IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 3, 21 pages.

https://doi.org/10.62056/avivommol

Special Soundness in the Random Oracle Model

Douglas Wikström[a]

KTH Royal Institute of Technology, Stockholm, Sweden

Abstract. We generalize the optimal knowledge extractor for constant-round special sound protocols presented by Wikström (2018) to a knowledge extractor for the corresponding non-interactive Fiat-Shamir proofs in the random oracle model and give an exact analysis of the extraction error and running time.
Relative the interactive case the extraction error is increased by a factor ℓ and the running time is increased by a factor $O(\ell)$, where ℓ is the number of oracle queries made by the prover.
Through carefully chosen notation, novel concepts, and a technical lemma, we effectively recast the extraction problem of the notoriously complex non-interactive case to the interactive case. Thus, our approach may be of independent interest.

1 Introduction

Zero knowledge proofs and proofs of knowledge. Zero knowledge proofs were discovered by Goldwasser, Micali, and Rackoff [GMR89]. They allow a prover to interactively convince a verifier that a statement is true without disclosing anything else. A related notion discovered by Bellare and Goldreich [BG92] are proofs of *knowledge*. In such protocols the prover not only shows that a statement is true, but that it holds a witness of this fact.

The completeness of a protocol is the probability that it completes successfully when both parties follow the protocol on a valid common input. The *soundness error* of a protocol is the probability that a malicious prover convinces an honest verifier that a false statement is true.

If there is an extraction algorithm such that for every prover and every statement a witness is output in expected time (over the internal randomness of the extractor) $\mathsf{poly}/(\Delta - \epsilon)$, where Δ is the probability that the honest verifier is convinced and ϵ is the *knowledge error*, then the protocol is called a proof of knowledge. Thus, the knowledge error is an upper bound on the probability that a prover convinces a verifier without knowing a witness.

The extractor may rewind and complete multiple executions from any point of the execution, i.e., it treats the prover as a deterministic oracle. A knowledge error ϵ implies a soundness error of at most ϵ, since the analysis of the knowledge extractor may be seen as a probabilistic proof [AS08]. Due to the efficiency requirement on the extractor the reverse implication does not hold. Readers are referred to [Gol00] for a thorough discussion of variations of these notions.

Special soundness. A three-message public-coin protocol [GMR89, Bab85] is defined to be *special sound* if a witness can be computed efficiently from two accepting transcripts with a common first prover message, but distinct verifier messages. This notion was

E-mail: dog@kth.se (Douglas Wikström)
[a]Partially supported by the TrustFull project funded by the Swedish Foundation for Strategic Research.

Received: 2024-07-08 Accepted: 2024-09-02

introduced by Cramer et al. [CDS94] as a generalization of a property of Schnorr's proof of knowledge of a discrete logarithm [Sch91].

In the generalization of Wikström [Wik18] a $(2r+1)$-round protocol is special sound if: (1) the ith verifier message is chosen uniformly at random from the ground set S_i of a matroid $\mathbb{M}_i$ for $i \in [r]$, and (2) a witness can be computed from a tree of accepting transcripts such that for each $i \in [r]$ and each node at depth $i-1$ the verifier messages form a basis of $\mathbb{M}_i$.

We use matroids as a language to formalize independence abstractly, but examples from the literature include inequality [Sch91] or linear independence [BGR98]. It is common to restrict challenge spaces in real-world applications by truncation or by using a pseudo-random generator to expand challenges. Using the language of matroids this amounts to considering a natural induced submatroid. Our results remain rigorous and explicit even in this case. Matroids are commonly used and play an important role in the analysis of graph algorithms. Follow-up work focus on $(d_1, \ldots, d_r)$-special-soundness, i.e., the special case of *uniform matroids*.

Knowledge extraction. For special sound protocols knowledge extraction amounts to sampling interactions until two (or more) accepting transcripts with the required independence properties are found. A basic version of a forking lemma appears in Schnorr's analysis of an extractor of a discrete logarithm [Sch91] and was refined and generalized in later work [PS96, BN06]. A knowledge extractor may be an expected/strict polynomial time algorithm which always/mostly extracts a witness. Mathematically the difference is mostly a matter of taste, but it matters in some applications [BL04].

A forking lemma for interactive constant-round protocols appears in Bootle et al [BCC$^+$16]. The first exact result is given by Wikström [Wik18]. Note that for *generic* special sound protocols the knowledge error can never be smaller than $\max_{i\in[r]}\{|S_i|^{-1}\}$, since it may suffice to guess a single challenge correctly. Indeed, without additional assumptions a prover may be able to verify if it guessed correctly in each round. Thus, his analysis is essentially tight in this case.

However, the special case of protocols where a prover cannot determine if it guessed correctly until the last round remain important, since they admit using small challenge spaces and still have exponentially small knowledge error. In this case, it is important to consider sampling *without replacement*. The analysis in [Wik18] applies in a straightforward way by replacing geometric and negative binomial distributions by their hypergeometric and negative hypergeometric siblings and adjust the tail bounds.

Follow-up work [HKR19, dPLS19, JT20, AL21] may in hindsight loosely be thought of as performing this analysis from scratch. This is partly obscured by the different choice of notation.

They also focus on the special case of $(d_1, \ldots, d_r)$-special-soundness, which is captured in [Wik18] as uniform matroids where $\mathsf{rank}(\mathbb{M}_i) = d_i$. From a technical point of view the difference is minor. Their results can be adapted either by: (1) constraining sampling, or (2) showing that the output is random and iterate. However, one cannot apply an extractor for uniform matroids in a blackbox way to extract, e.g., linearly independent vectors, since the only guarantee for the output is that the vectors are pairwise distinct. To summarize, Attema et al [ACK21] provides a tighter analysis for a special case of special sound protocols, where additionally provers are *oblivious* to their conditional success probability conditioned on the interaction so far.

Fiat-Shamir heuristic. Recall that a public coin protocol may be converted into a non-interactive protocol using the Fiat-Shamir transform [FS86]. This replaces each verifier message by the output of a hash function evaluated on the common input and the current partial transcript. This is important in practice to reduce the number of rounds.

The Fiat-Shamir heuristic suggests that we may analyze such protocols by replacing the hash function by a random oracle. The random oracle model was generalized and formalized by Bellare and Rogaway [BR93]. When we analyze the protocol in the random oracle model we are effectively assuming that it suffices to consider adversaries which treat the hash function as if it was ideal and never inspect its definition.

However, the adversary may still exploit the fact that it may query the random oracle repeatedly on inputs of its choice. This enables it to probe a tree of partial executions until it can extend at least one interaction with the random oracle to a complete accepting transcript that it outputs as its non-interactive proof.

Grafting protocols. It is more convenient to think of the interaction between the prover and the random oracle as a *grafting protocol* where the prover may extend the execution from any previous existing verifier message by grafting a new branch, i.e., a reply, to a verifier message. The verifier is easily adapted correspondingly to give the prover this ability. An execution is then considered to be accepting if any path from the root to a leaf in the resulting tree of transcripts, corresponding to a transcript of the basic protocol, is accepting.

We stress that neither party rewinds to a previous point in the execution; branches are *grafted* to the existing tree of executions which remains part of the view. Furthermore, the branches are added in a particular order by the prover, i.e., each prover-verifier message exchange may be associated with an integer index which orders them chronologically. This means that an execution of the grafting protocol may be identified with a topologically ordered subtree of the tree of all possible executions of the basic protocol.

Provided that the verifier messages have high entropy the computation of a Fiat-Shamir proof is statistically close in distribution to the execution of the corresponding grafting protocol. Thus, we study knowledge extraction for grafting protocols.

1.1 Contribution

The main contributions of our work are: (1) an exact security analysis of Fiat-Shamir protocols, and (2) a novel technique for analyzing protocols in the random oracle model.

We state the former result informally below for easy reference and focus on the application of the novel technique that we use to handle the main technical problem that differentiate the non-interactive case from the interactive case. We believe that this can be adapted to analyze other constructions in the random oracle model as if they where interactive protocols.

1.1.1 Exact security of Fiat-Shamir proofs.

It is well known [Sch91] that if a prover convinces a verifier with probability Δ in a three-message special sound protocol, then a witness can be extracted in expected time $p/(\Delta - \epsilon)$, where p is polynomial and ϵ is the knowledge error. Wikström [Wik18] generalized the notion to constant-round special sound protocols and gave an exact and tight bound. The concrete contribution of this work is a corresponding theorem for grafting protocols.

Theorem 1 (Informal). *Let $(\mathcal{P}, \mathcal{V})$ be a $(2r+1)$-message $(\mathbb{M}_1, \ldots, \mathbb{M}_r)$-special sound protocol with soundness error ϵ_S and knowledge error ϵ_K for a knowledge extractor that for any instance and prover that convinces the verifier with probability $\Delta > \epsilon_K$, for a constant c is expected to execute the protocol $c/(\Delta - \epsilon_K)$ times.*

Then its $(2\ell+1)$-message grafting protocol has soundness error $\ell\epsilon_S$ and knowledge error $\ell\epsilon_K$ for a knowledge extractor that for any instance and prover that convinces the verifier with probability $\Delta > \ell\epsilon_K$ is expected to execute the grafting protocol $4 \cdot 3^{r+1}\ell \cdot c/(\Delta - \ell\epsilon_K)$ times.

