CONJURA · OPEN PROBLEM

Ruling Out Computationally Unique VDFs in the Random Oracle Model

Parallel random oracle model, perfect completeness, standard (computational) uniqueness

Status. Statement: AI-written from Mohammad Mahmoody, Caleb Smith and David J. Wu, *Can Verifiable Delay Functions be Based on Random Oracles?*, Cryptology ePrint Archive 2019 (full version of the ICALP 2020 paper, subsuming the earlier note *A Note on the (Im)possibility of Verifiable Delay Functions in the Random Oracle Model*), not yet checked by a human. The paper settles the perfectly unique case with perfect completeness, and the tight regime $\sigma > T \cdot (1 - 1/2(s+t))$ or $\sigma = T - T^{\rho}$ even without uniqueness; what is open, and what is conjectured below, is the same impossibility under computational uniqueness, the notion the definition of a VDF actually requires.

Category. *Idealized Models & Non-Uniformity.*

Conjecture (informal). *A verifiable delay function takes a long, inherently sequential time to evaluate, can be checked quickly, and has a unique accepted output. Can such an object be built out of nothing but an ideal hash function, modelled as a random oracle, where sequential time is measured by the number of rounds of adaptive oracle queries an algorithm needs? The answer is no when uniqueness is perfect, meaning that no accepting proof for a wrong output exists at all. What remains open is the standard notion, where wrong outputs with accepting proofs may well exist but no efficient adversary can find one. The conjecture is that the impossibility survives this weakening: relative to a random oracle, no construction can be both computationally unique and genuinely sequential.*

1 The setting

Verifiable delay functions were introduced by Boneh, Bonneau, Bünz and Fisch [BBBF18]: a function that requires T *sequential* steps to evaluate, whose output can be verified in time $t \ll T$, and whose

accepted output is unique. Every efficient construction known at the time of this work rests on structured algebraic assumptions — groups of unknown order [Wes19, Pie19] or isogenies — which raises the question of whether VDFs can instead be built generically from unstructured, symmetric primitives. The natural formalisation of that question is the *parallel* random oracle model, where the total running time of an algorithm is its number of oracle queries and its parallel running time is its number of adaptive query *rounds*; this is the model in which Mahmoody, Moran and Vadhan ruled out time-lock puzzles [MMV11].

The source paper settles two special cases. First, VDFs with *perfect* uniqueness are impossible in this model: the attacker repeatedly simulates the honest evaluation in its head, answering oracle queries itself, and after each simulated execution asks all previously unasked queries to the real oracle in a single round; after $2(s+t)+1$ such simulated executions, using $2(s+t)$ rounds of real-oracle queries, a majority vote over the simulated outputs returns the true value. The counting argument is that a round can go wrong only by hitting, for the first time, one of the at most $s+t$ queries made by the real setup and verification, so at most $s+t$ rounds are bad; in every good round the simulated execution is consistent with *some* oracle O', and perfect uniqueness relative to O' forces the simulated output to be the correct one. This also rules out strongly decodable VDFs and, as a warm-up via [MMV11], the permutation VDFs that arise from combining reversible proofs of sequential work [AKK19] with the sloth function [LW15], showing that some structured assumption is necessary there.

Second, in the *tight* regime studied concurrently by Döttling, Garg, Malavolta and Vasudevan [DGMV19], where σ is very close to T, the paper rules out even proofs of sequential work [MMV13, CP18], which are VDFs without uniqueness: the attacker simulates a random subset of G of the T oracle queries and answers the rest honestly, succeeding unless one of the simulated queries is touched by setup or verification. This yields impossibility for $\sigma = T \cdot (1 - 1/2(s+t))$ and for $\sigma = T - T^{\rho}$ with constant $0 < \rho < 1$, and it is extended in the paper to an expensive private setup phase and to the random permutation oracle model.

