Electronic Colloquium on Computational Complexity, Report No. 6 (2026)

Separating RAM and Multitape Turing Machines with Short Random Oracles

Lijie Chen* Yichuan Wang†

2026-01-04

Abstract

We prove that relative to a random oracle answering $O(\log n)$-bit queries, there exists a function computable in $O(n)$ time by a random-access machine (RAM) but requiring $n^2/\mathsf{polylog}(n)$ time by any multitape Turing machine. This provides strong evidence that simulating RAMs on multitape Turing machines inherently incurs a nearly quadratic time overhead, since the random oracle can be heuristically instantiated by a concrete hash function such as SHA-256.

*UC Berkeley. lijiechen@berkeley.edu.
†UC Berkeley. yichuan-21@berkeley.edu.

ISSN 1433-8092

1 Introduction

Random-access machines (RAMs) [CR73] and multitape Turing machines represent two fundamental models of computation that differ significantly in their memory access capabilities. A RAM can access any memory location in constant time, allowing it to jump directly to arbitrary positions in its memory array. In contrast, a multitape Turing machine must move its tape heads sequentially, requiring linear time to traverse between distant memory locations. Indeed, multitape Turing machines are the standard definition of computation in complexity theory (see, e.g., [AB09]), while RAMs are more similar to how modern CPUs operate in practice.

It is well known that RAMs can simulate a T-time multitape Turing machine in $O(T)$ time (throughout the paper, we make the standard assumption that the word size of a RAM is $O(\log n)$), whereas the best-known simulation of a RAM on multitape Turing machines runs in T^2 time [CR73]. This naturally raises the following question:[1]

Can multitape Turing machines simulate T-time RAMs in $T^{1.99}$ time?

Additionally, a recent breakthrough by [Wil25] (following the tree-evaluation algorithm of Cook and Mertz [CM24]) shows that a T-time multitape Turing machine can be simulated by a $\widetilde{O}(\sqrt{T})$-space Turing machine.[2] This provides an additional motivation to study the question above—if the answer is yes, then one can also simulate T-time RAMs in $T^{1-\epsilon}$ space, which is the major open question left by [Wil25].

1.1 Our Results

In this work, we show a tight separation between RAMs and multitape Turing machines in the *random oracle model*.

Theorem 1.1 (Informal version of Theorem 4.3)**.** *With high probability over a uniformly random oracle $\mathcal{O} : \{0,1\}^{O(\log n)} \to \{0,1\}^{O(\log n)}$, there exists a function that is computable in $O(n)$ time by an $\mathcal{O}$-oracle RAM but requires at least $n^2/\mathsf{polylog}(n)$ time by an $\mathcal{O}$-oracle multitape Turing machine.*

We believe our result provides strong evidence that simulating RAMs on multitape Turing machines inherently incurs a quadratic time overhead (i.e., the answer to the motivating question is negative), since the random oracle can be heuristically instantiated by a concrete hash function, such as SHA-256. This gives a concrete candidate function that separates RAMs and multitape Turing machines.

An immediate question is whether one can remove the short random oracle and prove the separation in the *plain model* (i.e., unconditionally). We note that this might be challenging—it is known that a nondeterministic multitape Turing machine can simulate T-time deterministic RAMs in $T \cdot \mathsf{polylog}(T)$ time [GS89]. Hence, if one can prove an unconditional separation between RAMs and multitape Turing machines similar to Theorem 1.1, then one would be able to show that $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTIME}[n^{1.99}]$, a significant breakthrough result in complexity theory.[3]

[1] We note that the answer is obviously no for sublinear T, since a RAM can implement the $O(\log n)$-time *binary search* for finding an element in a sorted input array, while a multitape Turing machine must scan all elements of the array in linear time. Here we are interested in the setting in which T is at least linear, meaning both models have time to read the whole input.

[2] For space, multitape Turing machines and RAMs can simulate each other with only linear overhead [vEB90, SvEB88]. Therefore, the choice of a particular Turing machine model is not that consequential for space complexity.

[3] Here, $\mathsf{DTIME}[T(n)]$ and $\mathsf{NTIME}[T(n)]$ are defined with respect to multitape Turing machines. The best we know is $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTIME}\left[o\left(n \cdot (\log^\star n)^{1/4}\right)\right]$ [PPST83, Wil09].

Remark on the locality barrier. [CHO+22, Yao89] show that many complexity results *localize* in the sense that they hold for any *short* oracle. Indeed, Williams's result [Wil25] also *localizes*: $\mathsf{DTIME}[T] \subseteq \mathsf{SPACE}[\widetilde{O}(\sqrt{T})]$ holds for *any* oracle of input length at most $\sqrt{T}$. Our result unconditionally shows that there is a *locality barrier* for simulating RAMs on multitape Turing machines in subquadratic time (if one believes such simulation is indeed possible despite the random oracle separation).

2 Proof Overview

Below we give an overview of our proof of Theorem 1.1.

2.1 Defining the Hard Function

We first present a function Hard that is computable in linear time on RAMs but requires quadratic time on multitape Turing machines.

We partition the input string $x \in \{0,1\}^{10n\log n}$ as

$$x = (x_1, x_2, \ldots, x_n), \quad |x_i| = 10\log n.$$

The random oracle $\mathcal{O}$ is modeled as a uniformly random function

$$\mathcal{O} : \{0,1\}^{30\log n} \to [n]^3.$$

Thus, each query of the form (x_a, x_b, x_c) maps to a random triple $(a', b', c') \in [n]^3$.

