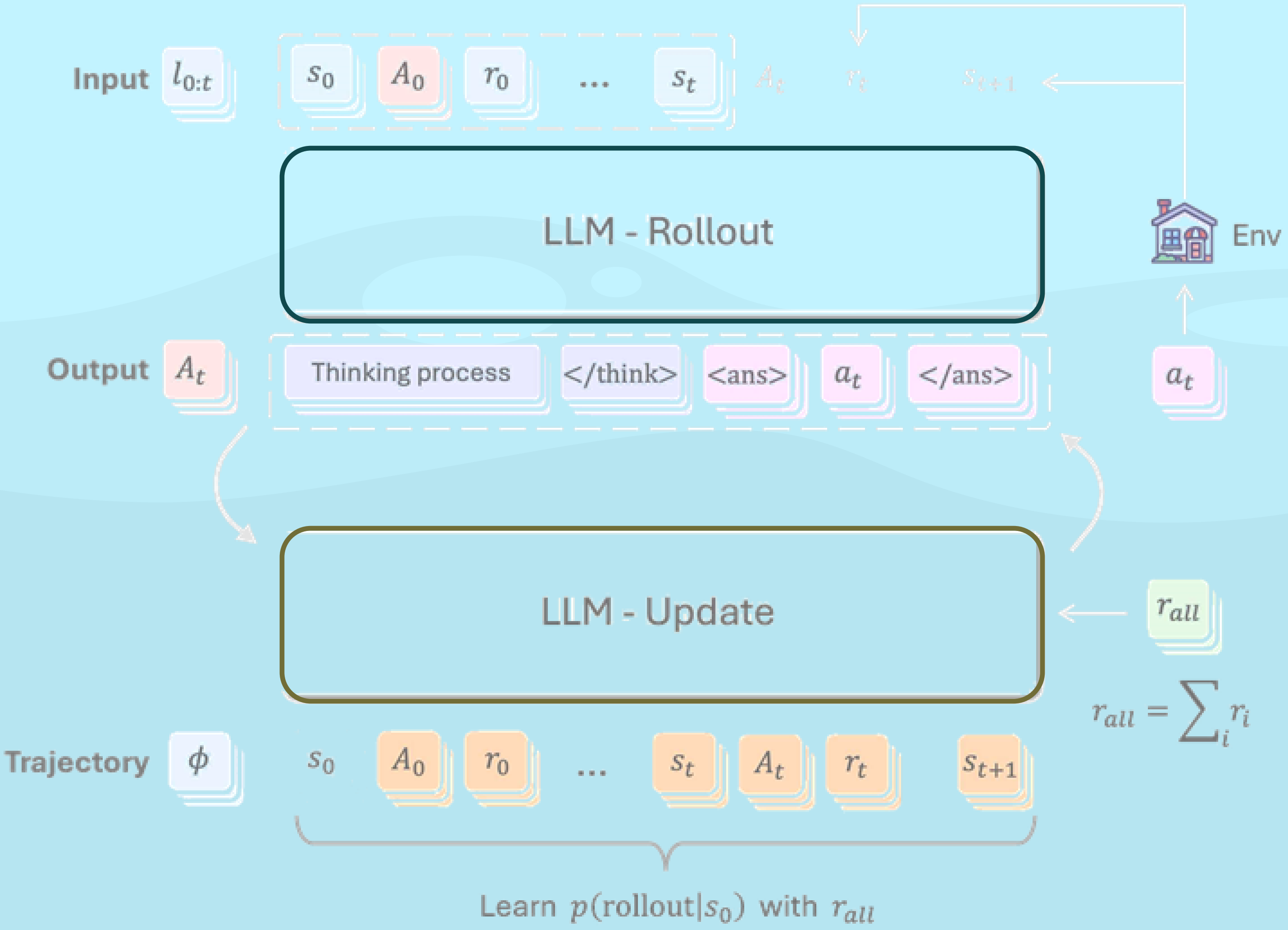


Diagram credit Zihan Wang based on R1 paper.

