How RL Elicits Long-horizon Reasoning and Search: A Learning Dynamics Perspective

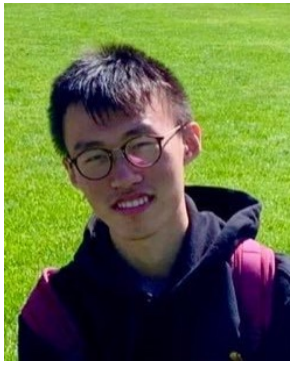
Yuejie Chi

Yale Statistics & Data Science

Seoul, July 2026

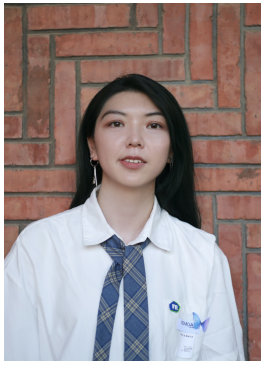
Tong Yang

CMU



Zixin Wen

CMU



Yu Huang

Penn

Aarti Singh

CMU

Yingbin Liang

OSU

Yuxin Chen

Penn

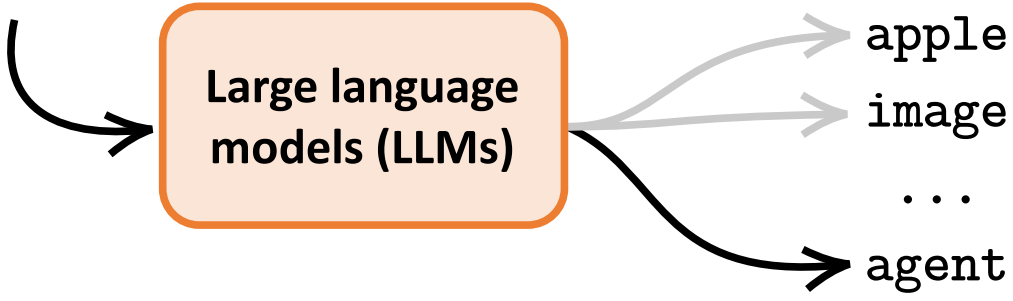
Yuting Wei

Penn

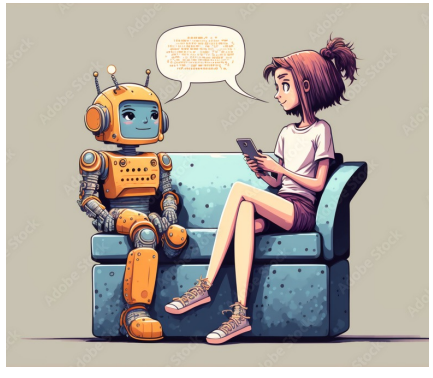
Large language models

The language models predict the next word based on previous words.

```
Prompt: Explain reinforcement learning (RL).

Answer: Reinforcement learning (RL) is a type of
 machine learning where an...
```



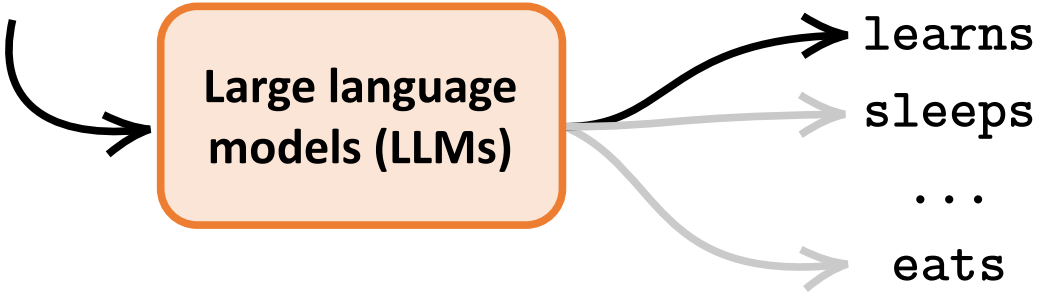
LLMs can do amazing things beyond their original conception.

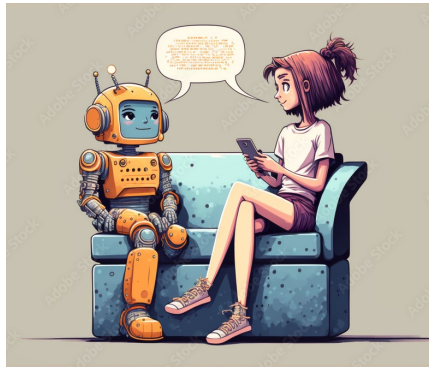


Large language models

The language models predict the next word based on previous words.

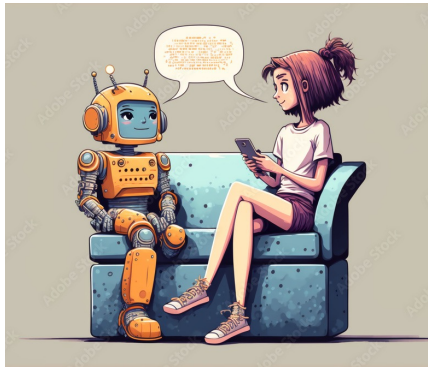
```
Prompt: Explain reinforcement learning (RL).

Answer: Reinforcement learning (RL) is a type of
 machine learning where an agent
```

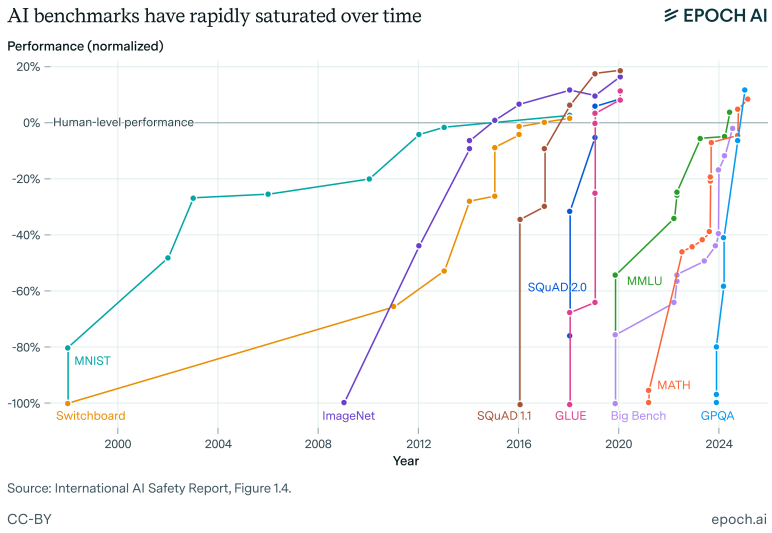


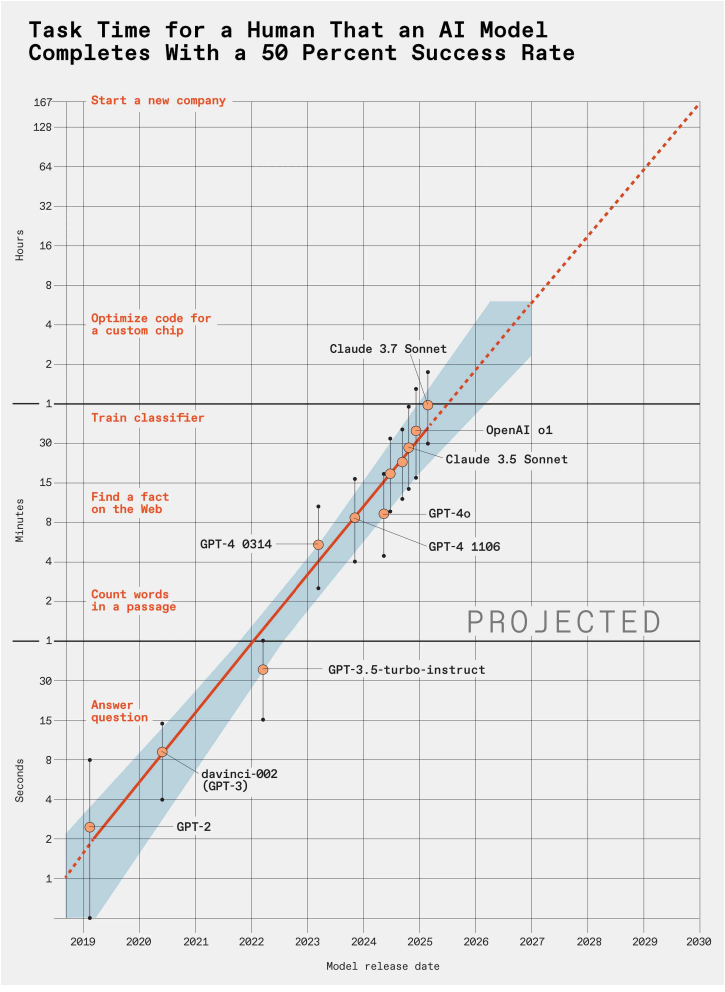


LLMs can do amazing things beyond their original conception.

