IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 3, 29 pages.

https://doi.org/10.62056/aep2c3w9p
Check for updates

Special Soundness Revisited

Douglas Wikström[a]

KTH Royal Institute of Technology, Stockholm, Sweden

Abstract. We generalize and abstract the problem of extracting a witness from a prover of a special sound protocol into a combinatorial problem induced by a sequence of matroids and a predicate, and present a parametrized algorithm for solving this problem.
The parametrization provides a tight tradeoff between the running time and the extraction error of the algorithm, which allows optimizing the parameters to minimize: the soundness error for interactive proofs, or the extraction time for proofs of knowledge.
In contrast to previous work we bound the distribution of the running time and not only the expected running time. Tail bounds give a tighter analysis when applied recursively and a concentrated running time.

1 Introduction

The notion of zero-knowledge proofs is fundamental in cryptography. It was introduced by Goldwasser, Micali, and Rackoff [GMR89] as a way for one party to interactively convince another party that a given statement is true without disclosing anything else. Another related notion formalized by Bellare and Goldreich [BG92], are proofs of *knowledge*. Here the prover convinces the verifier that it holds some piece of information, e.g., it could show that it holds a witness of an NP relation. This notion makes sense even for statements that are true by definition, but for which a witness is hard to compute.

The completeness of a protocol is the probability that an honest prover convinces an honest verifier on a true statement, i.e., the probability that the protocol serves its purpose for honest parties. The soundness of an interactive proof system is captured by its *soundness error*. This is an upper bound on the probability that an honest verifier in an interaction with a (possibly malicious) prover accepts a false statement as true.

A protocol is said to be a proof of knowledge if there is an extraction algorithm such that for every prover and every statement a witness is output in expected time $\mathsf{poly}/(\Delta-\epsilon)$, where Δ is the probability that the honest verifier is convinced and ϵ is the *knowledge error*. The expected value is taken only over the internal randomness of the extractor. The knowledge error captures the probability that a prover can convince a verifier in the protocol without *knowing* a witness of the statement in the sense that it can be extracted efficiently.

It is important to understand that the extractor has complete control over the prover, e.g., it may rewind and complete multiple executions from any point of the execution, and uses the prover as an oracle. A knowledge error ϵ implies a soundness error of at most ϵ, since we may view the existence of the extractor as a probabilistic proof [AS08] that a witness exists, but the reverse implication does not necessarily hold since the extractor must be efficient.

We refer the reader to [Gol00] for a thorough discussion and historial notes.

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

1.1 Special Soundness and Extraction

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 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 natural generalization of Cramer et al.'s notion we require that the accepting transcripts form a tree, i.e., the executions are identical to start with and then successively branch to form a tree, where the verifier messages at each branching point are "independent". The notion of independence is protocol dependent, but it is readily captured using matroids and usually corresponds to inequality [Sch91] or linear independence [BGR98].

Recall that in proofs of knowledge [BG92] we consider the prover as a *deterministic* next-message function to allow rewinding. Moreover, in public coin protocols the verifier's messages do not depend on the prover's messages, so we can consider the prover and verifier jointly as a predicate on a sequence of verifier messages.

We construct and analyze an algorithm that extracts a tree such that every path satisfies the predicate and the children of each node is a basis relative to a given matroid. This reduces the problem of constructing a knowledge extractor for a special-sound protocol to the inherently protocol-dependent construction of a procedure that computes a witness from such a tree of accepting transcripts.

2 Contributions

The main contribution of this work is an exact extraction theorem that can be applied directly and rigorously to special sound protocols to get exact security guarantees. The generalization of special soundness using matroids captures a natural class of protocols.

The concrete motivation of this work is the need for exact analysis of the real-world applications of generalizations of Schnorr's protocol [Sch91], e.g., batch proofs of Bellare et al. [BGR98] and proofs of shuffles, e.g., [Nef01, FS01, Gro03, TW10]. Exact reductions are *necessary* in such applications, since it is otherwise not possible to choose concrete security parameters with any real-world guarantee of security.

Our results reduce the exact analysis of soundness and knowledge extraction such protocols to proving that a witness can be computed from a tree of accepting transcripts that are sufficiently independent. Our main theorem is parametrized, since this is necessary to allow derivation of optimized concrete security parameters of any such protocol. Below we give an informal statement.

Theorem 1 (Informal)**.** *If $(\mathcal{P}, \mathcal{V})$ is an r-round public coin protocol such that a witness can be computed in polynomial time T from a rooted tree of accepting transcripts provided that: (a) each node at depth i has d_i children, (b) the transcripts are "independent", and (c) the probability that a randomly chosen completion from depth $i-1$ of a partial transcript is "independent" from previous transcripts in the ith round is at least $1 - \omega_i$, then:*

1. *There exists an extraction algorithm $\mathcal{X}$ such that if a prover $\mathcal{P}^*$ convinces $\mathcal{V}$ with probability Δ, then the expected number of queries to $\mathcal{P}^*$ needed by $\mathcal{X}$ to extract such a tree of accepting transcripts is $c \prod_{j \in [r]} d_j / (\Delta - \epsilon)$, where the constant $\sum_{i \in [r]} \omega_i < \epsilon < 1$ can be chosen arbitrarily and the constant $c > 0$ is derived explicitly from ϵ.*

2. *There is a knowledge extractor such that: if $\Delta > \epsilon$, then a witness can be extracted in time $c \prod_{j \in [r]} d_j / (\Delta - \epsilon) + T$.*

3. *The distribution of the running time of $\mathcal{X}$ satisfies a concentration bound determined by d_1, i.e., for large d_1 the running time is almost strictly polynomial.*

In applications the matroid subdensity ω_i is exponentially small, the extraction error ϵ must be chosen exponentially small, and c is a constant of practical size. The degree d_1 needed at the root of the transcript tree is large in many applications, e.g., it may be the number of ciphertexts decrypted in a given batch.

Implicit in our approach is that it may be applied rigorously to protocols embedded within larger protocols using extension rather than composition, in particular when instances are maliciously chosen. We give concrete examples and discuss the extension of protocols in Section B.

More broadly, matroids may be the right language to capture and analyze independence requirements which appear in security proofs in various guises. Similarly, arguing about distributions instead of expected values should be the norm in exact security analysis, since it gives tighter bounds, more insight, and simplifies rigorous reuse of previous work.

Related work. In prior work Bootle et al [BCC+16] gives a forking lemma for interactive constant-round protocols, but it is not exact.

Our result was made public 2018. Follow-up work by Attema et al [ACK21] study $(d_1, \ldots, d_r)$-special-soundness. Unfortunately, it is not clearly stated that this is equivalent to special soundness for uniform matroids as defined here, i.e., matroids where every set of at most r elements is an independence set.

For such uniform matroids the sample space can always be made small. Thus, their focus is on the *special case* of small sample spaces and uniform matroids. They also use the alternative definition of proofs of knowledge where the extractor may fail with small probability, but this is merely a presentational detail.

We analyze the *general case* where there is a requirement on independence between accepting transcripts to extract a witness, which is typical in proofs over groups. The motivation of our work was the need for an exact analysis of, and concrete security parameters for, batch proofs of decryption [BGR98] and proofs of shuffles [TW10] which are deployed in real-world national and local electronic voting systems, including rigorous analysis of restricted sample spaces. The special case studied in [ACK21] does not suffice, since it only considers uniform matroids, i.e., there is no guarantee that a witness can be computed from the extracted transcripts.

In the general case the difference between sampling with or without replacement is not essential, since an exponentially large set of values are eliminated from sampling with each additional accepting challenge found. In [ACK21] the method of sampling matters, since a single value is eliminated from sampling with each accepting transcript found when uniform matroids are considered.

However, our analysis applies in a straightforward way by replacing geometric and negative binomial distributions by their hypergeometric and negative hypergeometric siblings and adjust the tail bounds, which is effectively what is done in [ACK21] using combinatorial language instead of statistical language and the alternative definition of a proof of knowledge.

Other follow-up results such as [HKR19, dPLS19, JT20, AL21] are less relevant to understand our contribution, since they have a different or broader focus.

