Let Me Think! A Long Chain-of-Thought Can Be Worth Exponentially Many Short Ones

Parsa Mirtaheri*, Ezra Edelman*, Samy Jelassi, Eran Malach, Enric Boix-Adserà

UC San Diego

Penn University of Pennsylvania

HARVARD UNIVERSITY

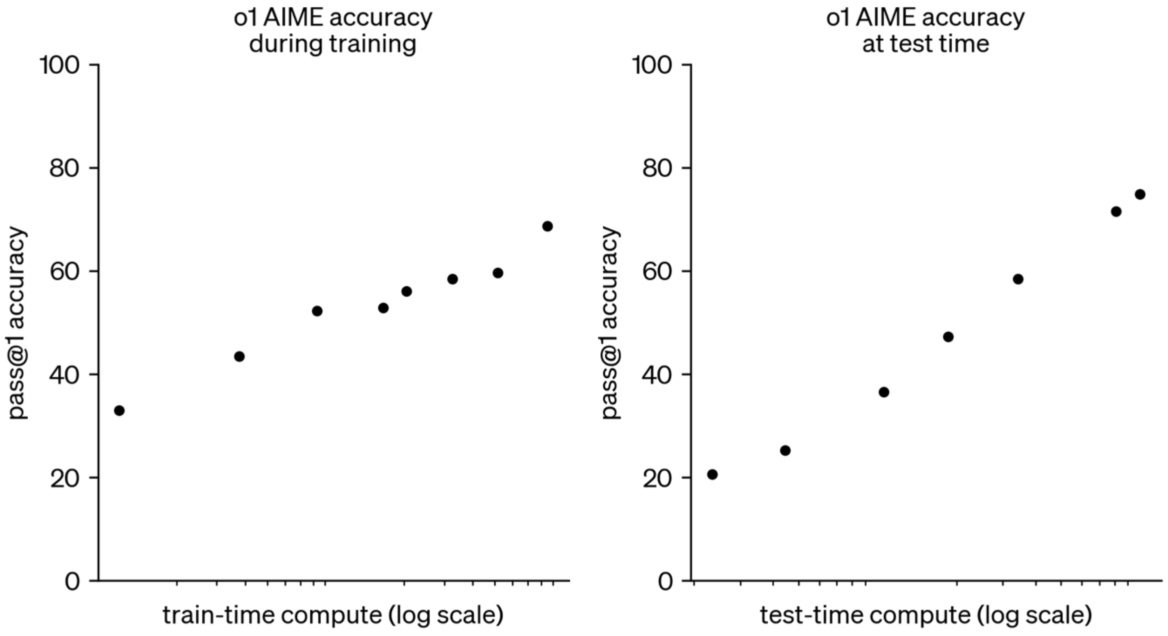
NEURAL INFORMATION PROCESSING SYSTEMS

* Equal contribution

Nov 24, 2025

Inference-Time Scaling

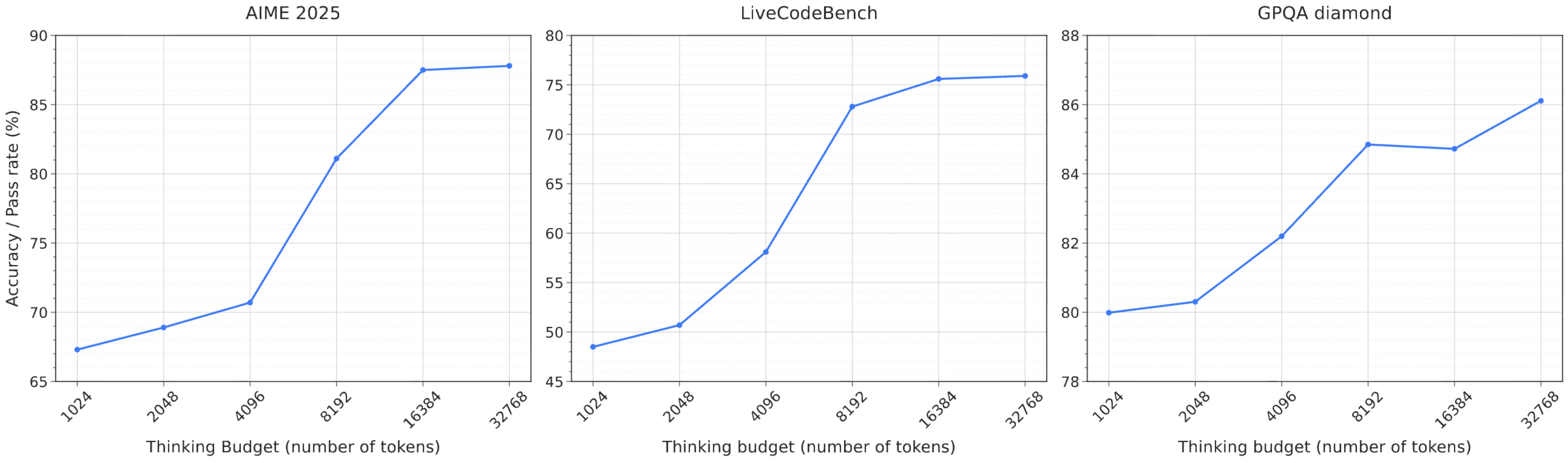
- Inference-time computation has emerged as a promising scaling axis for improving large language model reasoning.



OpenAI. Learning to reason with llms, September 2024.

Inference-Time Scaling

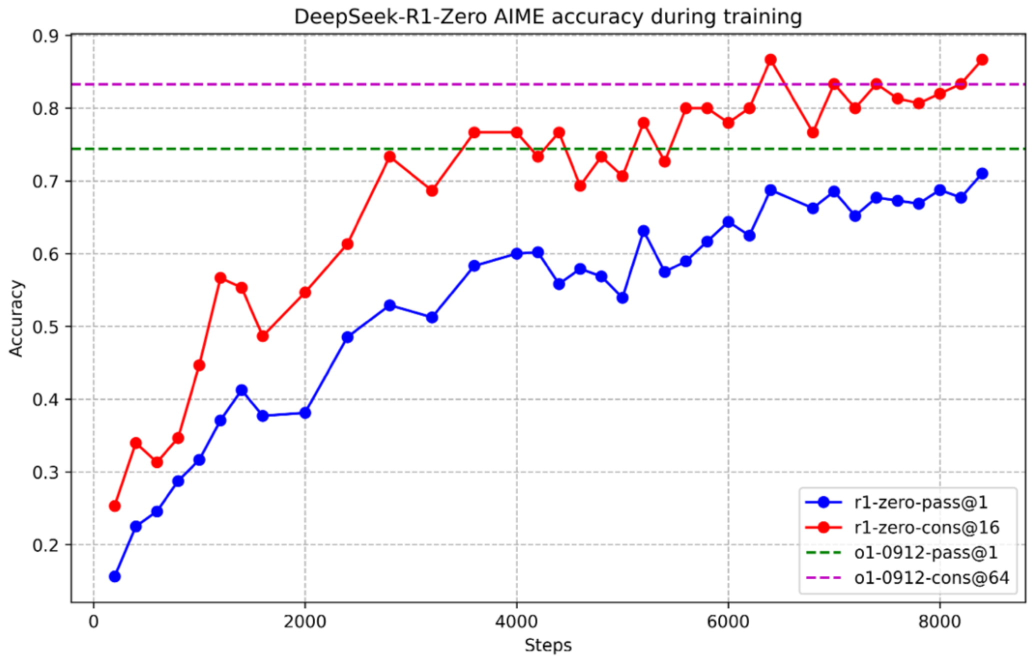
- Inference-time computation has emerged as a promising scaling axis for improving large language model reasoning.

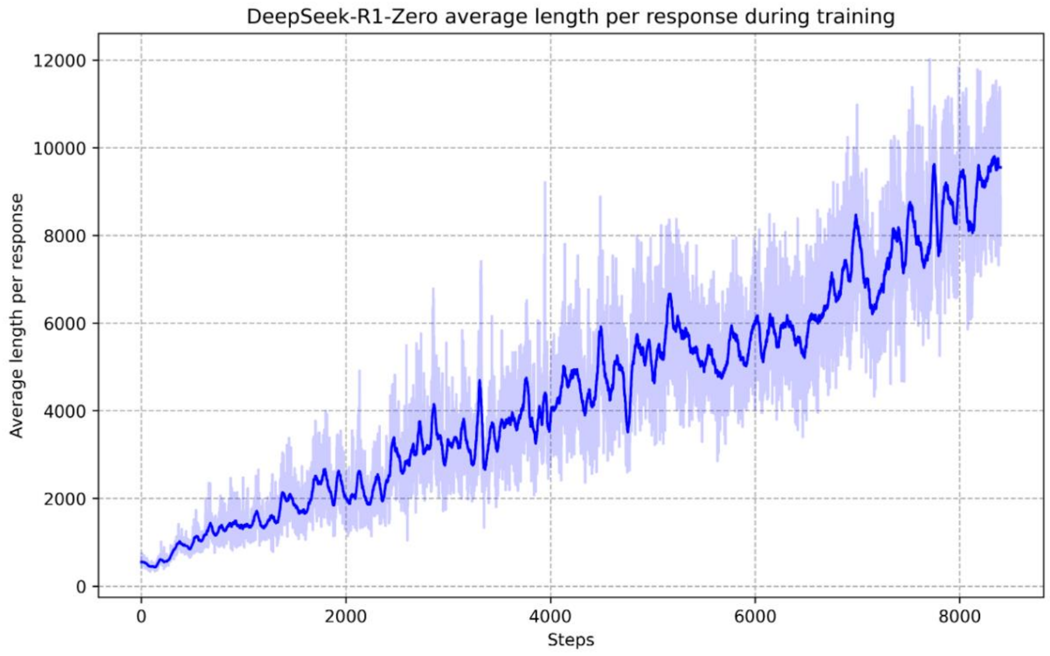


Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities., July 2025.

Inference-Time Scaling

- Inference-time computation has emerged as a promising scaling axis for improving large language model reasoning.