The rise of large language models





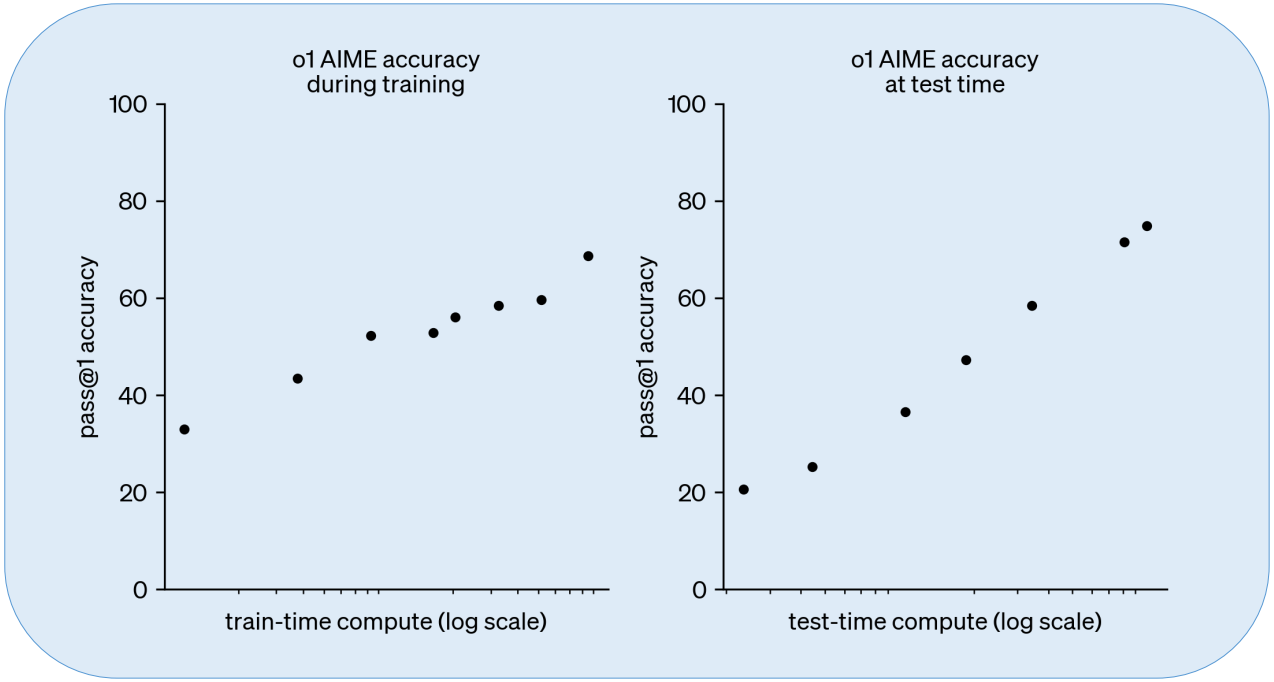


LLM benchmarking shows capabilities doubling every **7 months**.

`https://spectrum.ieee.org/large-language-model-performance`

Reasoning models and test-time scaling

Reasoning models: OpenAI released o1 in 2024, which was trained via reinforcement learning.



(Figure credit: o1)

Test-time scaling: reasoning performance improves with more test-time compute (e.g., thinking longer).

RL with verifiable rewards

RL with verifiable rewards (RLVR): In early 2025, Deepseek released R1, and revealed the training recipe called RLVR.

- The model is only rewarded for accuracy and format.
- No outcome or process neural reward model.

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a + x}} = x$ is equal to

Response: <think>

To solve the equation $\sqrt{a - \sqrt{a + x}} = x$, let's start by squaring both $\cdots$

$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2$.

Rearrange to isolate the inner square root term:

$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$

Wait, wait. Wait. That's an aha moment I can flag here.

Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$

We started with the equation:

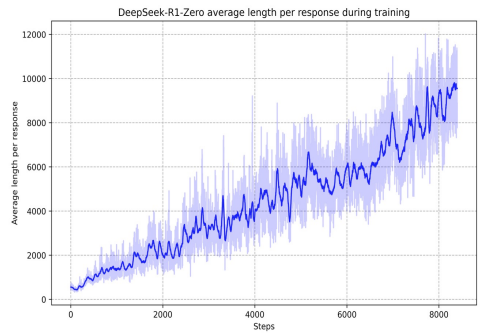
$\sqrt{a - \sqrt{a + x}} = x$

First, let's square both sides:

$a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$

Next, I could square both sides again, treating the equation: $\cdots$

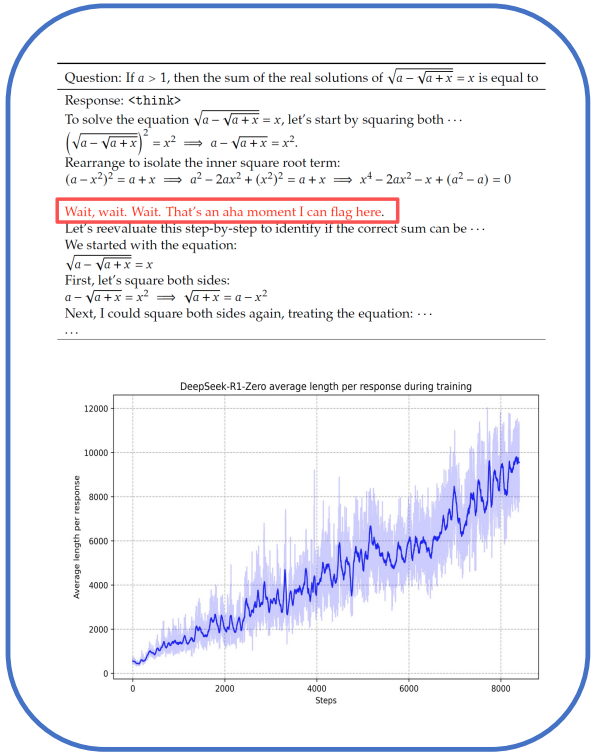
$\cdots$



RL with verifiable rewards

RL with verifiable rewards (RLVR): In early 2025, Deepseek released R1, and revealed the training recipe called RLVR.

- The model is only rewarded for accuracy and format.
- No outcome or process neural reward model.

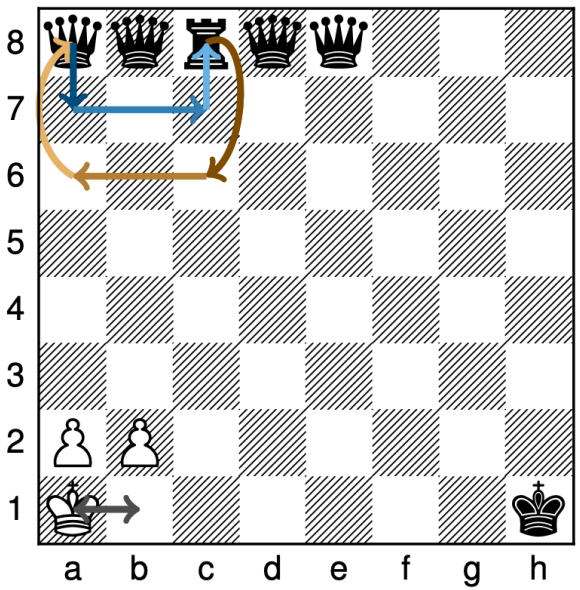


How does RLVR elicit long-horizon reasoning?

State tracking

Given an initial state and a sequence of *actions*, compute the final state.