In applications we may often choose parameters of the protocol to reduce the soundness and knowledge errors by a factor of $1/\ell$. Thus, in practice the Fiat-Shamir transform causes a loss of roughly $\log \ell + O(1)$ bits of security. The constant factor is well below constant factors due to implementation considerations and may be ignored in practice.

1.1.2 Novel proof technique.

A careful choice of notation, novel concepts, and a technical lemma, allow us to effectively reduce the problem of constructing an extractor to the combinatorial problem of finding an accepting basis in a suitably defined matroid tree, but the distribution of the verifier messages is influenced by the adversary and not necessarily uniform. The main technical challenge is to prove that this distribution can be sampled efficiently. Apart from this the analysis from [Wik18] applies mutatis mutandi. We think that this approach may be the main and lasting contribution of our work.

Remark. A few months after our discovery Attema, Fehr, and Klooss largely independently discovered a theorem [AFK22] similar to our main theorem. Interestingly, their work is based on Attema et al. [ACK21], while we rely on the work of Wikström [Wik18]. We are currently working on a general framework that generalizes these and other related results.

1.2 Proof Strategy

Grafting protocols. We first formalize the computation of a non-interactive Fiat-Shamir proof, based on a special sound protocol, as the execution of a *grafting protocol*. In such protocols the verifier allows the prover to repeatedly: (1) spawn a new execution of the basic protocol, or (2) extend an existing partial execution of the basic protocol by grafting an additional round to it.

Thus, at any point during execution the transcript may be viewed as a tree of partial executions that grows one edge for each round. We stress that the prover may choose the location of each grafted round and that this may depend on both the structure of the tree and the verifier messages seen so far. The verifier accepts if there exists an embedded accepting transcript of the basic protocol corresponding to a path from the root to a leaf.

This captures the computation of a Fiat-Shamir proof faithfully except that the prover may only execute a round after the previous rounds have been executed, i.e., it is effectively restricted to queries to the random oracle that do not amount to guessing any reply correctly. The entropy of verifier messages is typically high in applications, and applying the Fiat-Shamir transform at all to a round with small entropy is pointless unless we assume additional properties about the protocol. Thus, for generic protocols a prover can only guess correctly with exponentially small probability, and the grafting protocol is essentially a faithful model.

Extraction problem for grafting protocols. An extractor for a special sound protocol repeatedly and recursively: (1) samples an accepting transcript, and (2) samples other accepting transcripts from a prefix of the first. If this fails within reasonable time it gives up and rewinds before restarting the recursive procedure. Additionally, the verifier messages at a given depth are sampled such that the children of each node form a basis of a matroid determined by the protocol.

An extractor that treats the prover as a blackbox must rewind it to extract a suitable tree of transcripts. Rewinding is easy to visualize for interactive protocols, but for a grafting protocol this means that leaves are pruned in the reverse order in which they were grafted to the tree of partial executions. Furthermore, when the protocol is executed with fresh randomness from a partial grafting transcript the tree of partial transcripts may regrow into a differently shaped tree and not only have different verifier messages associated with the nodes.

To understand the added complexity in the extraction problem for grafting protocols it is worthwhile to consider an embedded accepting transcript in a grafting transcript. An example of a tree of transcripts with an embedded transcript is given at the top of Figure 1.2. To rewind the execution of the embedded execution to a given round requires rewinding the execution of the grafting protocol.

The problem is that even if we complete an accepting execution from the rewinded state there is no guarantee that the resulting embedded accepting transcript is a completion of the prefix of the original embedded accepting transcript. Indeed, the prover may spawn a new execution of the special sound protocol, or graft additional rounds to any of the existing partial executions to form a new embedded accepting transcript.

Linearization and grafted sequences. The prover messages are a deterministic function of the verifier messages induced by the prover, which means that we can view: (1) the entire protocol execution, and (2) the verdict of the verifier as a single predicate and focus on the verifier messages.

The added complexity that comes with a tree of partial transcripts is partially superficial, since the actual execution of the grafting protocol proceeds linearly and the transcript is simply a list of messages that appears in topological order with respect to the tree if we encode the position of a grafted round as an integer index.

Furthermore, although each index for grafting a round is adversarially controlled, it is a deterministic function of the verifier messages thus far in the execution. Thus, the distribution of the list of verifier messages is induced by the prover and can be efficiently sampled. We call the sequence of verifier messages that is generated by this process a *grafted sequence*. An example corresponding to a tree of transcripts ordered topologically is given in the middle of Figure 1.2.

Shadow sequences and sampling. From a *complete* accepting grafted sequence z of the verifier we know the positions of the verifier messages belonging to the corresponding embedded accepting transcript, and the corresponding prover messages can be computed deterministically.

Thus, given a grafted sequence z, we can partition it into a *shadow sequence* of the form $w = (w_1, \ldots, w_r)$, where w_i ends with the ith verifier message of the embedded transcript (except w_r which is slightly different). This is illustrated at the bottom of Figure 1.2. We think of w as a shadow, since prefixes are not stable under the addition of elements. More precisely, suppose that z is a grafted sequence with shadow sequence w, and that the shadow prefix $w_{[i]}$ equals the prefix $z_{[k]}$ if we concatenate its components. If z' is a grafted sequence with the same prefix $z_{[k]}$, then the prefix $w'_{[i]}$ of its shadow sequence may have no common elements with $w_{[i]}$.

We may still think of each shadow element w_i as sampled from a ground set of a *shadow matroid* $\mathbb{M}_i^*$, which inherits the essential combinatorical structure from the corresponding matroid $\mathbb{M}_i$ of the special sound protocol, but the distribution is influenced by the prover.

If we from any prefix $w_{[i]}$ of an accepting shadow sequence w with reasonable probability could sample a complete shadow sequence w', then we would have reduced the extraction problem for grafting protocols to that of basic special sound protocols, i.e., the analysis from [Wik18] would apply with minor syntactical changes.

Sampling shadow sequences. Unfortunately, we can only sample *grafted* sequences directly. Suppose that w is the shadow sequence of a grafted sequence z and that the prefix $w_{[i]}$ corresponds to a prefix $z_{[k]}$ of z. Then if a randomly sampled z' conditioned on $z'_{[k]} = z_{[k]}$ is accepting with probability Δ we can certainly sample an accepting grafted sequence z' from the prefix $z_{[k]}$ in time roughly $1/\Delta$, but in general the probability that its shadow sequence w' satisfies $w'_{[i]} = w_{[i]}$ may be arbitrarily low.

Similarly, if we ignore the requirement on acceptance, and focus on keeping the prefix of the shadow sequence it is not hard to see that a random prefix can be extended with conditional probability roughly $1/\ell$ throughout the recursive process.

We show that both properties can be maintained simultaneously throughout an execution with constant probability of failure in each step of the process and thereby allow sampling accepting shadow sequences that extend the prefixes that appear in the algorithm.

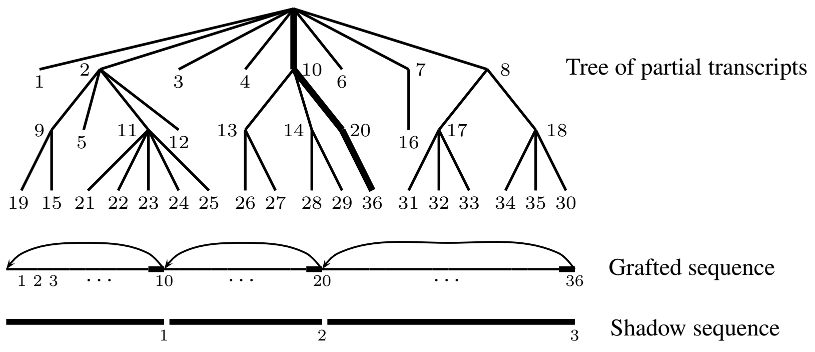

Figure 1: A tree of partial transcripts for a 7-message grafting protocol with enumerated grafted branches, the corresponding grafted sequence with predecessor pointers for an embedded accepting transcript of the basic protocol (using thicker lines), and the corresponding shadow sequence.

2 Background

We need a number of definitions and concepts from [Wik18], but the reader may in a first reading think of linear independence over a vector space instead of independence in a matroid. This is natural and common in applications [BGR98, Nef01, FS01, Wik05]. Truncated challenges, which is common in real applications, means considering the induced submatroid of a vector space. Thus, the language of submatroids not only captures the type of independence we care about, it captures an important practical optimization rigorously as a special case.

Recall that a matroid $\mathbb{M} = (S, I)$ consists of a ground set S and a set I of subsets of S that is closed downwards and satisfy the independent set exchange property. A basis is a set $B \in I$ such that $B \cup \{x\} \notin I$ for every $x \in S \setminus B$. The rank $\mathsf{rank}(\mathbb{M})$ of a matroid is the unique maximal number of elements in a basis. The span of a set A is defined by $\mathsf{span}(A) = \{x \in S \mid \mathsf{rank}(A \cup \{x\}) = \mathsf{rank}(A)\}$. A flat is a set which is its own span. Appendix F provides explicit definitions. Throughout d_i denotes the rank of a matroid $\mathbb{M}_i$.

A matroid tree $(\{v_0\}, \mathbb{M}_1, \ldots, \mathbb{M}_r)$, where v_0 is an arbitrary singleton, represents the set of verifier messages in a special sound protocol as well as the independence relations needed from a set of accepting transcripts to allow computation of the witness. A subtree is a basis if for each node at depth $i-1$ its children form a basis of $\mathbb{M}_i$.