We now define a sequence of indices $\{(i_{k,1}, i_{k,2}, i_{k,3})\}_{k=1}^{n}$ as follows:

- Start with $(i_{1,1}, i_{1,2}, i_{1,3}) = (1,1,1)$.
- For each $k \in \{2,3,\ldots,n\}$, set

$$(i_{k,1}, i_{k,2}, i_{k,3}) := \mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}}).$$

Finally, define Hard as the first bit of $i_{n,1}$. (Here we encode $i_{n,1}$ as a $\lceil \log n \rceil$-bit string.) Note that Hard indeed depends on $\mathcal{O}$ and should have been denoted as $\mathsf{Hard}^{\mathcal{O}}$; we write Hard for simplicity in the overview.

This function can be evaluated in $O(n\log n) = O(|x|)$ time on a RAM: each oracle query yields the next indices, and random access allows immediate retrieval of the required array entries.

Intuitively, a multitape Turing machine must perform the queries sequentially. To compute $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$, it must first locate the entries $(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$. Since the indices $(i_{k-1,1}, i_{k-1,2}, i_{k-1,3})$ are only revealed after the previous query, each step forces a fresh search across (most of) the input, requiring $\widetilde{\Omega}(n)$ time per query. Summed over n steps, this yields an almost quadratic lower bound. Below we elaborate on these two points separately.

Notation. Let M be a multitape Turing machine that computes Hard. Now we fix an input x of nearly maximal Kolmogorov complexity (i.e., $K(x) \approx |x|$; see Section 3.1 for the definition), and consider randomness only from the oracle $\mathcal{O}$. We will show that M must spend almost quadratic time to compute Hard on such x.

2.2 Sequential Necessity of Oracle Queries

We first show that, with high probability over the choice of $\mathcal{O}$, M must perform all queries $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$, and exactly in the prescribed order.

- **No queries can be skipped.** Note that the answer depends on $(i_{n,1}, i_{n,2}, i_{n,3})$ (i.e., it is the first bit of $i_{n,1}$), and every $(i_{k,1}, i_{k,2}, i_{k,3})$ is only revealed after querying $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$.[4]

  If even one such $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$ query is skipped (i.e., the machine never queries $\mathcal{O}$ at $x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}}$), then we can show that, conditioned on the machine's computation history, the machine has no information at all about the subsequent tuples $(i_{t,1}, i_{t,2}, i_{t,3})$ $(t \geq k)$. Therefore, the output of Hard also remains almost uniformly random. In this case, M succeeds with probability at most $1/2 + o(1)$.

- **Correct order of queries.** Similarly, we can argue that the queries $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$ must be performed in the prescribed order, with high probability over $\mathcal{O}$. The intuition is that M is very unlikely to query $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ before $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$, since, as discussed before, $i_{k,1}, i_{k,2}, i_{k,3}$ can only be revealed by querying $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$; without knowing the values $i_{k,1}, i_{k,2}, i_{k,3}$, the probability that the machine queries this particular oracle input by chance is small.

2.3 Lower Bounding the Time Gap Between Queries

Following the discussions above, we can from now on assume that M has made all oracle queries and that they are in the correct order (i.e., we condition on this event, which happens with high probability over $\mathcal{O}$).

Define stopping times

$$0 < T_1 < T_2 < \cdots < T_{n-1},$$

where T_k is the step when M makes the query $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$. We note that here $T_1, \ldots, T_{n-1}$ are random variables that depend on $\mathcal{O}$.

The central part of our proof is to show that for each $p = 2, 3, \ldots, n-1$, the gap $T_p - T_{p-1}$ must be large (say, $\geq n$) with high probability over $\mathcal{O}$.

We recall that T_{p-1} is the timestamp at which M queries

$$\mathcal{O}(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}}) = (i_{p,1}, i_{p,2}, i_{p,3}),$$

and T_p is the time when M next queries

$$\mathcal{O}(x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}}).$$

Fix the computation history Hist up to time T_{p-1}, and, specifically, Hist does not yet contain the outcome of the query at time T_{p-1},[5] and thus the triple $(i_{p,1}, i_{p,2}, i_{p,3})$ is still completely unknown.

Suppose, toward contradiction, that with high probability $T_p - T_{p-1} \leq n$. This means that after learning the triple $(i_{p,1}, i_{p,2}, i_{p,3})$ at time step $T_{p-1} + 1$, the machine M is able to prepare the subsequent query $(x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}})$ in at most n steps. But in n steps, M can only move its tape heads within distance $\leq n$ of their previous positions. Hence, the portions of the tapes that

[4] Ending up with $(i_{k,1}, i_{k,2}, i_{k,3})$ by chance is unlikely since there are n^3 possible triples.

[5] Precisely, the machine makes the query to $\mathcal{O}(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}})$ at time T_{p-1}, and *receives the answer* at time $T_{p-1} + 1$.

influence this computation have size $O(n)$ and therefore Kolmogorov complexity at most $O(n)$ (note that there are at most a constant number of tapes in a multitape Turing machine).

Now consider the following procedure: Given the tape contents visible within distance n at time T_{p-1}, one can enumerate all possible outcomes of the oracle call $\mathcal{O}(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}}) = (i_{p,1}, i_{p,2}, i_{p,3})$ (there are n^3 possibilities), simulate M's next n steps for each case, and observe the resulting access pattern. In particular, this process reveals $x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}}$ (with high probability over the choice of $(i_{p,1}, i_{p,2}, i_{p,3})$). Thus, we obtain a procedure that reconstructs a large fraction of the x_i's ($i \in [n]$, and recall that each $|x_i| = 10 \log n$) from only $O(n)$ bits of initial information.

But this contradicts our assumption that x was chosen to have near-maximal Kolmogorov complexity ($\approx 10n \log n$). Therefore, the assumption that $T_p - T_{p-1} \leq n$ with high probability must be false. We conclude that, with constant probability, each gap $T_p - T_{p-1}$ is at least $\Omega(n)$, establishing the quadratic lower bound.