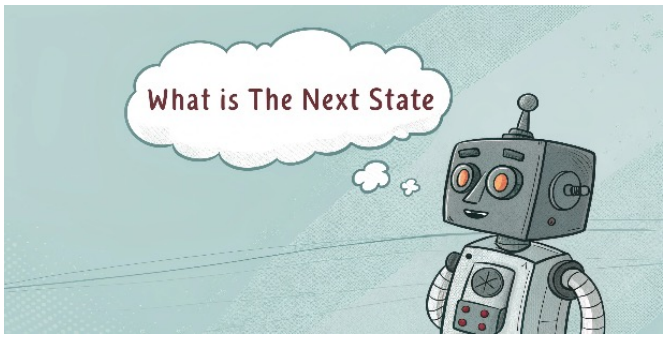
Playing Chess



Images modified from Merrill et al., 2024

Code evaluation

```
x = [0, 0, 1, 0, 0]
x[1], x[3] = x[3], x[1] # Swap 1, 3
```

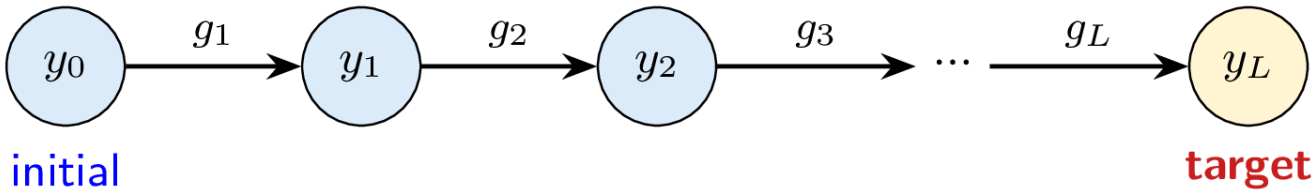


Entities tracking

Alice, Bob, Carl, Dan, and Emma each have a coin. All are dimes except Carl's. Alice and Carl trade coins.

State tracking as compositional reasoning

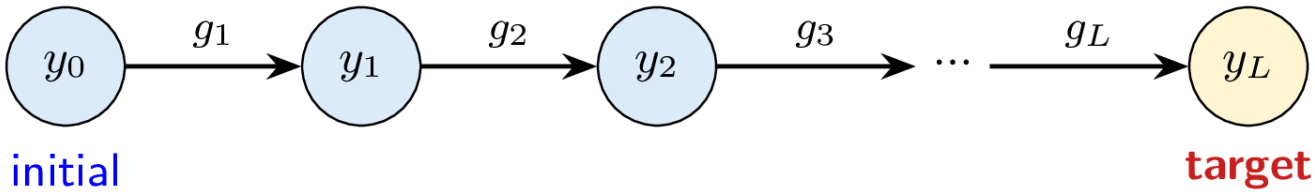
Given a group $\mathcal{G}$ acting on a finite value set $\mathcal{Y}$:



Goal: calculate the last value $y_L = g_L(g_{L-1}(\cdots g_1(y_0)))$ given the sequence of transitions $g_i \in \mathcal{G}$ with an initial value $y_0 \in \mathcal{Y}$.

State tracking as compositional reasoning

Given a group $\mathcal{G}$ acting on a finite value set $\mathcal{Y}$:



Goal: calculate the last value $y_L = g_L(g_{L-1}(\cdots g_1(y_0)))$ given the sequence of transitions $g_i \in \mathcal{G}$ with an initial value $y_0 \in \mathcal{Y}$.

Example: modulo sum

$$y_1 = y_0 \oplus_5 4, \qquad y_2 = y_1 \oplus_5 2, \qquad y_0 = 0$$

where

$$y_2 = y_1 \underbrace{\oplus_5 2}_{g_2} = (y_0 \underbrace{\oplus_5 4}_{g_1}) \oplus_5 2 = (\underbrace{0}_{y_0} \oplus_5 4) \oplus_5 2 \equiv 1 \pmod 5.$$

How does the transformer reason sequentially?

Transformer as autoregressive policy: given (prompt, y_0),

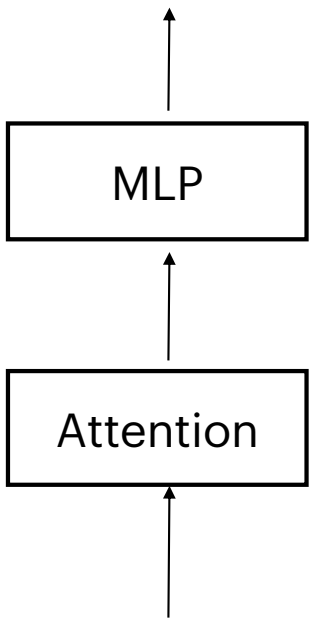
$$\pi_\theta(\widehat{y}^\ell \mid \text{prompt}, y_0) = \prod_{k=1}^{\ell} \pi_\theta(\widehat{y}_k \mid \text{prompt}, y_{k-1}), \qquad \ell = 1, \ldots, L,$$

where $\widehat{y}^\ell = (\widehat{y}_1, \ldots, \widehat{y}_\ell)$.

Model mechanism: TF = MLP (executor) $\circ$ Attention (retrieval)

Recursion of L-step reasoning: at each step ℓ, recall the target $y_\ell = g_\ell(y_{\ell-1})$.

- **Attention** = retriever: find correct g_ℓ from prompt at step ℓ – horizon-variant
- **MLP** = executor: $(g_\ell, y_{\ell-1}) \mapsto g_\ell(y_{\ell-1})$ with near-perfect accuracy – horizon-invariant



Outcome-based RL objective and policy gradient

Outcome-based RL objective:

$$\mathcal{J}_L(\theta) = \mathbb{E}_{\text{prompt},y_0}\Big[\,\mathbb{E}_{\widehat{y}^L\sim\pi_\theta(\cdot|\text{prompt},y_0)}\big[r(\widehat{y}^L \mid \text{prompt}, y_0)\big]\Big],$$

which uses only the terminal reward:

$$r(\widehat{y}^L \mid \text{prompt}, y_0) \triangleq \mathbf{1}\big\{\widehat{y}_L = y_L\big\}.$$

Outcome-based RL objective and policy gradient

Outcome-based RL objective:

$$\mathcal{J}_L(\theta) = \mathbb{E}_{\text{prompt},y_0}\Big[\mathbb{E}_{\widehat{y}^L\sim\pi_\theta(\cdot|\text{prompt},y_0)}\big[r(\widehat{y}^L \mid \text{prompt}, y_0)\big]\Big],$$

which uses only the terminal reward:

$$r(\widehat{y}^L \mid \text{prompt}, y_0) \triangleq \mathbf{1}\{\widehat{y}_L = y_L\}.$$

Length-normalized policy gradient via REINFORCE:

$$\theta^{(t+1)} = \theta^{(t)} + \eta\nabla_\theta\widetilde{\mathcal{J}}_L(\theta)$$

$$\nabla_\theta\widetilde{\mathcal{J}}_L(\theta) = \frac{1}{L}\mathbb{E}_{\text{prompt},y_0}\Big[r(\widehat{y}^L \mid \text{prompt}, y_0)\nabla\log\pi_\theta(\widehat{y}^L \mid \text{prompt}, y_0)\Big],$$

where $\widetilde{\mathcal{J}}_L(\theta) = \frac{1}{L}\mathcal{J}_L(\theta)$, and η is the learning rate.

Pretrained atomic skills

Pretrained MLP: we assume MLP is pretrained to capture the atomic skills of group action

$$(g, y) \mapsto g(y)$$

with weight W fixed during RL training.

Pretrained atomic skills

Pretrained MLP: we assume MLP is pretrained to capture the atomic skills of group action

$$(g, y) \mapsto g(y)$$

with weight W fixed during RL training.

When MLP receives $\frac{1}{2}(g+y)$, then for $j = \tau(g(y))$

$$\text{softmax}\Big(\text{MLP}_W\Big(\frac{1}{2}(g+y)\Big)\Big)_j = 1 - \frac{1}{\text{poly}(d)}.$$

Here, $\tau : \mathcal{Y} \to [d]$ index the states for prediction.

