Can Verifiable Delay Functions be Based on Random Oracles?

Mohammad Mahmoody, Caleb Smith, and David J. Wu

ICALP 2020

Verifiable Delay Functions (VDF)

[BBBF18]

a deterministic function that is *slow* to compute, but *fast* to verify

time bound T

$$\mathrm{Setup}\left(1^{\lambda}, T\right) \to \mathrm{pp}$$

deterministic value y

$$\mathrm{Eval}(\mathrm{pp}, x) \to (y, \pi)$$

runs in time T

π can be randomized

$$\mathrm{Verify}(\mathrm{pp}, x, y, \pi) \to 0/1$$

runs in time $t = \mathrm{poly}(\lambda, \log T)$

Completeness:

$$\mathrm{Verify}(\mathrm{pp}, x, y, \pi) = 1$$

Uniqueness: no adversaries running in time $\mathrm{poly}(\lambda, T)$ can find (y', π') such that

$$\mathrm{Verify}(\mathrm{pp}, x, y', \pi') = 1$$

and

$$y' \neq \mathrm{Eval}(\mathrm{pp}, x)$$

Sequentiality: no adversary running in <u>parallel</u> time $\sigma \ll T$ can compute y where

$$y = \mathrm{Eval}(\mathrm{pp}, x)$$

Verifiable Delay Functions (VDF)

[BBBF18]

a deterministic function that is *slow* to compute, but *fast* to verify

time bound T

$\mathrm{Setup}(1^\lambda, T) \to \mathrm{pp}$

deterministic value y

$\mathrm{Eval}(\mathrm{pp}, x) \to (y, \pi)$

runs in time T

π can be randomized

$\mathrm{Verify}(\mathrm{pp}, x, y, \pi) \to 0/1$

runs in time $t = \mathrm{poly}(\lambda, \log T)$

Many applications:

- randomness beacons
- proofs of replication
- computational timestamping

Many constructions:

- groups of unknown order [Pie19, Wes19]
- incremental verifiable computation [BBBF18]
- pairings/isogenies [FMPS19, Sha19]

All of these constructions
rely on algebraic structure

Verifiable Delay Functions (VDF)

[BBBF18]

a deterministic function that is *slow* to compute, but *fast* to verify

$$\mathrm{Setup}\left(1^{\lambda}, T\right) \to \mathrm{pp}$$

$$\mathrm{Eval}(\mathrm{pp}, x) \to (y, \pi)$$

$$\mathrm{Verify}(\mathrm{pp}, x, y, \pi) \to 0/1$$

Many applications:

- randomness beacons
- proofs of replication
- computational timestamping

Many constructions:

- groups of unknown order [Pie19, Wes19]
- incremental verifiable computation [BBBF18]
- pairings/isogenies [FMPS19, Sha19]

Can we construct VDFs from an unstructured assumption?

This Work

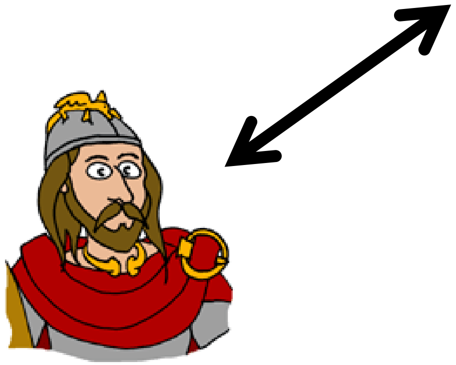
Can we construct VDFs from an <u>unstructured</u> assumption?

e.g., one-way functions,
collision-resistant hash functions

This Work

Can we construct VDFs from an unstructured assumption?

User only has oracle access to $\mathcal{O}$



Can model objects like one-way functions, collision-resistant hash functions by a random oracle

$$\mathcal{O}: \{0,1\}^n \rightarrow \{0,1\}^k$$

Metrics of interest:

- **Time complexity:** number of queries
- **Parallel time complexity:** rounds of queries

This work: *Can we construct VDFs from a random oracle?*

This Work

Can we construct VDFs from an unstructured assumption?

Reason for optimism?