Neither technique reaches the notion actually used in practice. The first spends perfect uniqueness at exactly the step where a con-

sistent simulated execution is declared correct, and offers nothing when wrong outputs with accepting proofs exist but are merely hard to find. The second cannot be pushed out of the tight regime on its own, because publicly-verifiable proofs of sequential work with $\sigma = T/2$ do exist in the random oracle model [MMV13]: any lower bound for non-tight parameters must therefore consume the uniqueness property, which is precisely what the second technique never touches. The paper poses closing this gap as its main open question. A proof would rule out black-box constructions of computationally unique VDFs with perfect completeness from ideal hash functions, leaving only the negligible-completeness-error case that the paper additionally asks for; a refutation — a computationally unique, sequential construction relative to a random oracle — would be a considerable surprise and would identify uniqueness error as the resource that makes random-oracle VDFs possible. The paper's own framing of the alternative is that any random-oracle construction of a VDF must either use non-black-box techniques or leverage its uniqueness error in a critical manner [MSW20].

Progress to date. Theorem 3.1 of the source gives, for any perfectly unique VDF with perfect completeness, an adversary asking $2T \cdot (s + t)$ queries in $2(s + t)$ rounds, which breaks σ-sequentiality for every $\sigma \geq 2(s + t)$. Proposition 3.2 derives the permutation-VDF case from the time-lock puzzle lower bound of Mahmoody, Moran and Vadhan [MMV11]. Theorem 3.6 and Corollary 3.7 rule out tight proofs of sequential work, hence tight VDFs, with no uniqueness assumption and tolerating completeness error; Theorem 4.1 and Corollary 4.2 extend this to a setup phase running in time $\mathrm{poly}(\lambda, T)$ with private coins, via a preprocessing step that learns the ϵ-heavy setup queries, and Section 4.2 extends it to the random permutation oracle model. The concurrent work of Döttling, Garg, Malavolta and Vasudevan [DGMV19] independently gives a random oracle lower bound for tight VDFs. No progress is reported on computational uniqueness itself.

2 Notation and parameters

- $\lambda \in \mathbb{N}$: the security parameter. $T = T(\lambda) \in \mathbb{N}$: the time (delay) bound; it is also the bound on the number of oracle queries and query rounds of the honest evaluation algorithm.

- $\mathcal{R}$: a finite range. $O\colon \{0,1\}^* \to \mathcal{R}$: a random oracle, i.e. a uniformly random function, to which all algorithms have oracle access.
- An oracle-aided algorithm may issue many queries *in parallel* within a single round. Its *total time* is its total number of oracle queries; its *parallel time* is its number of query rounds. This is the parallel random oracle model.
- $\Pi = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$: the VDF, a triple of oracle-aided algorithms. pp: the public parameters output by Setup; $\mathcal{X}_{\mathsf{pp}}$ and $\mathcal{Y}_{\mathsf{pp}}$: the input and output spaces determined by pp, with $\mathcal{X}_{\mathsf{pp}}$ uniformly samplable.
- s: an upper bound on the number of oracle queries made by Setup; t: an upper bound on the number of oracle queries made by Verify.
- γ: the completeness error of Π; σ: the sequentiality parameter, measured in query rounds of the online adversary.
- p: a universal polynomial, quantified before Π; $\mathrm{negl}(\lambda)$: a function that is $\lambda^{-\omega(1)}$; $\mathrm{poly}(\cdot)$: a fixed polynomial in its arguments.

3 The conjecture

Definition 1 (VDF in the parallel random oracle model). A *verifiable delay function* in the parallel random oracle model is a triple $\Pi = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ of oracle-aided algorithms:

- $\mathsf{Setup}^O(1^\lambda, 1^T) \to \mathsf{pp}$ makes at most s oracle queries and outputs public parameters pp, which determine a uniformly samplable input space $\mathcal{X}_{\mathsf{pp}}$ and an output space $\mathcal{Y}_{\mathsf{pp}}$;
- $\mathsf{Eval}^O(\mathsf{pp}, x) \to (y, \pi)$, for $x \in \mathcal{X}_{\mathsf{pp}}$, makes at most T oracle queries in at most T rounds and outputs a value $y \in \mathcal{Y}_{\mathsf{pp}}$ together with a (possibly empty) proof π. The value y is required to be determined by pp, x and O alone, independently of the coins of Eval; we write $y =$

$\mathsf{Eval}^{\mathcal{O}}(\mathsf{pp}, x)$ for this value;

- $\mathsf{Verify}^{\mathcal{O}}(\mathsf{pp}, x, y, \pi) \to \{0,1\}$ makes at most t oracle queries.