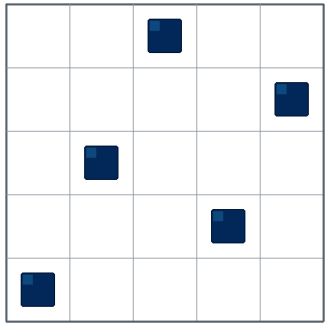
- Our prior work [Huang et al, 2025] provides guarantees for learning such atomic structures via overparameterized ReLU network, with activation level

$$B = C_{\text{mlp}} \log d,$$

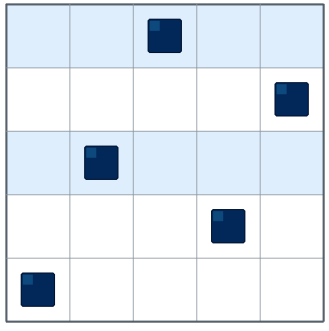
where C_{mlp} is a constant determining the **skill sharpness**.

Attention training

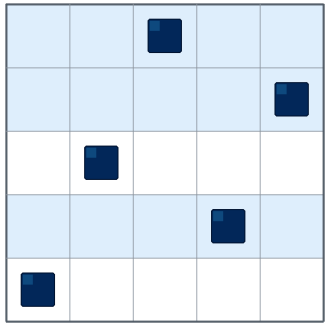
Attention mechanism reused across lengths: the ground truth retrieval pattern is a permutation at the longest possible reasoning length, measuring the *positional alignment*.



ground truth



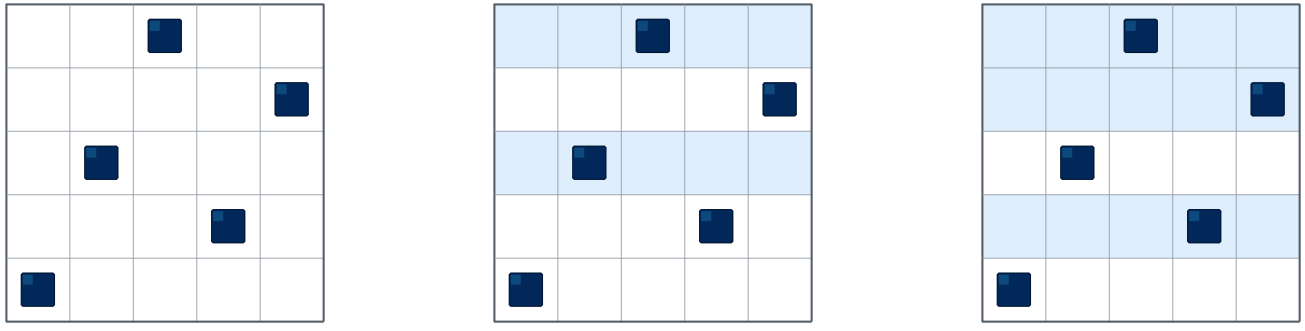
length-2



length-3

Attention training

Attention mechanism reused across lengths: the ground truth retrieval pattern is a permutation at the longest possible reasoning length, measuring the *positional alignment*.



ground truth length-2 length-3

Uniform attention at initialization: We train an attention matrix Q to learn the permutation pattern with

$$Q^{(0)} = 0 \qquad \text{(uniform attention)}.$$

Training dynamic tracks how the attention pattern concentrates throughout training.

Horizon barrier of compositional reasoning

Our first result reveals that RLVR is efficient for **short-horizon** reasoning, but suffers from **flat landscape** beyond a critical horizon.

Theorem (Horizon Barrier)

Assume $\mathcal{G}$ is non-abelian and simply transitive, then

- **Success of RL within short horizon $L < C_{\mathsf{mlp}}$:** *for large enough iterations $T = \widetilde{O}(\frac{\mathrm{poly}(d)}{\eta})$ with $\eta = \frac{1}{\mathrm{poly}(d)}$, the reward*

$$\mathcal{J}_L^{(T)} \geq 1 - \frac{1}{\mathrm{poly}(d)}.$$

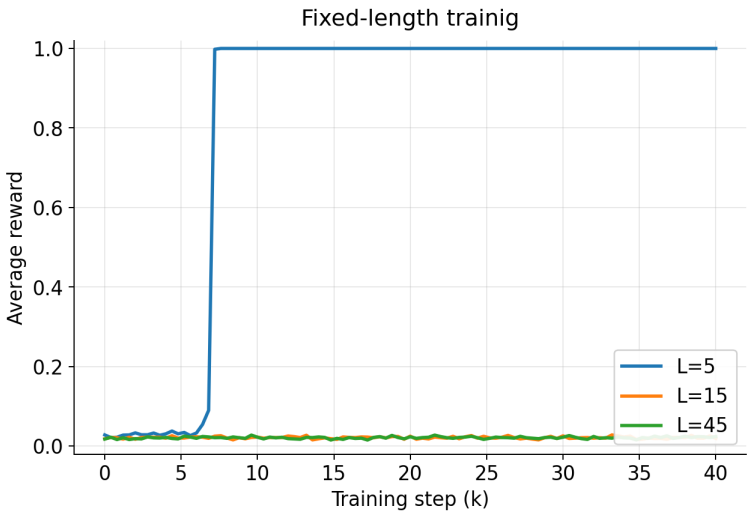
- **Exponentially flat region when $L \geq 2C_{\mathsf{mlp}}$:** *the gradients of $\mathcal{J}_L$ and rewards $\mathcal{J}_L^{(t)}$ satisfy*

$$\left|\left\langle \left[\nabla_Q \mathcal{J}_L^{(t)}\right] x, x' \right\rangle\right| \leq d^{-\Omega(L)}, \qquad \mathcal{J}_L^{(t)} = \frac{1}{d}(1 + o(1)),$$

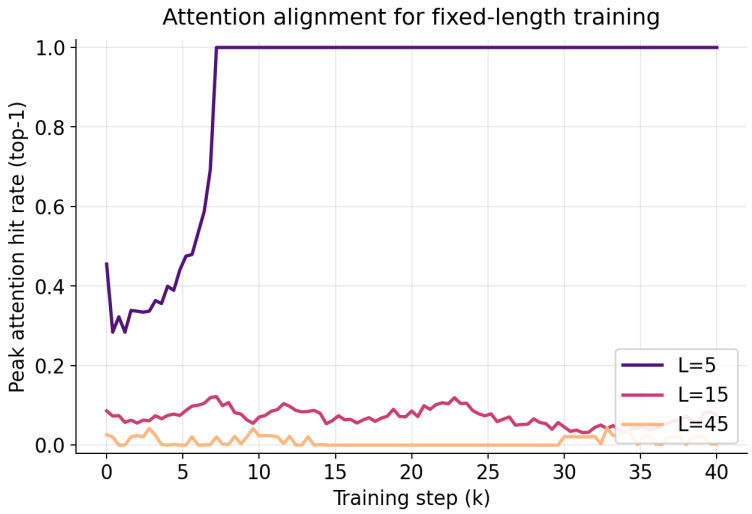
whenever $\max_{x,x' \in \mathcal{X}} \langle Q^{(t)} x, x' \rangle \leq 0.01$.

Single-length training

Experiment setup: we assume a cyclic group $\mathbb{Z}_{96}$ for reasoning length $L = 5, 15, 45$.



Average reward



Attention concentration

RL rapidly learns **short-horizon compositions**, whereas longer horizons exhibit a **near-flat reward plateau.**

Mixed-difficulty RLVR

The previous result showed that RL cannot operate beyond a critical horizon, even when the reward is non-zero by random guesses.

What if we **mix different horizons** together?

Mixed-difficulty distribution: we uniformly mix the horizons

$$\mathcal{L}_R = \{L_1, L_2, \ldots, L_K\}, \quad L_k = \min\{\lceil RL_{k-1} \rceil, L_{\max}\}.$$

Here, $R = L_{k+1}/L_k$ is the difficulty ratio, which controls how **smooth** the curriculum structure in the distribution.