Publicly-verifiable proofs of sequential work (PoSW) do exist in the random oracle model [MMV13]

Proofs of sequential work: VDFs without a uniqueness requirement

Can model objects like one-way functions, collision-resistant hash functions by a random oracle

$$\mathcal{O}: \{0,1\}^n \to \{0,1\}^k$$

Metrics of interest:

- **Time complexity:** number of queries
- **Parallel time**

Random oracle is the only source of hardness

This work: *Can we construct VDFs from a random oracle?*

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several <u>specific</u> settings):

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles (e.g., [LW15, AKKPW19])

Perfect uniqueness: for all $y' \neq \mathrm{Eval}(\mathrm{pp}, x)$ and $\pi \in \{0,1\}^*$, $\mathrm{Verify}(\mathrm{pp}, x, y', \pi) = 0$

"Every input x has at most one output y that verifies"

Computational uniqueness: no efficient adversary running in time $\mathrm{poly}(\lambda, T)$ can find (y', π') with $y' \neq \mathrm{Eval}(\mathrm{pp}, x)$ such that $\mathrm{Verify}(\mathrm{pp}, x, y', \pi) = 1$

"Efficient adversary cannot find different output $y' \neq \mathrm{Eval}(\mathrm{pp}, x)$ that verifies"

Corollary. "Permutation VDFs" cannot be built from random oracles

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several specific settings):

Theorem. VDFs with perfect uniqueness cannot be based solely on random oracles

(e.g., [LW15, AKKPW19])

Perfect uniqueness: for all $y' \neq \mathrm{Eval}(\mathrm{pp}, x)$ and $\pi \in \{0,1\}^*$, $\mathrm{Verify}(\mathrm{pp}, x, y', \pi) = 0$

"Every input x has at most one output y that verifies"

Permutation VDF: VDF with an efficiently-computable inverse Eval^{-1} (i.e., "reversible" proof of sequential work) [LW15, AKKPW19]

g in time $\mathrm{poly}(\lambda, T)$ can find (y', π')

$y' \neq \mathrm{Eval}(\mathrm{pp}, x)$ *that verifies"*

Corollary. "Permutation VDFs" cannot be built from random oracles

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several <u>specific</u> settings):

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles
(e.g., [LW15, AKKPW19])

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles
(e.g., [DGMV19]) *(lower bound also appeared in concurrent work of [DGMV19])*

Tight sequentiality: parallel adversary running in time $\sigma = T - T^{\rho}$ for $\rho < 1$ cannot find $y = \mathrm{Eval}(\mathrm{pp}, x)$

Sequentiality: parallel adversary running in time $\sigma \ll T$ cannot find $y = \mathrm{Eval}(\mathrm{pp}, x)$

e.g., $\sigma = T/2$ or $\sigma = \sqrt{T}$

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several <u>specific</u> settings):

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

(e.g., [LW15, AKKPW19])

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles

(e.g., [DGMV19]) *(lower bound also appeared in concurrent work of [DGMV19])*

Tight sequentiality: parallel adversary running in time $\sigma = T - T^{\rho}$ for $\rho < 1$ cannot find $y = \mathrm{Eval}(\mathrm{pp}, x)$

Impossibility also extends to tight publicly-verifiable proofs of sequential work (PoSW)

Note: Non-tight PoSWs ($\sigma = T/2$) are known in the random oracle model [MMV13]

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several <u>specific</u> settings):

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

(e.g., [LW15, AKKPW19])

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles

(e.g., [DGMV19]) *(lower bound also appeared in concurrent work of [DGMV19])*

Conclusions:

- Lower bounds exist for certain types of VDFs in the random oracle model
- <u>Non-tight</u> VDFs with <u>computational uniqueness</u> still plausible from random oracles!