3 Preliminaries

3.1 Kolmogorov Complexity

For a pair of strings $x, y \in \{0,1\}^*$, we define the following encoding of the pair (x, y):

$$\langle x, y \rangle \coloneqq 1^{\lceil \log |x| \rceil} 0 \, |x| \, xy,$$

whose length is $|x| + |y| + O(\log |x|)$.[6]

We briefly recall standard definitions from Kolmogorov complexity, following the plain (non-prefix-free) variant.

Definition 3.1 (Kolmogorov complexity)**.** *For any binary string x, the Kolmogorov complexity of x is*

$$\mathrm{K}(x) := \min\{|\langle M, w \rangle| : M \text{ is a Turing machine, } w \in \{0,1\}^*, \text{ and } M(w) = x\}.$$

Here $|M|$ denotes the description length of M in a fixed encoding, and $|w|$ is the length of the input.

Definition 3.2 (Conditional Kolmogorov complexity)**.** *For strings x, y, the conditional Kolmogorov complexity of x given y is*

$$\mathrm{K}(x|y) := \min\{|\langle M, w \rangle| : M \text{ is a Turing machine, } w \in \{0,1\}^*, \text{ and } M(\langle w, y \rangle) = x\}.$$

Proposition 3.3 (Basic properties)**.**

- *For any string x, $\mathrm{K}(x) \leq |x| + O(1)$.*
- *For any integers $n \geq k > 0$,*

$$\Pr_{x \sim \{0,1\}^n} \left[\mathrm{K}(x) \leq n - k\right] \leq 2^{-k+O(1)}.$$

  Thus, most n-bit strings have complexity at least $n - O(1)$.

- *For any integer $n > 0$,*

$$\mathop{\mathbb{E}}_{x \sim \{0,1\}^n} [\mathrm{K}(x)] \geq n - O(1).$$

[6] Any reasonable encoding suffices, as it only changes complexities by an additive $O(\log(|x| + |y|))$.

- *For any string y and integers $n \geq k > 0$,*

$$\Pr_{x \sim \{0,1\}^n} \left[\mathrm{K}(x|y) \leq n - k \right] \leq 2^{-k+O(1)}.$$

- *For any string y and integer $n > 0$,*

$$\mathop{\mathbb{E}}_{x \sim \{0,1\}^n} [\mathrm{K}(x|y)] \geq n - O(1).$$

- *For any x, y, we have $\mathrm{K}(x|y) \leq \mathrm{K}(x) + O(1)$.*

We also need the following well-known result regarding Kolmogorov complexity.

Theorem 3.4 (Symmetry of Information [ZL70])**.** *For any $x, y \in \{0,1\}^*$,*

$$\left| \mathrm{K}(xy) - \mathrm{K}(x) - \mathrm{K}(y|x) \right| \leq O(\log |xy|).$$

3.2 Oracle Machines

An oracle machine (either a multitape machine or a RAM) has an additional read-write oracle tape and two special states q_{query} and q_{answer}. To make a query to the oracle $\mathcal{O}$, the machine first writes something to its oracle tape,[7] and then enters the query state q_{query}. Then, at the next time step, the machine enters the answer state q_{answer}, with the content of the oracle tape replaced by $\mathcal{O}(y)$, where y is the content of the oracle tape upon entering q_{query}.

We say the machine makes a query w at time step t if, upon entering the state q_{query} at time step t, the content of the oracle tape is w. Crucially, at time t the machine *does not know* the answer to the query w —it only receives the answer at time step $t+1$ when it enters q_{answer}.

4 Separation of RAM and Multitape Turing Machines

In this section, we prove Theorem 1.1.

4.1 The Hard Function

We first define the hard function.

Definition 4.1 (The hard function)**.** *Let $n, \ell \in \mathbb{Z}^+$ and let $\mathcal{O} : \{0,1\}^{3\ell} \to [n]^3$ be an oracle. The function $\mathsf{Hard}^{\mathcal{O}}_{n,\ell} : (\{0,1\}^\ell)^n \to \{0,1\}$ is defined by the following procedure:*

Algorithm 1 Definition of $\mathsf{Hard}^{\mathcal{O}}_{n,\ell}$

1: **Input:** $x_1, x_2, \ldots, x_n$ with $x_i \in \{0,1\}^\ell$.
2: Initialize $(i_{1,1}, i_{1,2}, i_{1,3}) \leftarrow (1,1,1)$.
3: **for** $k = 2$ to n **do**
4: $\quad (i_{k,1}, i_{k,2}, i_{k,3}) \leftarrow \mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$.
5: **end for**
6: **Output:** The first bit of $i_{n,1}$.

[7] In this paper, we assume the oracle tape has a fixed length $\Theta(\log n)$.

For brevity, we will often write $\mathsf{Hard}^{\mathcal{O}}$ when n and ℓ are clear from context.

The following proposition follows immediately by running the algorithm in Algorithm 1 on a RAM.

Proposition 4.2 (Easy for RAMs)**.** *There is an oracle RAM computing* $\mathsf{Hard}^{\mathcal{O}}_{n,\,10\log n}$ *in* $O(|x|)$ *time.*

4.2 Multitape Turing Machine Lower Bound

We are now ready to prove the multitape Turing machine lower bound.

Theorem 4.3 (Formal version of Theorem 1.1)**.** *Let* M *be an oracle multitape Turing machine running in time* $n^2/(\log n)^{\omega(1)}$*. For sufficiently large* n*, we have*

$$\Pr_{\mathcal{O},x}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}_{n,10\log n}(x)\right] \le 1/2 + o(1),$$

where the probability is over a uniformly random oracle $\mathcal{O} : \{0,1\}^{30\log n} \to [n]^3$ *and a uniformly random input* $x \sim \{0,1\}^{10n\log n}$*.*

Proof.