Definition 1 (Matroid Tree)**.** The *matroid tree* associated with a list of matroids $\mathbb{M} = (\{v_0\}, \mathbb{M}_1, \ldots, \mathbb{M}_r)$ is the vertex-labeled rooted unordered directed tree of depth r such that: the root is labeled v_0 and every node at depth $i-1$ has edges to $|S_i|$ children which are uniquely labeled with the elements of the ground set S_i.

Definition 2 (Basis)**.** A *basis* of a matroid tree $\mathbb{M}$ of depth r is a maximal subgraph such that for every $i \in [r]$ the set of children of every node at depth $i-1$ is a basis of $\mathbb{M}_i$.

The subdensity captures the fraction of elements of the ground set which is outside a flat. This was introduced in [Wik18] to allow analysis of protocols where verifier messages are chosen from a subset of the algebraic structure which defines the independence sets.

Definition 3 (Subdensity)**.** Let $\mathbb{M} = (S, I)$ be a matroid of rank d. Then its ith *subdensity* is $\omega_{\mathbb{M},i}$ if $|A|/|S| \leq \omega_{\mathbb{M},i}$ for every flat A of rank $i-1$, and it has *maximal subdensity* $\omega_{\mathbb{M}} = \omega_{\mathbb{M},d}$.

After abstracting the execution of a protocol and the verdict of the verifier as a predicate ρ on verifier messages the extraction problem amounts to finding a basis of a matroid tree. Let $S = \times_{i\in[r]} S_i$ and $\Delta_\rho(\mathbb{M}) = \Pr[\rho(v) = 1]$, where v is sampled uniformly over S.

Definition 4 (Accepting Basis Extractor)**.** A probabilistic polynomial time algorithm $\mathcal{X}_\kappa$ parametrized by $\kappa \in \{0,1\}^*$ is a $(\epsilon_\kappa, \mathsf{D}_\kappa(\Delta))$-*accepting basis extractor* with *extraction error* ϵ_κ for a matroid tree $\mathbb{M}$, where $\mathsf{D}_\kappa(\Delta)$ for fixed κ is a family of distributions on $\mathbb{N}$ parametrized by $\Delta \in [0,1]$, if for every $\mathbb{M}$-predicate $\rho : S \to \{0,1\}$ and $\Delta_\rho(\mathbb{M}) \geq \Delta_0 > \epsilon_\kappa$: $\mathcal{X}_\kappa^{\rho(\cdot)}(\mathbb{M}, \Delta_0)$ outputs a ρ-accepting basis of $\mathbb{M}$, where the distribution of the number of $\rho(\cdot)$-queries is bounded by $\mathsf{D}_\kappa(\Delta_0)$.

In other words, an accepting basis extractor finds a tree of accepting interactions, but only outputs the corresponding tree of challenges, since the prover messages are defined by the verifier's challenges. Furthermore, the challenges which are children of a node at depth $i-1$ are independent with respect to the ith matroid. The running time is expressed in terms of a distributional bound, since we typically have a concentrated distribution for free, which is useful to derive exact bounds in applications.

3 Grafting Protocols

Before we introduce grafting protocols we need some notation. The ith message of the prover is a pair (p_i, a_i), where p_i is the index of a previous verifier message onto which the new branch is grafted and a_i is a prover message of the basic protocol. The verifier always sends its next challenge message immediately, so there is no need for an additional index for the verifier's messages. One round of interaction is therefore always a triple (p_i, a_i, v_{i+1}), where $p_i = 0$ if the prover starts a fresh execution of the basic protocol. Every path in the tree of partial executions of the basic protocol then has the form $(a_{j_1-1}, v_{j_1}, \ldots, a_{j_i-1}, v_{j_i})$, where $p_{j_i} = j_{i-1}$, i.e., it is an embedded transcript of the basic protocol. To ensure that the distribution of verifier messages is correct, the grafting verifier keeps state and samples each message from the appropriate ground set.

3.1 Functions of Transcripts

After each prover message during an execution of the grafting protocol on common input x the current transcript has the form $\big(x, (p_1, a_1, v_2), \ldots, (p_{i-1}, a_{i-1}, v_i), (p_i, a_i)\big)$ for some i. We call this a *truncated transcript* and denote it by $t_{[i]}$, where t may be a complete or truncated transcript itself. This allows us to define a natural depth function.

Definition 5 (Depth Function)**.** The *depth function* δ takes a truncated transcript of a grafting protocol as input and is defined by

$$\delta(t_{[i]}) = \left\{ \begin{array}{cl} 1 & \text{if } p_i = 0 \\ 1 + \delta(t_{[p_i]}) & \text{otherwise} \end{array} \right. .$$

When the truncated transcript is clear from the context we abuse notation and simply write $\delta(p_i)$ to mean $\delta(t_{[i]})$.

Definition 6 (Index Function)**.** The *index function* $\iota(\cdot)$ takes a truncated transcript of a grafting protocol as input and is defined by $\iota(t_{[i]}) = (j_1, \ldots, j_d)$, where $d = \delta(t_{[i]})$, $j_d = p_i$, and $j_l = p_{j_{l+1}}$ for $l = d-1, \ldots, 1$.

These functions merely gives a way to refer to the unique embedded partial transcript that was most recently extended by a round of interaction. Finally, we introduce notation for extracting the embedded transcript itself using the index function.

Definition 7 (Path Projection)**.** The *path projection* $\tau(\cdot)$ takes a truncated transcript of a grafting protocol as input and is defined by

$$\tau(t_{[i]}) = (x, a_{j_1-1}, v_{j_1}, \ldots, a_{j_d-1}, v_{j_d}, a_i) \ , \quad \text{where } (j_1, \ldots, j_d) = \iota(t_{[i]}) \ .$$

Note that if $d = r$, then the embedded transcript is complete and is either accepting or rejecting.

3.2 Grafting Verifier

We give an explicit transformation of a public coin verifier into a grafting verifier for completeness. It is implicit that it rejects if an index p_i provided by the prover is invalid, i.e., if there does not exist a verifier message with the index p_i in the existing partial transcript onto which a round can be grafted.

Without loss of generality we assume that the verifier sends exactly ℓ messages and that the prover's final message corresponds to a final reply of an accepting execution of the basic protocol, if any exists at all.

Definition 8 (Grafting Verifier)**.** If $(\mathcal{P}, \mathcal{V})$ is a $(\mathbb{M}_1, \ldots, \mathbb{M}_r)$-special sound protocol, then on common input x its *ℓ-grafting verifier* $\mathsf{G}[\mathcal{V}]$ proceeds as follows:

1. Initialize an empty table $H[\cdot]$
2. For $i = 0, \ldots, \ell - 1$:
   (a) Wait for a message (p_i, a_i) from the prover.
   (b) If $H[\tau(t_{[i]})] \neq \varnothing$, then set $v_{i+1} = H[\tau(t_{[i]})]$, and otherwise choose $v_{i+1} \in S_{\delta(p_i)}$ randomly, set $H[\tau(t_{[i]})] = v_{i+1}$, and hand v_{i+1} to the prover.
3. Wait for a message (p_ℓ, a_ℓ) from the prover.
4. Return the verdict $\mathcal{V}\big(\tau(x, (p_i, a_i, v_{i+1})_{i \in [0, \ell-1]}, p_\ell, a_\ell)\big)$ of the verifier $\mathcal{V}$.

Each verifier message is chosen from the ground set associated with the appropriate round in the basic protocol due to the depth function. To avoid grafting more than once at a given index with the *same* prover message the verifier uses a table. This mirrors the same property of a random oracle, i.e., once sampled it returns the same output every time it is queried on the same input.

We are interested in malicious provers which graft branches during the execution, but for completeness we describe in Appendix D the corresponding honest prover which is merely a wrapper of the honest prover of the basic protocol.

3.3 Grafting Protocols vs Non-interactive Fiat-Shamir Proofs

We may interpret the execution of a grafting protocol as the prover computing a Fiat-Shamir proof in the random oracle model using a random oracle $\mathcal{RO}$ as follows. Relative to the current truncated transcript $t_{[c]}$ a prover message (p_i, a_i) with $i \leq c$ uniquely identifies an embedded transcript $\tau(t_{[i]})$. For this embedded transcript the next verifier message

v_{i+1} is independently and uniformly distributed in the appropriate matroid ground set and sampled exactly once.

When the min entropy of the verifier message in each round is high this is essentially equivalent to the computation of a Fiat-Shamir proof, where the next verifier message is defined by $v_{i+1} = \mathcal{RO}(\tau(t_{[i]}))$. Indeed, if the min entropy η is high, then the probability that a prover queries the random oracle in advance at a point partially defined by random verifier messages that it has not yet received is bounded by $\ell 2^{-\eta}$.

In general we cannot expect that it is infeasible to: (1) determine if a prover message is likely to be part of an accepting execution, or (2) use re-randomization to generate arbitrarily many such prover messages from one. This means that a prover can probe up to ℓ verifier messages. Thus, if the min entropy is not exponentially smaller than $1/\ell$ the protocol may loose all soundness, i.e., it is unwise to apply the Fiat-Shamir transform at all.