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with perfect uniqueness cannot be based solely on random oracles

Argument uses similar ideas as lower bound for time-lock puzzles in the random oracle model [MMV11]

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

x — sequence of T oracle queries made by $\mathrm{Eval}^{\mathcal{O}}(x)$ — $y = \mathrm{Eval}^{\mathcal{O}}(x)$

$$x_1 \to x_2 \to x_3 \to x_4 \to x_5 \to x_6 \to x_7 \to \cdots \to x_T$$

query $\mathcal{O}$ on input x_1 (function of x)

query $\mathcal{O}$ on input x_i (function of x and and outputs of previous queries)

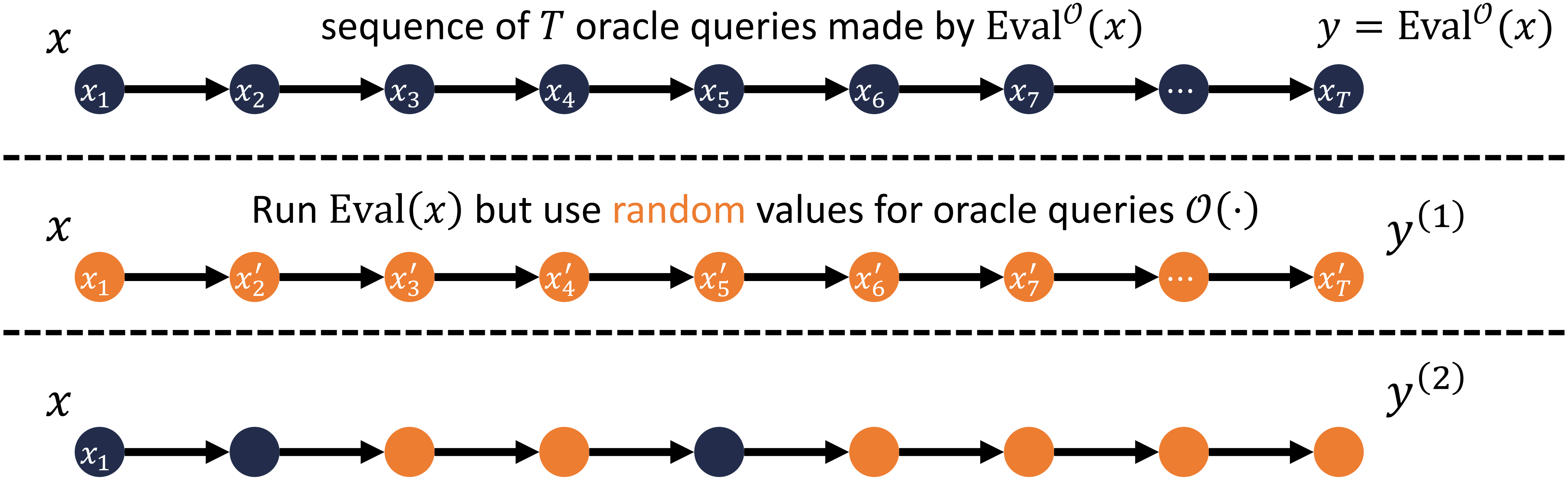
For simplicity, assume there are no public parameters pp or proof π

[Same argument works in general case; see paper for details]

Approach: construct algorithm that uses honest evaluation algorithm, but substitutes "fake" responses for <u>some</u> of the random oracle queries