3 Knowledge Extraction

Before we introduce a definition of special soundness based on matroids we introduce notation and discuss knowledge extraction in general. Recall that $\langle \mathcal{P}^*, \mathcal{V}_c \rangle(x)$ denotes the verdict of $\mathcal{V}$ regarding an interaction with a prover $\mathcal{P}^*$ on common input x and using a random tape $c = (v_1, \ldots, v_r)$ of challenges. We may define a predicate on a sequence of verifier messages that captures the verdict of the verifier at the end of an interaction.

Definition 1 (Prover Predicate)**.** The *prover predicate* $\rho[\mathcal{P}^*]$ for a public-coin protocol $(\mathcal{P}, \mathcal{V})$ is defined by $\rho[\mathcal{P}^*](v) = \langle \mathcal{P}^*, \mathcal{V}_c \rangle(v_0)$, where $v = (v_0, \ldots, v_r)$ and $c = (v_1, \ldots, v_r)$, i.e., v_0 is the instance and c is the list of challenges of the verifier.

We state the definition for *public coin* protocols for clarity, but it is natural for any proof of knowledge by considering c to be the random tape of the verifier instead. In fact, any non-trivial proof of knowledge requires the extractor to find a set of transcripts, since otherwise it is possible to compute the witness directly from a single execution, or no execution at all. It is conceivable that even partial or rejecting transcripts are useful to the extractor, but in the worst case only accepting transcripts give any knowledge of the witness. For zero-knowledge protocols it is easy to see that it does not suffice to collect a random set of accepting transcripts, since the witness could then be computed after a number of sequential executions. Thus, we consider only accepting transcripts that form a tree.

Definition 2 (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 $\ell(u_i) = (v_i, a_i)$.

Note that v_0 corresponds to the instance of the execution that may be chosen jointly. This notation makes more sense when one considers the protocol as embedded into a larger protocol where the instance is chosen as the result of a random process under the influence of the adversary. Furthermore, if the prover starts, then the first verifier message is empty. Thus, for any proof of knowledge we may divide the problem of extracting a witness into: (i) the problem of finding a suitable accepting transcript tree, and (ii) computing the witness from such an accepting transcript tree.

In a suitable accepting transcript tree the paths from the root to the leaves must be in "generic position" with respect to a protocol-dependent equation system that determines when a witness can be computed efficiently.

Special sound protocols is a class of protocols where the independence requirements on an accepting transcript tree can be described concisely in terms of the verifier messages as elements of matroids. Thus, we introduce notation to project the labels of a transcript tree to their verifier message parts to allow us to express such conditions.

Definition 3 (Challenge Tree)**.** The *challenge tree* $\mathbb{V}(T)$ of an accepting transcript tree T with vertex labels $\ell(\cdot)$ has the same nodes and vertices, but labels defined by $\ell'(u) = v$, where $\ell(u) = (v, a)$.

4 Matroids

Recall that a matroid $\mathbb{M} = (S, I)$ consists of a *ground set* S and a non-empty set I. Each element of I is called an *independence set* and is a set of elements from the ground set S which satisfy natural independence properties: (i) if the elements of a set are independent, then so are the elements of any subset, and (ii) we can extend a set of independent elements until it forms a *basis*, which is a set of independent elements of maximal size. The number of elements needed to form a basis is called the *rank* of the matroid, since all bases have the same number of elements. A subset of the ground set induces a submatroid. A *flat* is a maximal submatroid for a given rank. For completeness we provide one standard way of formalizing matroids in Appendix D and concrete examples below.

The two most common examples of matroids in the literature are essentially vector spaces over finite fields and matroids that capture inequality, but the ground sets may be restricted for practical reasons.

Example 1 (Inequality Matroid)**.** The inequality matroid (S, I) over a ground set S has as independent set I the set of all subsets of S of size at most two.

Example 2 (Vector Space as Matroid)**.** A vector space $\mathbb{Z}_q^N$ over a finite field $\mathbb{Z}_q$, where q is prime, can be viewed as a matroid $\left(\mathbb{Z}_q^N, I\right)$, where I is the set of all sets of linearly independent vectors.

Example 3 (Flats and Submatroids)**.** Consider a vector space $\mathbb{Z}_q^N$ as a matroid. Then a flat is simply a linear subspace. A subset $S = [0, 2^t]^N$ of its groundset, where $t < \log q$, induces a submatroid (S, I) where I is the set of all sets of linearly independent vectors in S. The induced matroid is not a vector space or even a group, but it inherits the combinatorial properties of the vector space to some extent.

We denote the *singleton matroid* with a singleton ground set $\{u\}$ and independence set $\{\varnothing, \{u\}\}$ by $\{u\}$ and introduce some additional notation that allow us to consider a list of matroids as a tree.

Definition 4 (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.

A matroid tree is unordered, but the children of each node are labeled uniquely. Thus, we abuse notation and identify a node with its label. We use $\mathbb{M}$ to denote both the matroid tree and the list of matroids.

Definition 5 (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 $\mathbb{M}_i$.

4.1 Restricted Verifier Messages and Submatroids

In the above examples every submatroid of the same rank has the same cardinality, which means that the fraction $|A|/|S|$ is the same for every flat A of a given rank. In our applications we need this fraction to remain exponentially small, but we need to relax the requirement to make room for oddities introduced in cryptographic protocols.

Consider a protocol where, conceptually, the verifier picks challenges from the vector space $\mathbb{Z}_q^N$, i.e., the ground set of the matroid in Example 2. This set may be unnecessarily large in practice, so to improve efficiency the verifier messages could instead be chosen over $\{0,1\}^{n_v \times N} \subset \mathbb{Z}_q^N$ for some $n_v < \log q$.

In our formalization this simply means that we consider the submatroid induced by the subset $\{0,1\}^{n_v \times N}$. The submatroid inherits the relevant independence sets, but is "ragged" in that elements are eliminated from the ground set.

Definition 6 (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* $\omega_{\mathbb{M}} = \omega_{\mathbb{M},d}$.

Above we intentionally refrain from defining the ith subdensity to be $|A|/|S|$ and settle for an upper bound, since this quantity may be hard to derive exactly.

Note that we have $\omega_{\mathbb{M},1} = 0$ for every non-trivial matroid $\mathbb{M}$. In Example 2 the ith subdensity is q^{i-N-1} and in Example 1 the 2nd subdensity is $1/|S|$. In the induced submatroid discussed above this is instead $2^{n_v(i-N-1)}$.

5 Special Soundness

Accepting transcript trees of Schnorr's protocol [Sch91] (see Definition 17) from which a witness can be computed efficiently can be characterized as those where the verifier messages are independent with respect to the inequality matroid of Example 1.

This characterization generalizes in a natural way to any set of verifier messages, any number of rounds, and any number of independent accepting transcripts needed for extraction, and given our extraction theorem the matroids capture the soundness and knowledge properties of the protocol. Furthermore, it allows the extraction theorem to be applied directly to the analysis of protocols embedded in larger protocols, and even more broadly as explained in Section B.

Definition 7 (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.

In Section C we generalize the definition to capture what we call *piece-wise* special sound protocols. The knowledge extractor for such a protocol invokes a constant number of extractors of the form we have considered above, but it may also use information gathered from previous calls in the next in order to extract all the information needed to compute the witness.

6 The Extraction Problem

Suppose that $\mathbb{M}_0, \ldots, \mathbb{M}_r$ are matroids with ground sets $S_0, \ldots, S_r$ respectively. We consider an unordered complete tree such that the children of a node at depth $i-1$ are identified (or more precisely labeled) with the elements of S_i. To ensure that this is a tree and not a forest we require that S_0 is a singleton set. The element it contains is mostly used as a placeholder for the root, but it is essential that it remains a variable for applications in general settings.

The predicate from Definition 1 that captures both the computations performed by the prover during execution and the computations performed by the verifier to reach a verdict has the following form.

Definition 8 (Predicate)**.** An $\mathbb{M}$-*predicate*, where $\mathbb{M} = (\mathbb{M}_0, \ldots, \mathbb{M}_r)$ is a matroid tree, is a function of the form $\rho : \bigtimes_{i \in [0,r]} S_i \to \{0, 1\}$.

The goal is to find a basis subtree such that: for every inner node at depth $i-1$ its children is a basis of $\mathbb{M}_i$, and the predicate ρ evaluates to 1 on every path in the subtree from the root to a leaf. We call this an accepting basis.

Definition 9 (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.