More precisely, let $\mathsf{G}[\mathcal{V}]_{\mathcal{RO}}$ be the verifier that simply replaces the use of the table $H[\cdot]$ by an application of the random oracle $\mathcal{RO}$ (for a suitable range) and derives a challenge $v_{i+1} \in S_{\delta(p_i)}$ deterministically from the output of the random oracle identically to how v_{i+1} is derived from a uniformly distributed random tape by $\mathsf{G}[\mathcal{V}]$. It is easy to see that transcripts of executions with $\mathsf{G}[\mathcal{V}]_{\mathcal{RO}}$ and $\mathsf{G}[\mathcal{V}]$ are identically distributed. Indeed, we may think of the random oracle as a table.

All we need to show is that a Fiat-Shamir prover may be seen as a prover interacting with $\mathsf{G}[\mathcal{V}]_{\mathcal{RO}}$ except with statistically small probability, but this is standard. If a verifier has min entropy η, then the probability that an adversary queries $\mathcal{RO}$ on a transcript $t_{[i]}$ without querying it on the prefix $t_{[i-1]}$ first is bounded by $2^{-\eta}$, and it makes at most ℓ queries. Thus, the statistical distance between the two settings is at most $\ell 2^{-\eta}$ by the union bound.

4 Grafted Sequences

Recall that in [Wik18] the extraction problem is reduced to the problem of extracting an accepting basis of a matroid tree relative a prover predicate that captures both the execution of the protocol and the verdict of the verifier.

We proceed similarly to abstract the extraction of a tree of transcripts of a grafting protocol which correspond to an accepting basis tree in the basic protocol, but in our case the distribution of verifier messages depends on the prover.

A grafting function determines, from the list of verifier messages so far, at which point an additional branch is grafted to a sequence, i.e., given a sequence as input it outputs an integer index of an existing element in the sequence. This abstracts the choice made by the prover in a grafting protocol. A depth function makes explicit the depth at which a branch is grafted.

Definition 9 (Grafting Function)**.** A function f such that $f(\emptyset) = 0$ and $f(z_1, \ldots, z_i) \in [0, i]$ for every $z_1, \ldots, z_i \in \{0,1\}^*$ and every $i \in \mathbb{N}$ is a *grafting function.*

Definition 10 (Depth Function)**.** The *depth function* δ_f of a grafting function f is defined as follows:

$$\delta_f(z_1, \ldots, z_i) = \left\{ \begin{array}{cl} 1 & \text{if } f(z_{[i]}) = 0 \\ 1 + \delta_f(z_{[f(z_{[i-1]})]}) & \text{otherwise} \end{array} \right. .$$

A grafted sequence is an abstraction of a transcript of a grafted protocol where the verifier messages are explicit, and the prover messages are implicit.

Definition 11 (Grafted Sequence)**.** An $(\mathbb{M}, f)$*-grafted sequence of length* ℓ, where $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ is matroid tree with $\mathbb{M}_i = (S_i, I_i)$ is a sequence $z = (z_1, \ldots, z_\ell)$ such that

$\delta_f(z_{[i-1]}) \leq r$ and $z_i \in S_{\delta_f(z_{[i-1]})}$ for every $i \in [\ell]$. We denote the set of $(\mathbb{M}, f)$-grafted sequences of length ℓ by $G_{\mathbb{M},f,\ell}$.

Similarly to how we extracted indices from a grafted protocol transcript we extract indices of the verifier messages of an (implicitly defined) embedded truncated transcript of the basic protocol. It is not meaningful to define a path projection since the prover messages are defined by the complete grafted sequence.

Definition 12 (Index Function)**.** The *index function* ι_f takes a grafted sequence $z \in G_{\mathbb{M},f,\ell}$ as input and outputs indices $(j_1, \ldots, j_d)$ defined by $j' = f(z)$, $d = \delta_f(z_{[j'-1]})$, $j_d = j'$, and $j_i = f(z_{[j_{i+1}-1]})$ for $i = d-1, \ldots, 1$.

4.1 Shadow Sequences

We introduce shadow sequences and shadow matroids as a conceptual step to emphasize the similarity with the analysis in [Wik18]. The goal is to define a shadow sequence in such a way that we (almost) may think of it as a sequence of verifier challenges in an *interactive* protocol.

Definition 13 (Shadow Sequence)**.** If z is a grafted sequence over $G_{\mathbb{M},f,\ell}$, $(j_1, \ldots, j_d) = \iota_f(z)$, $j_0 = 0$, and $w_i = (z_{j_{i-1}+1}, \ldots, z_{j_i})$ for $i \in [1, d-1]$, and $w_d = (z_{j_{d-1}+1}, \ldots, z_\ell)$, then $\sigma_f(z) = (w_1, \ldots, w_d)$ is its *shadow sequence.*

The last shadow element ends at index ℓ and not index j_d, so there may be some spurious elements beyond the last embedded verifier message of the basic protocol. This is necessary, since they influence the prover's last message, but the reader may safely ignore this since it does not influence the analysis in any significant way.

For every grafted sequence z there is a shadow sequence $w = \sigma_f(z)$ and we may view a predicate ρ over grafted sequences as a predicate ρ^* over shadow sequences. The last component of the ith shadow element is an element from S_i. Thus, the ith shadow element is contained in the following matroid.

Definition 14 (Shadow Matroid)**.** If $\mathbb{M} = (S, I)$ is a matroid, then its *shadow matroid* $\mathbb{M}^* = (S^*, I^*)$ is defined by $S^* = \{0,1\}^* \times S \times \{0,1\}^*$ and letting C be an independence set in I^* if and only if $B = \{b \mid (a, b, c) \in C\}$ is an independence set in I and $|C| = |B|$.

Intuitively, we would like to think of a shadow sequence as a redundant representation of a list of verifier messages in the basic special sound protocol, but the prefixes/postfixes influence the implicitly defined prover messages and the output of the grafting function, so this remains an intuitive view.

4.2 Grafting Function and Predicate of a Prover

We define a grafting function in terms of a grafting protocol and use the definition of a prover predicate from [Wik18], which is restated below with adapted notation for easy reference. The purpose of these definitions is to abstract from protocols and consider a pair of a grafting function and a predicate instead. This leaves us with a clean combinatorial problem.

Definition 15 (Grafting Function of Prover)**.** The *grafting function* $f[\mathcal{P}^*, \mathcal{V}]$ *of* $\mathcal{P}^*$ for an ℓ-grafting protocol of a public-coin protocol $(\mathcal{P}, \mathcal{V})$ and common input x is defined as follows: On input $z = (z_1, \ldots, z_i)$, simulate $(\mathcal{P}^*, \mathsf{G}[\mathcal{V}])$ using z as the random tape for $\mathsf{G}[\mathcal{V}]$ until $\mathcal{P}^*$ outputs its ith message (p_i, a_i) and output p_i.

In other words the grafting function is defined by simply running the verifier and observing the index where it grafts each new round in the tree of interactions.

Definition 16 (Prover Predicate)**.** The *prover* $\mathbb{M}$*-predicate* $\rho[\mathcal{P}^*, \mathcal{V}, x]$ *of* $\mathcal{P}^*$ for the ℓ-grafting protocol of a public-coin protocol $(\mathcal{P}, \mathcal{V})$ and common input x is defined by $\rho[\mathcal{P}^*, \mathcal{V}, x](z) = \langle \mathcal{P}^*, \mathsf{G}[\mathcal{V}]_z \rangle(x)$, where $z = (z_1, \ldots, z_\ell)$.

The prover predicate is satisfied on a sequence of verifier messages if the grafting verifier would accept in an interaction with the prover.

4.3 Random Grafted Sequences

Suppose that $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ is a matroid tree and let f be a grafting function. A random variable over $G_{\mathbb{M},f,\ell}$ representing the verifier messages of an execution of a grafting protocol is readily defined by stipulating that each verifier message is uniformly and independently distributed over a ground set of the matroid identified by the depth function.

Definition 17 (Random Grafted Sequence)**.** The distribution of a random grafted sequence Z over $G_{\mathbb{M},f,\ell}$ is defined by $\mathsf{P}_{Z_i \mid Z_{[i-1]}}\left(\cdot \,\middle|\, z_{[i-1]} \right) = |S_{\delta_f(z_{[i-1]})}|^{-1}$.

Although each element Z_i is uniformly and independently distributed, the sequence is not necessarily uniformly distributed, since the choice of ground set is determined by previous elements and the grafting function. Consequently, the distribution of the shadow sequence $W = \sigma_f(Z)$ is not necessarily uniform.

5 Random Shadow Sequences

In each recursive call of the extractor the probability that the current prefix of a shadow sequence leads to an accepting sequence is assumed to be some quantity Δ, but we must also be able to efficiently sample extensions of the prefix.

The problem is that even if the prefix has probability Δ to lead to an accepting grafted sequence it may be the case that the resulting grafted sequence does not have the same prefix viewed as a shadow sequence. Indeed, the partitioning of the grafted sequence into a shadow sequence is determined by the grafted sequence as a whole. Conversely, if we focus on sampling a shadow sequence with a given prefix, then the acceptance probability under this conditioning may be significantly lower than Δ. Thus, we must prove that a random prefix has both properties at once with reasonable probability. We formalize the property we need below.