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

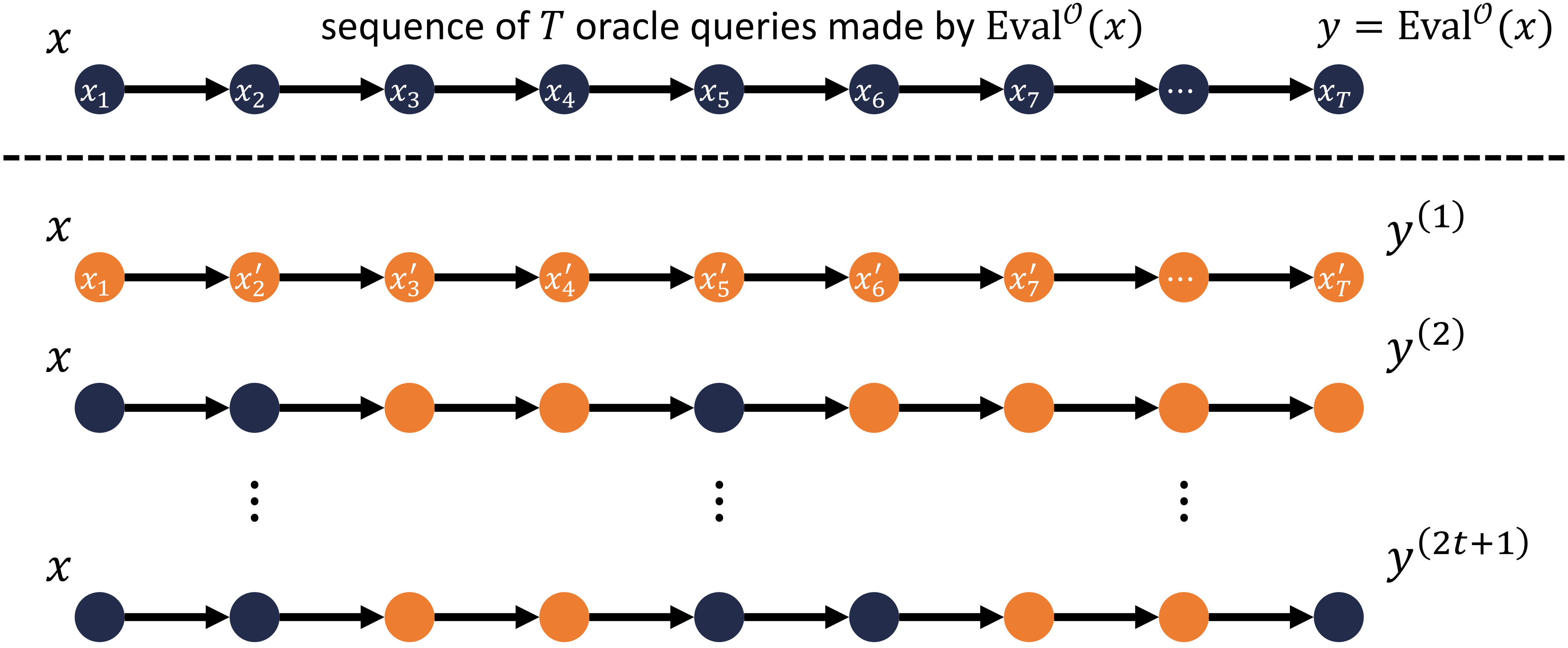


At end of each round, query $\mathcal{O}$ on all x_i appearing in previous round

Use <u>real</u> oracle values for values that have been queried in the past, and <u>random</u> values for the rest

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with perfect uniqueness cannot be based solely on random oracles



Ruling out VDFs with Perfect Uniqueness

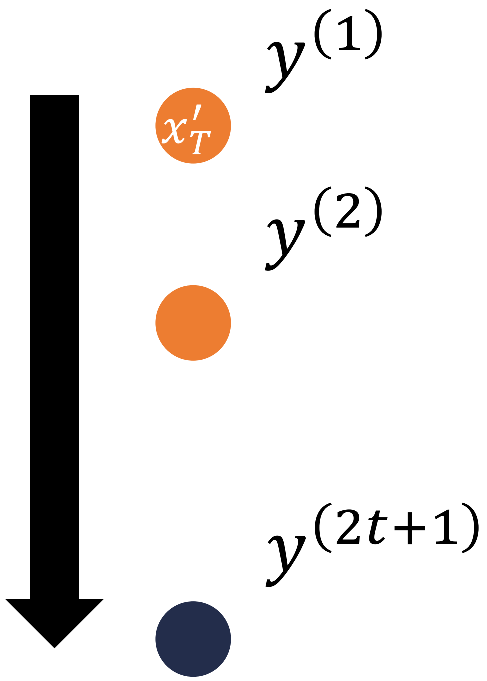
Theorem. VDFs with perfect uniqueness cannot be based solely on random oracles

x sequence of T oracle queries made by $\mathrm{Eval}^{\mathcal{O}}(x)$ $y = \mathrm{Eval}^{\mathcal{O}}(x)$

$$x_1 \to x_2 \to x_3 \to x_4 \to x_5 \to x_6 \to x_7 \to \cdots \to x_T$$

If algorithm succeeds, then
break sequentiality of the VDF

$2t + 1$ rounds of queries
$\leq (2t + 1)T$ queries in total



Output: majority value y' of $y^{(1)}, \ldots, y^{(2t+1)}$

Claim: $y' = \mathrm{Eval}^{\mathcal{O}}(x)$

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

x sequence of T oracle queries made by $\mathrm{Eval}^{\mathcal{O}}(x)$ $y = \mathrm{Eval}^{\mathcal{O}}(x)$