6.1 What Can We Expect?

The required tree structure implies that any extractor must find at least $d = \prod_{i \in [r]} d_i$ accepting executions, where d_i is the rank of $\mathbb{M}_i$. If we treat ρ as an oracle, then a first guess might be that the expected number of queries of an optimal extractor is $O\left(d/\Delta\right)$, where $\Delta = \Pr\left[\rho(v) = 1\right]$.

However, we also need to take into account the restrictions imposed by the matroids on the nodes at each level. Consider a node u at depth $i-1$. In general we cannot expect to simply pick a basis of $\mathbb{M}_i$ to be children of u and extend them to paths accepted by ρ, since the conditional probabilities of success for the children are not necessarily sufficiently large. One natural idea is to sequentially identify children and build both a basis B for $\mathbb{M}_i$ and recursively build subtrees for which the children are the roots.

Suppose that we are in the process of doing this and have an independence set B of $j < d_i$ children for which we have extracted subtrees. Then the next child of u must be chosen in $S_i \setminus \mathsf{span}(B)$, so without a deeper understanding of the distribution of accepting paths we must accept an additive loss of $\omega_{\mathbb{M}_i,j} = |\mathsf{span}(B)|/|S_i|$ in the success probability. Collecting statistics about the distribution of accepting paths precise enough to avoid this has similar complexity as solving the extraction problem itself and is therefore too costly. Thus, a somewhat more realistic goal for an efficient extractor is an expected running time of $O\left(d/(\Delta - \epsilon)\right)$, where $\epsilon = \sum_{i\in[r]} \omega_{\mathbb{M}_i}$ and $\omega_{\mathbb{M}_i} = \max_{j\in[d_i]} \omega_{\mathbb{M}_i,j}$.

Minimizing ϵ is important for protocols where we do not care about the running time of the extractor and only use it as a probabilistic proof to argue that a witness exists. This establishes ϵ as the soundness error of the protocol viewed as an interactive *proof.* However, if we view the protocol as a proof of *knowledge*, then the running time of the extractor plays an important role, since it influences the running time of a security reduction of an invoking protocol, which in turn determines the running time of an algorithm that breaks a complexity assumption. Minimizing ϵ increases the constant factor drastically.

6.2 On the Distribution of the Running Time

In the discussion above we have only considered the *expected* running time μ of a potential extractor $\mathcal{X}$. When this is not enough, the standard approach is to apply Markov's inequality and conclude that $\mathcal{X}$ completes within time 2μ with probability at least $1/2$, so the number of attempts we need to extract a witness has geometric distribution with probability $1/2$.

However, the discussion above suggests that the running time of an extractor may be concentrated due to the large number of relatively independent samples needed to extract the tree. To see this, consider a tree where the accepting paths are uniformly distributed. Then we would expect the number of samples needed by the extractor to have (almost) negative binomial distribution with parameter $d = \prod_{i\in[r]} d_i$ and probability $\Delta - \epsilon$.

Unfortunately, the tree is constructed by the adversary, so for many nodes the conditional probability Δ' of finding an extension to an accepting path may be very low. Any unsupervised attempt to simply call a recursive routine that extracts a subtree at such a point will give a running time that is at best geometrically distributed with probability Δ'. Given even a moderately large d, the probability of encountering such a node is high, and the total running time would then be a sum that is dominated by the running times of the extractions from such nodes (see Jansson [Jan17] for how bounds on sums of geometric distributions with different probabilities behave).

Thus, every strategy must have a mechanism to interrupt *all* subroutine calls that take too long to complete. Thus, we may hope that the number of queries made is essentially a constant times a random variable with negative binomial distribution with parameter d_1 and probability $\Delta - \epsilon$, where the scaling factor depends on the cost of a recursive call, and this is the type of result we achieve.

6.3 Strategy

In Section 6 we abstracted a prover and a verifier as a predicate. An *accepting basis extractor* extracts a set of transcripts from which a witness can be computed. We give a rigorous definition Section 7. In general any strategy that treats the prover as a blackbox has limited information about the suitability of verifier messages beyond if a complete transcript is accepting or not. Thus, we are restricted to try to extract a tree of transcripts in some topological order.

The base case at depth $r-1$ for a depth r tree consists of simply sampling accepting leafs. It is easy to see that this requires a number of queries that is bounded by the negative binomial distribution. This is described in Section 8.2, where we also describe how a sample can be validated probabilistically.

The general strategy is natural and described in Section 8.3. Since it is recursive we describe it as if we start at the root. We sample paths until we find an accepting path v. Consider the child u of the root in such a path. If the conditional probability of finding an accepting path through u is δ_u, then Markov's inequality implies that it is at least $\alpha\Delta$ with probability $1-\alpha$ for $\alpha \in (0,1)$.

We can invoke the algorithm recursively using u as a root and interrupt the execution if it takes too long, but if the recursive call is costly it is worthwhile to first validate that u is reasonably good.

We can do this by sampling random paths through u and sample a new node if we do not encounter sufficiently many accepting paths within a given number of attempts. We then balance the cost of sampling against the cost of failed attempts to execute the strategy recursively. For nodes close to the root sampling is relatively cheap compared to the cost of an interrupted execution.

This strategy gives a number of parameters for each recursive call. A parameter α determines how close to Δ we want δ_u to be. A parameter β captures the additional loss we have if we validate the candidate, since we cannot do this perfectly. The probability that validation gives the right result is determined by a parameter γ, and finally a parameter λ determines the probability that a recursive call completes.

We derive expressions for the extraction error and the expected value, and give a tail bound for the number of queries made by the extractor in terms of these parameters. This allows choosing good parameters for an exact security analysis of any special-sound protocol.

7 Accepting Basis Extractors

For a matroid tree $\mathbb{M}$ we let $S = \bigtimes_{i\in[0,r]} S_i$, where $\mathbb{M}_i = (S_i, I_i)$ and $S_0 = \{v_0\}$ for some v_0. Recall that S_i is the set of possible ith verifier messages, so S is the set of possible lists of verifier messages. We define the probability that the predicate ρ is satisfied for a list of verifier messages chosen uniformly at random by $\Delta_\rho(\mathbb{M}) = |\{\rho(v) = 1 \mid v \in S\}|/|S|$. When $\mathbb{M}$ is clear from the context we drop $\mathbb{M}$ and write Δ_ρ.

Let D and D' be distributions over $\mathbb{N}$. We say that D is *bounded* by D' if D stochastically dominates D', i.e., if for random variables X and X' distributed according to D and D', respectively, and every $x \in \mathbb{N}$: $\Pr[X \le x] \ge \Pr[X' \le x]$.

Definition 10 (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 $\Delta_\rho(\mathbb{M}) \ge \Delta_0 > \epsilon_\kappa$: $\mathcal{X}_\kappa^{\rho(\cdot)}(\mathbb{M}, \Delta_0)$ outputs a ρ-accepting basis of $\mathbb{M}$, where the distribution of the number of $\rho(\cdot)$-queries is bounded by $\mathsf{D}_\kappa(\Delta_0)$.

This definition is more precise than the definition of a proof of knowledge in that it bounds the distribution, and not only the expected value, of the number of queries made by the extractor. The extractor and the corresponding extraction error ϵ_κ and distribution $\mathsf{D}_\kappa(\Delta_0)$ can be modified using the parameter κ.

We require a lower bound Δ_0 on the accept probability Δ_ρ as an explicit input to the extractor to guarantee the expected behavior, but this is typically known as the result of a contrapositive assumption about a given prover.

We stress that the extraction error ϵ_κ is a property related to a particular algorithm and parameter κ and is neither necessarily the soundness error nor the knowledge error of the protocol from which the matroid tree is derived. It merely provides an upper bound on both when the running time is not too large and a witness can be efficiently computed from an extracted accepting basis.

8 Constructing Accepting Basis Extractors

We split the description of extractors into subroutines and analyze them separately to emphasize the structure of the main algorithm and the interplay between the parameters that we consider below. When convenient we use generating functions to describe distributions, e.g., a distribution D over $\mathbb{N}$ has probability generating function $\mathcal{G}_{\mathsf{D}}(z)$.