Mixed-difficulty training objective:

$$\mathcal{J}_{\mathrm{mix,R}} = \mathbb{E}_{L\sim\mathsf{Unif}(\mathcal{L}_R)}[\mathcal{J}_L(\theta)]$$

$y_0 \xrightarrow{g_1} \cdots \xrightarrow{g_{L_1}} y_{L_1}$

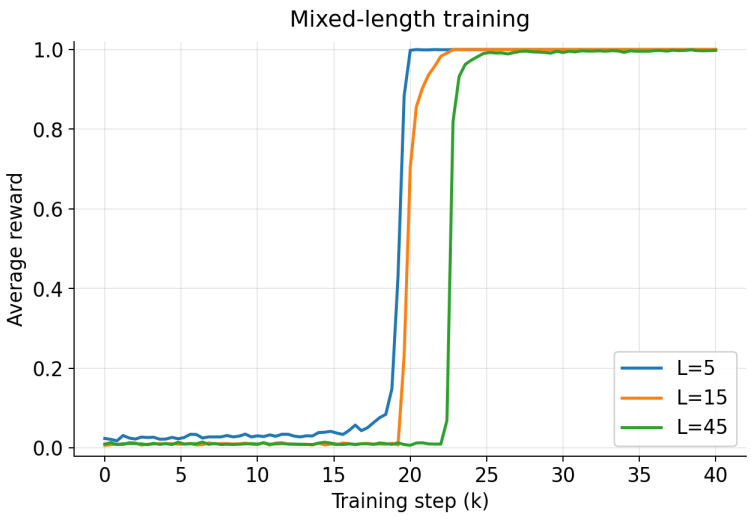
$y_0 \xrightarrow{g_1} y_1 \xrightarrow{g_2} y_2 \xrightarrow{g_3} \cdots \xrightarrow{g_{L_2}} y_{L_2}$

$\vdots$

$y_0 \xrightarrow{g_1} y_1 \xrightarrow{g_2} y_2 \xrightarrow{g_3} y_3 \xrightarrow{g_4} y_4 \xrightarrow{g_5} \cdots \xrightarrow{g_{L_K}} y_{L_K}$

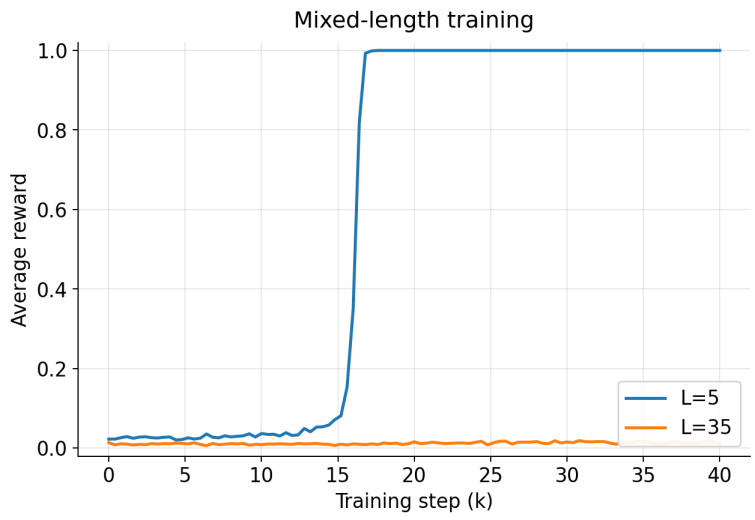
Learning behavior at different difficulty ratios

Relay



$R = 3$: a smoother difficulty spectrum facilitates relay dynamics across horizons.

Grokking



$R = 7$: a larger difficulty ratio yields a prolonged plateau at longer horizons.

The shared reasoning pattern generalizes across horizons through an implicit curriculum provided by the data mix.

RLVR dynamic under mixed difficulty

Stopping times: define the following stopping times for

- *Visible return*: $T_{\mathrm{vis},L} := \min_t\{\mathcal{J}_L^{(t)} \geq 0.01\}$;
- *Mastery*: $T_{\mathrm{mas},L} := \min_t\{\mathcal{J}_L^{(t)} \geq 0.99\}$.

Theorem (Huang et al., 2026)

Suppose $L_1 \leq C_{\mathsf{mlp}}$. Assume $\mathcal{G}$ is non-abelian and simply transitive,

- **Grokking (Long plateaus when $R \geq \omega(1)$):** *before the next horizon L_{k+1} sees visible returns, there is a long plateau:*

$$T_{\mathrm{vis},k+1} - T_{\mathrm{mas},k} \gtrsim \frac{L_{\max}}{\eta} \cdot d^{C_{\mathsf{mlp}}-1}.$$

- **Relay (Short plateaus when $R \leq O(1)$):** *before L_{k+1} sees visible returns, the plateau period is short:*

$$T_{\mathrm{vis},k+1} - T_{\mathrm{mas},k} \lesssim \frac{L_{\max}}{\eta} \cdot d^{C_{\mathsf{mlp}}\left(1-\frac{C_{\mathsf{mlp}}}{C_{\mathsf{mlp}}+R}\right)-1}.$$

- **(Phase transitions)** *Once horizon L_{k+1} reaches visible return, it arrives at mastery time quickly: $T_{\mathrm{mas},k} - T_{\mathrm{vis},k} \lesssim \frac{L_{\max}}{\eta} \cdot L_{k+1}$.*

Gradient characterization

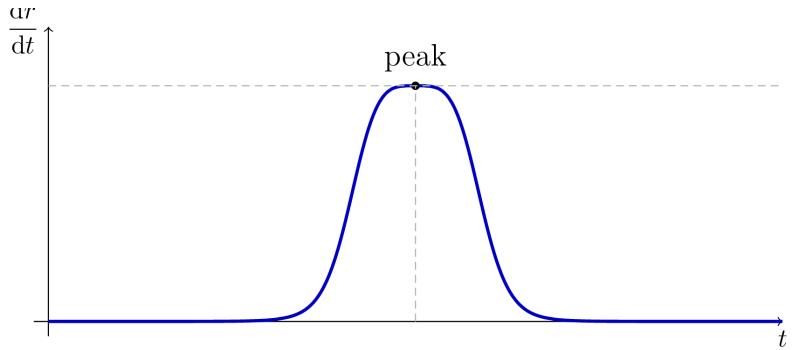
Step-wise probability margin: how much target dominates

$$\Delta_L :\approx p_{L,\text{target}} - p_{L,\text{non-target}}.$$

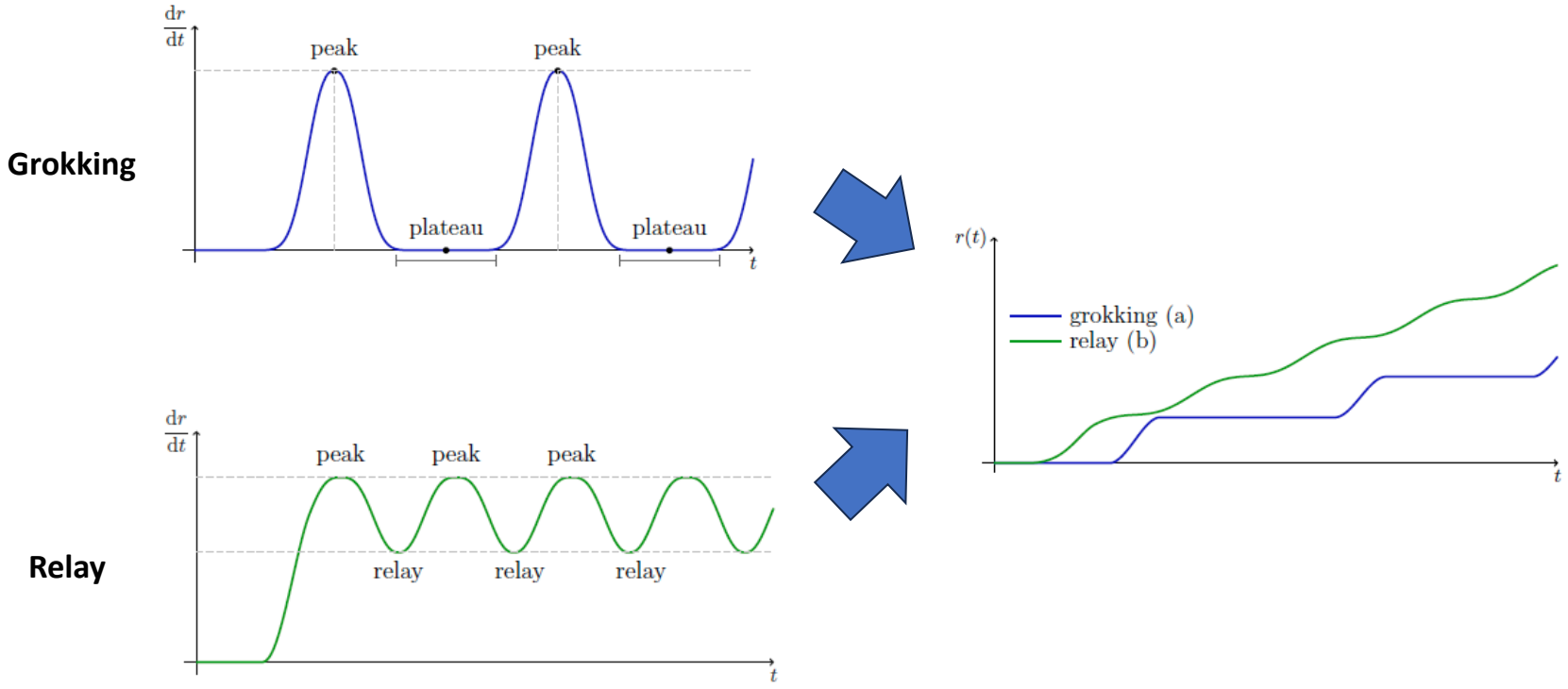
A key policy gradient characterization for aggregated attention q:

$$\nabla_q \widetilde{\mathcal{J}}_L \propto \underbrace{(\Delta_L)^L}_{\text{success over } L \text{ steps}} \cdot \underbrace{(1-\Delta_L)}_{\text{room to improve}}$$