Let $\mathrm{time}_M := n^2/(\log n)^{\omega(1)}$ be the upper bound of M's running time on x (i.e., the maximum number of steps that M takes on x before it halts).

We restrict attention to inputs x of (almost) maximal Kolmogorov complexity. Specifically, fix $x \in \{0,1\}^{10n\log n}$ with

$$\mathrm{K}(x) \ge 10n\log n - \log n.$$

We will show that for such x,

$$\Pr_{\mathcal{O}}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}_{n,10\log n}(x)\right] \le 1/2 + o(1). \tag{1}$$

Since almost all (more precisely, $\ge 1 - o(1)$ fraction of) inputs satisfy the above complexity bound, the theorem follows directly.

x_i-s are pairwise distinct. It is easy to show that if $x = x_1x_2\cdots x_n$ has Kolmogorov complexity $\mathrm{K}(x) \ge 10n\log n - \log n$, then all x_i-s ($1 \le i \le n$) are pairwise distinct. In fact, if $x_p = x_q$ for some $p < q$, we obtain the following short description of x: encode x into a string $x_1\cdots x_{q-2}x_{q-1}x_{q+1}x_{q+2}\cdots x_n$ and a tuple $(p,q) \in [n]^2$. Since each x_i has length $10\log n$, we have $\mathrm{K}(x) \le (n-1)\cdot 10\log n + 2\log n + O(\log\log n) < 10n\log n - \log n$, which leads to contradiction.

Notations. From now on, we only consider a fixed input x and oracle TM M, and we only consider the randomness over the random oracle $\mathcal{O}$. Now the tuples $(i_{1,1}, i_{1,2}, i_{1,3}), \cdots, (i_{n,1}, i_{n,2}, i_{n,3})$ in the definition of Hard (Definition 4.1) are random variables that are functions on $\mathcal{O}$.

Let's define some more random variables:

- For $t = 0, 1, \cdots, \mathrm{time}_M$, let Hist_t be M's computation history by time t (including the t-th step). Specially, if M enters q_{query} at the t-th step and enters q_{answer} at the $(t+1)$-th step, then Hist_t contains the query string y but not the answer $\mathcal{O}(y)$, while Hist_{t+1} contains the answer $\mathcal{O}(y)$. Let $\mathsf{Hist}_{+\infty}$ be M's full computation history.

- Define stopping times $0 = T_0 < T_1 < \cdots < T_{n-1}$ as follows: for $1 \le k \le n-1$, let T_k be the first time step M queries $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ after T_{k-1}. (M enters q_{query} at the T_k-th step and enters q_{answer} at the (T_k+1)-th step. Thus, Hist_{T_k} contains the value of $(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ but not the value of $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$, while Hist_{T_k+1} contains the value of $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$.) If M never makes that query, set $T_k := +\infty$.
- For $k = 1, 2, \cdots, n-1$, let Good_k denote the event that M has not queried $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ before T_k. It is clear that Good_k is a function on Hist_{T_k}, since the value of $(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ is revealed in Hist_{T_k}, and Hist_{T_k} contains all query histories before T_k. Let $\mathsf{Good} := \mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \cdots \wedge \mathsf{Good}_{n-1}$. (Intuitively, Good captures that M made all queries in the correct order.)

We remark that all Hist_t ($0 \le t \le \mathrm{time}_M$), $\mathsf{Hist}_{+\infty}$, T_k ($0 \le k \le n-1$), Good_k ($1 \le k \le n-1$), Good are random variables that are functions of $\mathcal{O}$ (since M and x are fixed).

Our proof is divided into proving the following three claims:

Claim 4.3.1.
$$\Pr_{\mathcal{O}}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}(x) \middle| (T_{n-1} = +\infty) \wedge \mathsf{Good}\right] \le 1/2 + o(1).$$

Claim 4.3.2.
$$\Pr_{\mathcal{O}}\left[\neg\mathsf{Good}\right] \le o(1).$$

Claim 4.3.3. *For any $p = 2, 3, \cdots, n-1$, we have*
$$\Pr_{\mathcal{O}}\left[(T_p - T_{p-1} \le n) \wedge (T_{p-1} < +\infty) \wedge \mathsf{Good}_{p-1}\right] \le o(1).$$

Intuitively, Claims 4.3.1, 4.3.2 state that M cannot compute Hard without making all queries, and with high probability, M makes queries in order. Claim 4.3.3 is the core claim, showing that the expected gaps between $T_0, T_1, \cdots, T_{n-1}$ are large. We first show why Claims 4.3.1–4.3.3 imply Theorem 4.3, then we prove Claim 4.3.3 (the core step), and finally we prove Claims 4.3.1, 4.3.2.

Proof of Theorem 4.3 assuming Claims 4.3.1–4.3.3.

By Claim 4.3.1, to prove
$$\Pr_{\mathcal{O}}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}_{n,10\log n}(x)\right] \le 1/2 + o(1),$$
it suffices to show $\Pr_{\mathcal{O}}\left[(T_{n-1} = +\infty) \wedge \mathsf{Good}\right] \ge 1 - o(1)$. By Claim 4.3.2, we then only need to show $\Pr_{\mathcal{O}}\left[(T_{n-1} < +\infty) \wedge \mathsf{Good}\right] \le o(1)$.