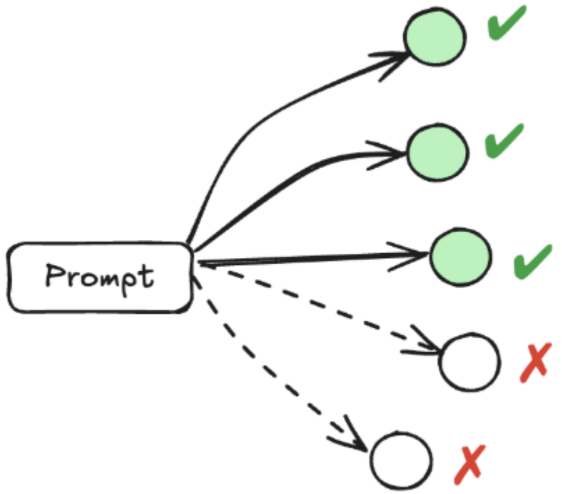


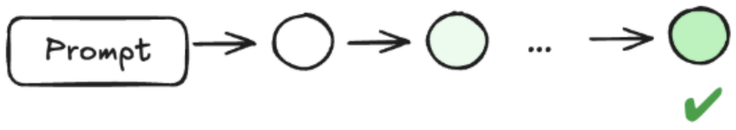


DeepSeek-AI. DeepSeek-R1: incentivizing reasoning capability in LLMs via reinforcement learning, January 2025

Sequential vs Parallel Scaling

- There are various ways of utilizing test time resources for improving reasoning.
- Two main approaches are **parallel** and **sequential** scaling.

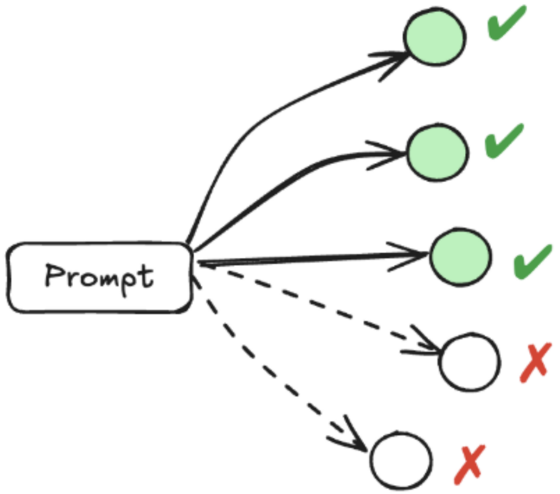


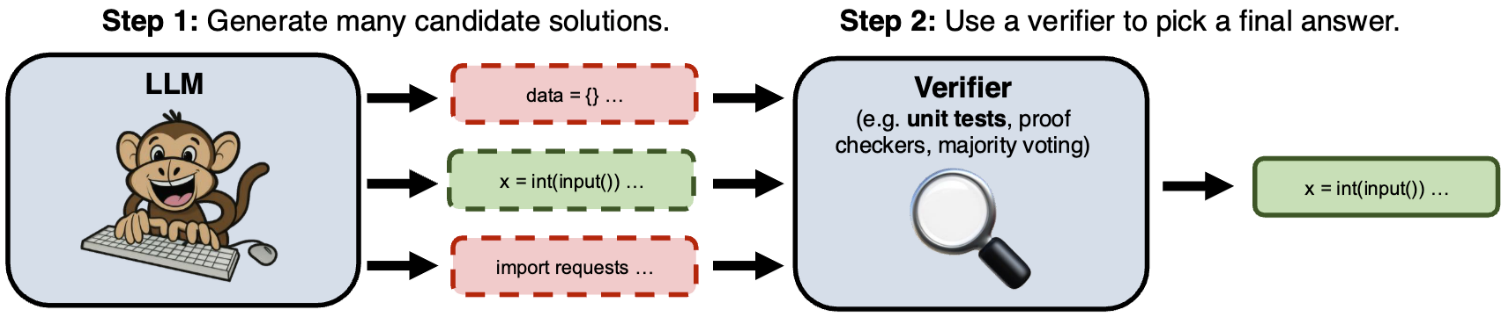


Snell, et al. "Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning." 2025.
Weng, Lilian. "Why We Think". Lil'Log (2025). https://lilianweng.github.io/posts/2025-05-01-thinking/

Sequential vs Parallel Scaling

- There are various ways of utilizing test time resources for improving reasoning.
- Two main approaches are **parallel** and **sequential** scaling.

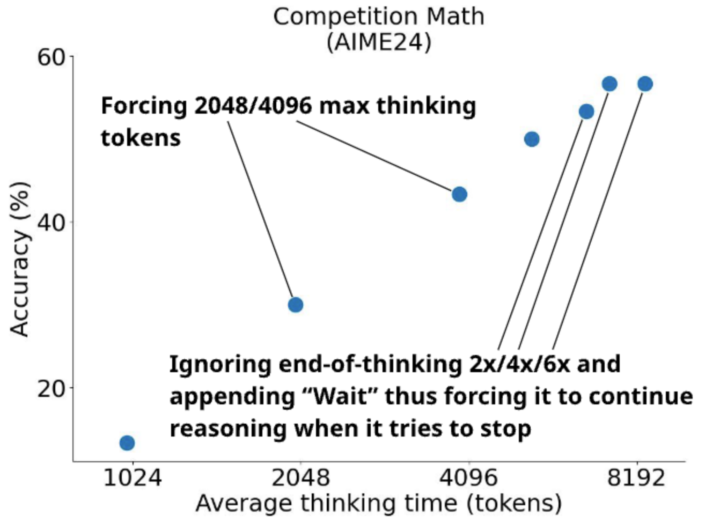


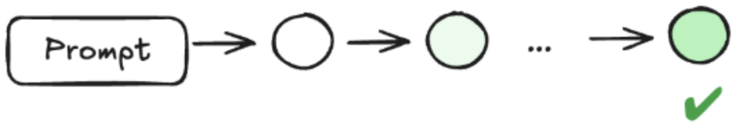


Snell, et al. "Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning." 2025.
Weng, Lilian. "Why We Think". Lil'Log (2025). https://lilianweng.github.io/posts/2025-05-01-thinking/
Brown, et al. "Large Language Monkeys: Scaling Inference Compute with Repeated Sampling", 2024

Sequential vs Parallel Scaling

- There are various ways of utilizing test time resources for improving reasoning.
- Two main approaches are **parallel** and **sequential** scaling.





Snell, et al. "Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning." 2025.
Weng, Lilian. "Why We Think". Lil'Log (2025). https://lilianweng.github.io/posts/2025-05-01-thinking/
Muennighoff, et al. "s1: Simple test-time scaling." (2025).

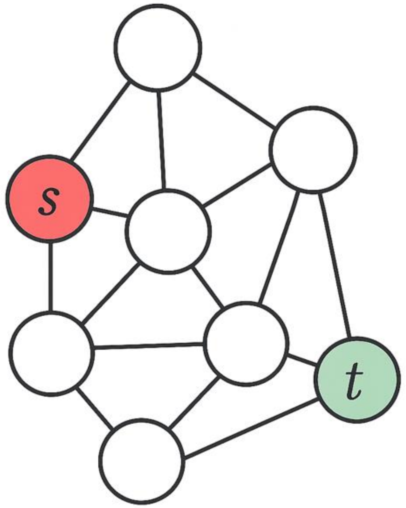
Sequential vs Parallel Scaling

- There are various ways of utilizing test time resources for improving reasoning.
- Two main approaches are **parallel** and **sequential** scaling.

Can we quantify the trade-off between sequential and parallel scaling for reasoning problems?

Graph Connectivity Tasks

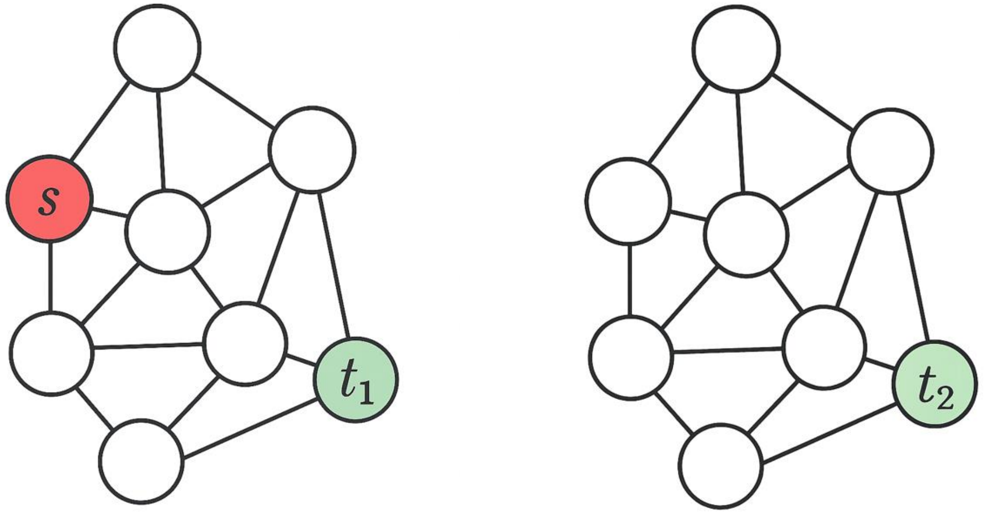
- Our reasoning task is a symmetric variant of the graph connectivity task.
- **(s, t)-Connectivity:** Are nodes s and t connected in a given graph G?



Is s connected to t?

Graph Connectivity Tasks

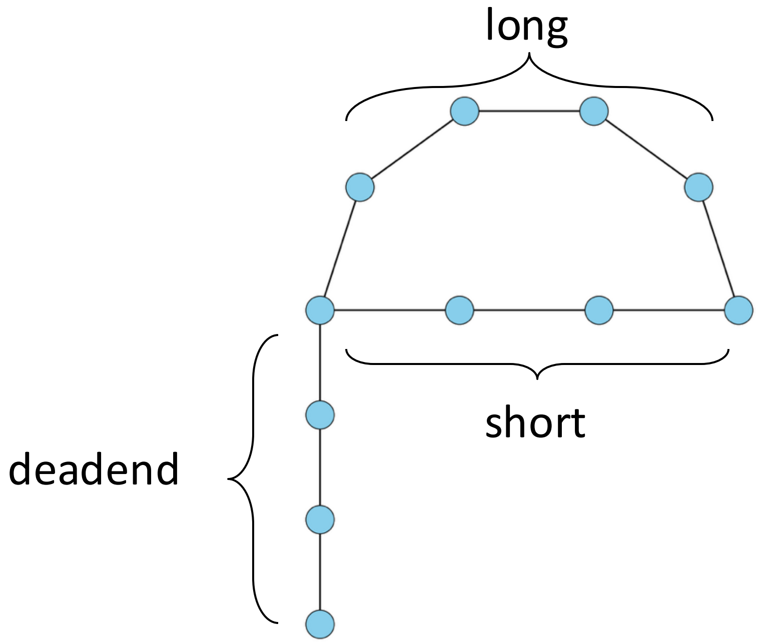
- Our reasoning task is a symmetric variant of the graph connectivity task.
- **(s, t)-Connectivity:** Are nodes s and t connected in a given graph G?
- **(s, t_1, t_2)-Connectivity:** Is node s connected to t_1 or t_2 in a given graph G?



