Efficient Turing Machine Simulation with Transformers

Qian Li
SICIAM, Shenzhen Research Institute of Big Data

Yuyi Wang
CRRC Zhuzhou Institute & Tengen Intelligence Institute

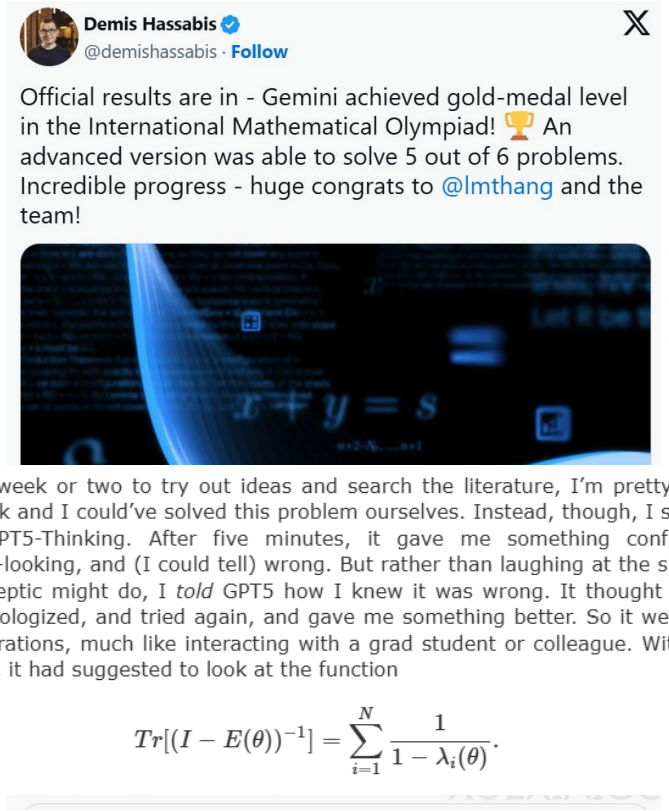
ICLR 2026

Large Language Models (LLMs)

Transformer-based LLMs shows amazing **general reasoning** abilities

- International Mathematical Olympiad (IMO) gold medal.
- helped crack the long-standing quantum computing problems.
- ICPC World Finals gold medal.

Question： can LLMs achieve AGI?



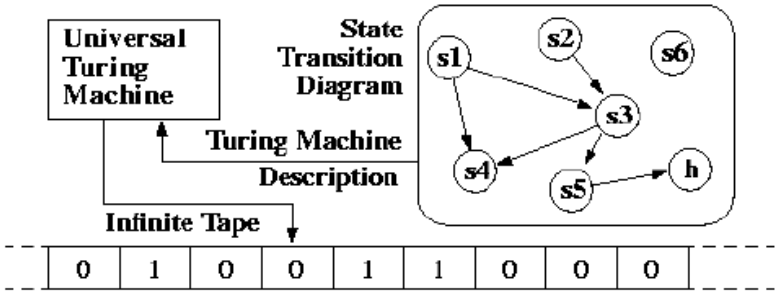
To Understand the Deduction Ability

Question: what enables transformers to support such reasoning capabilities?

A central result [1-5]: **Transformers (with COTs) are Turing complete**

- Any Turing machine can be simulated by a Transformer (with proper parameters).

Turing Machine



* https://web.mit.edu/manoli/turing/www/turing.html

To Understand the Deduction Ability

Question: what enables transformers to support such reasoning capabilities?

A central result [1-5]: **Transformers (with COTs) are Turing complete**

- Any Turing machine can be simulated by a Transformer (with proper parameters).

Church-Turing hypothesis: **Any algorithmic process can be simulated using a TM**.

- Algorithmic progress: a finite number of exact instructions
- Computable = Solvable by TMs.

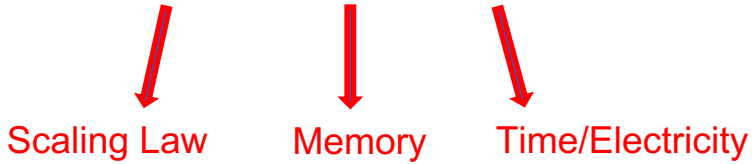
Motivation

Question: what enables transformers to support such reasoning capabilities?

A central result [1-5]: **Transformers (with COTs) are Turing complete**

- Any Turing machine can be simulated by a Transformer (with proper parameters).

Question: how many resourced, e. g. size/window/CoT, are required?



Our Results

Question: how many resourced, e.g. size/window/CoT, are required?