Note that Claim 4.3.3 gives that for any $2 \le p \le n-1$, we have
$$\Pr_{\mathcal{O}}\left[(T_p - T_{p-1} \le n) \wedge (T_{n-1} < +\infty) \wedge \mathsf{Good}\right] \le o(1).$$

Also note that when $(T_{n-1} < +\infty) \wedge \mathsf{Good}$ holds, we have $T_{n-1} \le n^2/(\log n)^{\omega(1)}$ from our assumption on the running time of M, and thus the fraction of p-s ($2 \le p \le n-1$) such that $T_p - T_{p-1} \le n$ is at least $(1 - o(1))$. So there exists $p^* \in \{2, 3, \cdots, n-1\}$ such that
$$\Pr_{\mathcal{O}}\left[T_{p^*} - T_{p^*-1} \le n \middle| (T_{n-1} < +\infty) \wedge \mathsf{Good}\right] \ge 1 - o(1).$$

Since Claim 4.3.3 shows
$$\Pr_{\mathcal{O}}\left[(T_{p^*} - T_{p^*-1} \le n) \wedge (T_{n-1} < +\infty) \wedge \mathsf{Good}\right] \le o(1),$$
we have $\Pr_{\mathcal{O}}\left[(T_{n-1} < +\infty) \wedge \mathsf{Good}\right] \le o(1)$.

□

Now, we present the core step, showing that the expected gaps between $T_0, T_1, \cdots, T_{n-1}$ are large.

Proof of Claim 4.3.3.

Denote the head state and the cells on the tapes of M that are within distance $\leq n$ of the heads at time T_{p-1} by Tape (it also includes the cells close to the corresponding head at the input tape of M). We have $\mathrm{K}(\mathsf{Tape}) \leq O(n)$. We remark that Tape is a random variable that is a function of $\mathcal{O}$. We also let

$$\mathcal{E}_{p-1} \coloneqq [(T_{p-1} < +\infty) \wedge \mathsf{Good}_{p-1}]$$

be the event that we will consider during the proof. It is easy to see that $\mathcal{E}_{p-1}$ and Tape are determined by $\mathsf{Hist}_{T_{p-1}}$.

$x_{i_{p,1}}$ can be easily extracted from Tape and $\mathcal{O}$ when $T_p - T_{p-1} \leq n$. Note that if $T_p - T_{p-1} \leq n$, then given Tape and $\mathcal{O}$, we can simulate the subsequent steps of M, and at time T_p (which is at most n steps later), the oracle query tape is $x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}}$. Thus, given Tape and $\mathcal{O}$, and also the description of M and the value of $T_p - T_{p-1}$ (which is $\leq n$), we can recover $x_{i_{p,1}}$. Therefore,

$$T_p - T_{p-1} \leq n \implies \mathrm{K}(x_{i_{p,1}} | \mathsf{Tape}, \mathcal{O}) \leq \log n + O(1) \leq 2 \log n. \tag{2}$$

Bounding the description length of a random x_a given Tape and $\mathcal{O}$. Note that $i_{p,1}$ is determined by Tape and $\mathcal{O}$. Next, we show that if $T_p - T_{p-1} \leq n$ occurs with high probability, then the same is true even for a uniformly random index $a \sim [n]$. In particular, we show that the probability that $\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O})$ is small is at least the probability that $\mathrm{K}(x_{i_{p,1}} | \mathsf{Tape}, \mathcal{O})$ is small.

Let $(a, b, c) \sim [n]^3$ be uniformly random and independent of $\mathcal{O}$. We prove that

$$\Pr_{\mathcal{O}} \left[(\mathrm{K}(x_{i_{p,1}} | \mathsf{Tape}, \mathcal{O}) \leq 2 \log n) \wedge \mathcal{E}_{p-1} \right] \leq \Pr_{\mathcal{O}, a, b, c} \left[(\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}) \leq 6 \log n) \wedge \mathcal{E}_{p-1} \right] \tag{3}$$

To show (3), consider another "punctured" random oracle $\mathcal{O}' \colon \{0,1\}^{30 \log n} \to [n]^3$ defined as follows: $\mathcal{O}'$ is identical to $\mathcal{O}$ except for the value of $\mathcal{O}'(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}})$: we set $\mathcal{O}'(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}}) := (a, b, c)$. Note that conditioned on $\mathsf{Hist}_{T_{p-1}}$, if $T_{p-1} < +\infty$ and Good_{p-1}, then $\mathcal{O}(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}})$ has not been queried in $\mathsf{Hist}_{T_{p-1}}$ and is distributed uniformly at random. So conditioned on $\mathsf{Hist}_{T_{p-1}}$, it holds that the two tuples $(x_{i_{p,1}}, \mathsf{Tape}, \mathcal{O})$ and $(x_a, \mathsf{Tape}, \mathcal{O}')$ have the same distribution, this further implies that $\mathrm{K}(x_{i_{p,1}} | \mathsf{Tape}, \mathcal{O})$ and $\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}')$ have the same distribution. Taking the expectation over $\mathsf{Hist}_{T_{p-1}}$ we get

$$\Pr_{\mathcal{O}} \left[(\mathrm{K}(x_{i_{p,1}} | \mathsf{Tape}, \mathcal{O}) \leq 2 \log n) \wedge \mathcal{E}_{p-1} \right] = \Pr_{\mathcal{O}, a, b, c} \left[(\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}') \leq 2 \log n) \wedge \mathcal{E}_{p-1} \right]. \tag{4}$$

Also note that

$$\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}) \leq \mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}') + 3 \log n + o(\log n). \tag{5}$$

This is because given $\mathsf{Tape}, \mathcal{O}$, to compute $\mathcal{O}'$, we only need the value of (a, b, c), which has length $3 \log n$, and $(x_{i_{p-1,1}}, x_{i_{p-1,2}}, x_{i_{p-1,3}})$, which can be directly retrieved from Tape. Combining Equations (4) and (5) we get (3).