Definition 18 (Extendable Shadow Prefix)**.** Let $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ be a matroid tree, let f be a grafting function, let Z be a random grafted sequence over $G_{\mathbb{M},f,\ell}$, define $J = \iota_f(Z)$, and $W = \sigma_f(Z)$, and let $\rho : G_{\mathbb{M},f,\ell} \to \{0, 1\}$ be a predicate. Define for every index $i \in [0, r-1]$, shadow sequence prefix $w \in [W_{[i]}]$, and bound $\beta \in (0, 1)$:

$$\zeta_w^\rho = \Pr\left[\rho(Z) = 1 \,\middle|\, Z_{[k]} = w\right] \qquad \text{Accept probability} \qquad (1)$$

$$\theta_w^\rho = \Pr\left[J_i = k \,\middle|\, \rho(Z) = 1 \wedge Z_{[k]} = w\right] \qquad \text{Probability of intact prefix} \qquad (2)$$

$$\xi_\rho(w, \Delta, \beta) = \left(\zeta_w^\rho \geq \Delta \wedge \theta_w^\rho \geq \beta^2/\ell\right) \qquad \text{Prefix } w \text{ is extendable} \qquad (3)$$

Thus, an extendable prefix w from the support $[W_{[i]}]$ of $W_{[i]}$ allows sampling a completion with notable probability such that the resulting shadow sequence has w as a prefix, i.e., we can sample a completion of w as a *shadow sequence*, and not only as a grafted sequence.

More precisely, we can always sample a complete grafting sequence Z starting with $w_{[i]}$ and if $\xi_\rho(w_{[i]}, \Delta, \beta) = 1$, then we have $\rho(Z) = 1 \wedge J_i = k$ with probability at least $\beta^2\Delta/\ell$. Below we show that this implies $\rho^*(W) = 1$ and $W_{[i]} = w_{[i]}$. Thus, we need roughly $\ell/(\beta^2\Delta)$ sampled grafted sequences starting from $w_{[i]}$ to find an accepting shadow sequence starting from $w_{[i]}$. We maintain a sufficient acceptance probability by application of the following well known lemma, which is proven in Appendix E.

Lemma 1 (Markov Conditioning). *If $H = (X, Y)$ is a random variable, E is an event in $[H]$, $\delta_x = \Pr_H [E \,|X = x]$, and $\Pr_H [E] \geq \Delta$, then $\Pr_H [\delta_X < \alpha\Delta \,|E] \leq \alpha$.*

5.1 Coinciding Indices

It should be clear that if z and z' share a prefix $z_{[k]}$ corresponding to a prefix $w_{[i]}$ of the shadow sequence w of z, and the ith element of the shadow sequence w' of z' end at index k, then $w'_{[i]} = w_{[i]}$.

Lemma 2 (Pinching). *For every $z, z' \in G_{\mathbb{M},f,\ell}$ and every $i \in [1, r-1]$, with $j = \iota_f(z)$ and $w = \sigma_f(z)$, and similarly for j' and w', we have*

$$z'_{[j'_i]} = z_{[j_i]} \text{ and } j'_i = j_i \quad \Longrightarrow \quad j'_{[i]} = j_{[i]} \text{ and } w'_{[i]} = w_{[i]} \quad .$$

Proof. If we define $p_t = f(z_{[t-1]})$ for $t \in [\ell]$ and similarly for p'_t and $z'_{[t-1]}$, then by assumption $p'_{[i]} = p_{[i]}$. Thus, if $j'_i = j_i$, then $j'_{[i]} = j_{[i]}$ which implies that $w'_{[i]} = w_{[i]}$. □

Suppose that we sample a grafted sequence z and let $w_{[i]}$ be a prefix of its shadow sequence w, which viewed as a prefix of the grafted sequence has the form $z_{[k]}$ for some k. If we sample a fresh completion $z'_{[k+1,\ell]}$ of $z_{[k]}$, and define $w' = \sigma_f(z_{[k]}, z'_{[k+1,\ell]})$ and $j' = \iota_f(z_{[k]}, z'_{[k+1,\ell]})$, then Lemma 2 says that it is sufficient to require that $j'_i = j_i$ to guarantee that $w'_{[i]} = w_{[i]}$. The next lemma is used to prove that over the random choice of $w_{[i]}$ this happens with reasonable probability.

Lemma 3 (Coinciding Indices). *Let $Z = (Z_1, \ldots, Z_\ell)$ be a random variable, let $\iota : [Z] \to [0, \ell-1]$ be a function from the support of Z to indices, define $K = \iota(Z)$, $X = (Z_1, \ldots, Z_K)$, and let Y be independently distributed with $\mathsf{P}_{Y|X}\ (\cdot\,|x) = \mathsf{P}_{Z_{[k+1,\ell]}|Z_{[k]}}\ (\cdot\,|x)$, where x has length k. If we define $\theta_x = \Pr [\iota(X, Y) = k \,|X = x]$, then for every $\beta \in (0, 1/2)$:*

$$\Pr \left[\theta_X < \beta^2/\ell\right] \leq 2\beta \quad . \tag{4}$$

Proof. We show that we can apply Lemma 1. By definition we have

$$\mathsf{P}_{X,K}\,(x, k) = \mathsf{P}_{Z_{[k]}|K}\ (x\,|k)\,\mathsf{P}_K\,(k) \quad , \tag{5}$$

where it is understood that $\mathsf{P}_{X,K}\,(x, k) = 0$ if the length of x is not equal to k. Thus, to sample x we may: sample a length k, sample $z_{[k]}$ as a prefix of a complete sequence z conditioned on $\iota(z) = k$, and set $x = z_{[k]}$. Furthermore, from independence we have

$$\Pr [\iota(X, Y) = k \,|X = x] = \Pr \left[\iota(Z) = k \,\middle|Z_{[k]} = x\right] \tag{6}$$

which means that $\theta_x = \Pr \left[\iota(Z) = k \,\middle|Z_{[k]} = x\right]$.

If we let $\beta \in (0, 1)$ and define $B = \{k \mid \mathsf{P}_K\,(k) < \beta/\ell\}$, then we trivially have $\Pr [K \in B] < \sum_{k\in[0,\ell-1]} \beta/\ell = \beta$. For every $k \notin B$ we have $\Pr [\iota(Z) = k] = \Pr [K = k] \geq \beta/\ell$ from the definitions of K and the set B. For $k \notin B$ and every $\alpha \in (0, 1)$ Lemma 1 then implies that

$$\Pr \left[\theta_{Z_{[k]}} < \alpha\beta/\ell \,\middle|\iota(Z) = k\right] \leq \alpha \quad , \tag{7}$$

which implies that

$$\begin{aligned}\Pr\left[\theta_X < \alpha\beta/\ell\right] &= \sum_{k\in[0,\ell-1]} \mathsf{P}_K\left(k\right)\Pr\left[\theta_{Z_{[k]}} < \alpha\beta/\ell \,\middle|\, \iota(Z)=k\right] &(8)\\ &\leq \Pr\left[K\in B\right] + \sum_{k\notin B} \mathsf{P}_K\left(k\right)\Pr\left[\theta_{Z_{[k]}} < \alpha\beta/\ell \,\middle|\, \iota(Z)=k\right] &(9)\\ &\leq \beta + \alpha\sum_{k\notin B}\mathsf{P}_K\left(k\right) \leq \alpha+\beta \enspace . &(10)\end{aligned}$$

The proof is completed by setting $\alpha = \beta$. □

5.2 Extendable Shadow Sequence

The following theorem follows from the two lemmas above and the union bound.

Theorem 2 (Extendable Shadow Sequence)**.** *Let $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ be a matroid tree, let f be a grafting function, let Z be a random grafted transcript over $G_{\mathbb{M},f,\ell}$, and define $J = \iota_f(Z)$ and $W = \sigma_f(Z)$. Let $\rho : G_{\mathbb{M},f,\ell} \to \{0, 1\}$ be a predicate. For every $i \in [r-1]$, $w \in [W_{[i-1]}]$ such that $\zeta^\rho_w \geq \Delta$, $\alpha \in (0, 1)$, and $\beta \in (0, (1-\alpha)/2)$.*

$$\Pr\left[\xi_\rho(W_{[i]}, \alpha\Delta, \beta) = 1 \,\middle|\, \rho^*(W) = 1, W_{[i-1]} = w\right] \geq 1 - \alpha - 2\beta \enspace . \qquad (11)$$

Proof. We show that we can apply Lemma 1 and Lemma 3 by choosing notation appropriately. We define the random variable (J_i, X), by

$$\mathsf{P}_{J_i,X}\left(\cdot\right) = \mathsf{P}_{J_i,W_{[i]} \mid W_{[i-1]}}\left(\cdot \mid w\right) \qquad (12)$$

In other words, X effectively captures the distribution of the ith shadow element and its ending index conditioned on the $i-1$ previous shadow elements in w. Next we define $H = (X, Y)$ by defining an independently distributed random variable Y

$$\mathsf{P}_{Y\mid X}\left(\cdot \mid x\right) = \mathsf{P}_{Z_{[k+1,\ell]} \mid Z_{[k]}}\left(\cdot \mid x\right) \enspace , \qquad (13)$$

where k denotes the length of x. The random variable Y represents the sampling of a completion of a grafting sequence starting with x. Finally, we define $K = \#(X)$, i.e., K is the index of what we expect to be the last element of the ith element of the shadow sequence.

By assumption $\Pr\left[\rho(H) = 1\right] \geq \Delta$. Thus, if we define $\zeta^\rho_x = \Pr\left[\rho(H) = 1 \mid X = x\right]$, then from Lemma 1 we have the bound $\Pr\left[\zeta^\rho_X < \alpha\Delta \mid \rho(H) = 1\right] \leq \alpha$.