Π has *completeness error* γ if for all $\lambda, T \in \mathbb{N}$,

$$\Pr\left[\, \mathsf{Verify}^{\mathcal{O}}(\mathsf{pp}, x, y, \pi) = 1 \,\right] \geq 1 - \gamma,$$

where the probability is taken over the experiment

$$\mathsf{pp} \leftarrow \mathsf{Setup}^{\mathcal{O}}(1^\lambda, 1^T), \qquad x \leftarrow \mathcal{X}_{\mathsf{pp}},$$
$$(y, \pi) \leftarrow \mathsf{Eval}^{\mathcal{O}}(\mathsf{pp}, x),$$

that is, over the choice of $\mathcal{O}$, the coins of Setup and Eval, and the uniform choice of x. *Perfect completeness* is the case $\gamma = 0$.

Definition 2 (computational uniqueness). Π is *computationally unique* if for every oracle $\mathcal{O}$ and every oracle-aided adversary Adv making at most $\mathrm{poly}(\lambda, T)$ oracle queries,

$$\Pr\left[\, y \neq \mathsf{Eval}^{\mathcal{O}}(\mathsf{pp}, x) \ \wedge\ \mathsf{Verify}^{\mathcal{O}}(\mathsf{pp}, x, y, \pi) = 1 \,\right] \leq \mathrm{negl}(\lambda),$$

where

$$\mathsf{pp} \leftarrow \mathsf{Setup}^{\mathcal{O}}(1^\lambda, 1^T),$$
$$(x, y, \pi) \leftarrow \mathsf{Adv}^{\mathcal{O}}(1^\lambda, 1^T, \mathsf{pp}),$$

and the probability is over the coins of Setup and of Adv only, and *not* over the choice of the oracle: the requirement is imposed for every fixed $\mathcal{O}$. (*Perfect uniqueness* is the stronger requirement that for every $\mathcal{O}$, every pp in the support of $\mathsf{Setup}^{\mathcal{O}}$ and every $x \in \mathcal{X}_{\mathsf{pp}}$, no pair (y, π) with $y \neq \mathsf{Eval}^{\mathcal{O}}(\mathsf{pp}, x)$ satisfies $\mathsf{Verify}^{\mathcal{O}}(\mathsf{pp}, x, y, \pi) = 1$.)

Definition 3 (σ-sequentiality). Π is *σ-sequential*, where σ may depend on λ, T, s and t, if for every pair $\mathsf{Adv} = (\mathsf{Adv}_0, \mathsf{Adv}_1)$ of oracle-aided algorithms making at most $\mathrm{poly}(\lambda, T)$ oracle queries in total and such that Adv_1 makes at most σ *rounds* of oracle queries,

$$\Pr\left[\, y = \mathsf{Eval}^{\mathcal{O}}(\mathsf{pp}, x) \,\right] = \mathrm{negl}(\lambda),$$

where

$$\mathsf{pp} \leftarrow \mathsf{Setup}^{\mathcal{O}}(1^\lambda, 1^T), \qquad \mathsf{st} \leftarrow \mathsf{Adv}_0^{\mathcal{O}}(1^\lambda, 1^T, \mathsf{pp}),$$
$$x \leftarrow \mathcal{X}_{\mathsf{pp}}, \qquad y \leftarrow \mathsf{Adv}_1^{\mathcal{O}}(\mathsf{st}, x),$$

and the probability is over the choice of $\mathcal{O}$, the coins of Setup and of Adv, and the uniform choice of x. Here Adv_0 is an unrestricted-round preprocessing algorithm and Adv_1 is the online, round-bounded adversary.