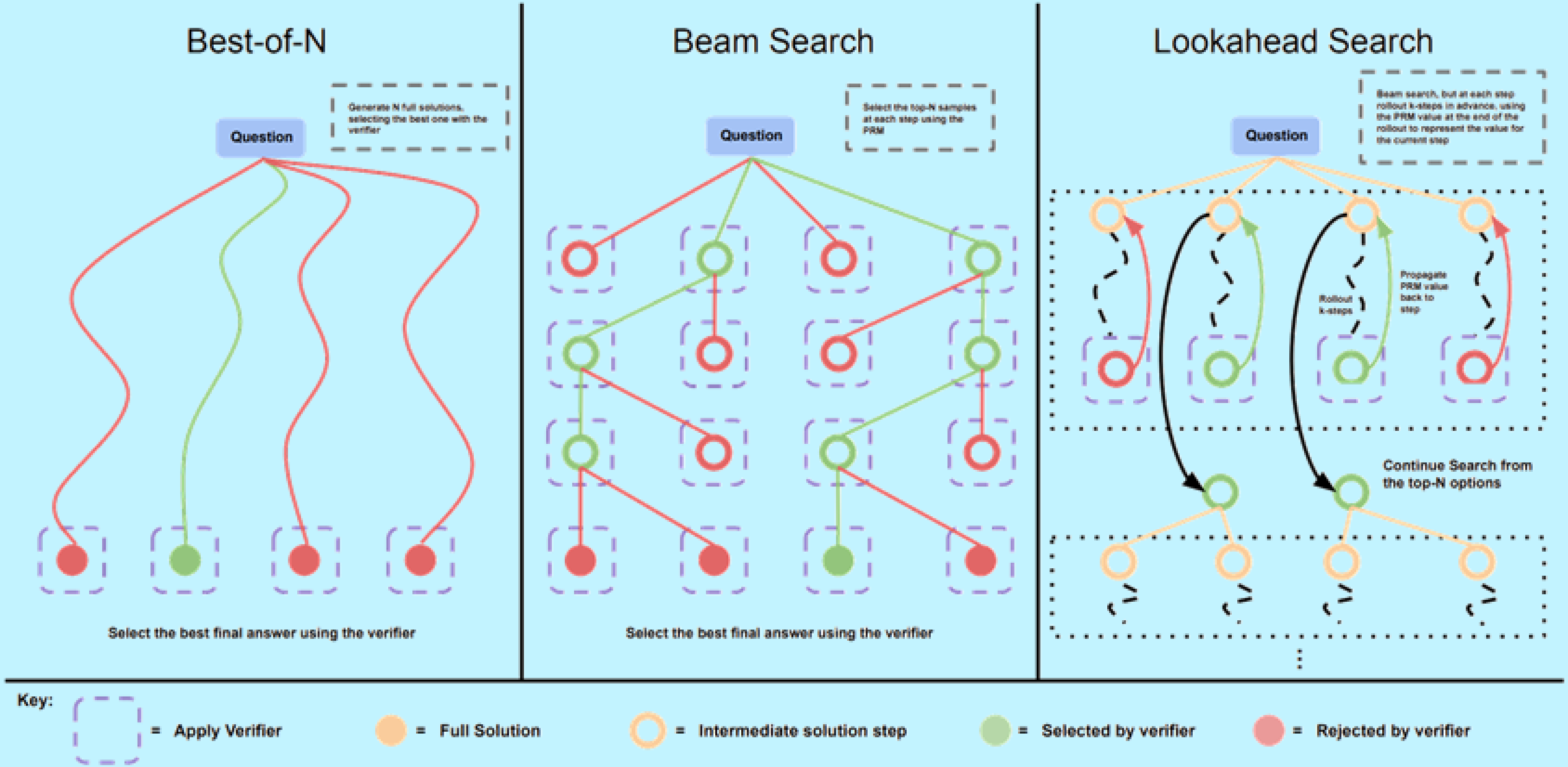
Effectiveness of
Extra Inference Time Compute