Is s connected to t_1 or t_2?

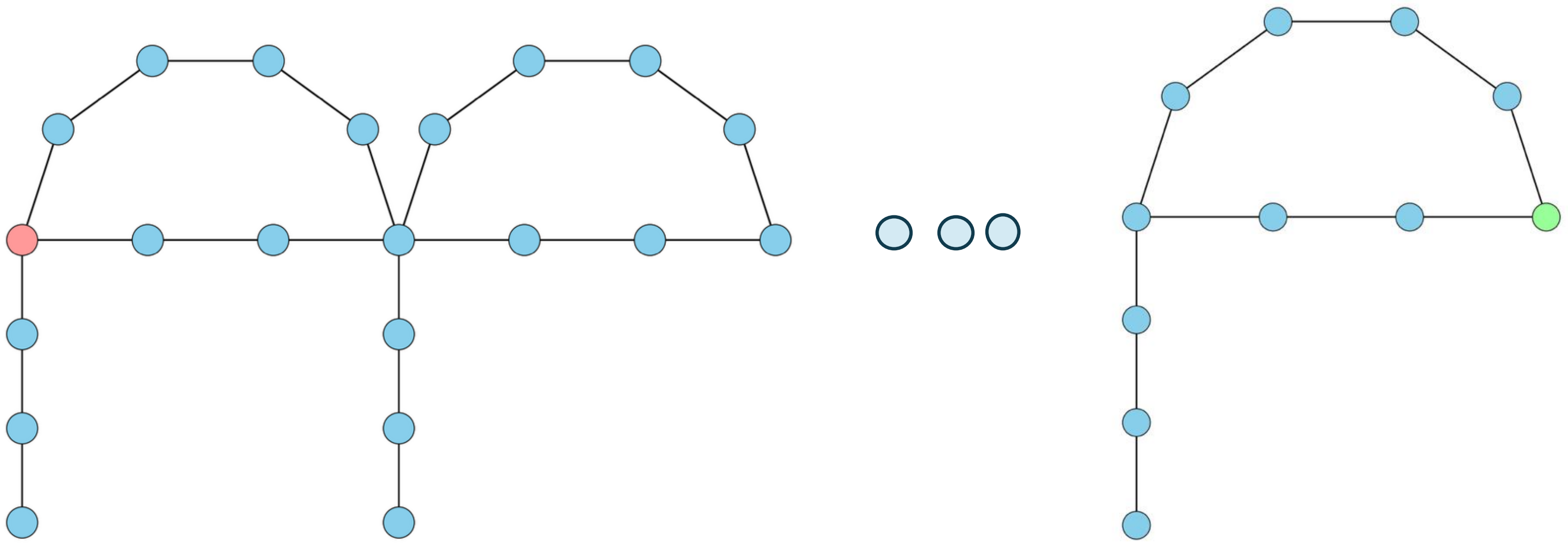
Graph Connectivity Tasks

- Our reasoning task is a symmetric variant of the graph connectivity task.
- **($\boldsymbol{s}, \boldsymbol{t}$)-Connectivity:** Are nodes s and t connected in a given graph G?
- **($\boldsymbol{s}, \boldsymbol{t_1}, \boldsymbol{t_2}$)-Connectivity:** Is node s connected to t_1 or t_2 in a given graph G?
- We focus on Bridge(**short**, **long**, **deadend**, depth) graphs:



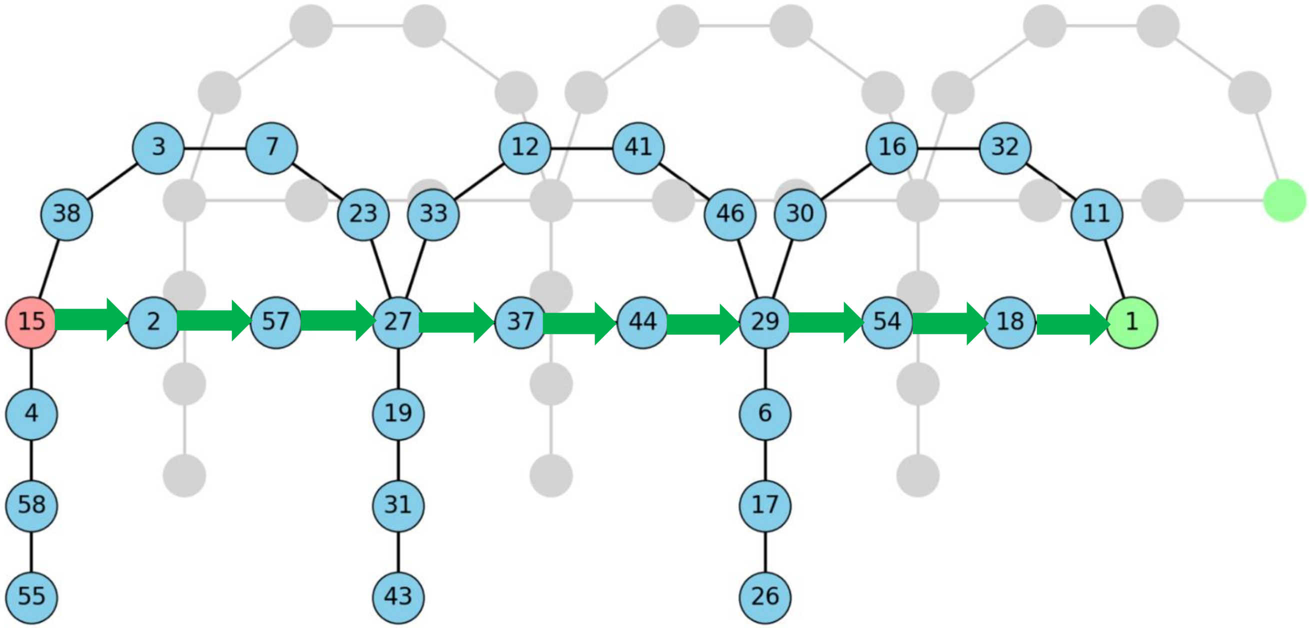
Graph Connectivity Tasks

- Our reasoning task is a symmetric variant of the graph connectivity task.
- **(s, t)-Connectivity:** Are nodes s and t connected in a given graph G?
- **(s, t_1, t_2)-Connectivity:** Is node s connected to t_1 or t_2 in a given graph G?
- We focus on Bridge(short, long, deadend, **depth**) graphs:



Training Models with Different Sequential Scales

- We fix the bridge graph and generate many randomly labeled instances of it.
- A graph is encoded as a randomly ordered list of edges.
- We train models with CoT strategies of different sequential scales and evaluate them.



Bridge(short=3, long=5, deadend=3, depth=3)

Input Prompt

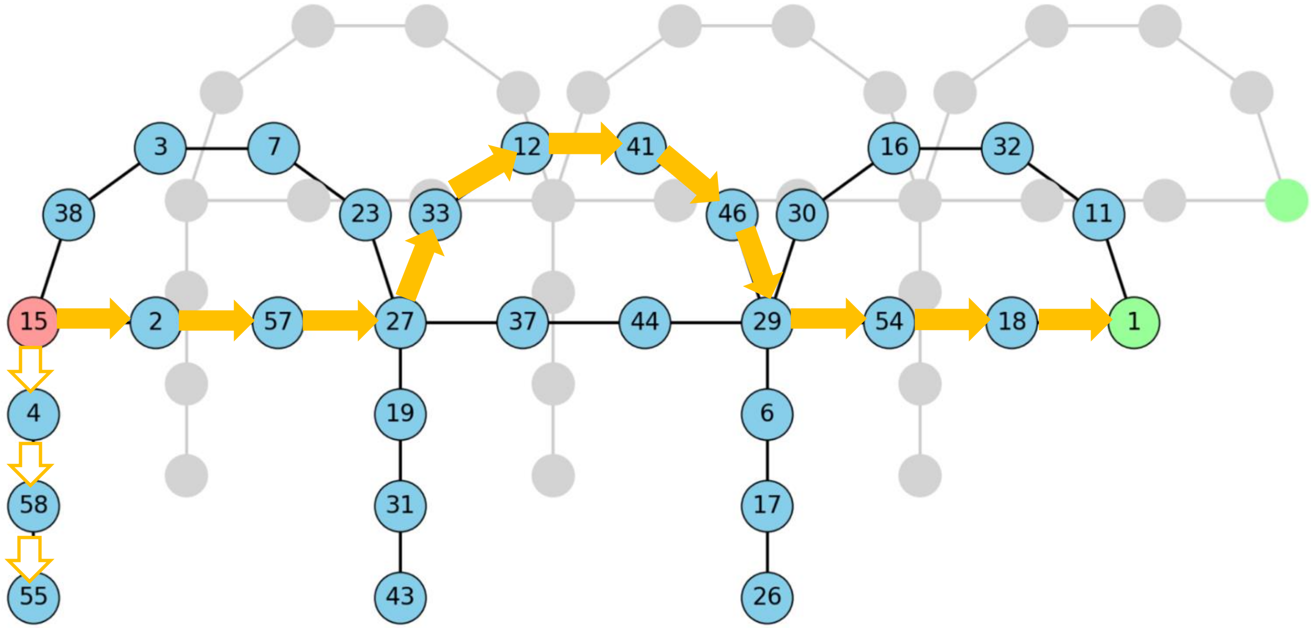
```
Graph:  [(29 54) (15 2) ...  (47 9) (32 16)]
Task:  15 to 8 or 1 ?
```

Chain-of-thought

```
Shortest-Path:  [15 2 57 27 37 44 29 54 18 1]
Decision:   [1]
```

Training Models with Different Sequential Scales

- We fix the bridge graph and generate many randomly labeled instances of it.
- A graph is encoded as a randomly ordered list of edges.
- We train models with CoT strategies of different sequential scales and evaluate them.