If we define $\theta^\rho_x = \Pr\left[J_i = k \mid \rho(H) = 1 \wedge X = x\right]$, where k is the length of x, then Lemma 3 implies that $\Pr\left[\theta^\rho_X < \beta^2/\ell \,\middle|\, \rho(H) = 1\right] \leq 2\beta$. The union bound finally gives

$$\Pr\left[\theta^\rho_X \geq \alpha\Delta \wedge \zeta^\rho_X \geq \beta \mid \rho(H) = 1\right] \geq 1 - \alpha - 2\beta \enspace , \qquad (14)$$

which concludes the proof. □

6 Accepting Basis Extractor for Shadow Sequences

To construct an extractor for grafting protocols we first show that shadow sequences can be sampled. This trivially gives a basic extractor and a basic sampler of shadow sequences corresponding to the basic algorithms in [Wik18]. The recursive extractor follows by syntactic changes, since it is is defined in terms of the expected value and tail bound for each recursive call and the tail bounds do not change.

6.1 Shadow Sampler

Theorem 2 says that from a suitable prefix $w_{[i]}$ of a shadow sequence we *can* sample a complete accepting shadow sequence w that keeps the prefix fixed, but we also need to ensure that $w_{i+1} \in \mathbb{M}^*_{i+1} \setminus \mathsf{span}(B^*)$, where B^* is a shadow version of an independent set $B \in I_{i+1}$, to ensure that we end up with a basis of $\mathbb{M}^*$. This can be accomplished by sampling every grafting element at depth $i+1$ from $S_{i+1} \setminus \mathsf{span}(B)$ instead of from S_{i+1}. There are at most ℓ such elements in a sequence so the statistical distance between this modified distribution and the original is at most $\ell\omega_{\mathbb{M}_{i+1}}$. Theorem 2 then implies the following lemma, which is proven in Appendix B for completeness. Here $\mathsf{Geo}(\Delta)$ denotes the geometric distribution.

Lemma 4 (Shadow Sampler)**.** *There exists a shadow sampler* $\mathsf{W}^{f,\rho}$ *such that for every grafting function* f*, predicate* ρ*, and* $\alpha \in (0,1)$*, and any input* $(\mathbb{M}, w_{[i]}, B, \Delta_0)$ *such that* $\xi_\rho\left(w_{[i]}, \Delta_0, \beta_0\right) = 1$ *with* $\Delta_0 > \ell\omega_{\mathbb{M}_{i+1}}$*:*

1. *the distribution of the number of calls to* ρ *is bounded by* $\mathsf{Geo}(\beta_0^2\Delta_1'/\ell)$*, and*

2. *the output* w *has prefix* $w_{[i]}$*,* $\rho^*(w) = 1$*, and* $w_{i+1} \in \mathbb{M}^*_{i+1} \setminus \mathsf{span}(B^*)$*, and* $\Pr\left[\xi_\rho\left(w_{[i+1]}, \Delta_1, \beta_1\right) = 1\right] \geq \beta_1$*,*

where $\Delta_1' = \Delta_0 - \ell\omega_{\mathbb{M}_{i+1}}$*,* $\Delta_1 = \alpha\Delta_1'$*, and* $\beta_1 = \frac{1}{3}(1-\alpha)$*:*

We need to change the syntax slightly to accommodate for prefixes needed to sample correctly, but the shadow sampler makes it trivial to construct a basic sampler $\mathcal{S}^{f,\rho}_\alpha$ that from an extendable shadow prefix samples the next shadow element conditioned on acceptance. We similarly denote by $\mathcal{B}^{f,\rho}_\alpha$ the basic extractor for shadow sequences that takes an input $w_{[r-1]}$ and invokes the shadow sampler d_r times with the parameter α, storing the new elements in an initially empty set B^*, to find accepting a set of transcripts with the prefix $w_{[r-1]}$ such that their rth elements form a basis over $\mathbb{M}^*_r$.

6.2 Accepting Basis Extractor

The recursive extractor $\mathcal{R}_\kappa[\mathcal{R}]$, parametrized by a parameter κ, and making recursive calls to $\mathcal{R}$, is virtually identical to that in [Wik18], since it is defined in terms of expected values and tail bounds of a recursive call or the basic extractor. Analytically the situation is equivalent to the original analysis of the basic strategy except for three changes: (1) $\ell\omega_{\mathbb{M}_i}$ replaces the subdensity $\omega_{\mathbb{M}_i}$, (2) we loose a factor 3 for each recursive call due to the threefold use of the union bound, and (3) the expected running time of the basic extractor increases by a factor of $\ell/(1-\alpha_{r-1})^2$. Thus, if we set $\nu_i = 1/\alpha_i$, then we may simply restate the main theorem from [Wik18] with these changes, but we provide a proof in Appendix C.

We consider families of distributions $\mathsf{D}(s,\Delta)$ parametrized by $s \in \mathbb{N}^+$ and $\Delta \in [0,1]$, which satisfy a tail bound of the form $\Pr\left[X \geq k\mu_{\mathsf{D}(s,\Delta)}\right] \leq t^{\mathsf{D}}_s(k)$, where X is distributed according to $\mathsf{D}(s,\Delta)$ and $\mu_{\mathsf{D}(s,\Delta)}$ is the expected value of $\mathsf{D}(s,\Delta)$. For compound geometric distributions we have $t^{\mathsf{CG}}_s(k) = e^{-(k-1-\ln k)s}$ from [Wik18].

Theorem 3 (Extractor)**.** *For every* $\nu_1, \ldots, \nu_{r-1} \in (1,\infty)$ *with* $\nu_i \geq \nu_{i+1}$ *there exist parameters* $\kappa_i = (\alpha_i, \lambda_i)$ *such that the algorithm* $\mathcal{X}_\kappa = \mathcal{R}_{\kappa_1}[\mathcal{R}_{\kappa_2}[\cdots\mathcal{R}_{\kappa_{r-2}}[\mathcal{B}]\cdots]]$*, is a*

$(\epsilon_0, \mathsf{D}_\kappa(\Delta_0))$*-accepting basis extractor for shadow matroid tree of* $\mathbb{M}$ *where:*

$$\epsilon_0 = \ell \sum_{i\in[r]} \omega_{\mathbb{M}_i} \prod_{j\in[i-1]} \nu_j \qquad \textit{(extraction error)} \tag{15}$$

$$\mu_{\mathsf{D}_0(\Delta_0)} \leq \ell \cdot \frac{c_0 \prod_{j\in[r]} d_j}{\Delta_0 - \epsilon_0} \qquad \textit{(expected number of queries)} \tag{16}$$

$$t^{\mathsf{D}_0}_{d_1}(k) \leq t^{\mathsf{CG}}_{d_1}(k) \quad \textit{for } k > 1 \ , \qquad \textit{(tail bound)} \tag{17}$$

where the constant c_0 *is defined by*

$$c_0 = 3^{r+1} \frac{\nu_{r-1}^2}{(\nu_{r-1}-1)^2} \prod_{i\in[r-1]} \frac{\nu_i^2}{(\nu_i - 1)} \cdot \min_{k_i\in(0,1)} \left\{ \frac{k_i}{h^{\mathsf{CG}}_{d_i}(k_i)} \right\} \ . \tag{18}$$

7 Interpretation

We refer the reader to [Wik18] for an in depth intuitive interpretation of the overall recursive formulas. To see that the extraction error ϵ_0 is tight up to a factor r first recall that it upper bounds the soundness error. Then consider a protocol such that: (1) guessing the verifier message of the special sound protocol in any round is sufficient to convince the verifier of a false statement, and (2) that the prover can determine if a guess is correct before the execution is continued. For such a protocol the soundness error is roughly $\epsilon = \sum_{i=1}^{r} \omega_{\mathbb{M}_i}$. In the grafting protocol the prover may probe any round independently with ℓ queries, so the soundness error is bounded by $\ell \max_{i\in[r]}\{\omega_{\mathbb{M}_i}\} \geq \epsilon_0/r$.

When $\Delta_0 \gg \epsilon_0$ the factor ℓ in the extraction error can be ignored and if we set $\nu_i = 2$ the running time is $4 \cdot 3^{r+1}\ell$ times the running time in the interactive case. Consider a prover that effectively executes the basic special sound protocol without grafting any forks until the $(r-2)$th round where it probes $\ell - r - 3$ round $r-1$ messages before completing the last round in exactly one randomly chosen partial execution. Then forking in round $r-2$ requires roughly ℓ samples. Thus, the factor ℓ is necessary and the running time is relatively tight up to a small constant $4 \cdot 3^{r+1}$ factor.

To summarize, our results show that for any constant r and any $(2r+1)$-message special sound protocol the application of the Fiat-Shamir heuristic comes at a cost of a factor ℓ in extraction error and $O(\ell)$ extraction time, respectively, but with the same type of distribution as in the interactive case. In practice the matroid subdensities can typically be decreased by a factor of $1/\ell$ by a different choice of parameters for the protocol and cancel the effect on the extraction error at modest cost in efficiency. Thus, the loss of security from the application of the Fiat-Shamir heuristic in practice is typically no more than $\log \ell + O(1)$ bits of security which is intuitively appealing.

References

[ACK21] Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed ς-protocol theory for lattices. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 549–579. Springer, 2021. doi:10.1007/978-3-030-84245-1_19.

[AFK22] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part I*, volume 13747 of *Lecture Notes in Computer Science*, pages 113–142. Springer, 2022. doi:10.1007/978-3-031-22318-1_5.

