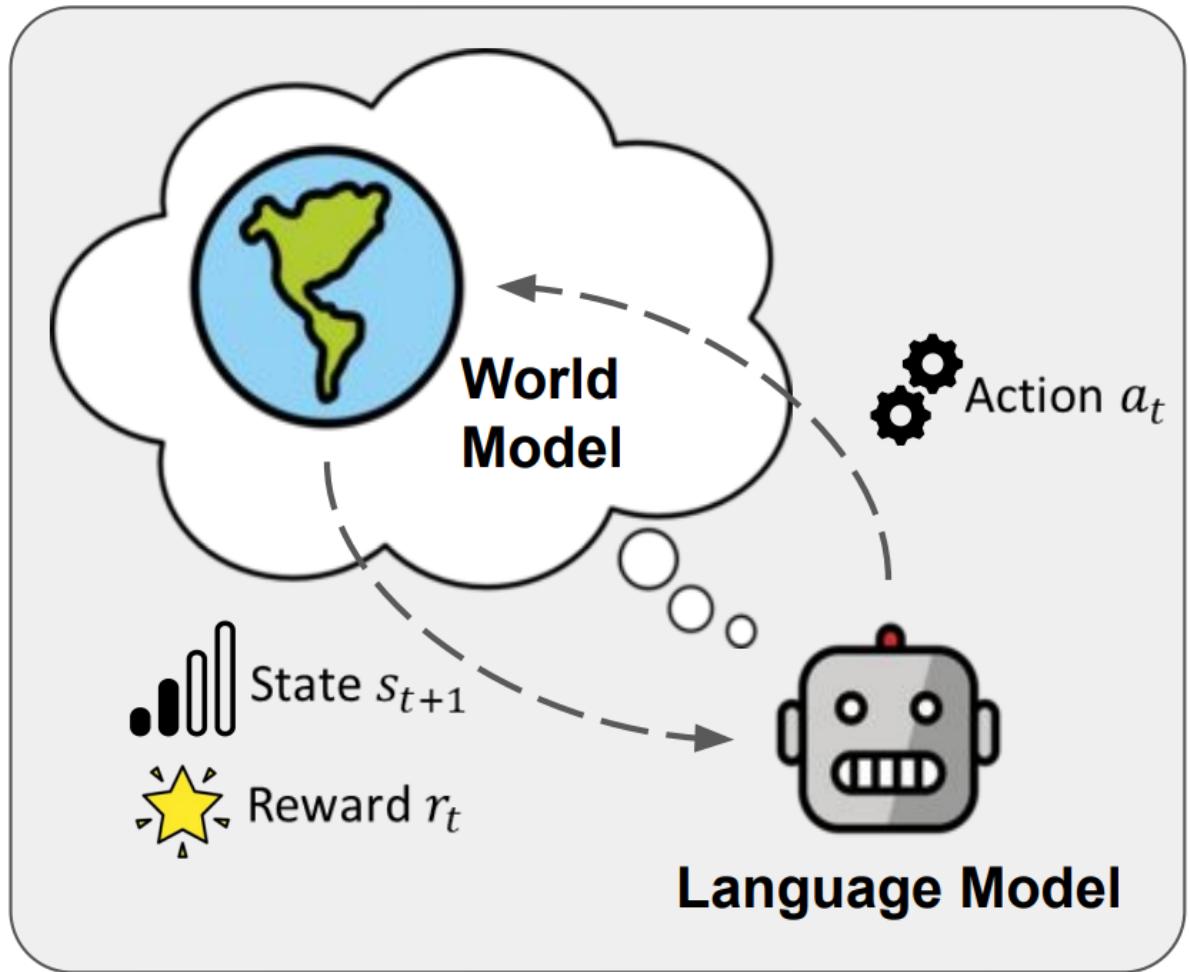


**Reasoning with Language  
Model is Planning with  
World Model**

# World Model

What is a world model and why is it useful?



- predicts the next state after applying an action to the current state
- enforces a more principled planning approach
- both the agent and the world are the same but repurposed LLM

## Prompt prefix

I am playing with a set of blocks where I need to arrange the blocks into stacks. Here are the actions I can do

Pick up a block  
Unstack a block from on top of another block

Put down a block  
Stack a block on top of another block

I have the following restrictions on my actions:  
<restrictions>

## Agent few-shot

[STATEMENT]

As initial conditions I have that, the orange block is clear, the yellow block is clear, the hand is empty, the blue block is on top of the red block, the orange block is on top of the blue block, the red block is on the table and the yellow block is on the table.

My goal is to have that the blue block is on top of the red block and the yellow block is on top of the orange block.

My plan is as follows:

[PLAN]

pick up the yellow block  
stack the yellow block on top of the orange block

[PLAN END]

## World few-shot

[SCENARIO 1]

[STATE 0] I have that, the white block is clear, the cyan block is clear, the brown block is clear, the hand is empty, the white block is on top of the purple block, the purple block is on the table, the cyan block is on the table and the brown block is on the table.