Now combining Equations (2) and (3), we have

$$\Pr_{\mathcal{O}} \left[(T_p - T_{p-1} \leq n) \wedge \mathcal{E}_{p-1} \right] \leq \Pr_{\mathcal{O}, a} \left[(\mathrm{K}(x_a | \mathsf{Tape}, \mathcal{O}) \leq 6 \log n) \wedge \mathcal{E}_{p-1} \right]. \tag{6}$$

Bounding the description length of x given $\mathcal{O}$. Note that for any $\mathsf{Tape}, \mathcal{O}$, we have

$$\begin{aligned}\mathrm{K}(x|\mathsf{Tape}, \mathcal{O}) &\leq \sum_{a\in[n]} \mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) + O(n \log\log n) \\ &\leq 10n\log n - 4\log n \cdot |\{a \in [n] \colon \mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) \leq 6\log n\}| + O(n\log\log n).\end{aligned}$$

To see the first inequality, given the descriptions of n machines $M_1, \ldots, M_n$ such that M_a outputs x_a given $\mathsf{Tape}, \mathcal{O}$, we can combine them into a single machine that outputs $x = (x_1, \ldots, x_n)$ given $\mathsf{Tape}, \mathcal{O}$ with an additional $O(n \log\log n)$ bits specifying the lengths of each machine M_i. The second inequality uses the fact that $\mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) \leq |x_a| + O(1) \leq 10\log n + O(1)$.

Also note that since $\mathrm{K}(\mathsf{Tape}) \leq O(n)$, we have

$$\begin{aligned}\mathrm{K}(x|\mathcal{O}) &\leq \mathrm{K}(x|\mathsf{Tape}, \mathcal{O}) + O(n) \\ &\leq 10n\log n - 4\log n \cdot |\{a \in [n] \colon \mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) \leq 6\log n\}| + O(n\log\log n). \qquad (7)\end{aligned}$$

Below we consider the following quantity:

$$\mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}])\right].$$

We give an upper bound using Equations (6) and (7) and a lower bound using Theorem 3.4, and putting them together proves Claim 4.3.3.

Upper Bounding $\mathbb{E}_{\mathcal{O}}\left[(\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}])\right]$ using Equations (6) and (7). In Equation (7), multiplying both side by $\mathbb{I}[\mathcal{E}_{p-1}]$ taking the expectation over $\mathcal{O}$, we get

$$\begin{aligned}&\mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}])\right] \\ &\leq 10n\log n \cdot \Pr_{\mathcal{O}}\left[\mathcal{E}_{p-1}\right] - 4n\log n \cdot \Pr_{\mathcal{O},a}\left[(\mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) \leq 6\log n) \wedge \mathcal{E}_{p-1}\right] + O(n\log\log n).\end{aligned}$$

Combining with Equation (6) (a lower bound of $\Pr_{\mathcal{O},a}\left[(\mathrm{K}(x_a|\mathsf{Tape}, \mathcal{O}) \leq 6\log n) \wedge \mathcal{E}_{p-1}\right]$), we get

$$\begin{aligned}&\mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}])\right] \\ &\leq 10n\log n \cdot \Pr_{\mathcal{O}}\left[\mathcal{E}_{p-1}\right] - 4n\log n \cdot \Pr_{\mathcal{O}}\left[(T_p - T_{p-1} \leq n) \wedge \mathcal{E}_{p-1}\right] + O(n\log\log n). \qquad (8)\end{aligned}$$

Lower Bounding $\mathbb{E}_{\mathcal{O}}\left[(\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}])\right]$ using Theorem 3.4 (Symmetry of Information) By Theorem 3.4 (Symmetry of Information), we have

$$\mathop{\mathbb{E}}_{\mathcal{O}}\left[\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}]\right] \geq \mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(\mathcal{O}|x) + \mathrm{K}(x) - \mathrm{K}(\mathcal{O}) - O(\log n)) \cdot \mathbb{I}[\mathcal{E}_{p-1}]\right].$$

For each part in the above equation:

- **The $\mathrm{K}(\mathcal{O}|x) - \mathrm{K}(\mathcal{O})$ part:** Note that $\mathrm{K}(\mathcal{O}|x) - \mathrm{K}(\mathcal{O}) \leq O(1)$, so

  $$\mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(\mathcal{O}|x) - \mathrm{K}(\mathcal{O})) \cdot \mathbb{I}[\mathcal{E}_{p-1}]\right] \geq \mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(\mathcal{O}|x) - \mathrm{K}(\mathcal{O}))\right] - O(1).$$

  By Proposition 3.3, we have $\mathbb{E}_{\mathcal{O}}\left[\mathrm{K}(\mathcal{O}|x)\right] \geq |\mathcal{O}| - O(1)$, and for any $\mathcal{O}$, $\mathrm{K}(\mathcal{O}) \leq |\mathcal{O}| + O(1)$, where $|\mathcal{O}| = n^{30} \cdot 3\log n$ is the length of $\mathcal{O}$ when encoded as a binary string. So we have

  $$\mathop{\mathbb{E}}_{\mathcal{O}}\left[(\mathrm{K}(\mathcal{O}|x) - \mathrm{K}(\mathcal{O})) \cdot \mathbb{I}[\mathcal{E}_{p-1}]\right] \geq -O(1).$$

- **The $\mathrm{K}(x)$ and $-O(\log n)$ parts:** From our assumption that x has almost maximal Kolmogorov complexity, we have $\mathrm{K}(x) \geq 10n \log n - \log n$. So

$$\mathop{\mathbb{E}}_{\mathcal{O}}[(\mathrm{K}(x) - O(\log n)) \cdot \mathbb{I}[\mathcal{E}_{p-1}]] \geq (10n \log n - O(\log n)) \cdot \Pr_{\mathcal{O}}[\mathcal{E}_{p-1}]$$