More inference compute → more ground truth feedback → more chances to learn "reasoning" behaviors

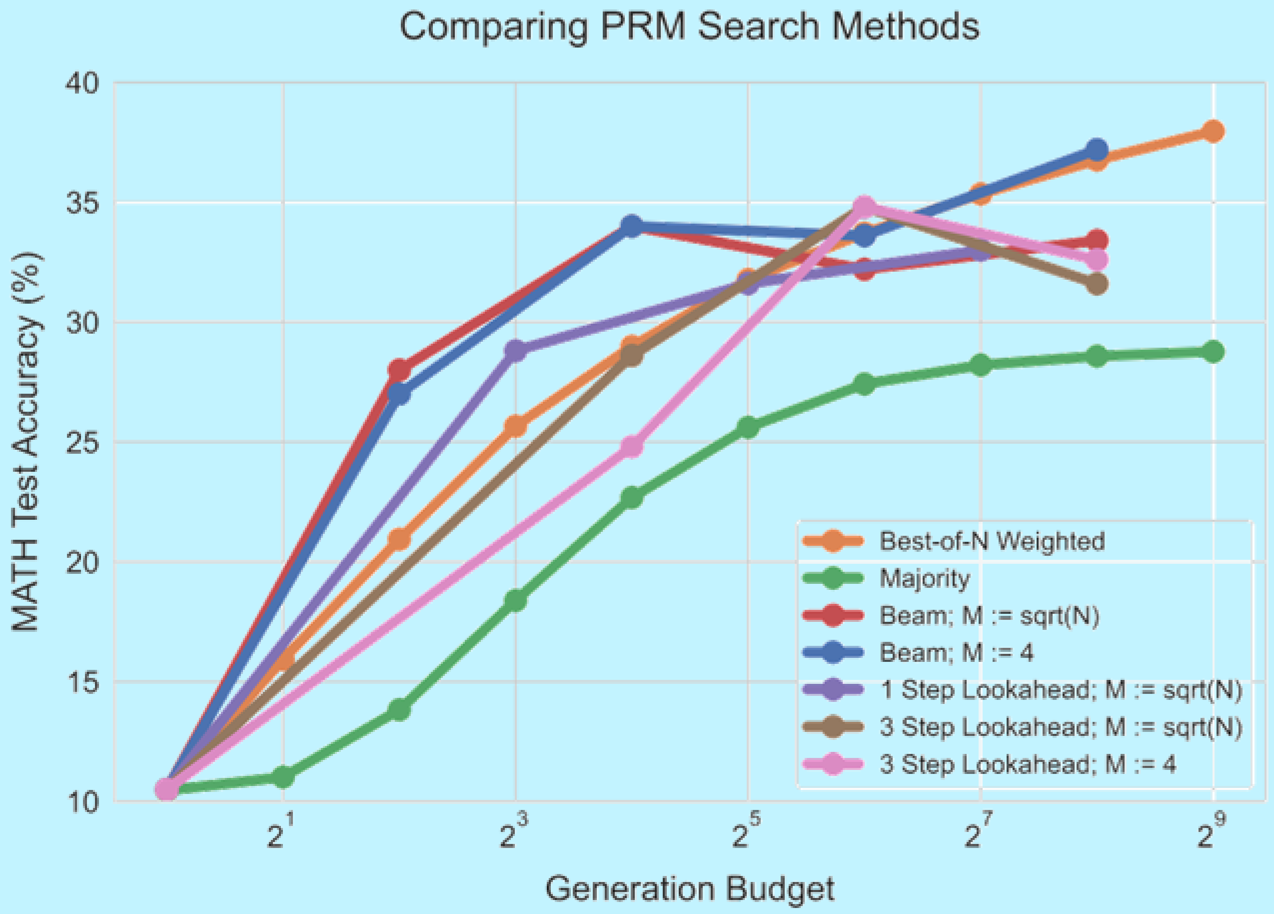
This is why RL scales with inference compute