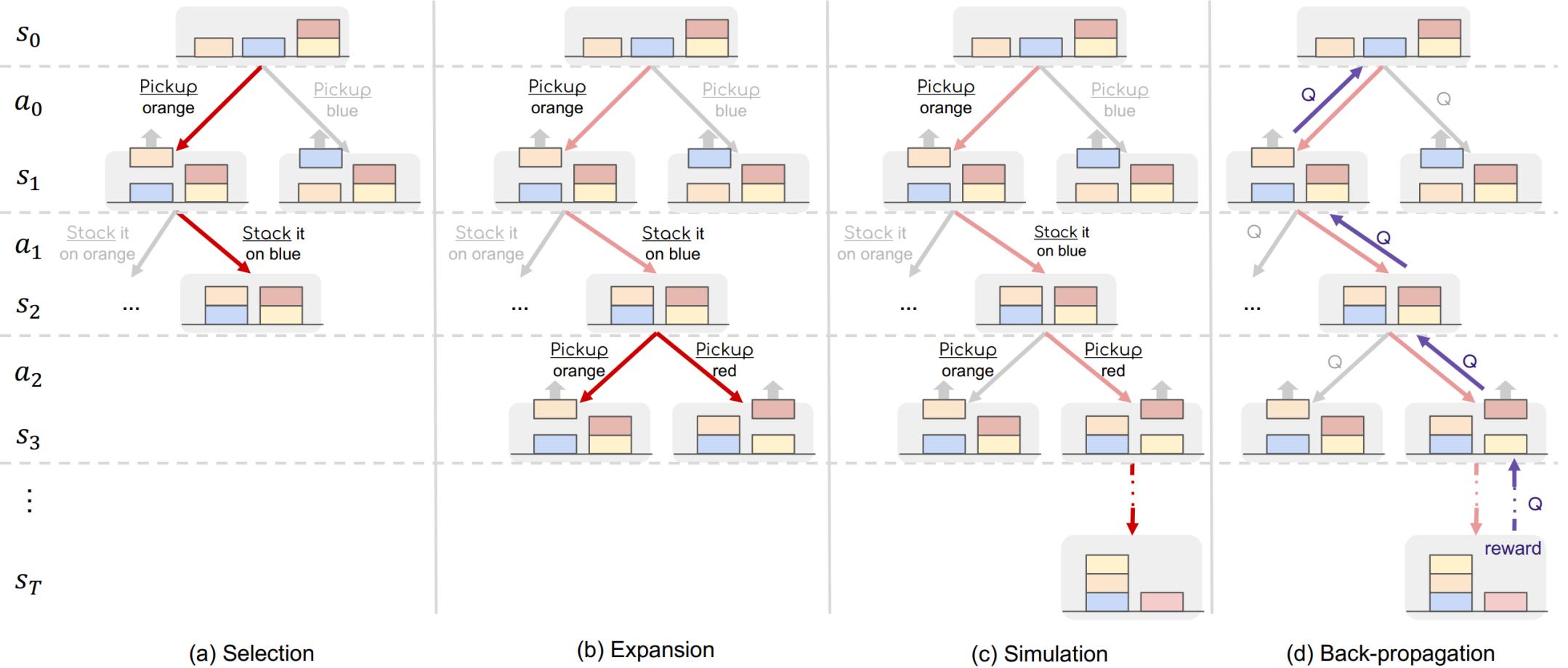
[ACTION] Pick up the brown block.

[CHANGE] The hand was empty and is now holding the brown block, the brown block was on the table and is now in the hand, and the brown block is no longer clear.