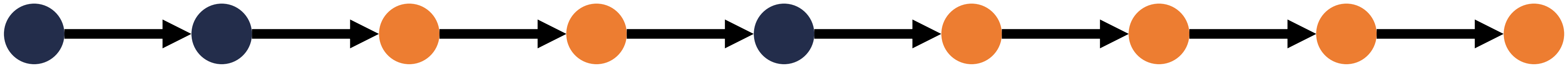
$$x_1 \to x_2 \to x_3 \to x_4 \to x_5 \to x_6 \to x_7 \to \cdots \to x_T$$

Consider $\mathrm{Verify}^{\mathcal{O}}(x, y)$:

sequence of t queries

$$z_1 \to z_2 \to \cdots \to z_t$$

Consider output in round i: $y^{(i)} = \mathrm{Eval}^{\mathcal{O}'}(x)$



Let $\mathcal{O}'$ be an oracle consistent with the above values

Suppose $\mathcal{O}(z_i) = \mathcal{O}'(z_i)$ for all $i \in [t]$

Then, $1 = \mathrm{Verify}^{\mathcal{O}}(x, y) = \mathrm{Verify}^{\mathcal{O}'}(x, y)$

Perfect uniqueness:

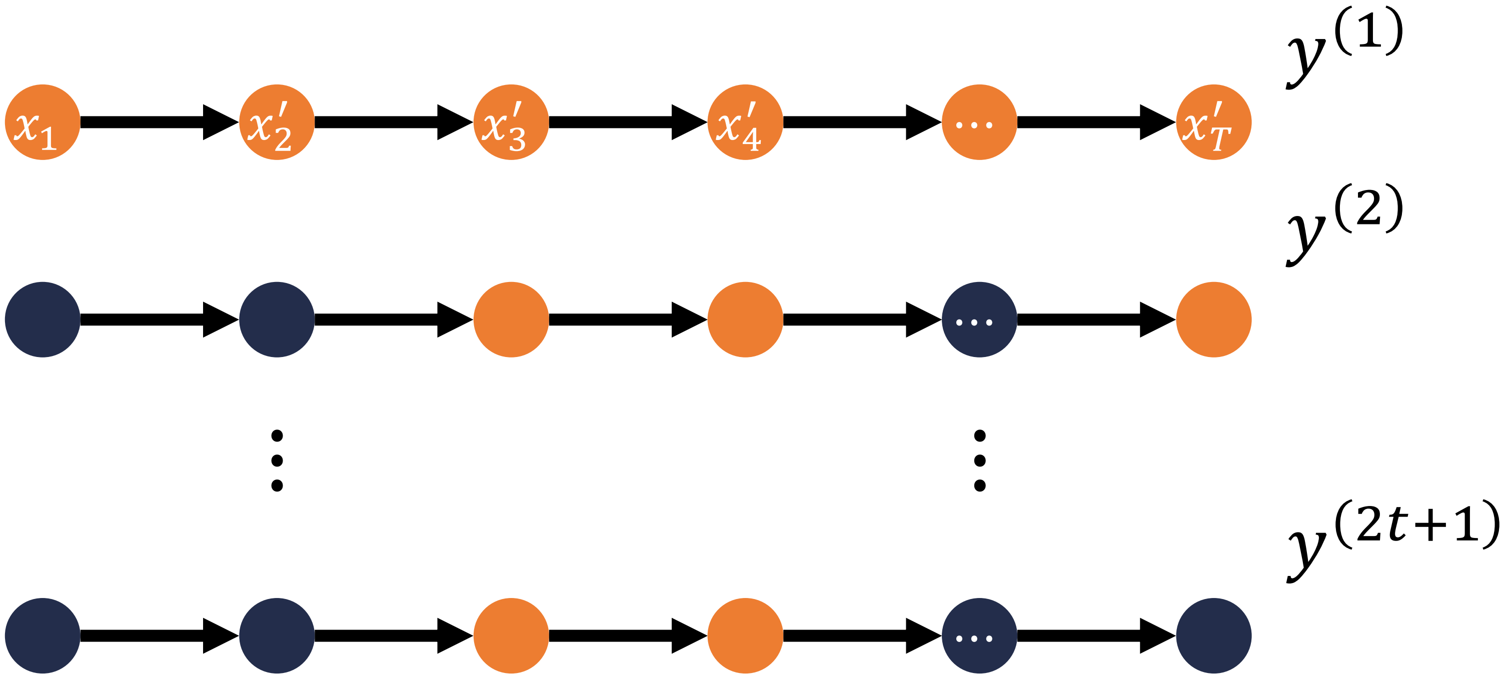
if $y^{(i)} = \mathrm{Eval}^{\mathcal{O}'}(x)$ and $\mathrm{Verify}^{\mathcal{O}'}(x, y) = 1$, then $y = y^{(i)}$