Conjecture 1 (no computationally unique VDF in the parallel random oracle model). *There exists a polynomial p such that the following holds. Let $\lambda \in \mathbb{N}$ be the security parameter and let $T = T(\lambda)$ be the time bound. Let $\Pi = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ be a VDF in the parallel random oracle model (Definition 1) with perfect completeness, i.e. completeness error $\gamma = 0$, in which* Setup *makes at most $s = s(\lambda, T)$ oracle queries,* Eval *makes at most T oracle queries, and* Verify *makes at most $t = t(\lambda, T)$ oracle queries. If Π is computationally unique (Definition 2), then Π is* not *σ-sequential (Definition 3) for $\sigma = p(\lambda, s, t)$.*

Remark 1 (the equivalent monotone form). Since σ-sequentiality is monotone in σ (an algorithm running in at most σ rounds also runs in at most σ' rounds for every $\sigma' \geq \sigma$), the conclusion is equivalent to: for every $\sigma \geq p(\lambda, s, t)$ there is no Π that simultaneously satisfies perfect completeness, computational uniqueness and σ-sequentiality in the parallel random oracle model. In particular, for the standard efficiency regime $s, t = \mathrm{poly}(\lambda, \log T)$ one has $p(\lambda, s, t) = \mathrm{poly}(\lambda, \log T) \ll T$, so no computationally unique VDF with perfect completeness in the parallel random oracle model

achieves any nontrivial sequentiality.

Remark 2 (the quantifier order in computational uniqueness). Following the source paper's definition, the uniqueness probability is taken over the coins of Setup and of Adv but *not* over the choice of the oracle, so Definition 2 imposes the bound for every fixed O against every query-bounded adversary. This is a strong reading. A version quantified instead with high probability over the choice of O is a formally different statement, and the two are not obviously equivalent; the model choice made here is the paper's.

Remark 3 (scope, and what counts as a resolution). Conjecture 1 keeps perfect completeness, matching the hypotheses of the paper's first lower bound. The paper asks, in the same sentence, that an extension also tolerate negligible completeness error; the statement above does not claim that strengthening. The polynomial p is a rendering of what extending the perfect-uniqueness attack would mean: the paper commits to no bound on the attacker's rounds, so an impossibility proved with a worse but still $o(T)$ round bound should count as settling the question.

4 Bibliography

[AKK19] H. Abusalah, C. Kamath, K. Klein, K. Pietrzak, M. Walter. *Reversible proofs of sequential work.* EUROCRYPT 2019, pages 277–291.

[BBBF18] D. Boneh, J. Bonneau, B. Bünz, B. Fisch. *Verifiable delay functions.* CRYPTO 2018, pages 757–788.

[CP18] B. Cohen, K. Pietrzak. *Simple proofs of sequential work.* EUROCRYPT 2018, pages 451–467.

[DGMV19] N. Döttling, S. Garg, G. Malavolta, P. N. Vasudevan. *Tight verifiable delay functions.* IACR Cryptology ePrint Archive, 2019:659.

[LW15] A. K. Lenstra, B. Wesolowski. *A random zoo: sloth, unicorn, and trx.* IACR Cryptology ePrint Archive, 2015:366.

[MMV11] M. Mahmoody, T. Moran, S. P. Vadhan. *Time-lock puzzles in the random oracle model.* CRYPTO 2011, pages 39–50.

[MMV13] M. Mahmoody, T. Moran, S. P. Vadhan. *Publicly verifiable proofs of sequential work.* ITCS 2013, pages 373–388.

[MSW20] M. Mahmoody, C. Smith, D. J. Wu. *Can verifiable delay functions be based on random oracles?* ICALP 2020.

[Pie19] K. Pietrzak. *Simple verifiable delay functions.* ITCS 2019, pages 60:1–60:15.

[Wes19] B. Wesolowski. *Efficient verifiable delay functions.* Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 379–407, Springer, 2019.