8.1 Notation for Bounds

Consider a random variable X taking values in $\mathbb{N}$. Furthermore, suppose that it has distribution $\mathsf{D}(s,\Delta)$ parametrized by $s \in \mathbb{N}^+$ and $\Delta \in [0,1]$. Recall that the negative binomial distribution is parametrized in this way and that its tail bound does not depend on Δ. This property is shared by the distributions we encounter, so we denote by $t_s^{\mathsf{D}}(k)$ a tail bound that satisfies $\Pr\left[X \geq k\mu_{\mathsf{D}(s,\Delta)}\right] \leq t_s^{\mathsf{D}}(k)$, where $\mu_{\mathsf{D}(s,\Delta)}$ is the expected value of $\mathsf{D}(s,\Delta)$. We similarly think of $h_s^{\mathsf{D}}(k) = 1 - t_s^{\mathsf{D}}(k)$ as a *head bound*. We need to express optimal parameters to head bounds as functions. We denote the smallest possible k that satisfies a lower bound λ by

$$k_s^{\mathsf{D}}(\lambda) = \min\{k \in (1,\infty) \mid h_s^{\mathsf{D}}(k) \geq \lambda\} \ . \tag{1}$$

When s is fixed we drop it from our notation and consider the distribution to carry this information, e.g., $\mathsf{D}_0(\Delta)$ could be defined as $\mathsf{D}(s,\Delta)$ in which case $t^{\mathsf{D}_0}(k) = t_s^{\mathsf{D}}(k)$ and similarly for other quantities.

However, if instead the value of k is fixed, and s appears as a parameter, then we can increase $h_s^{\mathsf{D}}(k)$ by increasing s which gives

$$s_\beta^{\mathsf{D}}(\gamma) = \min\{s \in \mathbb{N}^+ \mid h_s^{\mathsf{D}}(1/\beta) \geq \gamma\} \ , \tag{2}$$

where we replace k by $\beta = 1/k$ for notational convenience. Note that changing s changes the distribution. We have the following two concrete tail bounds

$$t_s^{\mathsf{NB}}(k) = e^{-(1-1/k)^2 ks/2} \tag{3}$$

$$t_s^{\mathsf{CG}}(k) = e^{-(k-1-\ln k)s} \tag{4}$$

for the negative binomial distribution $\mathsf{NB}(s,\Delta)$ for some probability Δ, and a product of s compound geometric distributions (see Definition 32), respectively. The (shifted) geometric distribution is denoted $\mathsf{Geo}(\Delta)$. At one point we also need to bound below the expected value, i.e., we need a bound of the form $\Pr\left[X \leq \mu_{\mathsf{NB}(s,\Delta)}/k\right] \leq n_s^{\mathsf{NB}}(k)$, where $n_s^{\mathsf{NB}}(k) = e^{-(k-1)^2\frac{s}{3k}}$. Due to asymmetry this bound is slightly weaker. (See Theorem 8 and Theorem 9.)

8.2 Basic Algorithms

Definition 11 (Basic Extractor)**.** The *basic extractor* algorithm $\mathcal{B}$ takes as input a tuple $(\mathbb{M},\Delta_0)$, where $\mathbb{M} = (\mathbb{M}_0,\mathbb{M}_1)$, and proceeds as follows:

1. Set $B = \varnothing$.
2. Repeat while $|B| < d_1$:
   (a) Sample $v \in S_0 \times (S_1 \setminus \mathsf{span}(B))$ randomly.
   (b) If $\rho(v) = 1$, then set $B = B \cup \{v_1\}$.
3. Return B.

Lemma 1 (Basic Extractor)**.** *The algorithm $\mathcal{B}$ is a $(\omega_{\mathbb{M}_1}, \mathsf{NB}(d_1, \Delta_1'))$-accepting basis extractor for $\mathbb{M}$, where $\Delta_1' = \Delta_0 - \omega_{\mathbb{M}_1}$.*

In the analysis of the basic extractor and the algorithms below we need to consider the conditional probability that a node can be extended to an accepting path. Thus, we define $\delta_u = \Pr[\rho(V) = 1 \,|V_1 = u]$, where V is uniformly distributed in S.

Definition 12 (Basic Sampler)**.** The *basic sampler* algorithm $\mathcal{BS}$ takes as input a tuple $(\mathbb{M}, B, \Delta_0)$, where $B \in I_1$ is not a basis and $\Delta_0 \in (0, 1]$, and repeats:

1. Sample $v \in S_0 \times (S_1 \setminus \mathsf{span}(B)) \times \bigtimes_{i\in[2,r]} S_i$ randomly.
2. If $\rho(v) = 1$, then return v_1.

Lemma 2 (Basic Sampler)**.** *If $\Delta_\rho \geq \Delta_0 > \omega_{\mathbb{M}_1}$, then the distribution of the number of queries to $\rho(\cdot)$ made by $\mathcal{BS}^{\rho(\cdot)}(\mathbb{M}, B, \Delta_0)$ is bounded by $\mathcal{G}_{\mathsf{BS}(\mathbb{M},\Delta_0)}(z) = \mathcal{G}_{\mathsf{Geo}(\Delta_1')}(z)$, where $\Delta_1' = \Delta_0 - \omega_{\mathbb{M}_1}$. Furthermore, if U denotes its output, then $\Pr[\delta_U \geq \alpha\Delta_1'] \geq 1 - \alpha$.*

Proof. A sample satisfies $\rho(v) = 1$ with probability at least Δ_1', so the number of samples needed is bounded by $\mathsf{Geo}(\Delta_1')$. For the second claim we let V be uniformly distributed in S and let U be the node denoted by v_1 in the algorithm. Set $B_\perp = S_1 \setminus \mathsf{span}(B)$. Then by definition we have

$$\Pr[U = u] = \Pr[V_1 = u \,|\rho(V) = 1 \wedge V_1 \in B_\perp] \quad \text{and} \tag{5}$$
$$\delta_u = \Pr[\rho(V) = 1 \,|V_1 = u \wedge V_1 \in B_\perp] \tag{6}$$

so

$$\Pr[U = u] / \delta_u = \frac{\Pr[V_1 = u \wedge V_1 \in B_\perp]}{\Pr[\rho(V) = 1 \wedge V_1 \in B_\perp]} \tag{7}$$

which implies

$$\mathrm{E}[1/\delta_U] = \sum_{u\in B_\perp} \Pr[U = u] / \delta_u \leq \frac{1}{\Delta_1'} \sum_{u\in B_\perp} \Pr[V_1 = u] \leq \frac{1}{\Delta_1'} \ . \tag{8}$$

Markov's inequality then implies that

$$\Pr[\delta_U < \alpha\Delta_1'] = \Pr\left[\frac{1}{\delta_U} > \frac{1}{\alpha\Delta_1'}\right] \leq \alpha \tag{9}$$

so $\Pr[\delta_U \geq \alpha\Delta_1'] \geq 1 - \alpha$ as claimed. □

Definition 13 (Sample Validator)**.** The *sample validator* algorithm $\mathcal{V}_{s,k}$ proceeds as follows on input $(\mathbb{M}, \Delta_0)$, where $s \in \mathbb{N}^+$, $k \in (1, \infty)$, and $\Delta_0 \in (0, 1]$:

1. Set $h = 0$ and $c = 0$.
2. While $h < s$ and $c < sk/\Delta_0$:

   (a) Sample $v \in \bigtimes_{i\in[0,r]} S_i$ randomly.
   (b) Set $h = h + \rho(v)$ and $c = c + 1$.

3. If $h = s$, then return 1 and otherwise return 0.

Lemma 3 (Sample Validator)**.** *The number of queries made by* $\mathcal{V}^{\rho(\cdot)}_{s,k}(\mathbb{M}, \Delta_0)$ *is bounded by* sk/Δ_0*. If* $\Delta_\rho \geq \Delta_0$*, then* $\Pr\left[\mathcal{V}^{\rho(\cdot)}_{s,k}(\mathbb{M}, \Delta_0) = 1\right] \geq h^{\mathsf{NB}}_s(k)$*, and if* $\Delta_\rho < \frac{\Delta_0}{k}$*, then* $\Pr\left[\mathcal{V}^{\rho(\cdot)}_{s,k}(\mathbb{M}, \Delta_0) = 1\right] \leq n^{\mathsf{NB}}_s(k)$*.*

Proof. The claims follow directly from the tail and head bounds of the negative binomial distribution (see Theorem 9). □

The validating sampling algorithm repeatedly samples paths until an accepting path is found. This is repeated until the first element of the found path is considered good by the validation algorithm.