- Δ_L small $\Rightarrow (\Delta_L)^L$ exponentially suppresses gradient (cold start)
- $\Delta_L \approx 1 \Rightarrow 1-\Delta_L$ vanishes (saturated)



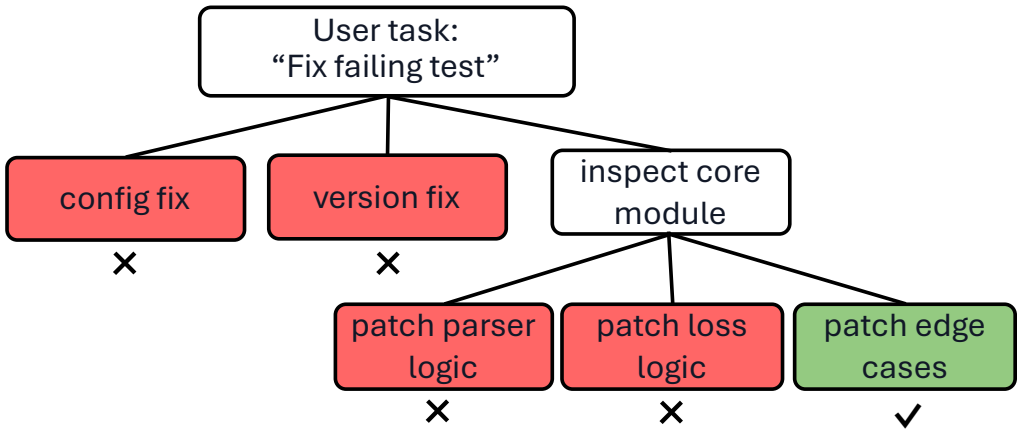
Cartoon illustration of RLVR dynamics



- **Grokking**: shorter-horizon saturates long before longer-horizon escapes random guessing.
- **Relay**: at the **edge of competence**, shorter-horizon still improves while longer-horizon already escapes randomness.

Search in agentic and reasoning tasks

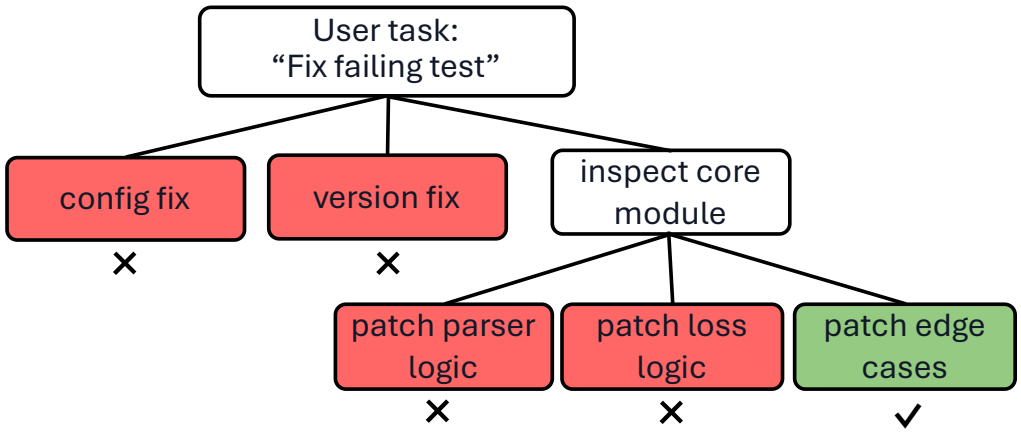
Search is a crucial algorithmic primitive in agentic and reasoning tasks.



- The structure of the search space and target of search are often a priori unknown to the agent.
- Agents need to perform backtracking to recover from failures.

Search in agentic and reasoning tasks

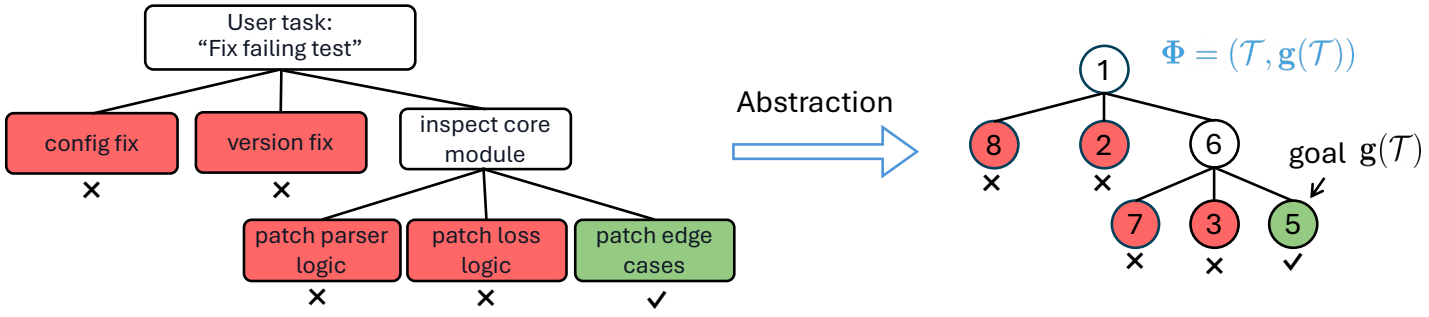
Search is a crucial algorithmic primitive in agentic and reasoning tasks.



- The structure of the search space and target of search are often a priori unknown to the agent.
- Agents need to perform backtracking to recover from failures.

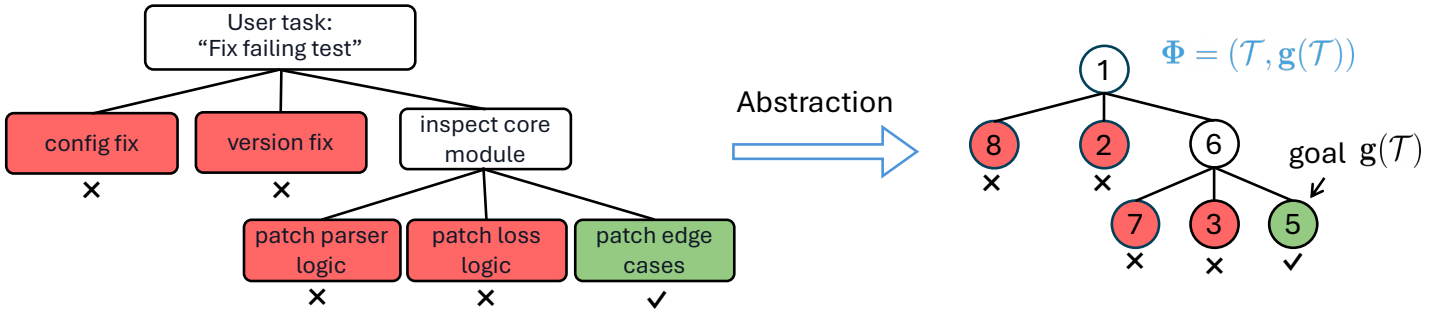
How does RL enable agents to search?

Abstraction as tree search



Goal: transformers act on the tree to find the goal node.