Policy Weight Initialization (i.e. base model) matters

- I have been doing this since 2018 with GPT-1/2, then T5, then Llama 2 / 3
- First time I saw it working "cleanly" was a few months ago when my students tried it on Qwen 2.5 Math
  - "clean" = kinda human readable CoT, backtracking, big perf boosts
  - Quiet-STaR and other RL with verifiable rewards didn't have it
- We also know Meta's post training team tried this with Llama 2, it didn't work and they dropped it – opting to use a DPO based strat for Llama 3

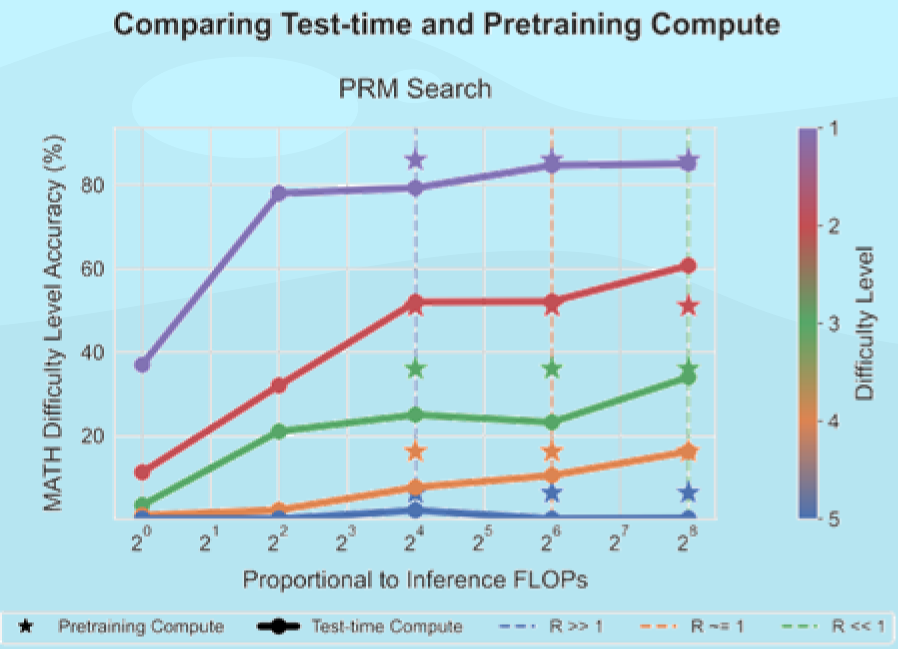


Snell et al. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters.



Snell et al. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters.

Test-time and pretraining compute are not 1-to-1 “exchangeable”. On easy and medium questions, test time compute can improve things a lot. With harder questions, you need better base models too. But after a certain point, inference compute scales better than train compute.



R = Tokens (Inf) / Tokens (train)

Snell et al. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters.

The main point of improving inference efficiency is speeding up online RL training by generating data

Online vs Offline RL

Online – you are improving a policy by observing feedback in close to real time

Offline – generate/use existing data and learn optimal policy from that

Online is Critical for all achieving reasoning behaviors. Personalized learning that can fix a model’s mistakes. Offline is useful for warm starting but isn’t enough to “discover” behaviors

What did Deepseek do differently?

GRPO – somewhat irrelevant, possibly more stable but you can get reasoning with PPO. Important thing is **Distributed Online RL**

Better base model somehow (no pre-training/SFT data released so unclear how exactly)

Imp to note that the continued pre-training vs SFT distinction is meaningless here. Only question is, did some kind of step by step data exist in the mix

Saw bad RL results for small models (no better than others) and decided to invest in **infra to scale it** anyways

What did Deepseek do differently?

Format rewards on top of accuracy rewards to make sure thinking was between <think> tokens (this isn't unique) but possibly important for maintaining human readability

Use small amount of human-filtered CoT as SFT first to do better policy init

There doesn't exist much open source data of this format out there right now

Immediate Barriers

Astute listener may notice that Deepseek method simplified is just policy gradient with Monte Carlo samples (alternatively framed as a bandit problem if you hate RL)

MC is high variance, relies heavily on stumbling across right trajectory

How to fix?