Definition 14 (Validating Sampler)**.** The *validating sampler* algorithm $\mathcal{VS}_{\alpha,\beta,\gamma}$ proceeds as follows on input $(\mathbb{M}, B, \Delta_0)$, where $\alpha, \beta, \gamma \in (0, 1)$, $\Delta_0 \in (0, 1]$, and $B \in I_1$ is not a basis:

1. Define $\Delta'_1 = \Delta_0 - \omega_{\mathbb{M}_1}$, $k = 1/\beta$, and $s = s^{\mathsf{NB}}_\beta(\gamma)$.
2. Repeat:
   (a) Compute $v_1 = \mathcal{BS}^{\rho(\cdot)}(\mathbb{M}, B, \Delta_0)$.
   (b) If $\mathcal{V}^{\rho(v_0,\cdot)}_{s,k}\big((\{v_1\}, \mathbb{M}_2, \dots, \mathbb{M}_r), \alpha\Delta'_1\big) = 1$, then return v_1.

Lemma 4 (Validating Sampler)**.** *If* $\Delta_\rho \geq \Delta_0 > \omega_{\mathbb{M}_1}$*, then the distribution of the number of queries of* $\mathcal{VS}^{\rho(\cdot)}_{\alpha,\beta,\gamma}(\mathbb{M}, B, \Delta_0)$ *is bounded by*

$$\mathcal{G}_{\mathsf{VS}(\mathbb{M},\alpha,\beta,\gamma,\Delta_0)}(z) = \mathcal{G}_{\mathsf{Geo}((1-\alpha)\gamma)}\left(\mathcal{G}_{\mathsf{BS}(\mathbb{M},\Delta_0)}(z) z^{s^{\mathsf{NB}}_\beta(\gamma)/\Delta_1}\right) \quad , \tag{10}$$

where $\Delta'_1 = \Delta_0 - \omega_{\mathbb{M}_1}$ *and* $\Delta_1 = \alpha\beta\Delta'_1$*, and its output* U *satisfies* $\Pr[\delta_U \geq \Delta_1] \geq \phi(\alpha, \beta, \gamma)$*, where* $\phi(\alpha, \beta, \gamma) = 1 - \alpha\beta n^{\mathsf{NB}}_s(1/\beta)/((1-\alpha)\gamma)$*.*

Proof. We know from Lemma 2 that the distribution of the number of queries made by $\mathcal{BS}^{\rho(\cdot)}$ in a given iteration is bounded by $\mathsf{BS}(\mathbb{M}, \Delta_0)$. Lemma 3 implies that the number of queries made by $\mathcal{V}^{\rho(v_0,\cdot)}_{s,k}$ is upper bounded by $ks/(\alpha\Delta'_1) = s^{\mathsf{NB}}_\beta(\gamma)/\Delta_1$ so the distribution of the total number of queries in the ith iteration is bounded by

$$\mathcal{G}(z) = \mathcal{G}_{\mathsf{BS}(\mathbb{M},\Delta_0)}(z) z^{s^{\mathsf{NB}}_\beta(\gamma)/\Delta_1} \quad . \tag{11}$$

Lemma 2 implies that if U_i denotes the output of $\mathcal{BS}^{\rho(\cdot)}$ in the ith iteration, then $\Pr[\delta_{U_i} \geq \alpha\Delta'_1] \geq 1 - \alpha$. From Lemma 3 and how k and s are defined in the algorithm we know that provided that $\delta_{U_i} \geq \alpha\Delta'_1$ the validator outputs 1 with probability at least $h^{\mathsf{NB}}_s(k) \geq \gamma$. Thus, the distribution of the number of samples considered is bounded by $\mathsf{Geo}((1-\alpha)\gamma)$. This implies that $\mathcal{G}_{\mathsf{Geo}((1-\alpha)\gamma)}(\mathcal{G}(z))$ bounds the distribution of the total number of queries as claimed.

Similarly to the previous claim we have $\Pr[\delta_{U_i} < \alpha\beta\Delta'_1] < \alpha\beta$ for every i from Lemma 2. Denote by A_i the output of $\mathcal{V}^{\rho(v_0,\cdot)}_{s,k}\big((\{U_i\}, \mathbb{M}_2, \dots, \mathbb{M}_r), \alpha\Delta'_1\big)$, i.e., it is the indicator variable for the event that $\mathcal{VS}_{\alpha,\beta,\gamma}$ returns U_i. Then Lemma 3 implies that $\Pr[A_i = 1 \,|\, \delta_{U_i} < \alpha\beta\Delta'_1] \leq n^{\mathsf{NB}}_s(k)$ which gives

$$\Pr[\delta_{U_i} < \alpha\beta\Delta'_1 \,|\, A_i = 1] < \frac{\alpha\beta n^{\mathsf{NB}}_s(1/\beta)}{\Pr[A_i = 1]} \leq \frac{\alpha\beta n^{\mathsf{NB}}_s(1/\beta)}{(1-\alpha)\gamma} \quad . \tag{12}$$

□

8.3 Recursive Algorithm

We now have the subroutines we need. We sequentially sample good candidates for roots of accepting bases of subtrees and make sure that the roots are independent with respect to $\mathbb{M}_1$. If extracting an accepting basis for a subtree takes too long, then we interrupt the execution and find a new candidate.

To be able to seamlessly talk about both the basic sampler and the validating sampler, and the distributions of queries we use the following notation. The sampling algorithm is defined by:

$$\mathcal{S}^{\rho(\cdot)}_{\alpha,\beta,\gamma}(\mathbb{M}, B, \Delta) = \begin{cases} \mathcal{BS}^{\rho(\cdot)}(\mathbb{M}, B, \Delta) & \text{if } \beta = 1 \\ \mathcal{VS}^{\rho(\cdot)}_{\alpha,\beta,\gamma}(\mathbb{M}, B, \Delta) & \text{otherwise} \end{cases} \tag{13}$$

The distribution of the number of queries (for a fixed oracle) is denoted by:

$$\mathsf{S}(\mathbb{M}, \alpha, \beta, \gamma, \Delta) = \begin{cases} \mathsf{BS}(\mathbb{M}, \Delta) & \text{if } \beta = 1 \\ \mathsf{VS}(\mathbb{M}, \alpha, \beta, \gamma, \Delta) & \text{otherwise} \end{cases} \tag{14}$$

The domain of the bounding function $\phi(\alpha, \beta, \gamma)$ is extended to $\beta \in (0, 1]$ by setting $\phi(\alpha, \beta, \gamma) = 1 - \alpha$ when $\beta = 1$.

Definition 15 (Recursive Extractor)**.** Let $\mathbb{M} = (\mathbb{M}_0, \ldots, \mathbb{M}_r)$ be a matroid tree and assume that $\mathcal{R}$ is a $(\epsilon_1, \mathsf{D}_1(\Delta))$-accepting basis extractor for matroid trees of the form $(\{v_1\}, \mathbb{M}_2, \ldots, \mathbb{M}_r)$, where $v_1 \in S_1$. The *recursive extractor* $\mathcal{R}_\kappa[\mathcal{R}]$, where $\kappa = (\alpha, \beta, \gamma, \lambda)$, $\alpha, \lambda, \gamma \in (0, 1)$, $\beta \in (0, 1]$, proceeds as follows on input $(\mathbb{M}, \Delta_0)$.

1. Set $\Delta_1 = \alpha\beta(\Delta_0 - \omega_{\mathbb{M}_1})$, $k = k^{\mathsf{D}_1}(\lambda)$, and $\mu = \mu_{\mathsf{D}_1(\Delta_1)}$.
2. Set $B = \varnothing$ and $T = \varnothing$.
3. While $|B| < d_1$:
   - (a) Compute $v_1 = \mathcal{S}^{\rho(\cdot)}_{\alpha,\beta,\gamma}(\mathbb{M}, B, \Delta_0)$.
   - (b) Extract subtree $t = \mathcal{R}^{\rho(v_0,\cdot)}\big((\{v_1\}, \mathbb{M}_2, \ldots, \mathbb{M}_r), \Delta_1\big)$, 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 $B = B \cup \{v_1\}$ and $T = T \cup \{t\}$.
4. Return the accepting basis tree T.

Remark 1 (Reusing Samples). The accepting paths with common prefix drawn by the sampling algorithm may be re-used by the extractor, but this will only make a noticable difference deep down in the tree due to the relatively large number of additional accepting paths that need to be found higher up in the tree and the fact that paths from the sampler may have to be discarded. To keep the presentation simple we only use this fact in the proof of Theorem 2.