Source	Precision	#Parameter	Window length	COT / TM step
Perez et al. 2021	$O(\log t)$	$O(1)$	$O(t)$	1
Bhattamishra et al. 2020	unbounded	$O(1)$	$O(t)$	1
Merrill & Sabharwal, 2024	$O(\log t)$	$O(1)$	$O(t)$	1
Li et al. 2024	$O(1)$	$O(\log t)$	$O(t \log t)$	$O(\log t)$
Qiu et al. 2024	$O(\log t)$	$O(1)$	$O(t \log t)$	$O(\log t)$
Li & Wang, 2025	$O(1)$	$O(1)$	$O(s)$	$\boldsymbol{O(s)}$
This Paper	$O(1)$	$O(1)$	$O(s)$	$\boldsymbol{O(s^c)}$

The exponent $c > 0$ can be made arbitrarily small by letting the #head*#layer sufficiently large constants.

Our Results

Source	Precision	#Parameter	Window length	COT / TM step
Perez et al. 2021	$O(\log t)$	$O(1)$	$O(t)$	1
Qiu et al. 2024	$O(\log t)$	$O(1)$	$O(t \log t)$	$O(\log t)$
Li & Wang, 2025	$O(1)$	$O(1)$	$O(s)$	$\boldsymbol{O(s)}$
This Paper	$O(1)$	$O(1)$	$O(s)$	$\boldsymbol{O(s^c)}$

Theorem: Any t-time s-space k-tape TM can be simulated by a constant bit-size Transformer with head-layer product K and window length $O(s)$, that takes $O(t \cdot s^{6k/K})$ CoT steps in total.

The exponent $c > 0$ can be made arbitrarily small by letting the #head*#layer sufficiently large constants.

Byproduct: Justification of LogSparse Attention

- **Quadratic Barrier of full attention**: it takes $O(n^2)$ to generate the n tokens
- Many **sparse attention** architectures are proposed:
  - BigBird (Deepmind), MoBA (Moonshot), DSA (Deepseek), NSA (Deepseek, ACL best paper)...
- However, almost all sparse attention are empirical and lack of theoretical justification.

- We give a justification of the **LogSparse attention** where each query attends only to tokens that are "1,2,4,8,..." steps earlier, truncated by the window size as a natural and promising sparse attention design in practice.
  - In our construction, the offsets are fixed to be $\lceil s^{6k/K} \rceil$, $\lceil s^{6k/K} \rceil^2$, ..., truncated by the window length. In particular, in the regime of practical interest, say when $s(n) \leq 2^{500}$ and $K \geq 6000$. It is lossless to focus on 2-tape TMs. Then the common ratio $\lceil s^{6k/K} \rceil$ =2.
  - It takes $O(\log w)$ time to generate a token.

Proof: Multi-queue TM as the Bridge

- **Multi-queue TM**: an automaton equipped with multiple queues.
  - Post Machine = 1-queue TM.
  - **Synchronous**: every queue pops and also appends exactly one symbol at each step.
- **Step I**: multi-tape TMs ⇒ synchronous multi-queue TMs