Bridge(short=3, long=5, deadend=3, depth=3)

Input Prompt

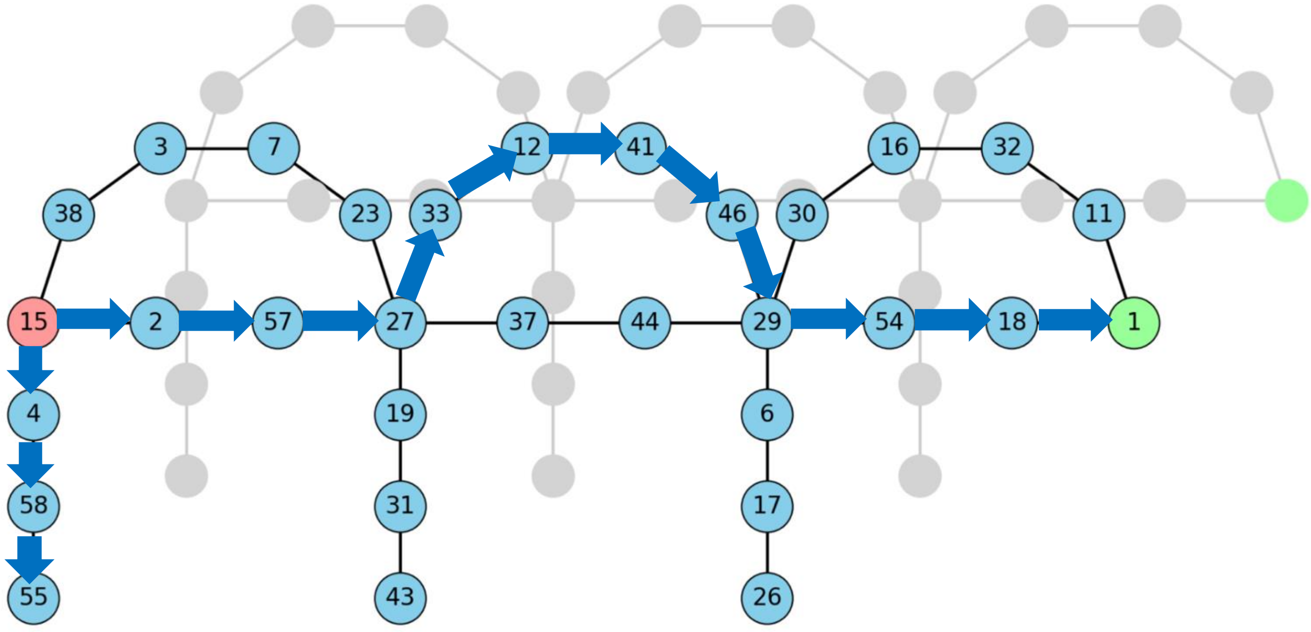
```
Graph:  [(29 54) (15 2) ...  (47 9) (32 16)]
Task:  15 to 8 or 1 ?
```

Chain-of-thought

```
Path:   [15 2 57 27 33 12 41 46 29 54 18 1]
Decision:   [1]
```

Training Models with Different Sequential Scales

- We fix the bridge graph and generate many randomly labeled instances of it.
- A graph is encoded as a randomly ordered list of edges.
- We train models with CoT strategies of different sequential scales and evaluate them.



Bridge(short=3, long=5, deadend=3, depth=3)

Input Prompt

```
Graph:  [(29 54) (15 2) ...  (47 9) (32 16)]
Task:  15 to 8 or 1 ?
```

Chain-of-thought

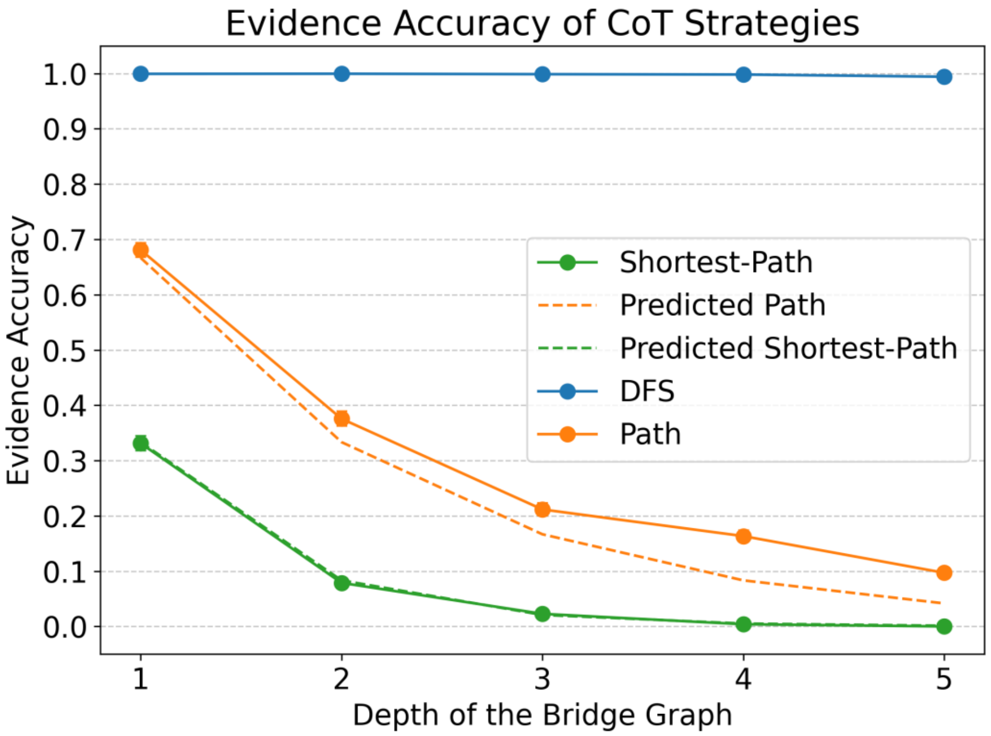
```
DFS: [15 4 58 55 2 57 27 33 12 41 46 29 54 18 1]
Decision:  [1]
```

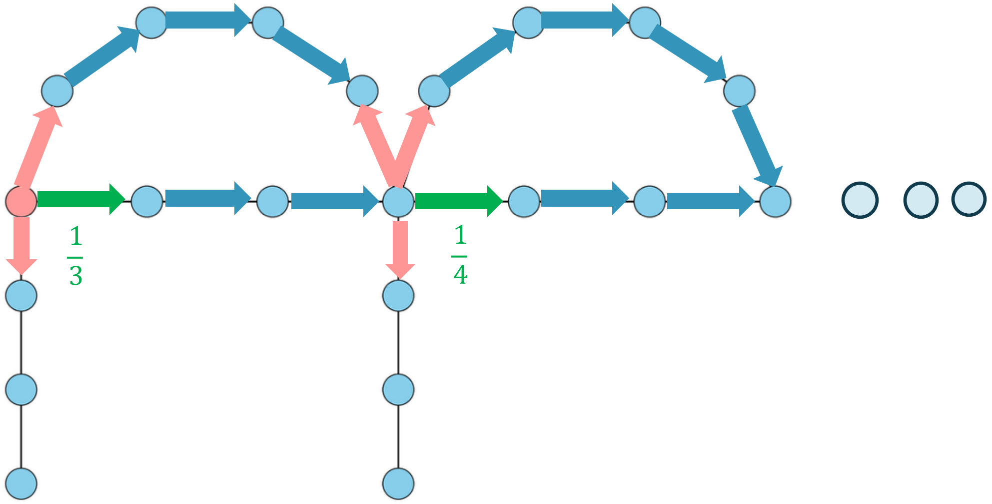
Verify(s, t_1, t_2, CoT)

if $s \neq \text{CoT}[1]$ or $\text{CoT}[L] \notin \{t_1, t_2\}$ **then**
  return false
for $i = 2$ to L **do**
  if CoT[i] has no neighbor in CoT[$1 : i-1$] **then**
    return false
return true

Experimental Results

- We train models for Bridge graphs with various depths.
- We observe that short CoT models have exponentially small accuracy.
- This accuracy is captured by the probability of in-distribution DFS traces.