Lemma 5 (Recursive Extractor)**.** *The algorithm $\mathcal{R}_\kappa[\mathcal{R}]$ is a $(\epsilon_0, \mathsf{D}_0(\Delta))$-accepting basis extractor, where $\epsilon_0 = \epsilon_1/(\alpha\beta) + \omega_{\mathbb{M}_1}$ and*

$$\mathcal{G}_{\mathsf{D}_0(\Delta_0)}(z) = \prod_{i=1}^{d_1} \mathcal{G}_{\mathsf{Geo}(\phi\lambda)}\left(\mathcal{G}_{\mathsf{S}(\mathbb{M},\alpha,\beta,\gamma,\Delta_0)}(z) z^{k^{\mathsf{D}_1}(\lambda)\mu_{\mathsf{D}_1(\Delta_1)}}\right) , \tag{15}$$

defined by $\phi = \phi(\alpha, \beta, \gamma)$ and $\Delta_1 = \alpha\beta(\Delta_0 - \omega_{\mathbb{M}_1})$.

Proof. From Lemma 2 and Lemma 4 we know that the number of queries made in Step 3a has distribution bounded by $\mathsf{S}(\mathbb{M},\alpha,\beta,\gamma,\Delta_0)$.

Denote the candidate node in the ith iteration by U_i, i.e., the value denoted v_1 in the algorithm, and denote the output of $\mathcal{R}^{\rho(v_0,\cdot)}\big((\{U_i\},\mathbb{M}_2,\ldots,\mathbb{M}_r),\Delta_1\big)$ in the ith iteration (or $\perp$ if it is interrupted) by T_i. Then both Lemma 2 and Lemma 4 imply that we have $\Pr[\delta_{U_i}\geq\Delta_1]\geq\phi$, and by our choice of scalar k we have $\Pr[T_i\neq\perp|\delta_{U_i}\geq\Delta_1]\geq\lambda$, so $\Pr[T_i\neq\perp]\geq\phi\lambda$. This means that the distribution of the number of iterations required to succeed once is bounded by $\mathsf{Geo}(\phi\lambda)$, and we need d_1 successes. $\square$

Corollary 1 (Recursive Extractor)**.** *The distribution* $\mathsf{D}_0(\Delta)$ *satisfies*

$$\mu_{\mathsf{D}_0(\Delta_0)}=\begin{cases}\frac{d_1}{(1-\alpha)\lambda}\left(\alpha\frac{1}{\Delta_1}+k^{\mathsf{D}_1}(\lambda)\mu_{\mathsf{D}_1(\Delta_1)}\right) & \textit{if } \beta=1\\ \frac{d_1}{\phi\lambda}\left(\frac{\alpha\beta+s_\beta^{\mathsf{NB}}(\gamma)}{(1-\alpha)\gamma}\frac{1}{\Delta_1}+k^{\mathsf{D}_1}(\lambda)\mu_{\mathsf{D}_1(\Delta_1)}\right) & \textit{otherwise}\end{cases}\tag{16}$$

$$t_{d_1}^{\mathsf{D}_0}(k)\leq t_{d_1}^{\mathsf{CG}}(k)\quad \textit{for } k\in(1,\infty)\ .\tag{17}$$

8.4 Accepting Basis Extractor

We now let the recursive extractor $\mathcal{R}$ invoke itself recursively until it suffices to invoke the basic extractor $\mathcal{B}$. For each additional recursive call needed, there is growth in the extraction error, but this only depends on the quantity $\nu=1/(\alpha\beta)$, apart from the matroid subdensities which are fixed. Given a fixed ν we may optimize all other parameters to minimize the expected number of queries, since our tail bound does not depend on the expected value. All we need to do this is the rank of the matroid and a bound on the distribution of the number of queries needed in the next recursive call.

Theorem 2 (Extractor)**.** *For every* $\nu_1,\ldots,\nu_{r-1}\in(1,\infty)$ *there exist parameters* $\kappa_i=(\alpha_i,\beta_i,\gamma_i,\lambda_i)$ *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 matroid trees of depth* r *where:*

$$\epsilon_0=\sum_{i\in[r]}\omega_{\mathbb{M}_i}\prod_{j\in[i-1]}\nu_j\qquad\textit{(extraction error)}\tag{18}$$

$$\mu_{\mathsf{D}_0(\Delta_0)}\leq\frac{c_0\prod_{j\in[r]}d_j}{\Delta_0-\epsilon_0}\qquad\textit{(expected number of queries)}\tag{19}$$

$$t_{d_1}^{\mathsf{D}_0}(k)\leq t_{d_1}^{\mathsf{CG}}(k)\quad\textit{for } k>1\ ,\qquad\textit{(tail bound)}\tag{20}$$

where the constant c_0 *is defined by* $c_{r-1}=1$ *and* $c_i=f_{\mathsf{S}}(d_{i+1},\nu_{i+1},c_{i+1})$ *for* $i=r-2,\ldots,0$*, using* $f_{\mathsf{S}}(d,\nu,c)=\min\{f_{\mathsf{BS}}(d,\nu,c),f_{\mathsf{VS}}(d,\nu,c)\}$*, where*

$$f_{\mathsf{BS}}(d,\nu,c)=\frac{\nu^2}{(\nu-1)}\cdot\min_k\left\{\frac{k}{h_d^{\mathsf{CG}}(k)}\right\}\cdot c\ ,\tag{21}$$

$$f_{\mathsf{VS}}(d,\nu,c)=\min_{\alpha,s,k}\left\{\frac{\nu}{\phi(\alpha,\nu\alpha,\gamma)}\left(\frac{1+s}{\alpha' h_s^{\mathsf{NB}}(\nu\alpha)h_d^{\mathsf{CG}}(k)}\cdot\frac{d_i}{D_{i+1,r}}+\frac{k}{h_d^{\mathsf{CG}}(k)}\cdot c\right)\right\}\tag{22}$$

with $D_{i,r}=\prod_{j\in[i,r]}d_j$*,* $\alpha\in(0,1/\nu)$*,* $\alpha'=1-\alpha$*,* $s\in\mathbb{N}^+$*, and* $k\in(1,\infty)$*.*

Both strategies give convoluted expressions, but we choose to not simplify, since they tell a story and are readily computed numerically. The first factor in $f_{\mathsf{BS}}(d,\nu,c)$ represents how aggressively we use Markov's bound, i.e., how good we want a sample to be. The second factor represents the tradeoff between the number of attempts needed to complete a recursive call and how long it is allowed to run. This factor appears as a term in the second factor of the validating sampling strategy as well, but here it is balanced with the first term where s represents how many samples are used for validation. This is only worthwhile when $D_{i+1,r}$ and c are large. Note that it is easy to compute optimal parameters for the algorithm for any concrete protocol.

Proof of Theorem 2. The tail bound follows directly from Corollary 1.

Bounding the extraction error. If we set $\zeta_i = \alpha_i\beta_i$ for $i \in [r-1]$ and $\zeta_r = 1$, then we have $\Delta_i = \zeta_i(\Delta_{i-1} - \omega_{\mathbb{M}_i})$ for $i \in [r]$ which expands to

$$\Delta_r = \Delta_0 \prod_{i\in[r]} \zeta_i - \sum_{i\in[r]} \omega_{\mathbb{M}_i} \prod_{j\in[i,r]} \zeta_j \quad . \tag{23}$$

The basic extractor requires that $\Delta_{r-1} - \omega_{\mathbb{M}_r} = \Delta_r > 0$ to work, so the extraction error is given by

$$\epsilon_0 = \sum_{i\in[r]} \frac{\omega_{\mathbb{M}_i}}{\prod_{j\in[i-1]} \zeta_j} = \sum_{i\in[r]} \omega_{\mathbb{M}_i} \prod_{j\in[i-1]} \nu_j \quad . \tag{24}$$

Deriving parameters. Next we consider the problem of deriving α_i, β_i, γ_i, and λ_i from ζ_i. Define $\mathcal{X}_{\kappa,i} = \mathcal{R}_{\kappa_i}[\mathcal{R}_{\kappa_{i+1}}[\cdots\mathcal{R}_{\kappa_{r-2}}[\mathcal{B}]\cdots]]$. We will express the expected running time of $\mathcal{X}_{\kappa,i}$ on the form $c_i D_{i+1,r}/(\Delta_i - \epsilon_i)$ for a constant c_i provided that its oracle and input (N_i, Δ_i) are reasonably good. More precisely, it is called with an oracle of the form $\rho(v_0, \ldots, v_{i-2}, \cdot)$ and a matroid tree $N_i = (\{v_{i-1}\}, \mathbb{M}_i, \ldots, \mathbb{M}_r)$ for some values $v_i \in S_i$. Denote by Δ_0 the original estimated probability used as input to $\mathcal{X}_{\kappa,1}$ and define

$$\Delta'_i = \Delta_{i-1} - \omega_{\mathbb{M}_i} \quad , \quad \Delta_i = \zeta_i\Delta'_i \quad , \quad \text{and} \quad \epsilon_i = \zeta_i(\epsilon_{i-1} - \omega_{\mathbb{M}_i}) \tag{25}$$

for $i \in [r]$. We set $\alpha'_i = 1 - \alpha_i$ for notational convenience below.