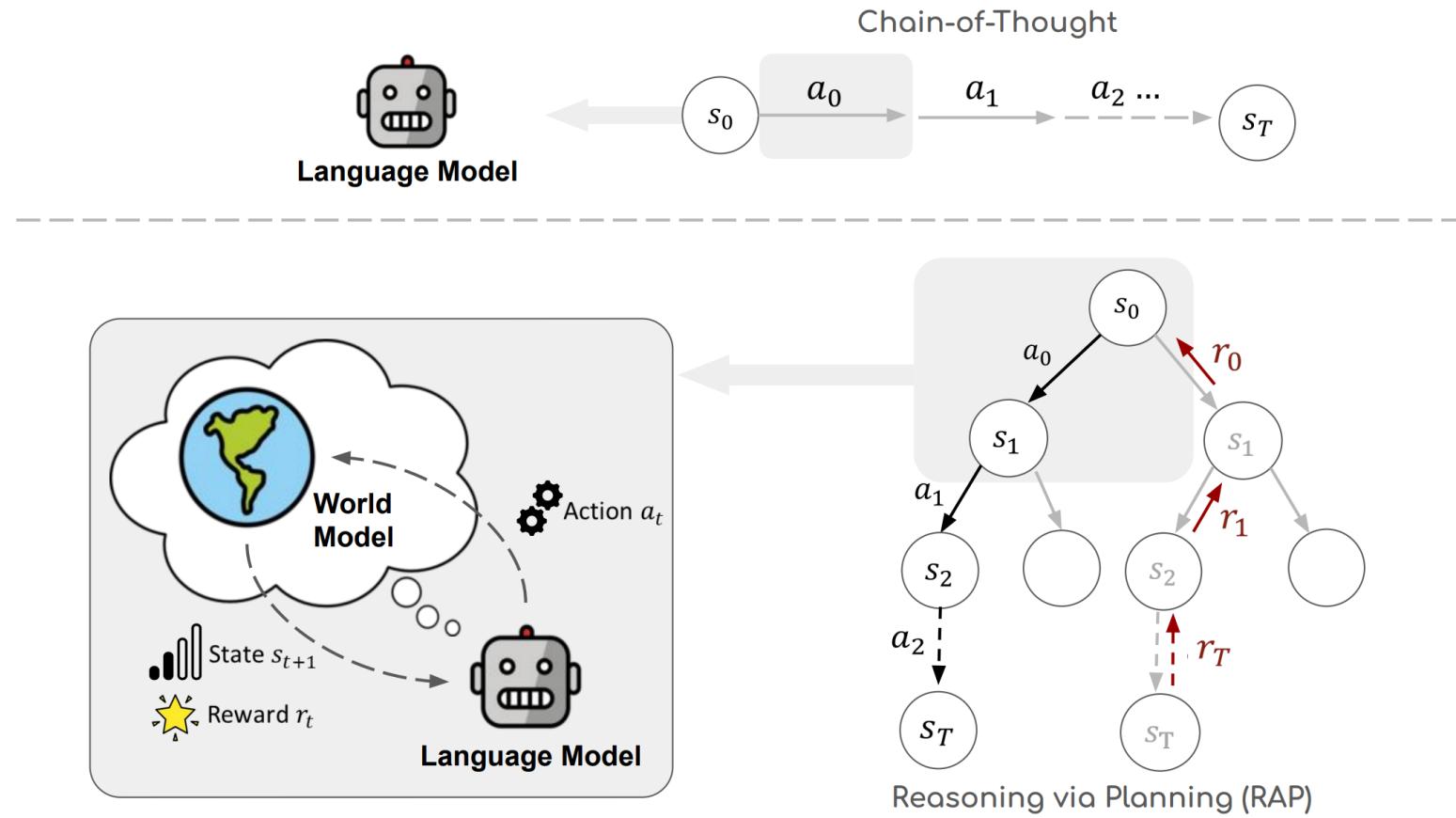
[STATE 1] I have that, the white block is clear, the cyan block is clear, the brown block is in the hand, the hand is holding the brown block,

# Monte Carlo Tree Search

How to reason with a world model?



$$a^* = \arg \max_{a \in A(s)} \left[ Q(s, a) + w \sqrt{\frac{\ln N(s)}{N(c(s, a))}} \right]$$



“ Planning, a central ability in intelligent agents, involves generating a series of actions to achieve a specific goal ”

## Reward design

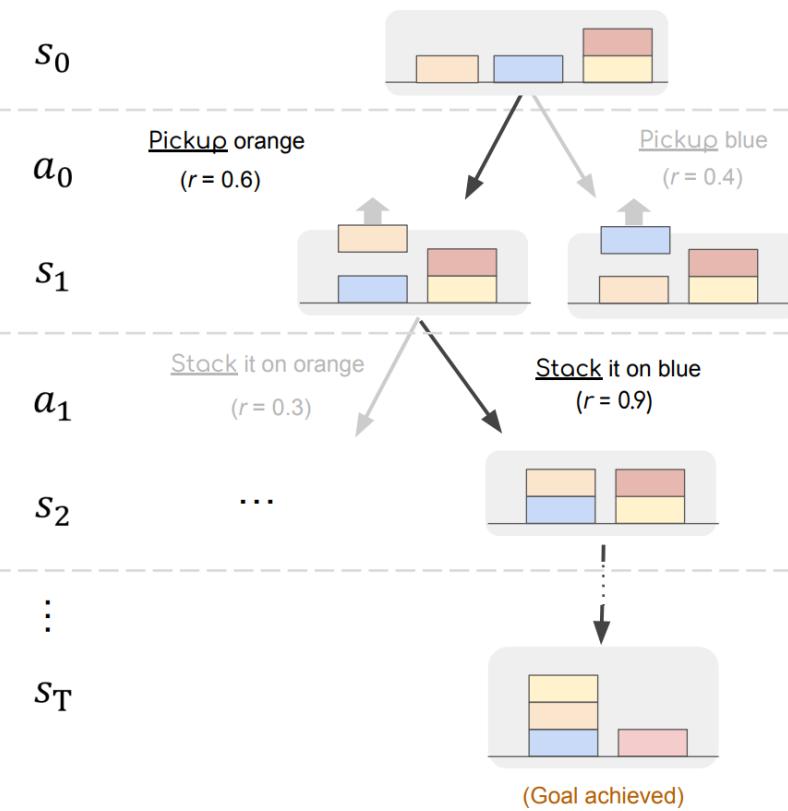
Table on GSM8k benchmarking

$R_1$	$R_2$	$R_3$	$\text{RAP}^{(1)}$	$\text{RAP}^{(10)}$	+aggr
✓	✗	✓	0.410	0.450	0.503
✓	✗	✗	0.350	0.447	0.490
✓	✓	✗	0.373	0.423	0.443