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with perfect uniqueness cannot be based solely on random oracles

If Eval queries z_j in some round, then real value $\mathcal{O}(z_j)$ used in all future rounds

At most t rounds can have incorrect value of z_i



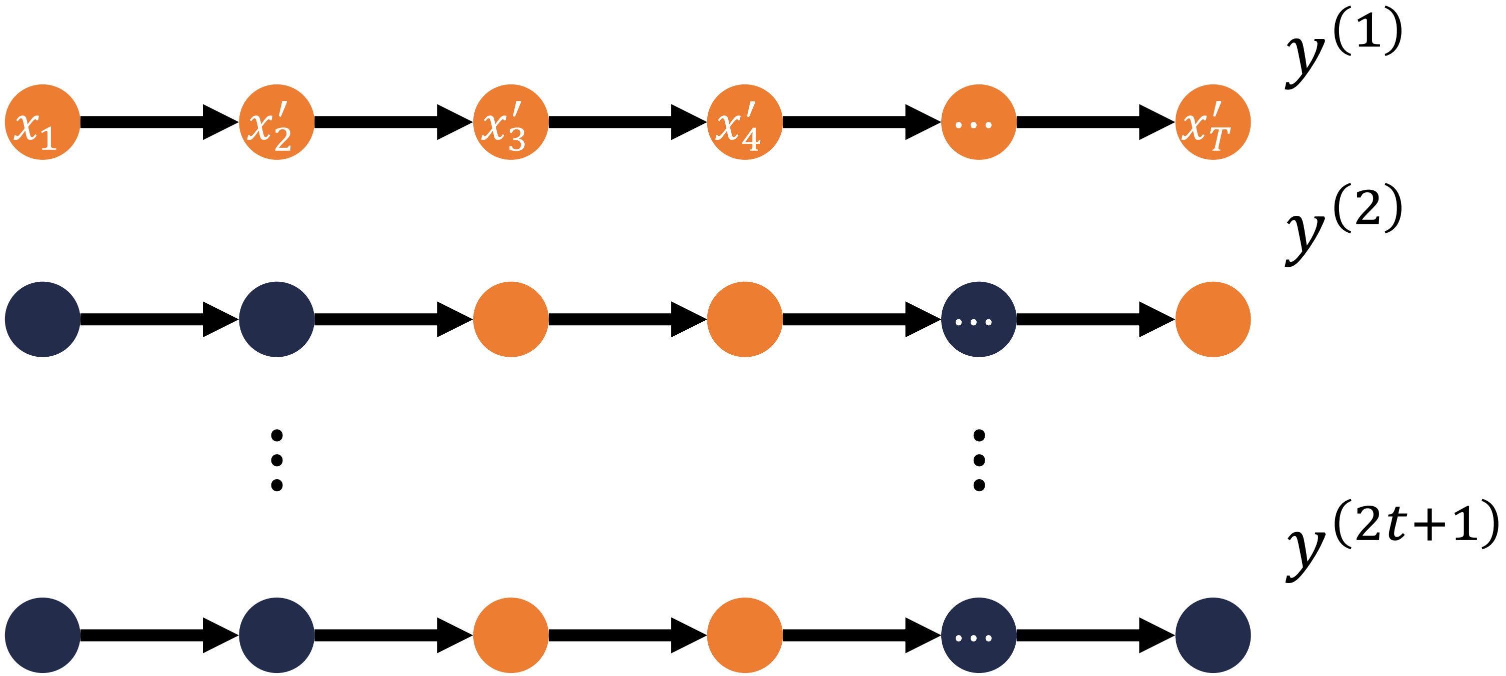
Property: If oracle values agree on $z_1, \dots, z_t$, then $y^{(i)}$ is correct