Abstraction as tree search

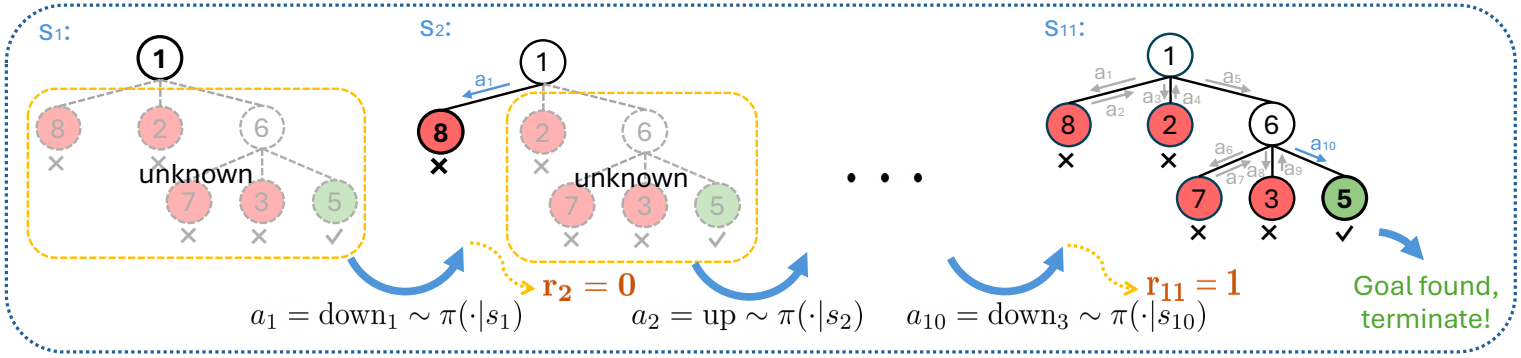


Goal: transformers act on the tree to find the goal node.

How does this different from supervised fine-tuning (SFT)?

- In SFT, the complete tree and entire root-to-goal path is used to supervise a path-finding mechanism (Yang et al., 2025).
- In RL, the tree is *revealed* as the agent explores it through trial-and-error, with feedback about if it is a leaf or goal node.

RL environment



Episodic setting:

- A new tree is generated at the beginning of each episode.
- Transformer as an autoregressive policy over the action space:

$$\pi_\theta(\cdot|\text{context}) \sim \Delta(\mathcal{A}), \qquad \text{where} \quad \mathcal{A} = \{\texttt{up}, \texttt{down}_k\}$$

  it can visit children nodes, backtrack to parent nodes, etc...
- Environment reveals the next node in the tree, and its nodal properties (leaf/non-leaf).

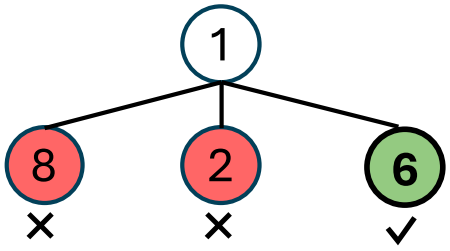
Can RL learn this search mechanism?

Yes!

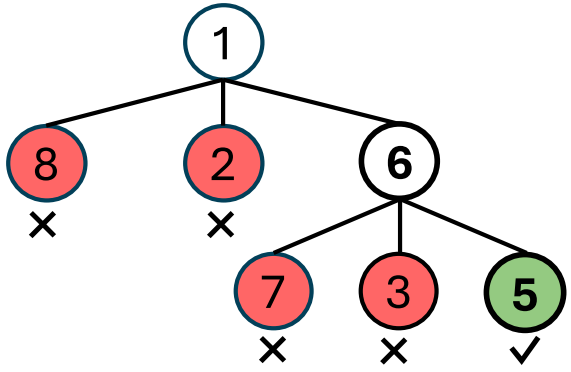
Theorem (Yang et al., 2026)

A one-layer two-head transformer learns to implement **depth-first search (DFS)** *via curriculum training on depth-1 and depth-2 trees sequentially. Furthermore, it generalizes directly to deeper trees.*

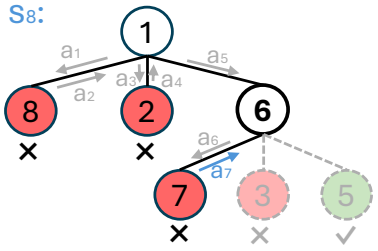
depth-1 trees



depth-2 trees



Attention head specialization

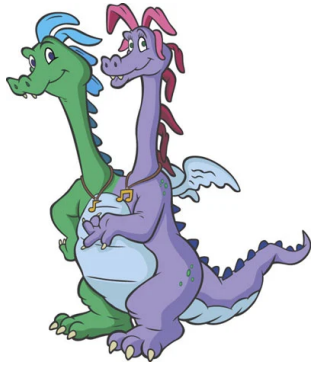


$$E^{(8)} = \begin{array}{c} \\ n \\ a \\ n' \\ c \end{array} \overset{\text{query}}{\begin{pmatrix} 0 & 1 & 8 & 1 & 2 & 1 & 6 & 7 \\ 0 & \text{down}_1 & \text{up} & \text{down}_2 & \text{up} & \text{down}_3 & \text{down}_1 & \text{up} \\ 1 & 8 & 1 & 2 & 1 & 6 & 7 & 6 \\ 0 & \times & 0 & \times & 0 & 0 & \times & 0 \end{pmatrix}}$$

$$a_8 = \text{down}_2 \sim \pi_\theta(E^{(8)}) \approx (0, 0.5, 0.5, 0)^\top$$

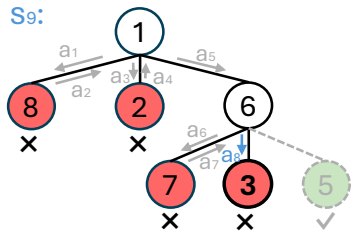
Task specialization of attention heads:

- One head retrieves action history and suppresses tried children from being selected again;
- One head uses label history to promote action up at non-goal leaves and suppress up at internal nodes.



The two heads coordinate to implement depth-first search.

Attention head specialization

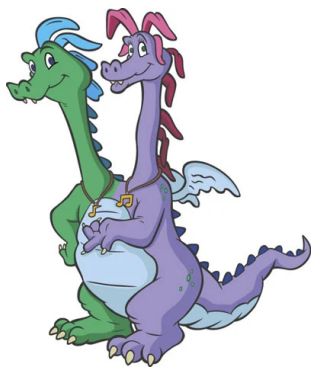


$$E^{(9)} = \begin{array}{c} \\ n \\ a \\ n' \\ c \end{array} \begin{pmatrix} 0 & 1 & 8 & 1 & 2 & 1 & 6 & 7 & 6 \\ 0 & \text{down}_1 & \text{up} & \text{down}_2 & \text{up} & \text{down}_3 & \text{down}_1 & \text{up} & \text{down}_2 \\ 1 & 8 & 1 & 2 & 1 & 6 & 7 & 6 & 3 \\ 0 & \times & 0 & \times & 0 & 0 & \times & 0 & \times \end{pmatrix} \begin{array}{c} \\ \\ \\ \\ \text{query} \end{array}$$

$$a_9 = \text{up} \sim \pi_\theta(E^{(9)}) \approx (0, 0, 0, 1)^\top$$

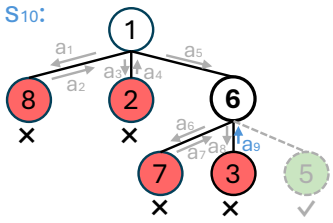
Task specialization of attention heads:

- One head retrieves action history and suppresses tried children from being selected again;
- One head uses label history to promote action up at non-goal leaves and suppress up at internal nodes.



The two heads coordinate to implement depth-first search.