- Small amount of human-filtered CoT data
- Finegrained / Process Reward Models

Human data lets us train both... but is expensive

More Immediate Barriers

How to fix?

Small amount of human-filtered CoT data

Finegrained / Process Reward Models

Human data lets us train both

RL easily overfits to noise in learned rewards

Preventing reward hacking is AGI-complete but there do exist some ways to progress

Reasoning Data Collection

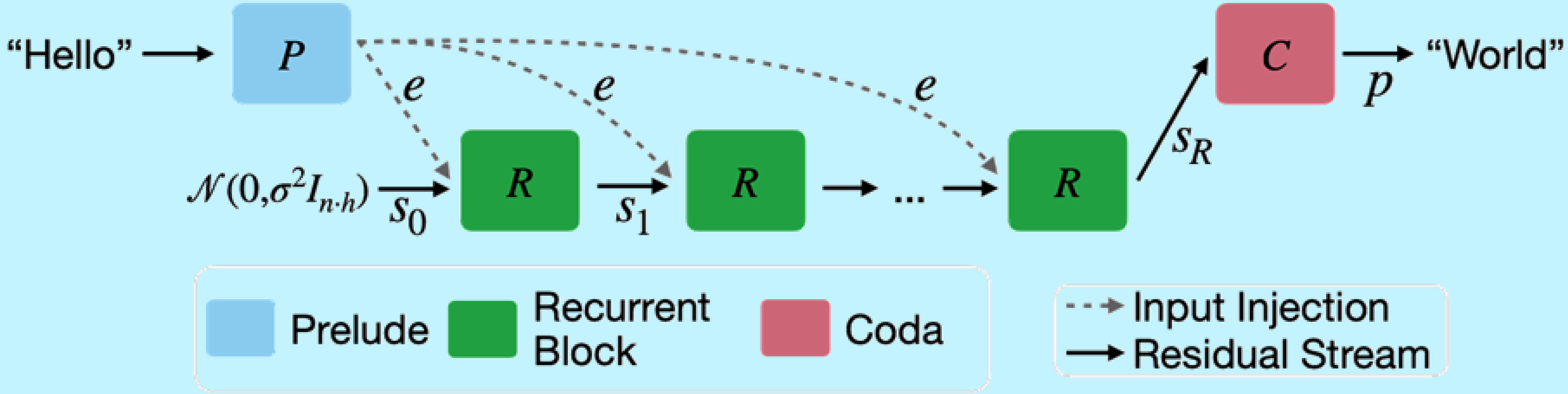
1. Human makes prompt
2. Model (partially tuned) produces CoT for it
3. Human finds first step CoT is wrong and rewrites just that step
4. Model generates again from there

2-4 repeat until correct answer. This is much more scalable way of doing human CoT filtering than people writing traces from scratch

Many Ways of Scaling Inference

Many ways of scaling inference compute

- Special ”thinking” tokens
- Yapping in language: wait …
- Maybe you don’t even need language at all?



$$\mathbf{e} = P(\mathbf{x})$$

$$\mathbf{s}_0 \sim \mathcal{N}(\mathbf{0}, \sigma^2 I_{n\cdot h})$$

$$\mathbf{s}_i = R(\mathbf{e}, \mathbf{s}_{i-1}) \qquad \text{for} \quad i \in \{1, \dots, r\}$$

$$\mathbf{p} = C(\mathbf{s}_r),$$

Laughs in Schmidthuber

Geiping et al. 2024. Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach.

But Raj do you really need MDPs? (Can’t you just do 1-step bandits?)

- Yes (No)

Takeaways

- If you have a true reward, have minimal inductive biases in your learning method
- Make sure your method scales along some axis and invest in engineering for it
- Get better rewards to close the gap to a true reward