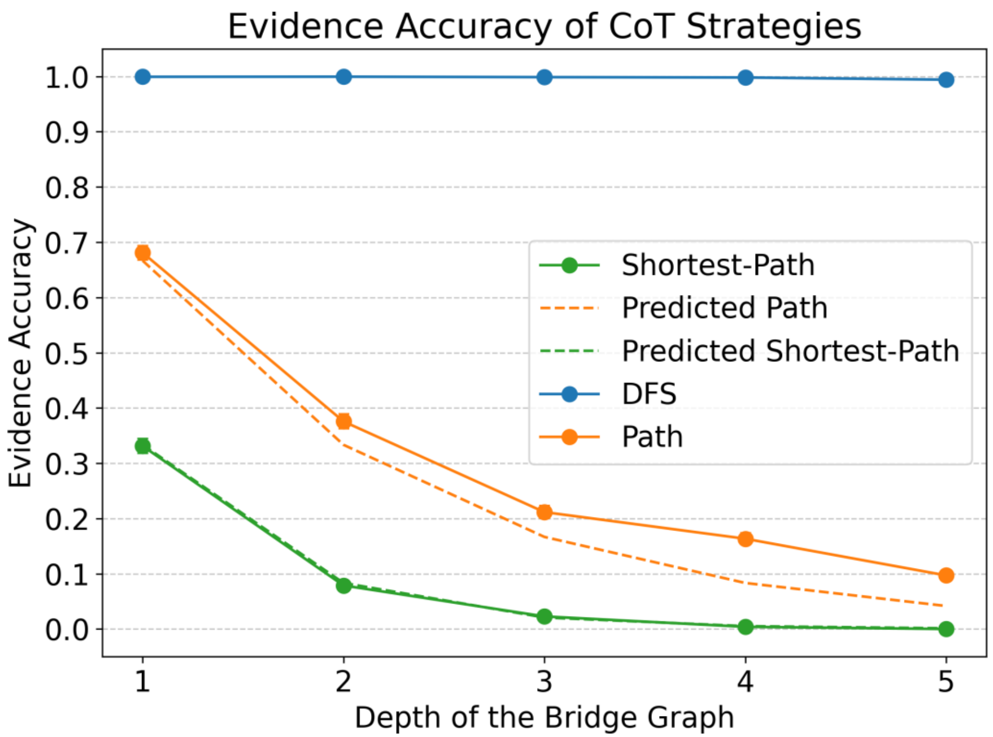


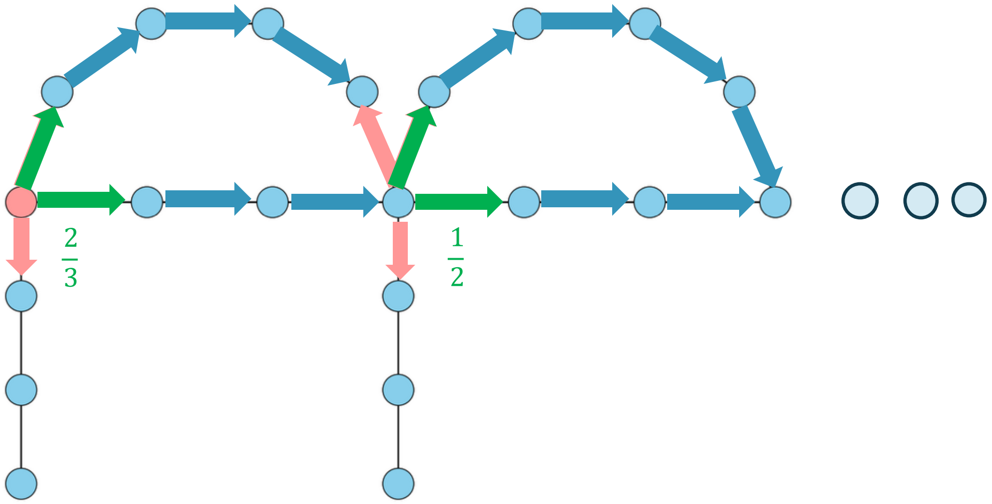


--- Predicted Shortest-Path $= \frac{1}{3} * \left(\frac{1}{4}\right)^{d-1}$

Experimental Results

- We train models for Bridge graphs with various depths.
- We observe that short CoT models have exponentially small accuracy.
- This accuracy is captured by the probability of in-distribution DFS traces.

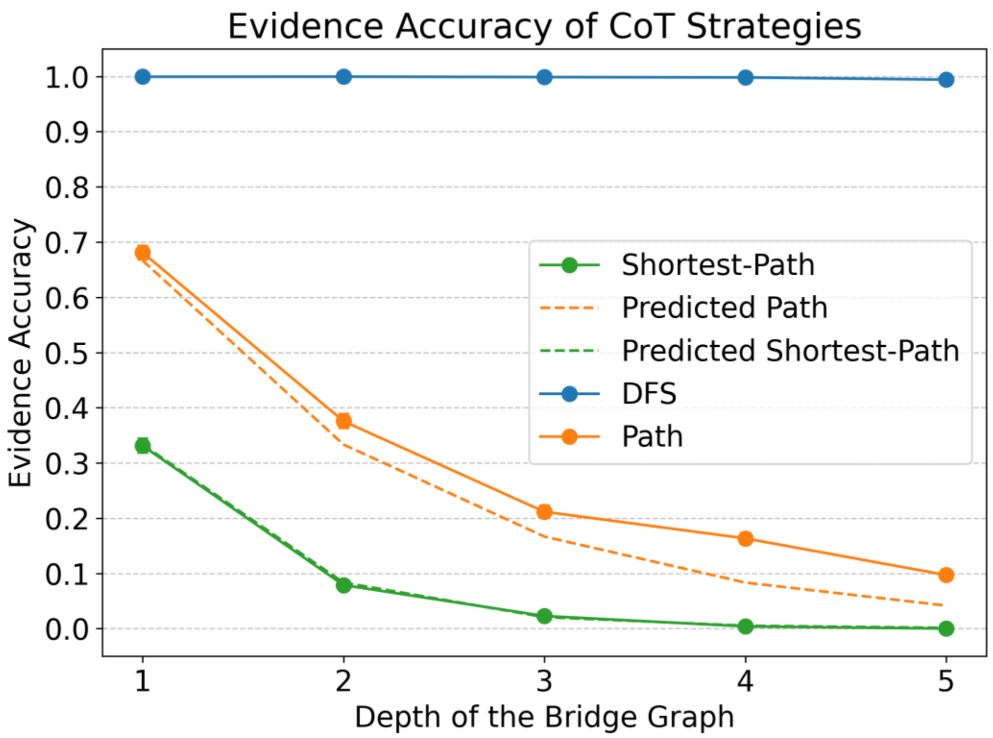


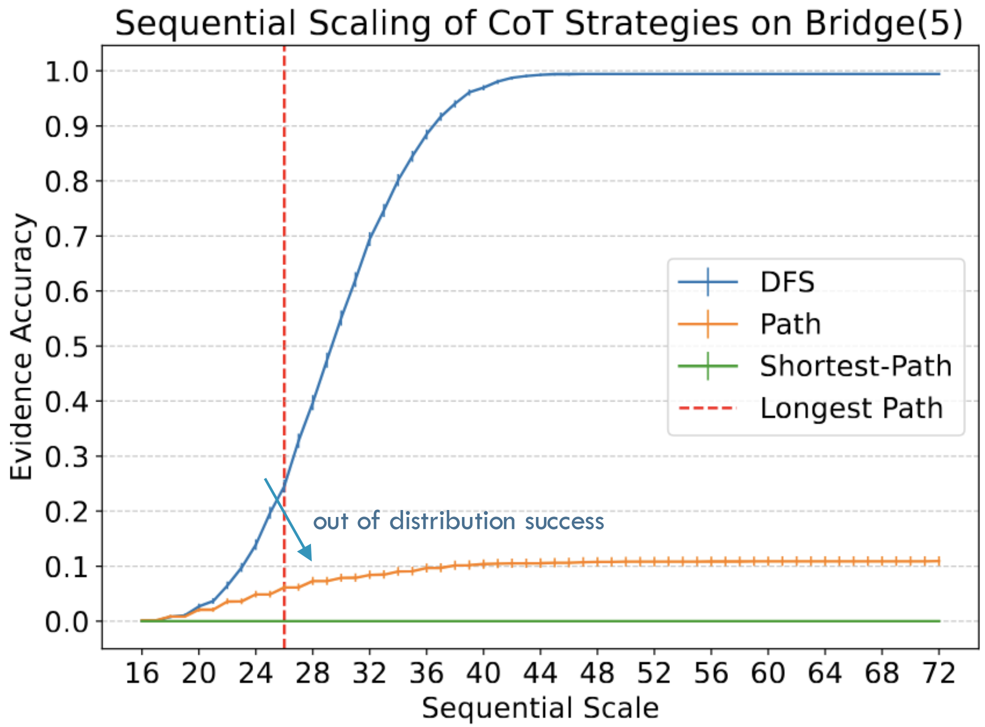


--- Predicted Path $= \frac{2}{3} * \left(\frac{1}{2}\right)^{d-1}$

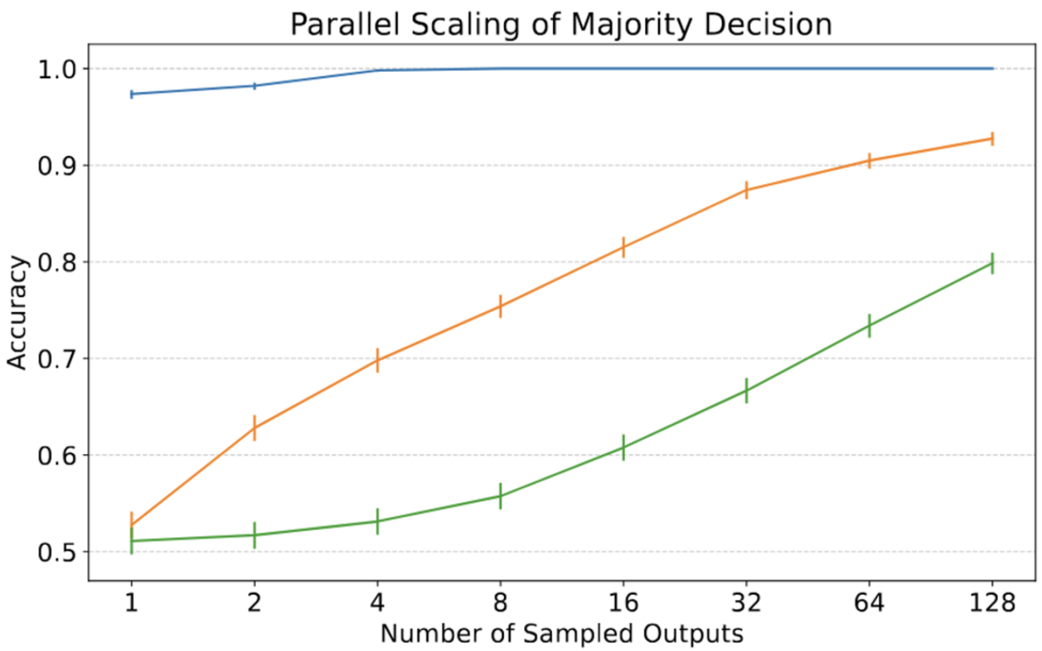
Experimental Results

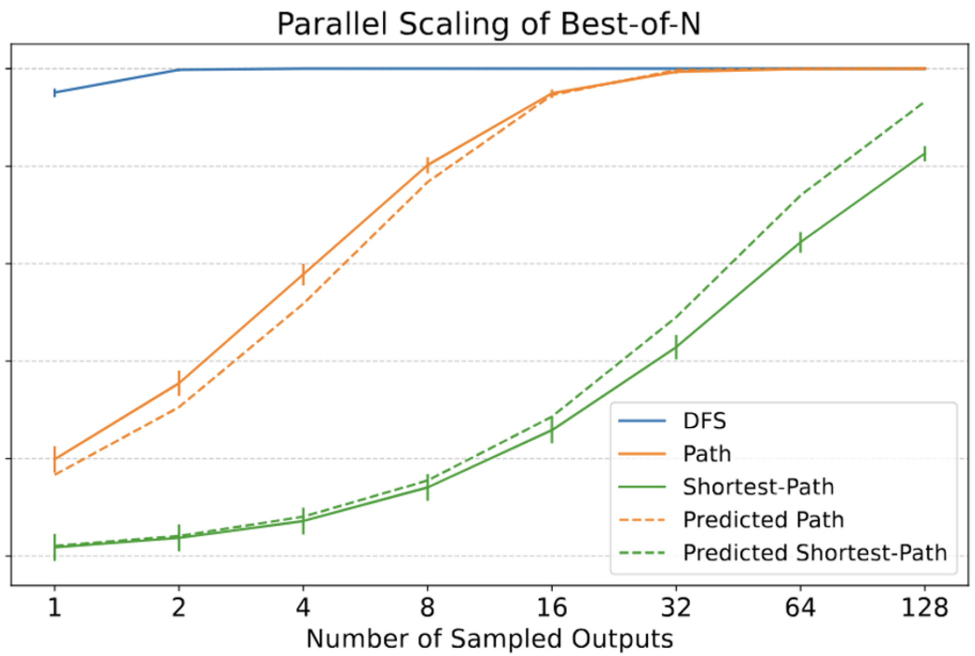
- We train models for Bridge graphs with various depths.
- We observe that short CoT models have exponentially small accuracy.
- This accuracy is captured by the probability of in-distribution DFS traces.