[AL21] Martin R. Albrecht and Russell W. F. Lai. Subtractive sets over cyclotomic rings - limits of schnorr-like arguments over lattices. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 519–548. Springer, 2021. doi:10.1007/978-3-030-84245-1_18.

[AS08] Noga Alon and Joel H. Spencer. *The Probabilistic Method, Third Edition*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 2008.

[Bab85] László Babai. Trading group theory for randomness. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 421–429. ACM, 1985. URL: http://doi.acm.org/10.1145/22145.22192, doi:10.1145/22145.22192.

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, 2016. doi:10.1007/978-3-662-49896-5_12.

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1992. doi:10.1007/3-540-48071-4_28.

[BGR98] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 236–250. Springer, 1998. doi:10.1007/BFb0054130.

[BL04] Boaz Barak and Yehuda Lindell. Strict polynomial-time in simulation and extraction. *SIAM J. Comput.*, 33(4):738–818, 2004. doi:10.1137/S0097539703427975.

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, pages 390–399. ACM, 2006. doi:10.1145/1180405.1180453.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi

Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73. ACM, 1993. URL: http://doi.acm.org/10.1145/168588.168596, doi:10.1145/168588.168596.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 1994. doi:10.1007/3-540-48658-5_19.

[dPLS19] Rafaël del Pino, Vadim Lyubashevsky, and Gregor Seiler. Short discrete log proofs for FHE and ring-lwe ciphertexts. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part I*, volume 11442 of *Lecture Notes in Computer Science*, pages 344–373. Springer, 2019. doi:10.1007/978-3-030-17253-4_12.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. doi:10.1007/3-540-47721-7_12.

[FS01] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 368–387. Springer, 2001. doi:10.1007/3-540-44647-8_22.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989. doi:10.1137/0218012.

[Gol00] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, New York, NY, USA, 2000.

[HKR19] Max Hoffmann, Michael Klooß, and Andy Rupp. Efficient zero-knowledge arguments in the discrete log setting, revisited. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 2093–2110. ACM, 2019. doi:10.1145/3319535.3354251.

[JT20] Joseph Jaeger and Stefano Tessaro. Expected-time cryptography: Generic techniques and applications to concrete soundness. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 414–443. Springer, 2020. doi:10.1007/978-3-030-64381-2_15.

[Nef01] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Se-*

curity, Philadelphia, Pennsylvania, USA, November 6-8, 2001., pages 116–125. ACM, 2001. URL: http://doi.acm.org/10.1145/501983.502000, doi:10.1145/501983.502000.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 1996. doi:10.1007/3-540-68339-9_33.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991. doi:10.1007/BF00196725.

[Wik05] Douglas Wikström. A sender verifiable mix-net and a new proof of a shuffle. In Bimal K. Roy, editor, *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, volume 3788 of *Lecture Notes in Computer Science*, pages 273–292. Springer, 2005. doi:10.1007/11593447_15.

[Wik18] Douglas Wikström. Special soundness revisited. *IACR Cryptol. ePrint Arch.*, 2018:1157, 2018. URL: https://eprint.iacr.org/2018/1157.

A Definitions

We recall the definitions introduced in [Wik18].

Definition 19 (Accepting Basis)**.** A basis B of a matroid tree $\mathbb{M}$ is *ρ-accepting* for an $\mathbb{M}$-predicate ρ if $\rho(v) = 1$ for each path v of maximal length in B.

Definition 20 (Accepting Transcript Tree)**.** Let $\mathcal{V}$ be a verifier of a $(2r+1)$-message public coin protocol. A rooted unordered directed tree T with vertex labels $\ell(\cdot)$ is an *accepting transcript tree* for $\mathcal{V}$ if every leaf has depth r and for every path $(u_0, \ldots, u_r)$ in T: $(v_0, a_0, \ldots, v_r, a_r)$ is accepting, where $\ell(u_i) = (v_i, a_i)$.

Definition 21 (Challenge Tree)**.** The *challenge tree* $\mathbb{V}(T)$ of an accepting transcript tree T with vertex labels $\ell(\cdot)$ has the same nodes and edges, but labels defined by $\ell'(u) = v$, where $\ell(u) = (v, a)$.

Definition 22 (Special Soundness)**.** A $(2r+1)$-message public coin-protocol $(\mathcal{P}, \mathcal{V})$ is $\big((\mathbb{M}_1, \ldots, \mathbb{M}_r), p\big)$*-special-sound* for an NP relation R, where $\mathbb{M}_i = (S_i, I_i)$ is a matroid, if the ith message of $\mathcal{V}$ is chosen randomly from S_i, and there exists a *witness extraction algorithm* $\mathcal{W}$ that given an accepting transcript tree T such that $\mathbb{V}(T)$ is basis subtree of $(\{x\}, \mathbb{M}_1, \ldots, \mathbb{M}_r)$ outputs a witness w such that $(x, w) \in \mathsf{R}$ in time p.

B Proof of Lemma 4

The extractor in [Wik18] repeatedly samples complete lists of accepting verifier messages with a slight bias to guarantee independence properties. We intend to essentially execute the original extractor with a shadow predicate ρ^* over a shadow matroid tree $\mathbb{M}^*$.

The original analysis assumes that the lists of verifier messages are sampled uniformly from a matroid, but what is actually necessary for the analysis to work is that they are sampled identically as in the protocol. Thus, nothing prevent us from invoking a modified extractor for our shadow matroid tree, provided that the verifier messages are sampled with the right distribution. Consider the following algorithm.

Definition 23 (Grafted Sequence Sampler)**.** The *grafted sequence sampler* Z^f for a grafting function f takes as input a tuple $(\mathbb{M}, \ell, z, b, B)$, where $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ is a matroid tree with $\mathbb{M}_i = (S_i, I_i)$, $z \in G_{\mathbb{M},f,k}$, and $B \in I_b$ is not a basis and proceeds as follows.

1. For $i = k+1, \ldots, \ell$:
   (a) Compute $d = \delta_f(z)$ and sample z_i randomly in $S_d \setminus \mathsf{span}(B)$ if $d = b$, and in S_d otherwise.
   (b) Append z_i to z.
2. Return z.

The running time of the algorithm is, apart from sampling in the complement of $\mathsf{span}(B)$, identical to executing the protocol, i.e., its running time corresponds to evaluating the predicate ρ if we ignore the cost of sampling verifier messages.

The algorithm is used below to sample an accepting shadow sequence which has a prefix $w_{[i-1]}$ and we need w_i to not be contained in a set $B^* \in I_b^*$, but at the time of sampling the grafted elements we do not know which sample from S_i will determine independence in I_b^*. Thus, we make sure that all grafted elements from S_b that make up w_i are from $S_b \setminus B$ instead, where B is the projection of B^* to their middle elements. It may seem that this approach introduces an unnecessarily large error in the distribution of the output, i.e., roughly $\ell\omega_{\mathbb{M}_b}$ instead of $\omega_{\mathbb{M}_b}$, but this seems unavoidable. Next we use the grafted sequence sampler to implement a shadow sampler.

Definition 24 (Shadow Sampler)**.** The *shadow sampler* algorithm $\mathsf{W}^{f,\rho}$, where f is a grafting function, takes as input a tuple $(\mathbb{M}, \ell, w_{[i]}, B)$, where $\mathbb{M} = (\mathbb{M}_0, \ldots, \mathbb{M}_r)$ is a matroid tree, $\rho : G_{\mathbb{M},f,\ell} \to \{0,1\}$, $w_{[i]}$ is a prefix of a shadow sequence corresponding to a grafted sequence $z_{[k]} \in G_{\mathbb{M},f,k}$, and $B \in I_{i+1}$ is not a basis. Repeat:

1. Compute $z = \mathsf{Z}^f(\mathbb{M}, \ell, z_{[k]}, i+1, B)$ and set $j = \iota_f(z)$ and $w' = \sigma_f(z)$.
2. If $\rho(z) = 1$ and $j_i = k$, then return w'.

Let z be the grafted sequence sampled by $\mathsf{W}^{f,\rho}$ such that $w' = \sigma_f(z)$ is returned, and set $j = \iota_f(z)$. By construction $w_{[i]}$ is a prefix of w' viewed as grafted sequences and it only returns if $j_i = k$. Thus, Lemma 2 implies that $w'_{[i]} = w_{[i]}$. Furthermore, $\mathsf{W}^{f,\rho}$ only returns if $\rho(z) = 1$ which implies that $\rho^*(w') = 1$. Finally, every element from S_{i+1} is sampled from the subset $S_{i+1} \setminus \mathsf{span}(B)$, which implies that $w_{i+1} \in S_{i+1}^* \setminus \mathsf{span}(B^*)$. This proves the first claim of Lemma 4.

If $\xi_\rho(w_{[i]}, \Delta_0, \beta_0) = 1$, then in each iteration the probability of returning is at least $\beta_0^2\Delta_0/\ell$. Thus, the distribution of the number of calls to ρ is bounded according to the second claim. The third claim of Lemma 4 follows directly from Theorem 2.

C Proof of Theorem 3

We now have the subroutines needed to derive a recursive extractor from the construction in [Wik18] almost by syntactic changes. The only essential difference is that all algorithms need the complete prefix of a partial shadow sequence as input to sample completions with the right distribution.

We need to modify the notion of an accepting basis extractor to allow for the additional parameter ℓ and the parameter β from Theorem 2.