Attention head specialization



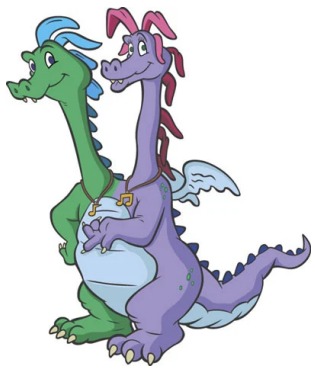
Goal found, terminate!

$$E^{(10)} = \begin{array}{c} \\ n \\ a \\ n' \\ c \end{array} \begin{array}{c} \text{query} \\ \begin{pmatrix} 0 & 1 & 8 & 1 & 2 & 1 & 6 & 7 & 6 & 3 \\ 0 & \text{down}_1 & \text{up} & \text{down}_2 & \text{up} & \text{down}_3 & \text{down}_1 & \text{up} & \text{down}_2 & \text{up} \\ 1 & 8 & 1 & 2 & 1 & 6 & 7 & 6 & 3 & 6 \\ 0 & \times & 0 & \times & 0 & 0 & \times & 0 & \times & 0 \end{pmatrix} \end{array}$$

$$a_{10} = \text{down}_3 \sim \pi_\theta(E^{(10)}) \approx (0, 0, 1, 0)^\top$$

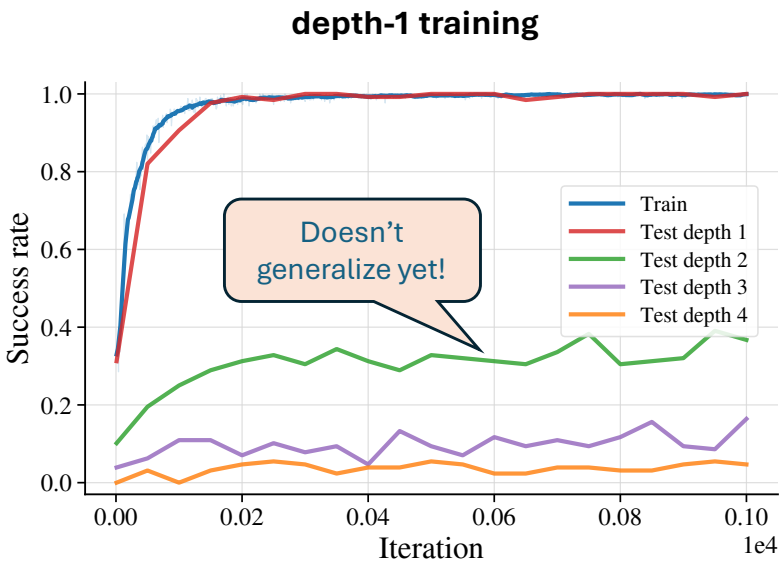
Task specialization of attention heads:

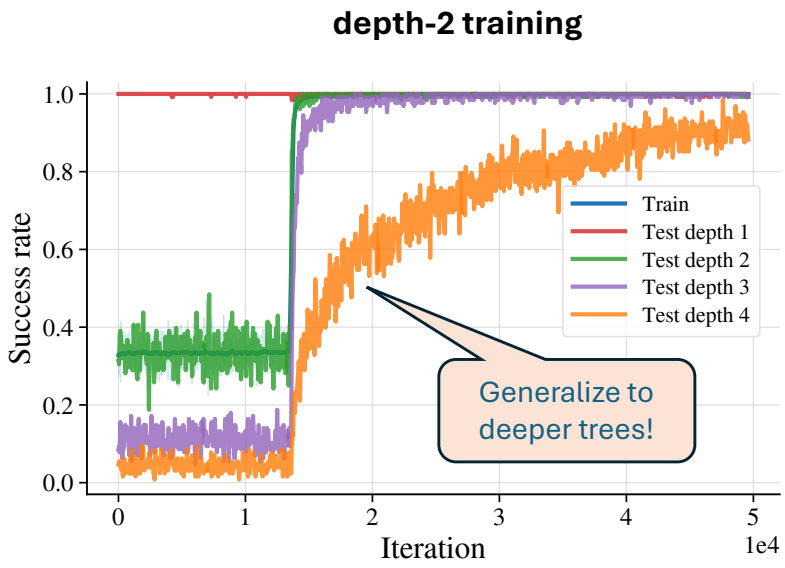
- One head retrieves action history and suppresses tried children from being selected again;
- One head uses label history to promote action up at non-goal leaves and suppress up at internal nodes.



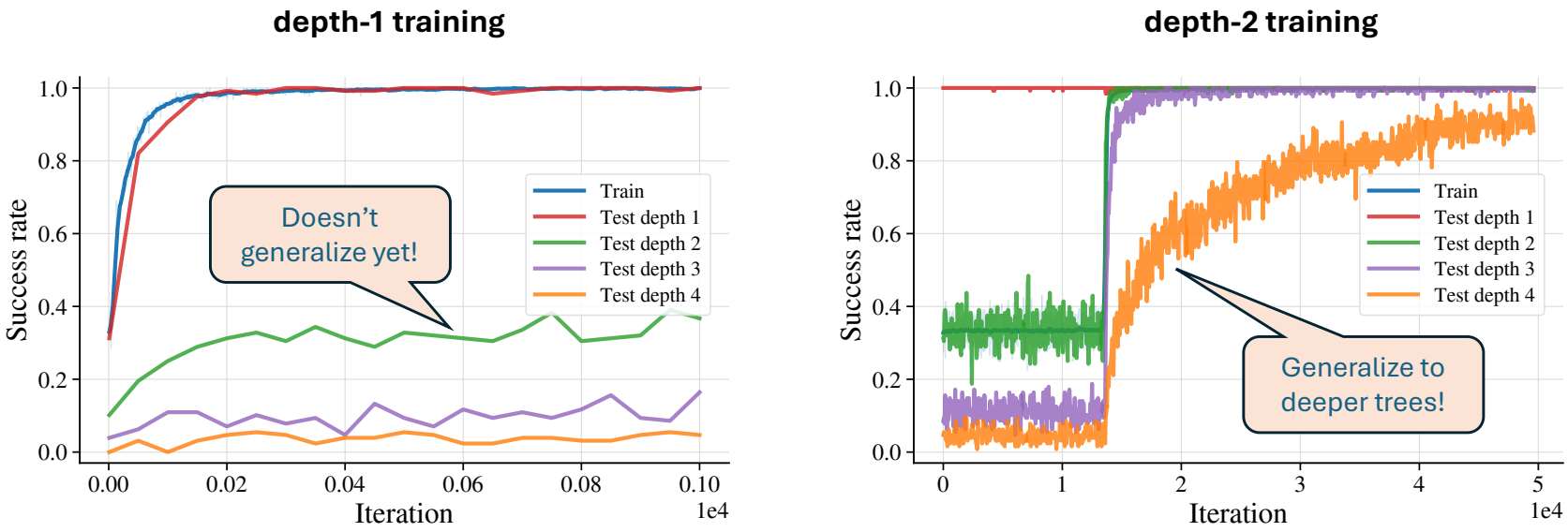
The two heads coordinate to implement depth-first search.

Numerical illustration





Numerical illustration

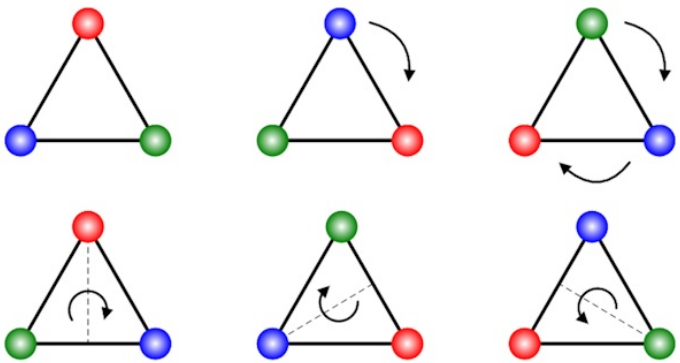


RL trains transformers to perform the depth-first search algorithm.

Summary

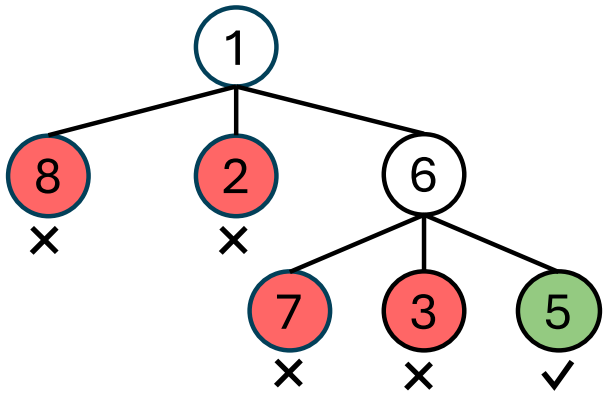
RL enables long-horizon reasoning and search in transformers through curriculum learning.

state tracking
w/ attention and MLP



long-horizon reasoning
(Huang et al., 2026)

tree search
w/ multi-head attention



Learn to search
(Yang et al., 2026)

Thanks!

- On the Emergence of Implicit Curriculum in RLVR Learning Dynamics, arXiv:2602.14872. ICML 2026.
- Agentic Transformers Provably Learn to Search via Reinforcement Learning, arXiv:2606.00183.

https://yuejiechi.github.io