Parallel Scaling of Models Trained on Short CoTs

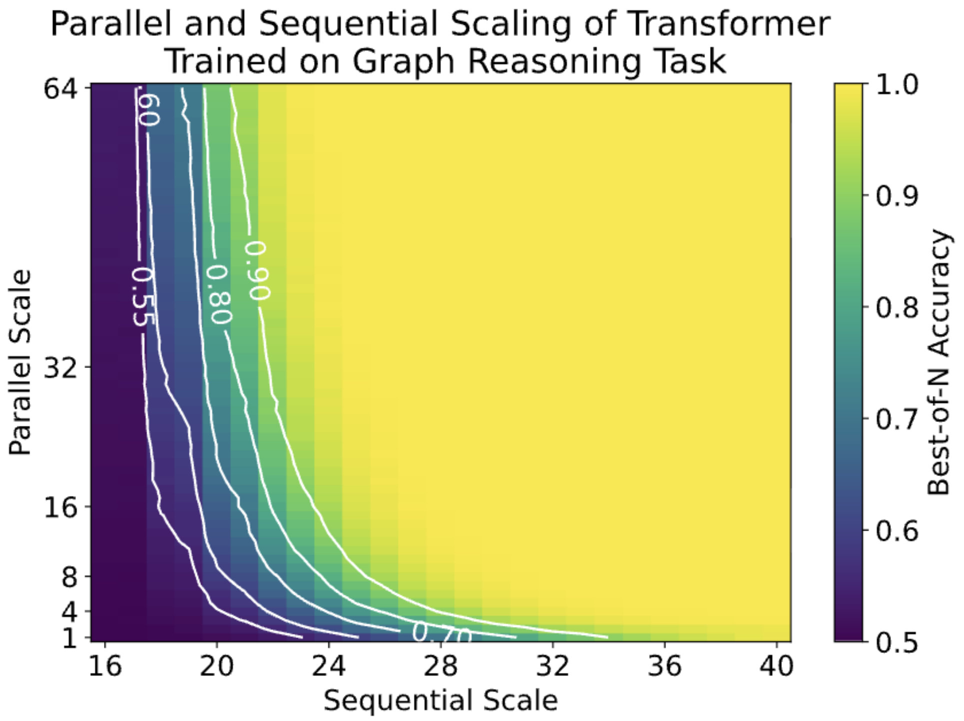




$\{CoT_i, decision_i\}_{i=1}^{n} \rightarrow \text{Maj}(\{decision_i\}_{i=1}^{n}) = t$

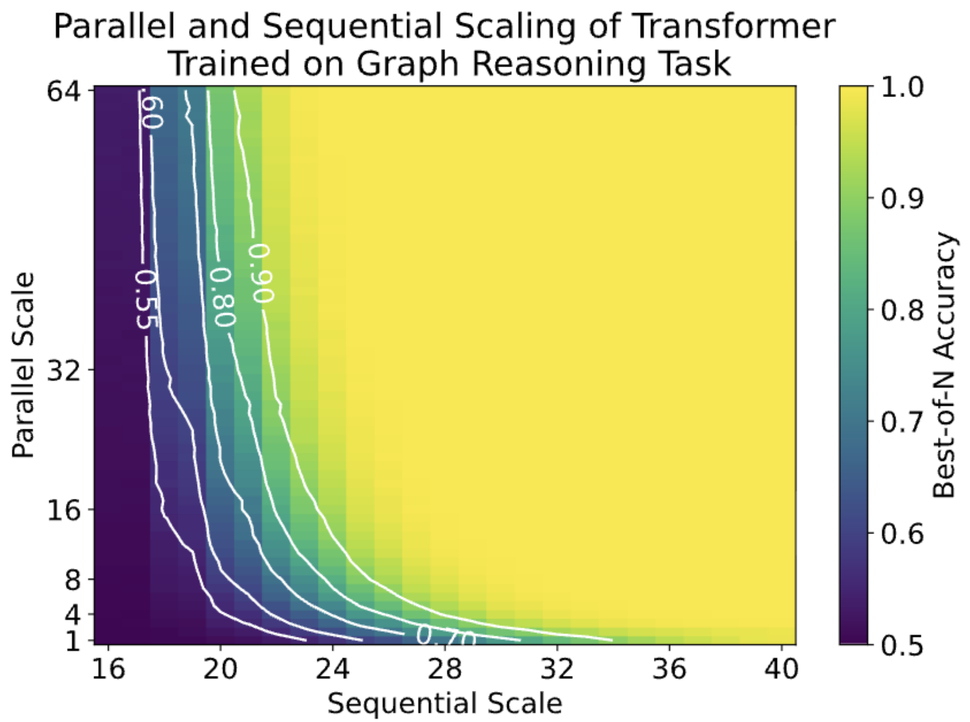
$\{CoT_i, decision_i\}_{i=1}^{n} \rightarrow \text{Any}(\{\text{Verify}(s, t_1, t_2, CoT_i)\}_{i=1}^{n})$

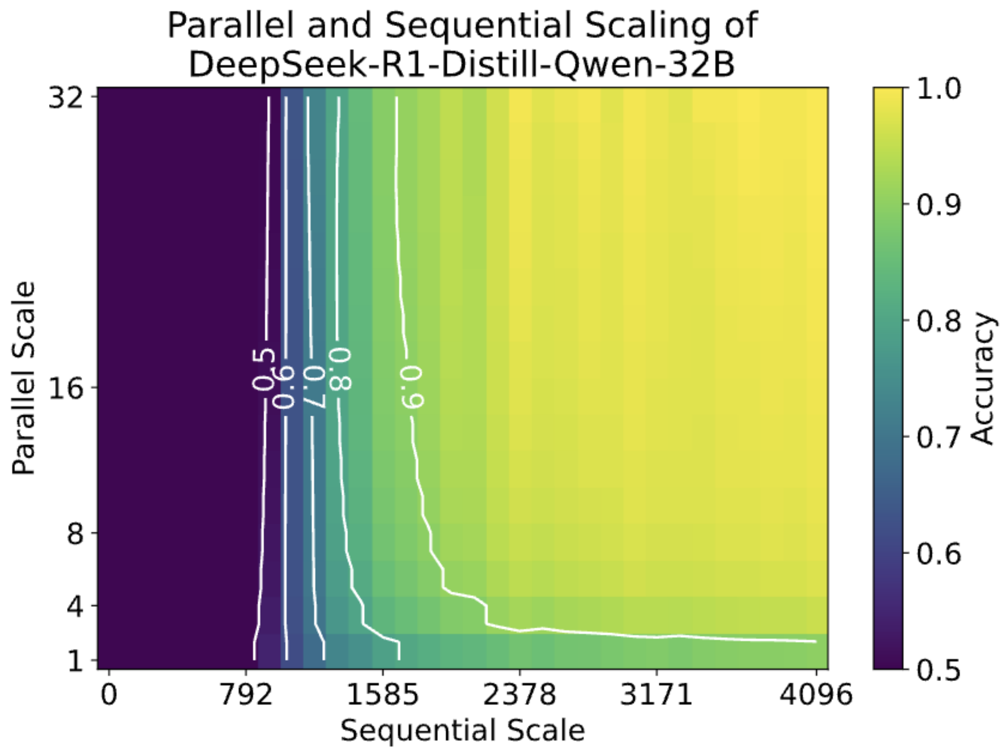
Sequential and Parallel Scaling of CoT Models



- We evaluate model's accuracy with combinations of parallel and sequential scaling.
- Sequential scale: token budget for each chain of thought.
- Parallel scale: number of independent chains of thought.

Sequential and Parallel Scaling of CoT Models





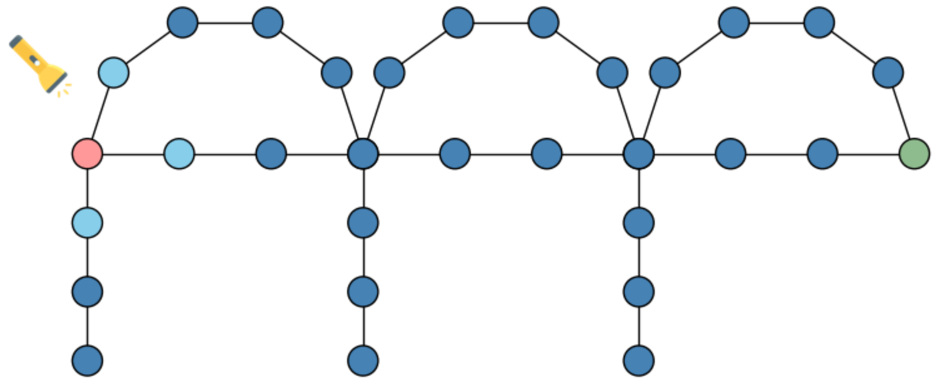
- We also evaluate frontier reasoning models on our graph connectivity task.
- We observe similar trends to our from-scratch training experiments.
- Sequential scaling has an exponential advantage in the short CoT regime.