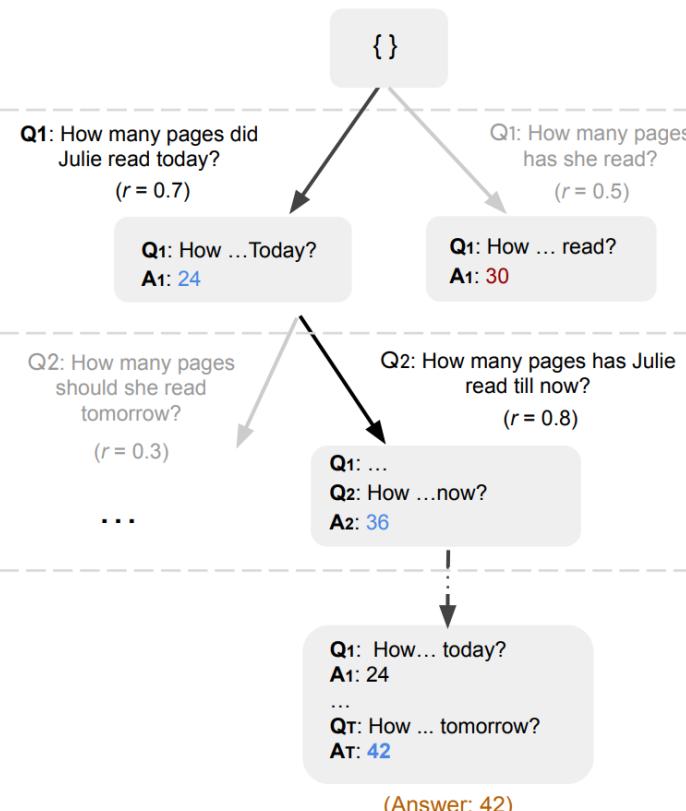
- **State confidence** ( $R_1$ )  
*the empirical probability of the state's validity*
- **Likelihood** ( $R_2$ )  
*likelihood of an action being chosen by the agent*
- **Self-evaluation** ( $R_3$ )  
*model's confidence in the reasoning step*
- **Heuristics**

# Examples

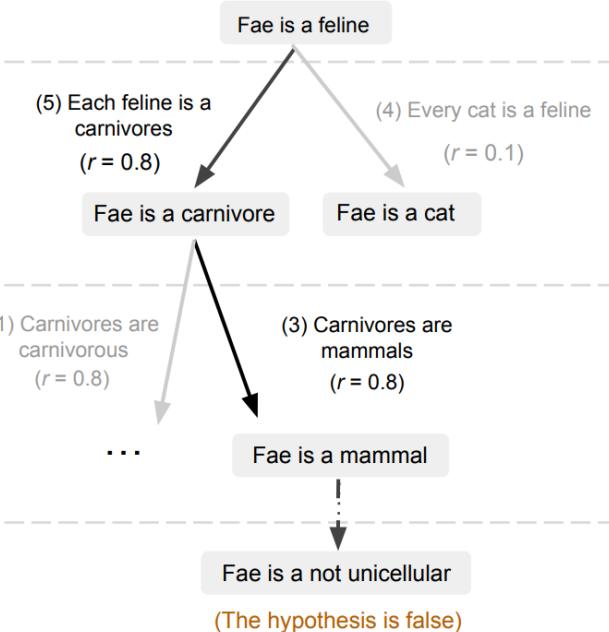
**Initial State:** The orange block is on the table, the blue block is on the table, and the red block...  
**Goal:** The orange block is on the blue block, and the yellow block is on the orange block.



Julie is reading... She wants to read half of the remaining pages tomorrow. How many pages should she read?



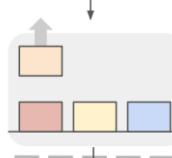
(1) Carnivores are carnivorous  
(2) Animals are not unicellular  
(3) Carnivores are mammals ...  
Fact: Fae is a feline  
**Hypothesis:** Fae is unicellular?



## CoT

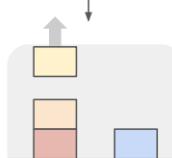
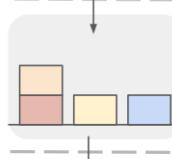
$a_0$

unstack the orange block from on top of the yellow block



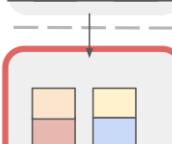
$a_1$

stack the orange block on top of the red block



$a_2$

pick up the yellow block



$a_3$

stack the yellow block on top of the blue block

## RAP

(Goal: The red block is on blue block; the orange block is on the red block)

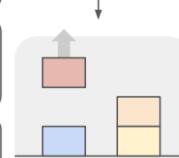
$s_0$

I have that, the red block is clear, the blue block is clear, the orange block is clear, the hand is empty, the orange block is on top of the yellow block, the red block is on the table, the blue block is on the table and the yellow block is on the table.



$a_0$

Pick up the red block.

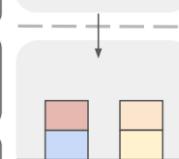


$s_1$

I have that, the blue block is clear, the orange block is clear, the red block is in the hand, the hand is holding the red block, the orange block is on top of the yellow block, the blue block is on the table, and the yellow block is on the table.

$a_1$

Stack the red block on top of the blue block.

