TM Simulations Results

September 29, 2025

Part I

Simulating Time with Square-Root Space — Ryan Williams 2025 [Wil25]

Theorem 1. *(Main Theorem) For every function $t(n) \geq n$,*

$$\mathrm{TIME}[t(n)] \subseteq \mathrm{SPACE}[\sqrt{t(n) \log t(n)}.$$

1 Proof Idea

I will reduce the simulation to an instance of the TreeEvaluation problem, which has been proven to be solvable in space $O(db + h \log(db))$, where d is the number of children of each node (fan-in), b is the number of bits of the output (bit-length), and h is the height of the tree (more about TreeEvaluation will be included (TODO)). Essentially, I will use the computation of a time $t(n)$ TM and turn it into a tree-like structure, by using the notion of block-respecting TM (more about block-respecting TM will be included (TODO)), of height $h = \Theta(t/b)$, bit-length b, and fixed fan-in d where I can choose $b = \Theta(\sqrt{t \log t})$ to make $h = \Theta(t/b) = \Theta(\sqrt{t/\log t})$. Applying the TreeEvaluation result, we have the space to be

$$O(db + h \log(db)) = O(\sqrt{t \log t} + \sqrt{t/\log t} \cdot \log(\sqrt{t \log t})) = O(\sqrt{t \log t}).$$

2 Previous Result: On Time Versus Space — Hopcroft, Paul, and Valiant 1977 [HPV77]

Theorem 2. $\mathrm{TIME}[t(n)] \subseteq \mathrm{SPACE}[t(n)/\log t(n)]$.

Proof. TODO $\square$

3 Warm-Up Result

Theorem 3. *For every function* $t(n) \geq n^2$, $\mathrm{TIME}[t(n)] \subseteq \mathrm{SPACE}[\sqrt{t(n)} \log t(n)]$.

Before I present the proof, I also need the following theorem.

Theorem 4. *Given a multitape TM M running in time $t(n)$, we can construct an equivalent* ***oblivious*** *2-tape TM M' running in time $T(n) \leq O(t(n) \log t(n))$. Furthermore, given n and $i \in [T(n)]$ specified in $O(\log t(n))$ bits, the 2 head positions of M' can be computed in* $\mathrm{poly}(\log t(n))$ *time.*

The proof of this theorem will be included later (TODO).

Using Theorem 4, I now have an oblivious 2-tape TM M' running in time $T \leq O(t \log t)$. I will now partition the computation of M' into time and tape blocks (see block-respecting TM). Specifically, the two tapes of M' will be split into **tape blocks** of $b(n)$contiguous cells and the $T(n)$ steps of M' are split into $B(n) := O(T(n)/b(n))$ **time blocks** of length up to $b(n)$. Because each time block is of length $\leq b$, at most two tape blocks can be accessed during a time block for each tape, four tape blocks for both tapes.

3.1 Computation Graph

I now construct a computation graph as follows. The graph $G_{M'}$ will have $B + 1 = O(T/b)$ nodes, where each node i for each time block $i \in 0, 1, \ldots, B$. Each node i will have a fixed d number of nodes directed at it, representing the time blocks that need to be read in order to compute the content of the tape blocks accessed during time block i. Since there are at most 2 tape blocks accessed during a time block for each tape (4 for both), I need at most 4 nodes to be connected to node i for each i. I will also connect node $i - 1$ with i to obtain the head positions of time block i. Thus, for each node i there are at most 5 nodes connected to it (in other words, $d = 5$).

Due to the obliviousness of M', I can claim that determining whether nodes i and j are connected cost only an additional $\mathrm{poly}(\log t)$ space. By Theorem 4, I can calculate the 2 tape head positions at any point in time using only $\mathrm{poly}(\log t)$ time. Thus, I can keep track of the tape head locations from time block i to time block j using $\mathrm{poly}(\log t)$ space.

Now, I need to compute the content of each tape blocks accessed during time block i, which will be the status of M' at the end of time block i. More specifically, let $\mathrm{content}(i)$ denote the content of time block i. I will include in $\mathrm{content}(i)$ the following: the state of M', the 2 tape head positions, and a list of the contents of those tape blocks accessed during time block i. As there are at most 4 such tape blocks, $\mathrm{content}(i)$ can be encoded in $O(b + \log t) \leq O(b)$ bits. I can also compute $\mathrm{content}(i)$ in $O(b)$ time and space, given the contents of all nodes j connected to i, by simply simulating M' for b steps.

3.2 TreeEvaluation **Instance**

Now, I'm ready to construct a TreeEvaluation instance. More precisely, I construct a tree $R_{M'}$ of height $B+1$ and fan-in at most 5, with a root node that will evaluate to content(B). Each tree node v of $R_{M'}$ will be labeled by a path from some graph node i to graph node B of $G_{M'}$. Inductively, I label the root node of $R_{M'}$ with the empty string. Then, for every tree node v labeled by the path P from graph node i to graph node B, and for every graph node j connected to i, I create tree node v' labeled by the path from j to i then path P.