Ruling out VDFs with Perfect Uniqueness

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles

If Eval queries z_j in some round, then real value $\mathcal{O}(z_j)$ used in <u>all</u> future rounds

At <u>most</u> t rounds can have incorrect value of z_i



There are $2t + 1$ rounds, at most t rounds incorrect ⇒ majority is correct

Ruling out Tight VDFs

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles

x sequence of T oracle queries made by $\mathrm{Eval}^{\mathcal{O}}(x)$ $y = \mathrm{Eval}^{\mathcal{O}}(x)$

$$x_1 \to x_2 \to x_3 \to x_4 \to x_5 \to x_6 \to x_7 \to \cdots \to x_T$$

Tight sequentiality: parallel adversary running in time
$\sigma = T - T^{\rho}$ for $\rho < 1$ cannot find $y = \mathrm{Eval}(\mathrm{pp}, x)$

Ruling out Tight VDFs

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles

Main idea:

- Choose random subset of the queries $S \subseteq [T]$
- Output $\mathrm{Eval}^{\mathcal{O}}(x)$ using $\mathcal{O}$ to answer queries outside S and random values for queries in S

Verification algorithm makes t queries to $\mathcal{O}$:

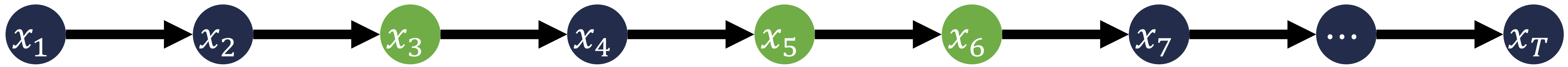
- Important queries that can "affect" verification are those queried by Verify
- With probability $1 - t \cdot |S|/T$, all queries Verify makes are outside S

Algorithm makes $T - |S|$ queries and succeeds with probability at least $1 - t \cdot |S|/T$

Ruling out Tight VDFs

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles

x sequence of T oracle queries made by $\mathrm{Eval}^{\mathcal{O}}(x)$ $y = \mathrm{Eval}^{\mathcal{O}}(x)$



Main idea:

- Choose random subset of the q
- Output $\mathrm{Eval}^{\mathcal{O}}(x)$ using $\mathcal{O}$ to ans

Verification algorithm makes t que

- Important queries that can "affe
- With probability $1 - t \cdot |S|/T$, a

For tight sequentiality, $\sigma = T - T^{\rho}$

- Set $|S| = T^{\rho}$
- Attack makes $T - T^{\rho}$ queries and succeeds with probability $1 - t/T^{1-\rho}$ which is noticeable since $t = \mathrm{polylog}(T) \ll T^{1-\rho}$

For non-tight sequentiality (e.g., $\sigma = T/2$), the success probability is vacuous

Algorithm makes $T - |S|$ queries and succeeds with probability at least $1 - t \cdot |S|/T$

Our Results

Can we construct VDFs from a random oracle?

Negative results (in several <u>specific</u> settings):

Theorem. VDFs with <u>perfect uniqueness</u> cannot be based solely on random oracles
(e.g., [LW15, AKKPW19])

Theorem. VDFs with <u>tight sequentiality</u> cannot be based solely on random oracles
(e.g., [DGMV19]) *(lower bound also appeared in concurrent work of [DGMV19])*

Open questions:

- Strengthen lower bounds to rule out VDFs in random oracle?
- Construct <u>non-tight</u> VDFs with <u>computational uniqueness</u> from random oracles?

Thank you!

`https://eprint.iacr.org/2019/663`