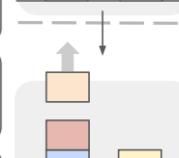


$s_2$

I have that, the orange block is clear, the red block is clear, the hand is empty, the orange block is on top of the yellow block, the red block is on top of the blue block, the blue block is on the table, and the yellow block is on the table.

$a_2$

Unstack the orange block from on top of the yellow block.



$s_3$

I have that, the orange block is in the hand, the red block is clear, the yellow block is clear, the hand is holding the orange block, the red block is on top of the blue block, the blue block is on the table, and the yellow block is on the table.

$a_3$

Stack the orange block on top of the red block.



$s_4$

I have that, the orange block is clear, the yellow block is clear, the hand is empty, the orange block is on top of the red block, the red block is on top of the blue block, the blue block is on the table, and the yellow block is on the table.



# Results

Blockworld				Math (GSM8k)		Logic (ProntoQA)		
Method	2-step	4-step	6-step	Method	Accuracy (%)	Method	Pred Acc	Proof Acc
CoT	0.17	0.02	0.00	Chain-of-Thought	29.4	CoT	87.8	64.8
CoT - pass@10	0.23	0.07	0.00	+ SC <sup>(10)</sup>	46.8	CoT + SC	89.8	-
CoT (GPT-4)	0.50	0.63	0.40	Least-to-Most	25.5	RAP (Ours)	<b>94.2</b>	<b>78.8</b>
RAP <sup>(10)</sup>	1.00	0.86	0.26	+ SC <sup>(10)</sup>	42.5			
RAP <sup>(20)</sup>	<b>1.00</b>	<b>0.88</b>	<b>0.42</b>	RAP <sup>(1)</sup>	40.0			
				RAP <sup>(10)</sup>	48.6			
				+ aggr	<b>51.6</b>			

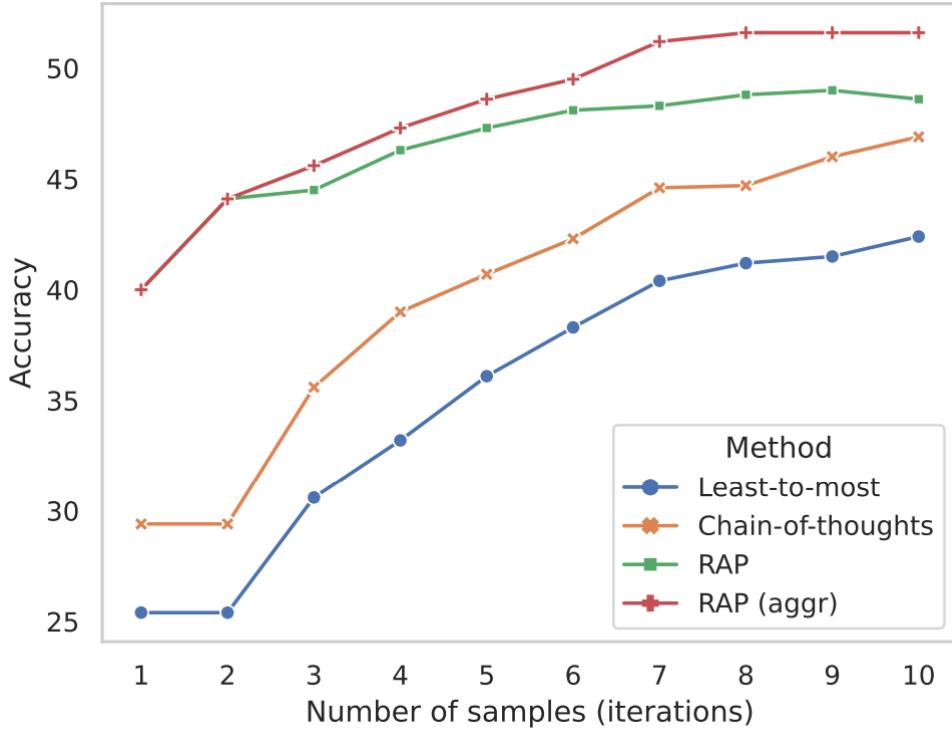
*The model used by authors is LLaMA-33B.*

*All figures refer to the success frequency.*

*The superscripts indicate the number of samples or iterations.*

<b>Setting</b>	<b>Method</b>	<b>2-step</b>	<b>4-step</b>	<b>6-step</b>	<b>8-step</b>	<b>10-step</b>	<b>12-step</b>	<b>All</b>
Easy	CoT	0.49	0.18	0.06	0.01	0.01	0.00	0.08
	RAP <sup>(10)</sup>	1.00	0.99	0.75	0.61	0.32	0.32	0.65
Hard	CoT	0.22	0.14	0.02	0.02	0.00	0.00	0.05
	RAP <sup>(10)</sup>	0.67	0.76	0.74	0.48	0.17	0.09	0.51