So we have

$$\mathop{\mathbb{E}}_{\mathcal{O}}[\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}]] \geq (10n \log n - O(\log n)) \cdot \Pr_{\mathcal{O}}[\mathcal{E}_{p-1}] - O(1). \tag{9}$$

Combining these two bounds about $\mathbb{E}_{\mathcal{O}}[\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}]]$ (Equations (8) and (9)), we have

$$\begin{aligned}
&(10n \log n - O(\log n)) \cdot \Pr_{\mathcal{O}}[\mathcal{E}_{p-1}] - O(1) \\
\leq &\mathop{\mathbb{E}}_{\mathcal{O}}[\mathrm{K}(x|\mathcal{O}) \cdot \mathbb{I}[\mathcal{E}_{p-1}]] \\
\leq &\, 10n \log n \cdot \Pr_{\mathcal{O}}[\mathcal{E}_{p-1}] - 4n \log n \cdot \Pr_{\mathcal{O}}[(T_p - T_{p-1} \leq n) \wedge \mathcal{E}_{p-1}] + O(n \log \log n),
\end{aligned}$$

so

$$\Pr_{\mathcal{O}}[(T_p - T_{p-1} \leq n) \wedge \mathcal{E}_{p-1}] \leq o(1). \qquad \square$$

Finally, we give the missing proofs of Claims 4.3.1, 4.3.2:

Proof of Claim 4.3.1.

Consider any fixed value H^* of $\mathsf{Hist}_{+\infty}$ such that $T_{n-1} = +\infty$ and Good holds. (It is easy to see that whether $T_{n-1} = +\infty$ and Good holds is determined by $\mathsf{Hist}_{+\infty}$.) We prove that conditioned on $\mathsf{Hist}_{+\infty} = \mathsf{H}^*$, the probability that $\mathsf{Hard}^{\mathcal{O}}(x) = 0$ is $1/2 \pm o(1)$, and thus, no matter what M outputs, the probability that $M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}(x)$ is at most $1/2 + o(1)$.

Let

$$S := \{(a, b, c) \colon M \text{ has queried } \mathcal{O}(x_a, x_b, x_c) \text{ in } \mathsf{H}^*\}.$$

Since M runs in time $\mathrm{time}_M = n^2/(\log n)^{\omega(1)}$, and all x_i-s are pairwise distinct, we have $|S| \leq n^2/(\log n)^{\omega(1)}$.

Since $T_0 = 0$ and $T_{n-1} = +\infty$, there must exist $1 \leq p \leq n-1$ such that $T_{p-1} < +\infty$ and $T_p = +\infty$. Then M has queried all $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ for $k = 1, 2, \cdots, p-1$, but has never queried $\mathcal{O}(x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}})$.

The remaining pointer chasing after p is unlikely to run into a collision. We prove that with high probability, the remaining pointer chasing procedure that is not queried:

$$(i_{p,1}, i_{p,2}, i_{p,3}) \to (i_{p+1,1}, i_{p+1,2}, i_{p+1,3}) \to \cdots \to (i_{n-1,1}, i_{n-1,2}, i_{n-1,3}),$$

does not go back into S or run into a cycle. More precisely, we use induction to prove that:

For $q = p, p+1, \cdots, n-1$, the following holds: conditioned on $\mathsf{Hist}_{+\infty} = \mathsf{H}^*$, with probability $\geq (1 - (|S| + n)/n^3)^{q-p}$, all triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, q)$ are not in S and are pairwise distinct.

- The case $q = p$:

  Because Good holds, $T_{p-1} < +\infty$, and $T_p = +\infty$, we know that M has never queried $\mathcal{O}(x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}})$. Therefore, $(i_{p,1}, i_{p,2}, i_{p,3}) \notin S$.

- The case $q > p$, assuming that the result holds for $q-1$:

  Consider conditioning on a fixed sequence of

  $$(i_{p,1}, i_{p,2}, i_{p,3}), (i_{p+1,1}, i_{p+1,2}, i_{p+1,3}), \cdots, (i_{q-1,1}, i_{q-1,2}, i_{q-1,3}),$$

  as well as $\mathcal{O}$'s corresponding entries except for the last one,[8] such that none of them is in S and they are pairwise distinct. Then the value of $\mathcal{O}(x_{i_{q-1,1}}, x_{i_{q-1,2}}, x_{i_{q-1,3}})$ is uniformly random. So the probability that $(i_{q,1}, i_{q,2}, i_{q,3})$ is in S or is equal to one of $(i_{k,1}, i_{k,2}, i_{k,3})$ $(p \le k \le q-1)$ is $\le (|S|+q-p)/n^3 \le (|S|+n)/n^3$.

  Also note that the induction hypothesis gives that with probability $\ge (1-(|S|+n)/n^3)^{q-p-1}$, all triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, q-1)$ are not in S and are pairwise distinct. Therefore, with probability $\ge (1-(|S|+n)/n^3)^{q-p}$, all triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, q)$ are not in S and are pairwise distinct.

Therefore, with probability $\ge (1-(|S|+n)/n^3)^{n-p-1}$, all triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, n-1)$ are not in S and are pairwise distinct.