The desired value to be computed at node v labeled by path from i to B is content(i). For $i = 0$, this value is the initial configuration of M', which can be produced immediately in $O(n)$ time and space. For $i > 0$, content(j) can be computed in $O(b)$ time and space given the contents of all children nodes.

Finally, although the tree has $2^{\Theta(B)} \leq 2^{\Theta(T/b)}$ nodes, I actually have random access to the every node and every function at each node, with an additional cost of only poly$(\log t)$ space. This is because given a node, I can compute the children nodes similarly to how I can determine whether two graph nodes are connected, mentioned earlier. And I can reuse this space once the children nodes have been determined. I can then use the TreeEvaluation result and let $b = \sqrt{t}\log t$ to get the desired result of $O(\sqrt{t}\log t)$. Now, to get the better bound in the main theorem, I need to reduce the height of the tree from $B+1 = O(T/b) = O((t\log t)/b)$ down to $O(t/b)$.

4 Main Result

The $t\log t$ blow up is mainly due to the simulation of oblivious TMs. In order to prevent it, I will no longer use such simulation, but then the difficulty becomes determining the edges of the computation graph efficiently. However, I can use more space to accomplish this. The idea is that I will enumerate over possible computation graphs G' and introduce a method for checking that $G' = G_{M'}$ in the functions of my TreeEvaluation instance.

To start, I am given a multitape TM M which runs in time t. I will use the following lemma to get a block-respecting TM M'.

Lemma 5. *For every time-constructible $b(n)$, $t(n)$ such that $\log t \leq b \leq t$ and every t-time l-tape TM M, there is an equivalent $O(t)$-time block-respecting $(l+1)$-tape TM M' with blocks of length b.*

Now, I have a multitape TM M' with $p := l+1$ tapes, runs in time $O(t)$, and has $B := O(t/b)$ time and tape blocks.

4.1 Computation Graph

I will construct a slightly different computation graph than the one in the warm-up example. Here, the set of nodes in G' is

$$S = \{(h,i),(h,0,i) \mid h \in [p], i \in [B]\}.$$

Intuitively, each (h, i) will correspond to the content of the relevant block of tape h after time block i, while each $(h, 0, i)$ corresponds to the content of the i-th block of tape h when it is accessed for the first time, i.e. the initial configuration. Each node (h, i) is labeled with an integer $m_{(h,i)} \in \{-1, 0, 1\}$, indicating the tape head movement at the end of time block i.

Next, there are two types of edges in G'. As before, each node (h, i) will be directed to by nodes that are needed to compute the content of (h, i). For each $h' \in [p]$, we connect $(h', i-1)$ to (h, i) for $i > 1$ and $(h', 0, 1)$ to $(h, 1)$. Additionally, for each $h' \in [p]$, we connect (h', i') (or $(h', 0, i')$) to (h, i) indicating the most recent time block that accessed the tape block of time block i. Ultimately, each node has in-degree of at most $2p$.

4.2 Succinct Graph Encoding

TODO

Part II

Improved Bounds on the Space Complexity of Circuit Evaluation — Yakov Shalunov 2025 [Sha25]

Theorem 6. *(Main Theorem) Given a size s circuit C on $n \leq s$ inputs and an input $x \in \{0,1\}^n$, the output of $C(x)$ can be evaluated in space $O(\sqrt{s \log s})$.*

1 Proof Idea

Similar to the previous paper by Williams, I will reduce this simulation to the TreeEvaluation problem.

TODO

Part III
Tree Evaluation Is In Space $O(\log n \cdot \log \log n)$ — James Cook, Ian Mertz 2023 [CM23]

Theorem 7. *(Main Theorem) Any* TreeEvaluation *instance can be computed in space $O((h + \log k) \cdot \log \log k)$, where h is the height and k is the alphabet size.*

NOTE: For the papers in this file, I use the improved result from the Improvement section of this paper, taken from Goldreich's survey [Gol24].

1 Proof Idea

2 Main Result

3 Improvement: On the Cook-Mertz Tree Evaluation Procedure — Goldreich 2024 [Gol24]

References

[CM23] James Cook and Ian Mertz. Tree evaluation is in space $o(\log n \cdot \log \log n)$, Nov 2023.

[Gol24] Oded Goldreich. On the cook-mertz tree evaluation procedure, Dec 2024.

[HPV77] John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *J. ACM*, 24(2):332–337, April 1977.

[Sha25] Yakov Shalunov. Improved bounds on the space complexity of circuit evaluation, Jun 2025.

[Wil25] Ryan Williams. Simulating time with square-root space, Feb 2025.