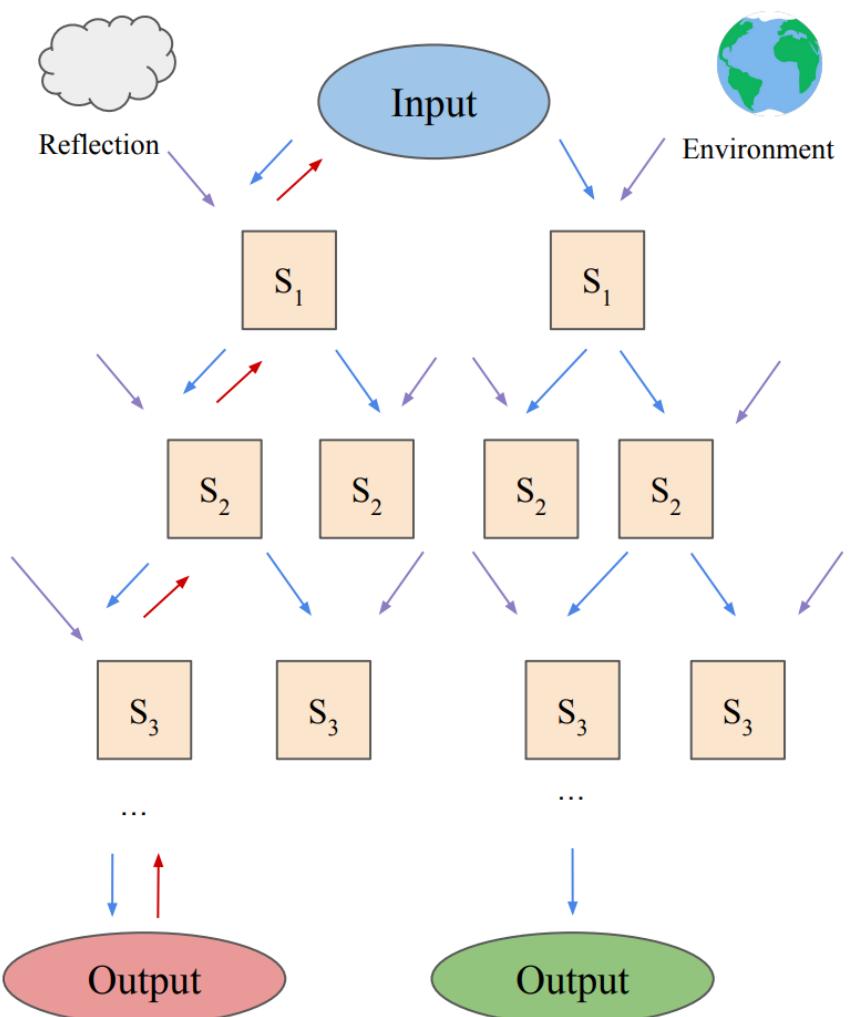
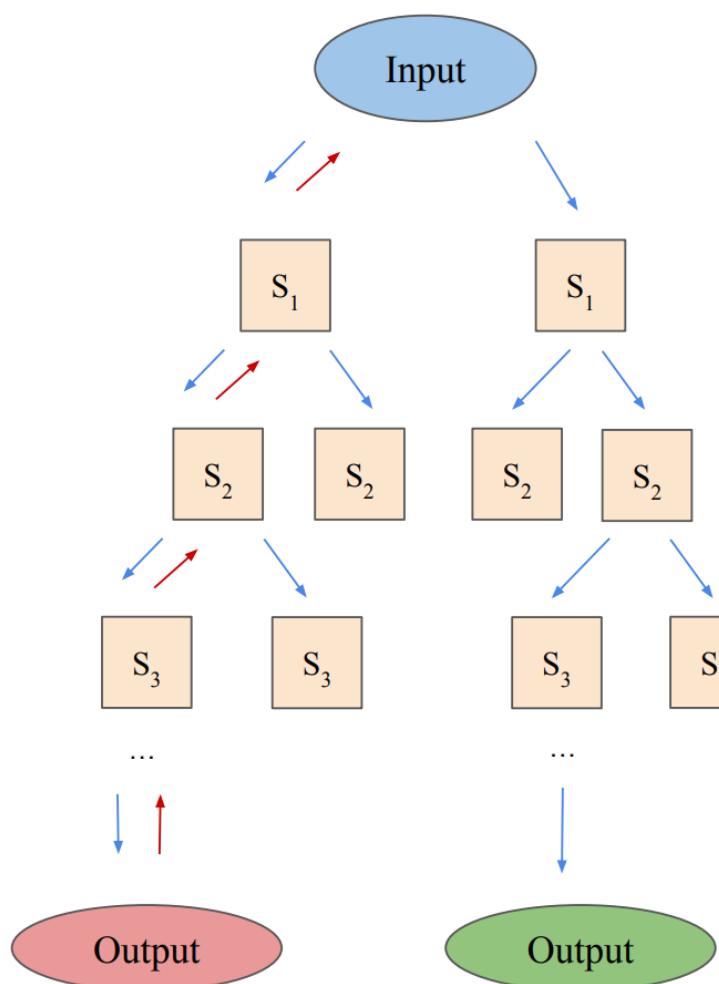
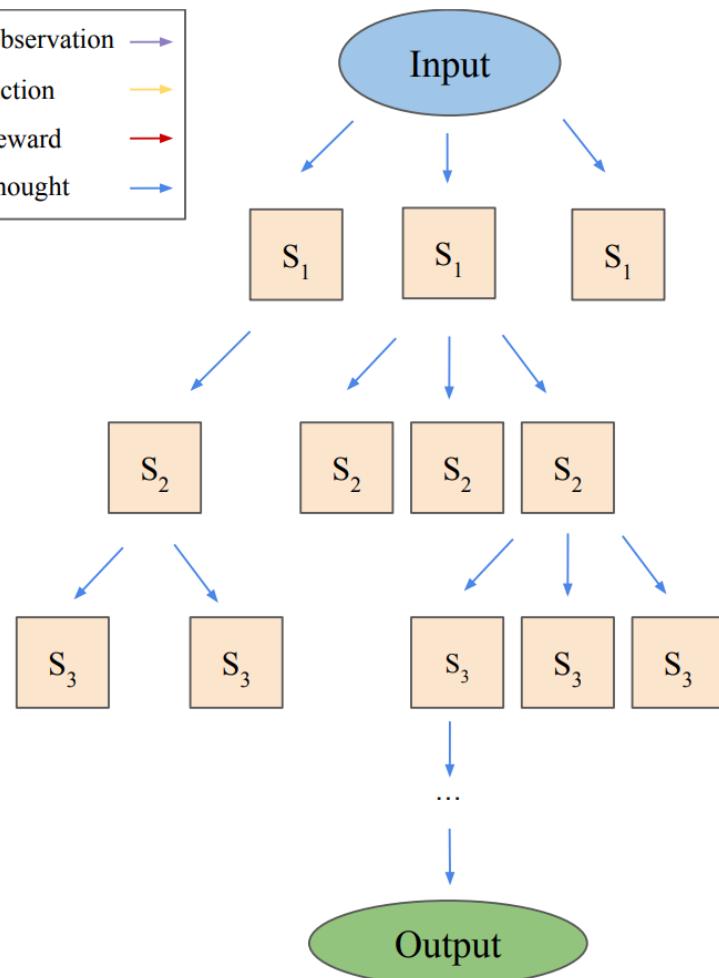
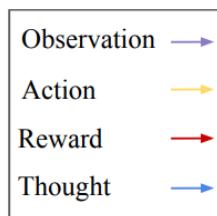
Chain of Thoughts drops significantly with time in comparison to RAP.