Definition 25 (Accepting Basis Extractor)**.** A probabilistic polynomial time algorithm $\mathcal{X}_\kappa$ parametrized by $\kappa \in \{0,1\}^*$ is a $(\epsilon_\kappa, \mathsf{D}_\kappa(\ell, \Delta, \beta))$-*accepting basis extractor* with *extraction error* ϵ_κ for the matroid tree $\mathbb{M}' = (\{w_{[i]}\}, \mathbb{M}_{i+1}^*, \ldots, \mathbb{M}_r^*)$, where $\mathsf{D}_\kappa(\ell, \Delta, \beta)$ for fixed κ is a family of distributions on $\mathbb{N}$ parametrized by $\ell \in \mathbb{N}$, $\Delta \in [0,1]$, $\beta \in (0,1)$, if for every grafting function f and predicate $\rho : G_{\mathbb{M},f,\ell} \to \{0,1\}$ the following holds.

If $\Delta_i > \epsilon_\kappa$, $\beta_i > 0$, and $\xi_\rho(w_{[i]}, \Delta_i, \beta_i) = 1$, then $\mathcal{X}_\kappa^{f,\rho}(\mathbb{M}, \ell, w_{[i]}, \Delta_i, \beta_i)$ outputs a ρ-accepting basis of $\mathbb{M}'$, where the distribution of the number of $\rho(\cdot)$-queries is bounded by $\mathsf{D}_\kappa(\ell, \Delta_i, \beta_i)$.

Definition 26 (Recursive Extractor)**.** Let $\mathbb{M} = (\mathbb{M}_1, \ldots, \mathbb{M}_r)$ be a matroid tree and assume that $\mathcal{R}$ is a $(\epsilon_i, \mathsf{D}_i(\ell, \Delta, \beta))$-accepting basis extractor for matroid trees of the form $(\{w_{[i]}\}, \mathbb{M}^*_{i+1}, \ldots, \mathbb{M}^*_r)$. The *recursive extractor* $\mathcal{R}_\kappa[\mathcal{R}]$, where $\kappa = (\alpha_i, \lambda_i)$ and $\alpha_i, \lambda_i \in (0,1)$ proceeds as follows on input $(\mathbb{M}, \ell, w_{[i-1]}, \Delta_{i-1}, \beta_{i-1})$.

1. Set $\Delta_i = \alpha_i(\Delta_{i-1} - \ell\omega_{\mathbb{M}_i})$, $\beta_i = \frac{1}{3}(1-\alpha_i)$, $k = k^{\mathsf{D}_i}(\lambda_i)$, and $\mu = \mu_{\mathsf{D}_i(\ell,\Delta_i,\beta_i)}$.
2. Set $B^* = \varnothing$ and $T = \varnothing$.
3. While $|B^*| < d_i$:
   (a) Compute $w = \mathcal{S}_{\alpha_i}^{f,\rho}(\mathbb{M}, \ell, w_{[i-1]}, B^*, \Delta_{i-1}, \beta_{i-1})$.
   (b) Extract subtree $t = \mathcal{R}^{f,\rho}(\mathbb{M}, \ell, w_{[i]}, \Delta_i, \beta_i)$, but interrupt the execution and set $t = \perp$ if it attempts to make more than $k\mu$ queries.
   (c) If $t \neq \perp$, then set $B^* = B^* \cup \{w_i\}$ and $T = T \cup \{t\}$.
4. Return the accepting basis tree T.

Above we abuse notation and set $B^* = B^* \cup \{w_i\}$ despite that the ground set of the shadow matroid $\mathbb{M}^*_i$ consists of triples (see Definition 14), i.e., strictly speaking we first turn w_i into a triple where the challenge from the basic protocol we focus on is the middle element.

The following lemma and corollary follows mutatatis mutandi from the corresponding proof in [Wik18], where we use indices to illustrate the similarity with the original recursive formulas.

Lemma 5 (Recursive Extractor)**.** *The algorithm $\mathcal{R}_\kappa[\mathcal{R}]$ is a $(\epsilon_{i-1}, \mathsf{D}_{i-1}(\ell, \Delta_{i-1}, \beta_{i-1}))$-accepting basis extractor, where $\epsilon_{i-1} = \epsilon_i/\alpha_i + \ell\omega_{\mathbb{M}_i}$ and*

$$\mathcal{G}_{\mathsf{D}_{i-1}(\ell,\Delta_{i-1},\beta_{i-1})}(z) = \prod_{i=1}^{d_i} \mathcal{G}_{\mathsf{Geo}(\beta_i\lambda_i)}\left(\mathcal{G}_{\mathsf{Geo}(\beta_{i-1}^2\Delta_{i-1}/\ell)}(z)\, z^{k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\ell,\Delta_i,\alpha_i)}}\right) \ , \tag{19}$$

defined by $\beta_i = \frac{1}{3}(1-\alpha_i)$ and $\Delta_i = \alpha_i(\Delta_{i-1} - \ell\omega_{\mathbb{M}_i})$.

Corollary 1 (Recursive Extractor)**.** *The distribution $\mathsf{D}_{i-1}(\ell, \Delta, \beta)$ satisfies*

$$\mu_{\mathsf{D}_{i-1}(\ell,\Delta_{i-1},\beta_{i-1})} = \frac{3d_i}{(1-\alpha_i)\lambda_i}\left(\frac{\ell}{\beta_{i-1}^2\Delta_i} + k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\ell,\Delta_i,\alpha_i)}\right) \tag{20}$$

$$t^{\mathsf{D}_{i-1}}(k) \leq t_{d_i}^{\mathsf{CG}}(k) \quad \textit{for } k \in (1,\infty) \ . \tag{21}$$

If $\beta_{i-1}^2 \geq \beta_i^2\alpha_i$, then the term $\ell/(\beta_{i-1}^2\Delta_i)$ can be dropped by observing that the initial sample can be reused in the recursive call and recursive calls are slightly more expensive. When $\beta_i = \frac{1}{3}(1-\alpha_i)$ this is always the case, since $(1-\alpha_{i-1})^2 \geq \alpha_i(1-\alpha_i)^2$ for $\alpha_{i-1}, \alpha_i \in (0,1)$.

Thus, the only change in the expected value compared to the basic case in [Wik18] is a factor of three in each recursive call (and there are $r-1$ levels of recursion), and that the expected value for the basic extractor is increased by a factor of $3^2\ell/(1-\alpha_{r-1})^2$. Setting $\nu_i = 1/\alpha_i$ gives the theorem.

D Grafting Prover

An honest grafting prover obviously does not need the liberty to graft additional branches to an execution.

Definition 27 (Grafting Prover)**.** If $(\mathcal{P}, \mathcal{V})$ is a $(\mathbb{M}_1, \ldots, \mathbb{M}_r)$-special sound protocol, then on common input x, and private input w such that $(x, w) \in \mathsf{R}$, the *grafting prover* $\mathsf{G}[\mathcal{P}]$ proceeds as follows.

1. Start a simulation of $\mathcal{P}$ on input (x, w) and when it outputs a message a_0, hand $(0, a_0)$ to the verifier.
2. For $i = 1, \ldots, r$:
   (a) Wait for a message v_i from the verifier.
   (b) Continue the simulation of $\mathcal{P}$ on input v_i, until it outputs a message a_i, hand (i, a_i) to the verifier.

E Omitted Proofs

Proof of Lemma 1. We have $\mathrm{E}\left[1/\delta_X \,|E\right] = \sum_{x\in[X]} \Pr\left[X = x \,|E\right]/\delta_X = 1/\Pr\left[E\right] \leq 1/\Delta$. Markov's inequality then implies $\Pr\left[\delta_X < \alpha\Delta \,|E\right] = \Pr\left[1/\delta_X > 1/(\alpha\Delta) \,|E\right] \leq \alpha$. □

F Basic Definitions

Readers who are uncomfortable with the language of matroids may safely think of a matroid $\mathbb{M} = (S, I)$ as a vector space $\mathbb{Z}_q^N$ for a prime integer q and a natural number N. Then independence means linear independence, the set I of sets of independent vectors is clearly non-empty and satisfies the requirements below, and a span is simply a linear subspace spanned by a set of vectors.

Definition 28 (Matroid)**.** A *matroid* is a pair (S, I) of a *ground set* S and a set $I \subset 2^S$ of *independence sets* such that:

1. I is non-empty,
2. if $A \in I$ and $B \subset A$, then $B \in I$, and
3. if $A, B \in I$ and $|A| > |B|$, then there exists an element $a \in A \setminus B$ such that $\{a\} \cup B \in I$.

Definition 29 (Submatroid)**.** Let (S, I) be a matroid and $S' \subset S$. The *submatroid* induced by S' is the pair (S', I') defined by $I' = I \cap 2^{S'}$.

Definition 30 (Basis)**.** Let (S, I) be a matroid. A set $B \in I$ such that $B \cup \{x\} \notin I$ for every $x \in S \setminus B$ is a *basis*.

Definition 31 (Rank)**.** The *rank* of a matroid (S, I) is the unique cardinality of each basis in I.

Definition 32 (Rank of Set)**.** Let (S, I) be a matroid and $A \subset S$. The *rank* $\mathsf{rank}(A)$ of A is the rank of the submatroid induced by A.

Definition 33 (Span and Flats)**.** Let (S, I) be a matroid and $A \subset S$. The *span* of A is defined by $\mathsf{span}(A) = \{x \in S \mid \mathsf{rank}(A \cup \{x\}) = \mathsf{rank}(A)\}$ and A is a *flat* if $\mathsf{span}(A) = A$.