Theoretical Evidence for Necessity of Long CoT

- Based on Vertex Query Model
- Based on Transformer Expressivity Limitations

Theoretical Separation based on Vertex Query Model

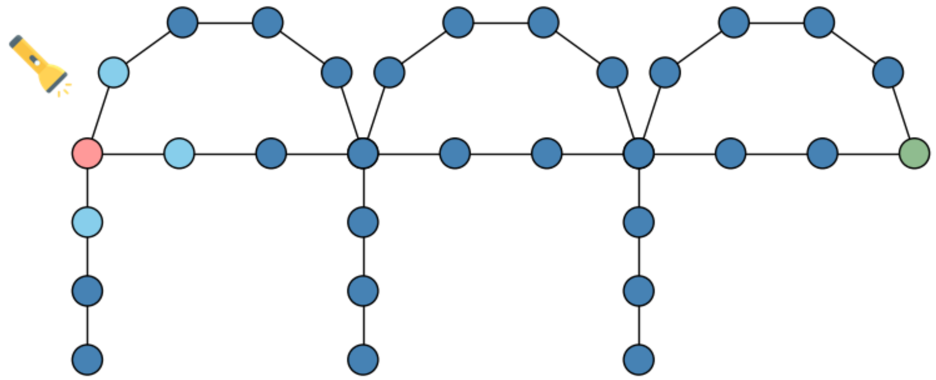
- We observed that the model's accuracy is captured by the probability of in-distribution DFS traces: $P_{G\sim\text{Bridge}}(\text{Verify}(M_S(G))) = P_{DFS}(D_S)$ for $S \in \{$S-Path, Path, DFS$\}$.
- The models cannot distinguish between the unvisited neighbors at inference-time, no matter how they were trained.
- Inspired by this observation, we introduce Vertex Query Model.



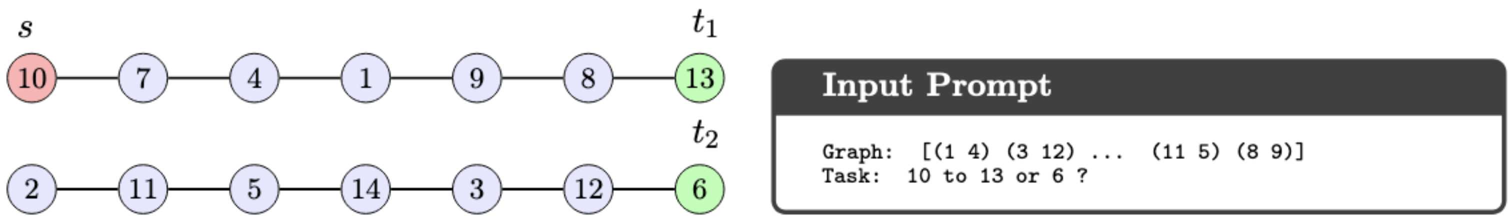
Theoretical Separation based on Vertex Query Model

- **Definition.** An algorithm for (s, t_1, t_2)-connectivity is implementable in the **Vertex Query Model (VQM)** if it takes as input s_1, t_1, t_2 ,and can only access the graph G through "neighborhood queries" N_G ,which given a vertex v, returns the set $N_G(v) = \{u : \exists (v, u) \in E\}$.

- We also define the **Restricted Vertex Query Model (RVQM)**, where the algorithm can only initially query s, and subsequently can only query vertices in the sets returned by previous queries.

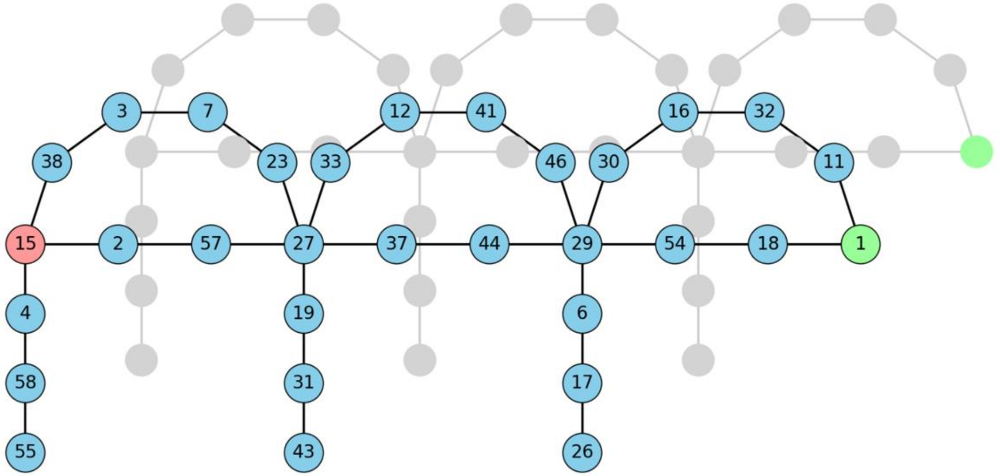


Two-Path Graph Connectivity with VQM



- Consider the graph G given by two disjoint paths of length $L \geq 3$. Suppose s, t_1, t_2 are three distinct endpoints of these paths. Then:
- $O(L)$ queries are sufficient: There is a VQM algorithm that executes $L - 1$ queries and solves the (s, t_1, t_2)-connectivity problem with probability 1.
- $\Omega(L)$ queries are needed: For any VQM algorithm that executes $q \leq (L - 2)/2$ queries, the probability of correctness of the algorithm on (s, t_1, t_2)-connectivity is exactly $1/2$.

Bridge Graph Connectivity with RVQM



- Consider an algorithm in the Restricted Vertex Query Model solving (s, t_1, t_2)-connectivity.
- Sequential scaling succeeds: There exists an algorithm which makes $(1+\delta)2\mathrm{ld}$queries and succeeds with probability at least $1-\exp\left(-\frac{1}{2}\mathrm{d}\delta^2\right)$.
- Parallel scaling fails: *Any algorithm which makes no more than* $(1-\delta)\frac{3}{2}ld$ *queries succeeds with probability at most* $\frac{1}{2}+exp\left(-\frac{1}{2}\delta^2\frac{3}{2}d\right)$ *.Thus, parallel scaling with majority vote needs independent runs to succeed with probability* $\geq 2/3$.

Theoretical Separation based on Transformer Expressivity

- We compare many chains of constant length to one long chain of polynomial length.

Theorem 1 (Informal statement of Theorem 4). *Assume the complexity-theoretic statement that* $\mathsf{TC}^0 \not\supseteq \mathsf{L}$. *Then the following is true for bounded-depth, limited-precision transformers.*

- ***Sequential scaling succeeds****: There is a constant* $c > 0$ *such that a transformer with a CoT of length* $\leq n^c$ *solves any* (s, t_1, t_2)*-connectivity problem.*
- ***Parallel scaling fails****: For any constants* $C_1, C_2 > 0$*, and any transformer architecture, majority vote over* $\leq n^{C_1}$ *independently-sampled CoTs of length* $\leq C_2$ *has accuracy* $\leq \frac{1}{2} + o(1)$ *for* (s, t_1, t_2)*-connectivity problems.*

Proof Ingredients

- **Sequential scaling succeeds**:
  - Corollary 2.1 from (Merrill and Sabharwal, 2023) : $\mathrm{TIME}\big(t(n)\big) \subseteq \mathrm{CoT}\big(t(n)\big)$.
  - Log-precision transformers with $t(n)$-length chain of thought can simulate Turing machines that run in time $t(n)$.
  - They can solve (s, t_1, t_2)-connectivity using standard graph traversal algorithms like depth-first search.

Merrill, William, and Ashish Sabharwal. "The expressive power of transformers with chain of thought." (2023).

Proof Ingredients

- **Parallel scaling fails**:

Definition 5 (TC^0 computational model). *A TC^0 circuit is a boolean circuit with AND, OR, NOT, and MAJORITY gates of potentially unbounded fan-in. A TC^0 circuit family is a collection of circuits indexed by the input size n, such that for each input size the circuit has polynomial width and bounded depth.*

Proposition 1 (Transformers are in TC^0; implied by Theorem 14 of (Chiang, 2024)). *For any bounded-depth softmax-attention transformer $T : \Sigma^* \to \mathbb{R}^{|\Sigma|}$ and any polynomial $p(n)$, there is a function $\hat{T} : \Sigma^* \to \mathbb{R}^{|\Sigma|}$ in TC^0 that approximates T to $2^{-p(n)}$ additive error on inputs of length n.*

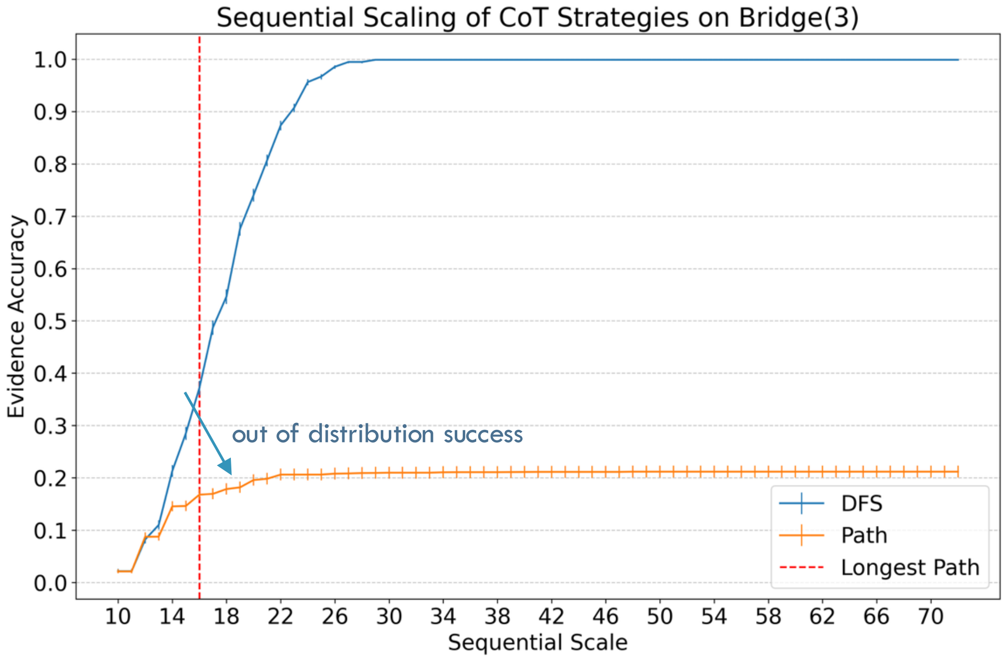
Chiang, David. "Transformers in uniform TC^0." (2024).