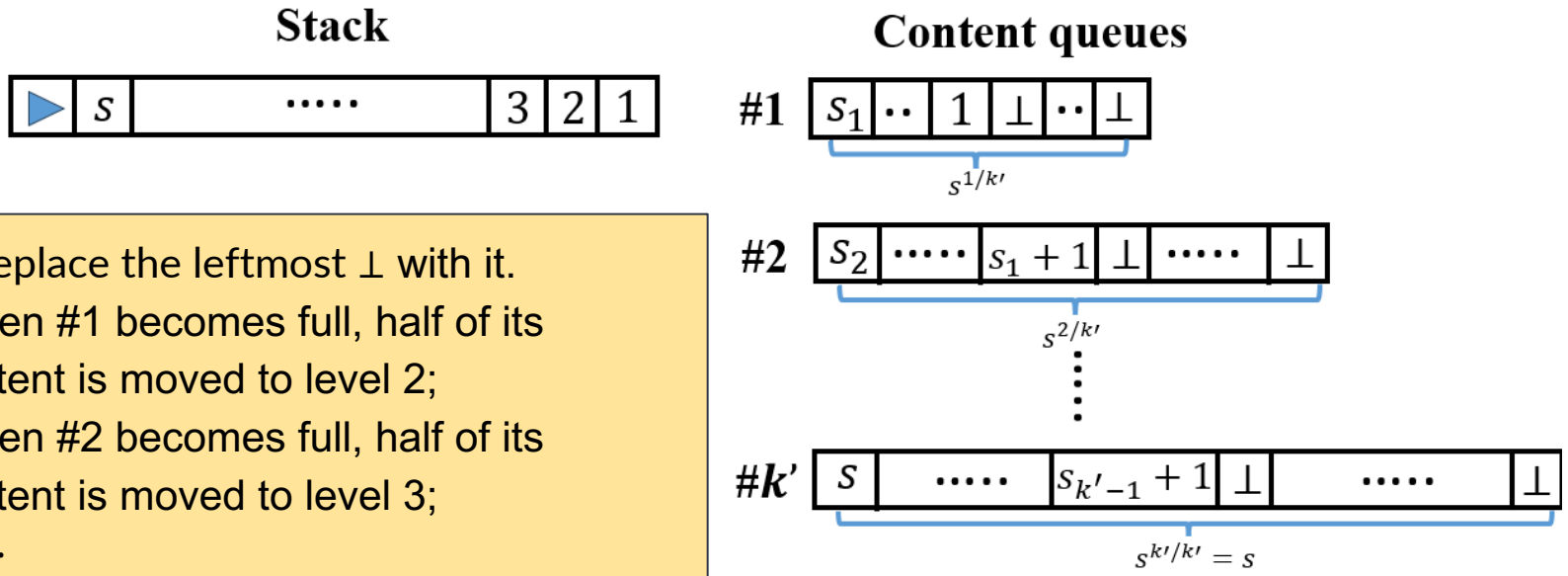
Lemma 1 (main technical part): t-time s-space k-tape TM ⇒ $O(ts^{k'})$-time s-space $(6kk')$-queue Synchronous TM.

  - Idea: one tape ⇒ two stacks ⇒ several queues whose sizes grow geometrically
- **Step II**: synchronous multi-queue TMs ⇒ Transformers.

Lemma 2: t-time s-space K-queue Synchronous TM ⇒ $O(t)$-CoT s-window K head-layer product constant bit-size Transformer

  - Idea: adapt the simulation technique from one queue to multiple queues.

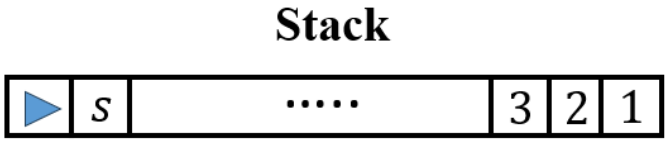
Step I: Simulating Stack with Queues

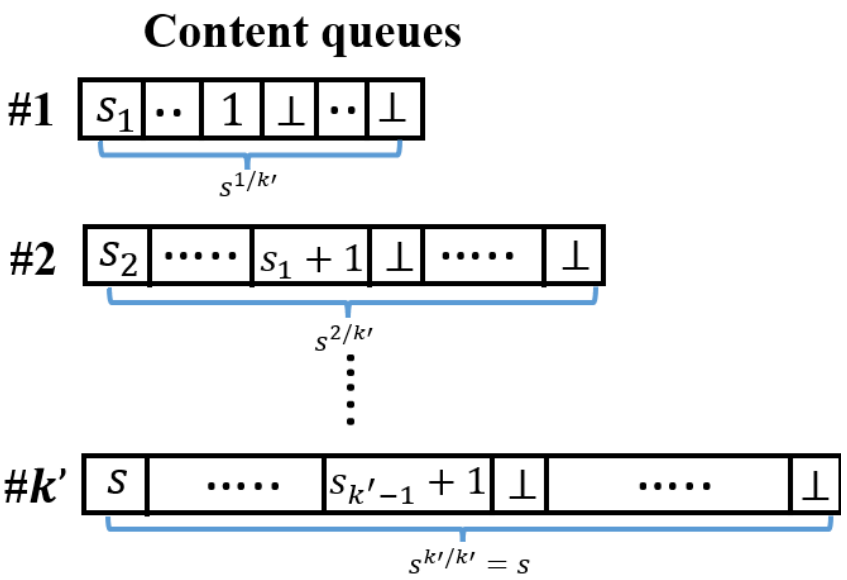


Push: replace the leftmost ⊥ with it.