The reward used is a geometric mean of state confidence and self evaluation.

# Tree search algorithms

Can we do better?



Methods comparison on coding dataset HumanEval

Approach	Self Refine	Memory	Reasoning	Decision Making	Planning
CoT (Wei et al., 2022)	✗	✗	✓	✗	✗
ReAct (Yao et al., 2023b)	✗	✗	✓	✓	✗
ToT (Yao et al., 2023a)	✗	✗	✓	✗	✓
RAP (Hao et al., 2023)	✗	✗	✓	✗	✓
Self-Refine (Madaan et al., 2023)	✓	✗	✓	✗	✗
Beam Search (Xie et al., 2023)	✓	✗	✓	✗	✗
Reflexion (Shinn et al., 2023)	✓	✓	✓	✓	✗
<b>LATS (Ours)</b>	✓	✓	✓	✓	✓

Prompt Method	Model	Pass@1
CoT (Wei et al., 2022)	GPT-3.5	46.9
ReAct (Yao et al., 2023b)	GPT-3.5	56.9
Reflexion (Shinn et al., 2023)	GPT-3.5	68.1
ToT (Yao et al., 2023a)	GPT-3.5	54.4
RAP (Hao et al., 2023)	GPT-3.5	63.1
<b>LATS (Ours)</b>	GPT-3.5	<b>86.9</b>

I/O	GPT-4	80.1
Reflexion	GPT-4	91.0
<b>LATS</b>	GPT-4	<b>94.4</b>

\*Language agent tree search unifies reasoning acting and planning in language models

# Self-Refine

(a) Dialogue:  $x, y_t$

User: I am interested in playing Table tennis.

Response: I'm sure it's a great way to socialize, stay active

(b) FEEDBACK  $\mathbf{fb}$

Engaging: Provides no information about table tennis or how to play it.

User understanding: Lacks understanding of user's needs and state of mind.

(c) REFINE  $y_{t+1}$

Response (refined): That's great to hear (...) ! It's a fun sport requiring quick reflexes and good hand-eye coordination. Have you played before, or are you looking to learn?

(d) Code optimization:  $x, y_t$

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

(e) FEEDBACK  $\mathbf{fb}$

This code is slow as it uses brute force. A better approach is to use the formula ...  $(n(n+1))/2$ .

(f) REFINE  $y_{t+1}$

```
Code (refined)

def sum_faster(n):
    return (n*(n+1))//2
```

# Decision-making

**Question:** Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

**Answer:** iPod

**Standard**



**Reason only**

**Thought:** Let's think step by step. Apple Remote was originally designed to interact with Apple TV. Apple TV can be controlled by iPhone, iPad, and iPod Touch. So the answer is iPhone, iPad, and iPod Touch.