Basic sampling strategy. Consider first the strategy where the non-validating sampler is used, i.e., we have $\beta_i = 1$ and $\zeta_i = \alpha_i$. If we exploit the fact that the path through the sampled node can be re-used we have

$$\mu_{\mathsf{D}_{i-1}(\Delta_{i-1})} = \frac{d_i}{\alpha'_i\lambda_i} k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\Delta_i)} \tag{26}$$

$$= \frac{1}{\alpha_i\alpha'_i} \cdot \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i \cdot \frac{D_{i,r}}{\Delta_{i-1} - \epsilon_{i-1}} \quad . \tag{27}$$

If we choose an optimal λ_i, then

$$c_{i-1} = \frac{1}{\alpha_i\alpha'_i} \cdot \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i = f_{\mathsf{BS}}(d_i, \nu_i, c_i) \quad . \tag{28}$$

Validating sampling strategy. Next we consider the strategy where samples are validated before use and we are not extracting leaves in the recursive call, i.e., we have $i < r-2$, $\beta_i < 1$, and $\zeta_i = \alpha_i\beta_i$. In this case re-using leaves has limited value and we have

$$\mu_{\mathsf{D}_{i-1}(\Delta_{i-1})} = \frac{d_i}{\phi_i\lambda_i}\left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{\alpha'_i\gamma_i} \cdot \frac{1}{\Delta_i} + k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\Delta_i)}\right) \tag{29}$$

$$= \frac{1}{\zeta_i\phi_i}\left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{\alpha'_i\gamma_i\lambda_i} \cdot \frac{d_i}{D_{i+1,r}} + \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i\right) \frac{D_{i,r}}{\Delta_{i-1} - \epsilon_{i-1}} \quad . \tag{30}$$

If we choose parameters optimally, then we have

$$c_{i-1} = \frac{1}{\zeta_i\phi_i}\left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{\alpha'_i\gamma_i\lambda_i} \cdot \frac{d_i}{D_{i+1,r}} + \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i\right) \leq f_{\mathsf{VS}}(d_i, \nu_i, c_i) \quad . \tag{31}$$

□

8.5 Interpretation

In the following we assume that Δ_0 is a tight lower bound for Δ_ρ. The expected running time of the extractor is $\mathsf{poly}/(\Delta_0 - \epsilon_0)$ as expected for a proof of knowledge with knowledge error ϵ_0.

We may choose ν_i arbitrarily close to one and conclude that a witness can be extracted provided that Δ_0 is slightly larger than $\epsilon = \sum_{i\in[r]} \omega_{\mathbb{M}_i}$, which coincides with our intuition about the soundness of special-sound protocols in general, i.e., to convince a verifier of a false statement it suffices in general to guess a challenge value correctly in at least one round. This is optimal in the sense that it is necessary to exploit dependencies between the rounds in the protocol to establish a smaller soundness error.

Note that Bellare and Goldreich's definition of a proof of knowledge [BG92] is satisfied regardless of how small we make $\nu_i > 1$. However, if we choose ν_i based on a given Δ_0 such that $\Delta_0 > \epsilon_0$, then the expected number of queries of the extractor has the form $f(\Delta_0)/(\Delta_0 - \epsilon_0)$, where $f(\Delta_0)$ grows superexponentially when Δ_0 approaches ϵ_0. Conversely, we may set $\nu_i \approx 2$ to minimize the expected running time of the extractor and accept an extraction error of the form $\sum_{i\in[r]} 2^{i-1}\omega_{\mathbb{M}_i}$. This begs the question: What is the knowledge error of the protocol?

A protocol is said to be a proof of knowledge with knowledge error ϵ_* if there is an extractor that outputs the witness in expected time $\mathsf{poly}/(\Delta_\rho - \epsilon_*)$. On the one hand we can make ϵ_* arbitrarily close to ϵ (and it cannot be smaller), but on the other hand this causes a drastic loss in security in terms of the running time of the extractor. Squeezing ϵ_* in this way is arguably an abuse of the definition, but we still think that it is more natural to view the knowledge error as a property of the extractor and not of the protocol.

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.

[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.

[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.

[Gro03] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In Yvo Desmedt, editor, *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *Lecture Notes in Computer Science*, pages 145–160. Springer, 2003. doi:10.1007/3-540-36288-6_11.

[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.

[Jan17] Svante Janson. Tail bounds for sums of geometric and exponential variables, 2017. arXiv:arXiv:1709.08157.

[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.

[KL81] Bernhard Korte and László Lovász. Mathematical structures underlying greedy algorithms. In Ferenc Gécseg, editor, *Fundamentals of Computation Theory, FCT'81, Proceedings of the 1981 International FCT-Conference, Szeged, Hungary, August 24-28, 1981*, volume 117 of *Lecture Notes in Computer Science*, pages 205–209. Springer, 1981. doi:10.1007/3-540-10854-8_22.

[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 Security, 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.

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

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004. URL: http://eprint.iacr.org/2004/332.

[TW10] Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In Daniel J. Bernstein and Tanja Lange, editors, *Progress in Cryptology - AFRICACRYPT 2010, Third International Conference on Cryptology in Africa, Stellenbosch, South Africa, May 3-6, 2010. Proceedings*, volume 6055 of *Lecture Notes in Computer Science*, pages 100–113. Springer, 2010. doi:10.1007/978-3-642-12678-9_7.

[Wal44] Abraham Wald. On cumulative sums of random variables. *Ann. Math. Statist.*, 15(3):283–296, 09 1944. doi:10.1214/aoms/1177731235.

A Postponed Proofs

Proof of Lemma 1. The probability that a randomly sampled v satisfies $\rho(v) = 1$ is at least $\Delta_0 - \omega_{\mathbb{M}_1} > 0$ by assumption from which the claimed distribution of the number of queries made follows immediately. $\square$

Proof of Corollary 1. The expected value follows from linearity and Wald's equation [Wal44] (or directly from Lemma 10). More precisely, it follows from the equalities $\mu_{\mathsf{Geo}(\phi\lambda)} = 1/(\phi\lambda)$, $\Delta_1' = \Delta_0 - \omega_{\mathbb{M}_1}$, $\Delta_1 = \alpha\beta\Delta_1'$, and

$$\mu_{\mathsf{S}(\mathbb{M},\alpha,\beta,\gamma,\Delta_0)} = \begin{cases} \frac{1}{\Delta_1'} & \text{if } \beta = 1 \\ \frac{1}{(1-\alpha)\gamma}\left(\frac{1}{\Delta_1'} + \frac{s_\beta^{\mathsf{NB}}(\gamma)}{\Delta_1}\right) & \text{otherwise} \end{cases} \tag{32}$$

and the fact that $\phi = 1 - \alpha$ when $\beta = 1$. The tail bound follows directly from Theorem 8 for this type of compound geometric distribution. ☐

B Additional Context

B.1 The Set of Accepting Transcript Trees is Not a Matroid

As explained in Section 3, for any knowledge extractor there is an implicitly defined set J of minimal accepting transcript trees from which a witness can be computed efficiently. Let S be the set of all possible transcripts and define I to consist of the union of J and all subsets of any element $A \in J$. This is a natural attempt to define a matroid (S, I) over the set of transcripts. The set I is non-empty, and $A \in I$ and $B \subset A$ implies that $B \in I$, but the augmentation property of matroids does not hold in general. Indeed, if $A, B \in I$ and $|A| > |B|$ there is nothing that guarantees that we can include any transcript $a \in A \setminus B$ into B and expect it to remain a subset of some set in J.