- If then #1 becomes full, half of its content is moved to level 2;
- If then #2 becomes full, half of its content is moved to level 3;
- ……

Step I: Simulating Stack with Queues

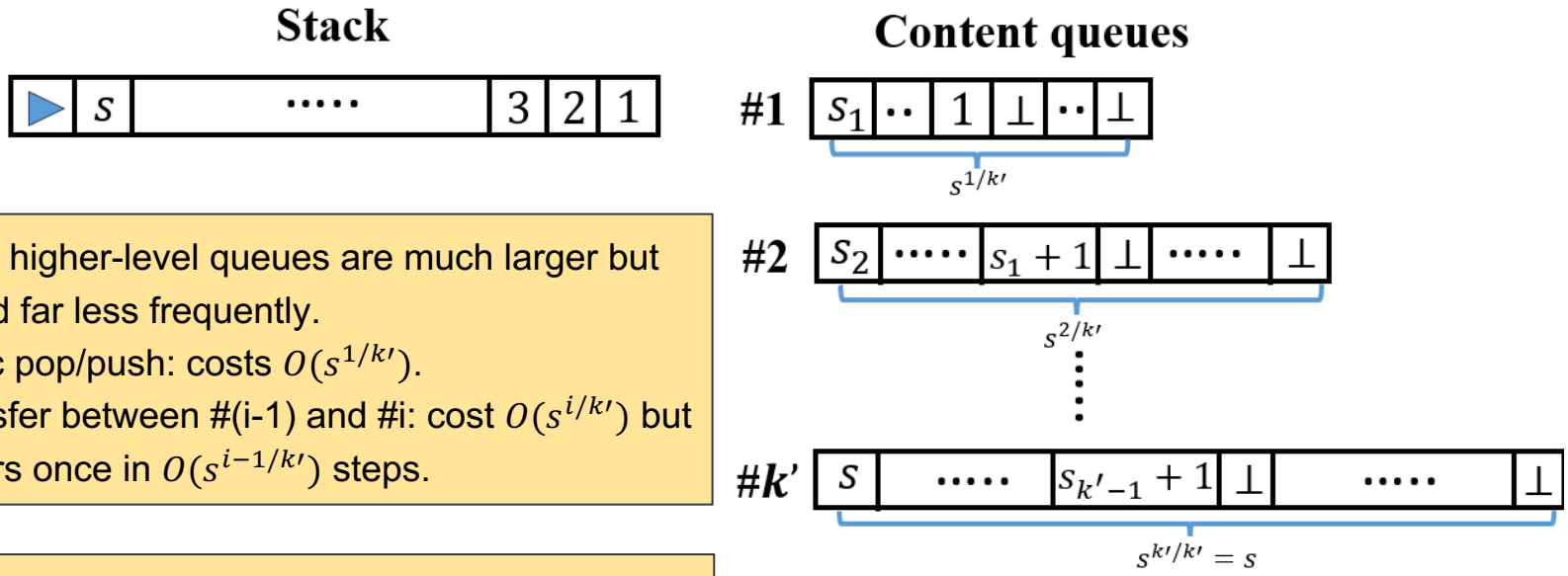




Pop: replace the rightmost 1 with ⊥.

- If then #1 becomes empty, refill the left half by pulling elements from #2.
- If then #2 becomes empty, refill the left half by pulling elements from #3.
- ……

Step I: Simulating Stack with Queues



Analysis: higher-level queues are much larger but accessed far less frequently.

- Basic pop/push: costs $O(s^{1/k'})$.
- Transfer between #(i-1) and #i: cost $O(s^{i/k'})$ but occurs once in $O(s^{i-1/k'})$ steps.

Remark: to make synchronous, we use some tricks

- auxiliary queues as buffers
- Split each content queue into two halves

Conclusion & Open Problems

Summary:

- Transformers are Turing complete even when:
  - Constant bit-size
  - Window length = space complexity.
  - LogSparse attention suffices.
  - $O(s^c)$ CoT per MT step
- Techniques: TM$\Longrightarrow$ multi-queue TM $\Longrightarrow$ Transformer.

Open problems:

- Can the per-step overhead be further reduced to O(1)?
- Our construction uses a nonstandard relative positional encoding that assumes the space bound is known in advance; how can positional encodings be designed to adapt dynamically when the space bound is unknown?

References

1. *Attention is Turing Complete*. Pérez, Barceló, and Marinkovic. JMLR-2021.
2. *On the Computational Power of Transformers and Its Implications in Sequence Modeling*. Bhattamishra, Patel, and Goyal. CoNLL-2020.
3. *The Expressive Power of Transformers with Chain of Thought*. Merrill, Sabharwal. ICLR-2024
4. *Chain of Thought Empowers Transformers to Solve Inherently Serial Problems*. Li, Liu, Zhou, and Ma. ICLR-2024.
5. *Ask, and It Shall Be Given: Turing Completeness of Prompting*. Qiu, Xu, Bao, and Tong. ICLR-2025.
6. *Constant Bit-size Transformers are Turing Complete*. Li, and Wang. NeurIPS-2025.