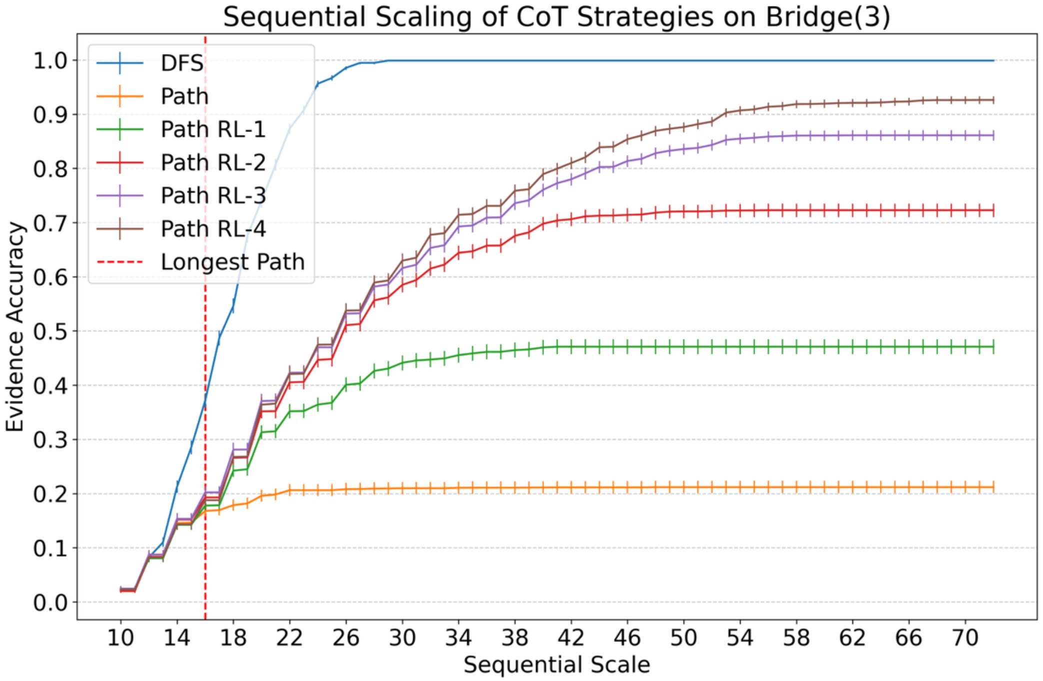
Proof Steps

- **Parallel scaling fails**:
  - Step 0: For transformer $T : \Sigma^* \to R^{|\Sigma|}$ and $x \in \Sigma^k$, let $D_{T,m}(x)$ denote the autoregressive distribution formed by sampling m tokens autoregressively from T.
  - Step 1: Find TC^0 function $\hat{T} : \Sigma^* \to R^{|\Sigma|}$, such that $D_{\hat{T},p(n)}(x)$ approximates $D_{T,p(n)}(x)$.
  - Step 2: Simulate constant-length CoT with a randomized TC^0 function $\tilde{T} : (\Sigma^* \cup \{0, 1\})^* \to \Sigma$.
  - Step 3: Take majority of constant-length CoTs and derandomize it to construct a TC^0 function that solves (s, t_1, t_2)-connectivity.
  - Step 4: Provide a TC^0 reduction from the L complete (s, t)-connectivity problem to the (s, t_1, t_2)-connectivity problem.

Merrill, William, and Ashish Sabharwal. "The expressive power of transformers with chain of thought." (2023).

Emergence of Long CoT





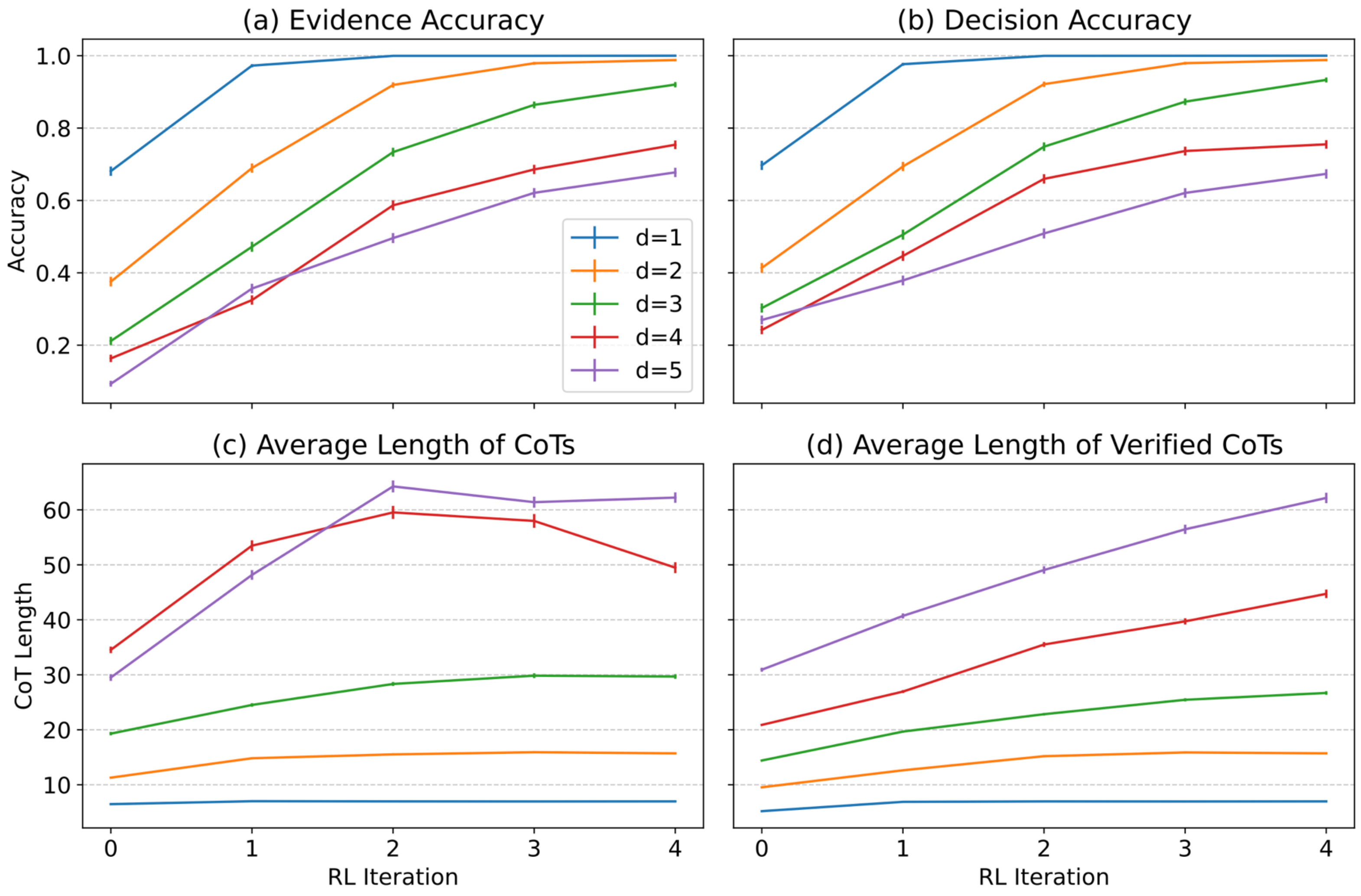
RL Iteration(M, D)

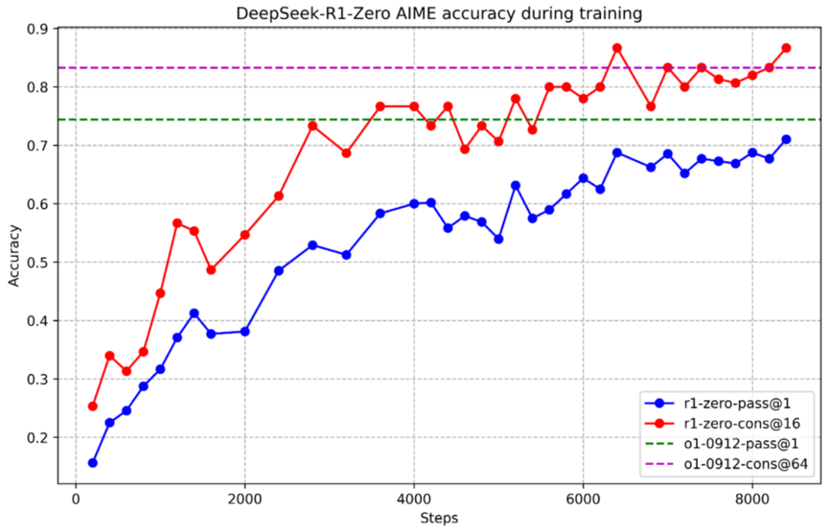
Zelikman, Eric, et al. "Star: Bootstrapping reasoning with reasoning." (2022)

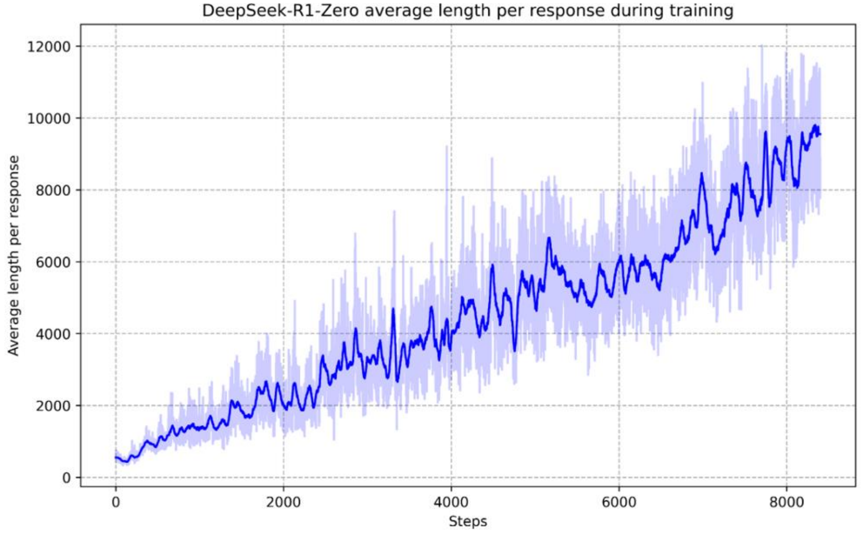
verified $\leftarrow$ empty list
for each task in D **do**
    (CoT, decision) $\leftarrow M$(task)
    if VERIFY(task, CoT) **then**
        add (task, CoT, decision) to verified
Fine-tune M on verified

Emergence of Long CoT

Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." (2025).







Parsa Mirtaheri

Ezra Edelman

Enric Boix

Eran Malach

Samy Jelassi