**Answer:** iPhone, iPad, iPod Touch



**Act only**

**Act 1:** `Search[Apple Remote]`

**Obs 1:** The Apple Remote is a remote control introduced...

**Act 2:** `Search[Front Row]`

**Obs 2:** Could not find [Front Row]. Similar: ...

**Act 3:** `Search[Front Row (software)]`

**Obs 3:** Front Row is a discontinued media center software ...

**Act 4:** `Finish[yes]`



**ReAct**

**Thought 1:** I need to search Apple Remote and find the program it was originally designed to interact with.

**Act 1:** `Search[Apple Remote]`

**Obs 1:** The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

**Thought 2:** Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

**Act 2:** `Search[Front Row]`

**Obs 2:** Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

**Thought 3:** Front Row is not found. I need to search Front Row (software) .

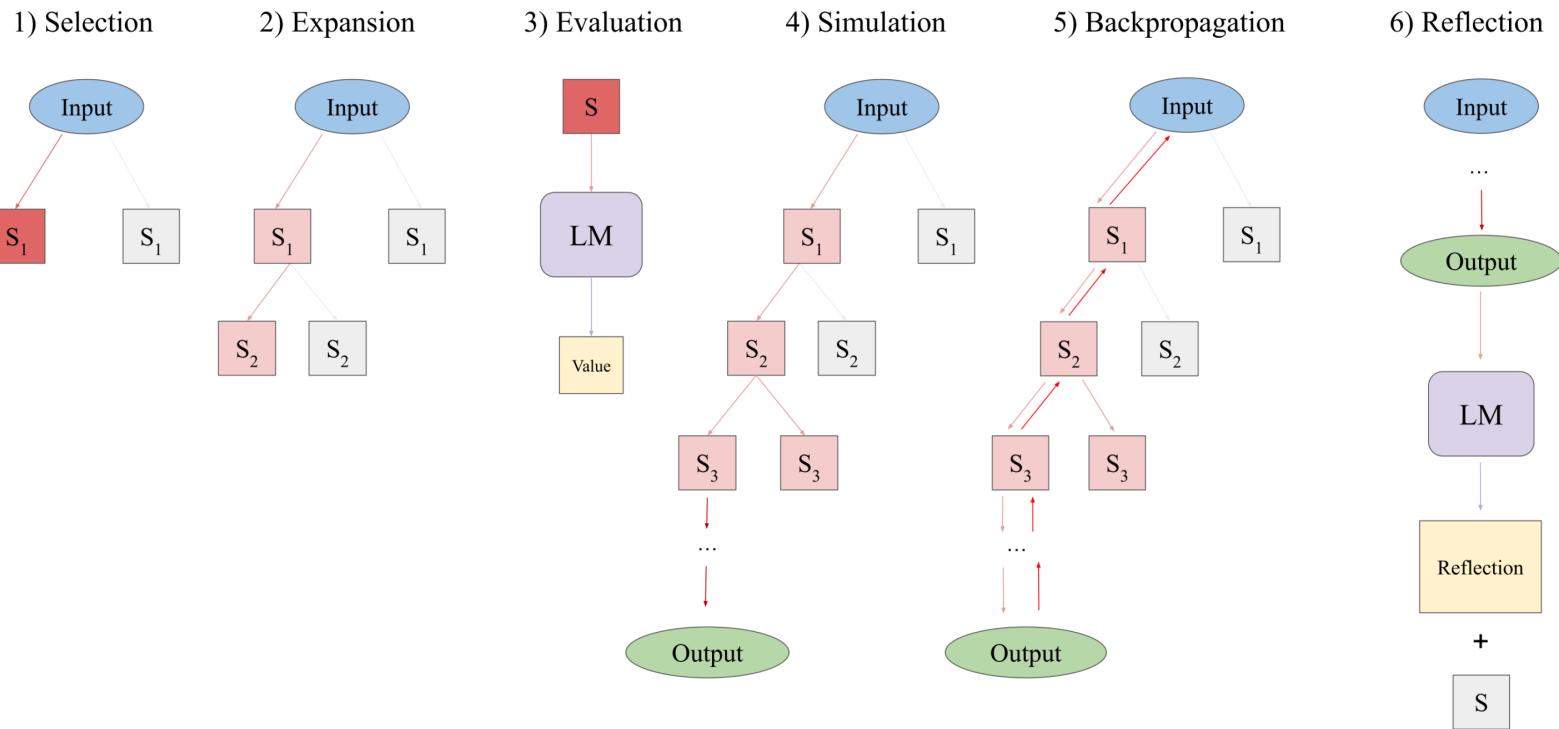
**Act 3:** `Search[Front Row (software)]`

**Obs 3:** Front Row is a discontinued media center software ...

**Thought 4:** Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

**Act 4:** `Finish[keyboard function keys]`





**“ Reflection.** Upon encountering an unsuccessful terminal node, the model is prompted with the trajectory and final reward to provide a verbal self-reflection that summarizes the errors in the reasoning or acting process and proposes superior alternatives. We store both failed trajectories and corresponding reflections in the memory. In subsequent iterations, these are integrated as additional context to the agent. This imparts a semantic gradient signal more useful than a scalar value, enabling the agent to learn from trial and error without the cost of expensive optimization processes such as reinforcement learning.



# LLM Reasoners

A library for advanced reasoning with Large Language Models

**End**