The distribution of the final $(i_{n,1}, i_{n,2}, i_{n,3})$**.** Note that conditioned on $\mathsf{Hist}_{+\infty} = \mathsf{H}^*$ and a fixed sequence of triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, n-1)$ with the above property, (all triples $(i_{k,1}, i_{k,2}, i_{k,3})$ $(k = p, p+1, \cdots, n-1)$ are not in S and are pairwise distinct,) the value of $(i_{n,1}, i_{n,2}, i_{n,3}) := \mathcal{O}(x_{i_{n-1,1}}, x_{i_{n-1,2}}, x_{i_{n-1,3}})$ is uniformly random, and thus, $\mathsf{Hard}^{\mathcal{O}}(x)$ is equally likely to be 0 or 1. Also note that

$$(1-(|S|+n)/n^3)^{n-p-1} \ge (1-1/(n(\log n)^{\omega(1)}))^n \ge 1-o(1),$$

hence the probability that $\mathsf{Hard}^{\mathcal{O}}(x) = 0$ is $1/2 \pm o(1)$. Therefore,

$$\Pr_{\mathcal{O}}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}(x) \middle| \mathsf{Hist}_{+\infty} = \mathsf{H}^*\right] \le 1/2 + o(1).$$

Taking the expectation over $\mathsf{Hist}_{+\infty}$ we get

$$\Pr_{\mathcal{O}}\left[M^{\mathcal{O}}(x) = \mathsf{Hard}^{\mathcal{O}}(x) \middle| (T_{n-1} = +\infty) \wedge \mathsf{Good}\right] \le 1/2 + o(1). \qquad \square$$

Proof of Claim 4.3.2.

If $\neg\mathsf{Good}$, then there exist $1 \le k \le n-1$ and $1 \le t \le \min\{T_k - 1, \mathsf{time}_M\}$, such that $\mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \cdots \wedge \mathsf{Good}_{k-1} \wedge \neg\mathsf{Good}_k$ holds, and M queried $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ at time t. Since T_k is the first time M queries $\mathcal{O}(x_{i_{k,1}}, x_{i_{k,2}}, x_{i_{k,3}})$ after T_{k-1}, we have $t \le T_{k-1}$.

Now, consider any fixed k, t and any fixed value H^* of $\mathsf{Hist}_{T_{k-1}}$ such that $\mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \cdots \wedge \mathsf{Good}_{k-1}$ holds. Recall that $\mathsf{Hist}_{T_{k-1}}$ does not contain the result of $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$, and whether $\mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \cdots \wedge \mathsf{Good}_{k-1}$ holds is determined by $\mathsf{Hist}_{T_{k-1}}$. Note that $\neg\mathsf{Good}$ holding with respect to k, t is equivalent to the event that the result of $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$ exactly matches M's query at time t. Since the result of $\mathcal{O}(x_{i_{k-1,1}}, x_{i_{k-1,2}}, x_{i_{k-1,3}})$ is uniformly random conditioned on $\mathsf{Hist}_{T_{k-1}} = \mathsf{H}^*$, and M's query at time t is fixed conditioned on $\mathsf{Hist}_{T_{k-1}} = \mathsf{H}^*$, we have

$$\Pr_{\mathcal{O}}\left[\neg\mathsf{Good} \text{ w.r.t. } k, t \middle| \mathsf{Hist}_{T_{k-1}} = \mathsf{H}^*\right] \le 1/n^3.$$

Taking the expectation over $\mathsf{Hist}_{T_{k-1}}$ and the sum over k, t, where $1 \le k \le n-1$ and $1 \le t \le \mathrm{time}_M$, we get

$$\Pr_{\mathcal{O}}[\neg\mathsf{Good}] \le n \cdot \mathrm{time}_M \cdot 1/n^3 \le o(1). \qquad \square$$

$\square$

[8] i.e., $\mathcal{O}(x_{i_{p,1}}, x_{i_{p,2}}, x_{i_{p,3}}), \mathcal{O}(x_{i_{p+1,1}}, x_{i_{p+1,2}}, x_{i_{p+1,3}}), \cdots, \mathcal{O}(x_{i_{q-2,1}}, x_{i_{q-2,2}}, x_{i_{q-2,3}})$

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[CHO+22] Lijie Chen, Shuichi Hirahara, Igor Carboni Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. Beyond natural proofs: Hardness magnification and locality. *ACM Journal of the ACM (JACM)*, 69(4):1–49, 2022.

[CM24] James Cook and Ian Mertz. Tree evaluation is in space O(log n · log log n). In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1268–1278. ACM, 2024.

[CR73] Stephen A. Cook and Robert A. Reckhow. Time bounded random access machines. *J. Comput. Syst. Sci.*, 7(4):354–375, 1973.

[GS89] Yuri Gurevich and Saharon Shelah. Nearly linear time. In *International Symposium on Logical Foundations of Computer Science*, pages 108–118. Springer, 1989.

[PPST83] Wolfgang J. Paul, Nicholas Pippenger, Endre Szemerédi, and William T. Trotter. On determinism versus non-determinism and related problems (preliminary version). In *24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, USA, 7-9 November 1983*, pages 429–438. IEEE Computer Society, 1983.

[SvEB88] Cees Slot and Peter van Emde Boas. The problem of space invariance for sequential machines. *Information and Computation*, 77(2):93–122, 1988.

[vEB90] Peter van Emde Boas. Machine models and simulations. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 1–66. Elsevier and MIT Press, 1990.

[Wil09] Ryan Williams. Answer to: Super-linear time complexity lower bounds for any natural problem in NP? MathOverflow, December 2009. Answer posted Dec 16, 2009.

[Wil25] R. Ryan Williams. Simulating time with square-root space. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 13–23. ACM, 2025.

[Yao89] Andrew C Yao. Circuits and local computation. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 186–196, 1989.

[ZL70] Alexander K Zvonkin and Leonid A Levin. The complexity of finite objects and the algorithmic concepts of randomness and information. *UMN (Russian Math. Surveys)*, 25(6):83–124, 1970.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il