This structure does not even necessarily satisfy the relaxed requirements of a greedoid [KL81], since the maximal elements of the set I may have different sizes and there is not necessarily a feasibility oracle. This should not be surprising to the reader, since we can express hard combinatorial optimization problems as knowledge extraction problems.

Thus, in this paper we show that a large and natural class of protocols can be described and analyzed exactly using the standard mathematical notion of matroids; not that the independence requirements of every protocol has this simple structure.

B.2 Counting Predicate Queries Suffices

We have no control over how the prover distributes its running time over the execution of the protocol. Counting queries may be viewed as the worst case where the vast majority of the work is performed right before the last round.

Above we have ignored all overhead costs in the extraction algorithms and focused on the number of oracle queries. The only potentially non-linear operation performed by the algorithms that is not already captured by the evaluation of the predicate is sampling from the complement of a flat in a matroid. Note that small subdensity does not imply that verifying independence is efficient. Consider the following definition.

Definition 16 (Sampling Cost)**.** Let $\mathbb{M}$ be a matroid of rank d. Then $\mathbb{M}$ has *sampling cost* $c_{\mathbb{M}}$ if there exists a probabilistic algorithm Alg with running time $c_{\mathbb{M}}$ such that setting $a_0 = \varnothing$ and computing $(a_i, b_i) = \mathsf{Alg}(a_{i-1})$ for $i = 1, \ldots, d$ gives a uniformly distributed basis $\{b_1, \ldots, b_d\}$ of $\mathbb{M}$.

The value a_i is used to store any pre-computation used by Alg to complete the task within the required time. One can give more precise running times for the extractors by including the cost for sampling in the the analysis, but we choose to not do this, since for the matroids of protocols the running time of the sampling algorithm is typically linear in i with a unit cost corresponding to a multiplication in a field or similar.

B.3 Direct Applications

We have phrased Theorem 2 in an abstract way not only to emphasize the simple combinatorial structure and to be able to give a rigorous proof, but also because it allows the theorem to be applied directly to a broad range of settings. Below we give two illustrative examples.

B.3.1 Schnorr's Protocol.

We consider a common practical variant of Schnorr's protocol [Sch91] where the verifier's challenge is sampled randomly from a subset $\{0,1\}^{n_v} \subset \mathbb{Z}_q$ instead of from the whole field, where n_v is an auxiliary security parameter. Consider a group homomorphism $\phi : \mathbb{Z}_q \to G_q$, where q is an odd prime, and G_q is a group of order q. The protocol can be used to prove knowledge of a pre-image $w \in \mathbb{Z}_q$ of $z \in G_q$, such that $\phi(w) = z$.

Definition 17 (Schnorr's Protocol)**.** On common input $z \in G_q$ and private input w, such that $\phi(w) = z$, the prover and verifier proceed as follows.

1. $\mathcal{P}$ chooses $r \in \mathbb{Z}_q$ randomly, computes $\alpha = \phi(r)$, and hands α to $\mathcal{V}$.
2. $\mathcal{V}$ chooses $c \in \{0,1\}^{n_v}$ randomly and hands c to $\mathcal{P}$.
3. $\mathcal{P}$ computes $d = wc + r \bmod q$.
4. $\mathcal{V}$ accepts if and only if $z^c\alpha = \phi(d)$.

The soundness of the protocol is captured by the inequality matroid $(\{0,1\}^{n_v}, I)$, where $I = \{X \subset \{0,1\}^{n_v} : |X| \leq 2\}$ is the set of subsets of $\{0,1\}^{n_v}$ of size at most two. A convenient notation for this matroid is $\{0,1\}_2^{n_v}$.

Lemma 6. *Schnorr's protocol is* $\left(\{0,1\}_2^{n_v}, O\left(n^2\right)\right)$*-special sound,*

Proof. An accepting transcript tree T such that $\mathbb{V}(T)$ is a basis subtree of the matroid tree is a pair of accepting transcripts (z, α, c, d) and (z, α, c', d') such that $\{c, c'\}$ is a basis of $\{0,1\}_2^{n_v}$. The witness extraction algorithm computes $w = (c - c')^{-1}(d - d')$, which takes time $O\left(n^2\right)$. The output w is a witness, since $\phi(w) = \phi((c - c')^{-1}(d - d')) = \phi(d - d')^{(c-c')^{-1}} = z$. □

Corollary 2. *There exists a* $(\epsilon, \mathsf{D}(\Delta))$*-accepting basis extractor* $\mathcal{X}$ *for Schnorr's protocol such that* $\epsilon = 2^{-n_v}$*,* $\mu_{\mathsf{D}(\Delta)} \leq \frac{1}{\Delta - \epsilon}$*, and* $t^{\mathsf{D}}(k) = e^{-2(k-1-\ln k)}$*.*

B.3.2 Batching protocol.

The ability to capture the soundness of Schnorr's protocol in a simple way may be viewed as a sanity check, since it inspired the definition of special soundness. Our next example is a batching protocol where multiple instances are combined into one before Schnorr's protocol is applied. We stress that this is an example from a broad class of protocols which illustrates how our approach captures concentration in the distribution of the running time.

More precisely, suppose that the prover holds a preimage x_i of a common input $y_i = \phi(x_i)$ for $i \in [N]$. We can of course simply construct product groups and apply Schnorr's protocol with minor syntactical changes, but the batching technique of Bellare et al. [BGR98] is more efficient.

A vector space $\mathbb{Z}_q^N$ is a matroid where a set of vectors is an independent set if and only if its members are linearly independent. Below we denote by $\{0,1\}^{n_e \times N}$ the matroid induced by restricting the ground set to $\{0,1\}^{n_e \times N}$.

Definition 18 (Batching Protocol)**.** On common input $y_i \in G_q$ for $i \in [N]$, and private input $x_i \in \mathbb{Z}_q$ such that $\phi(x_i) = y_i$ for $i \in [N]$ the prover and verifier proceed as follows.

1. $\mathcal{V}$ chooses $e \in \{0,1\}^{n_e \times N}$ randomly and hands e to $\mathcal{P}$.
2. $\mathcal{P}$ and $\mathcal{V}$ execute Schnorr's protocol on common input $z = \prod_{i\in[N]} y_i^{e_i}$ and private input $w = \sum_{i\in[N]} x_i e_i \bmod q$ and $\mathcal{V}$ outputs the resulting verdict.

Lemma 7. *The batching protocol is* $\left((\{0,1\}^{n_e\times N}, \{0,1\}_2^{n_v}), O\left(N^3 n^2\right)\right)$*-special sound.*

Proof. An accepting transcript tree T such that $\mathbb{V}(T)$ is a basis subtree of the matroid tree is a set of accepting transcripts

$$\left\{(e_j, \alpha_j, c_j, d_j), (e_j, \alpha_j, c'_j, d'_j)\right\}_{j\in[N]} \tag{33}$$

such that $\{e_1, \ldots, e_N\}$ is a basis of the vector space matroid and $\{c_j, c'_j\}$ is a basis of the inequality matroid for $j \in [N]$.

The witness extraction algorithm first uses the witness extraction algorithm of Schnorr's protocol for $j \in [N]$ to compute: w_j such that $\phi(w_j) = z_j$ from $(z_j, \alpha_j, c_j, d_j)$ and $(z_j, \alpha_j, c'_j, d'_j)$, where $z_j = \prod_{i\in[N]} y_i^{e_{j,i}}$.

Then it defines the vector $w = (w_1, \ldots, w_N)$ and the matrix $E = (e_{j,i})_{j,i\in[N]}$, and computes the matrix $A = (a_{l,j})_{l,j\in[N]}$ in $\mathbb{Z}_q^{N\times N}$ such that $AE = (\langle a_l, e'_i\rangle)_{l,i\in[N]}$ is the identity matrix, where: a_l denotes the lth row of A, e'_i denotes the ith column of E, and $\langle\cdot,\cdot\rangle$ is the inner product over $\mathbb{Z}_q^N$. The matrix A exists since $\{e_1, \ldots, e_N\}$ is a basis, and it can be computed in time $O\left(N^3 n^2\right)$ using Gaussian elimination.

The witness extraction algorithm computes $x_l = \langle w, a_l\rangle$ for $l \in [N]$ and outputs $x_1, \ldots, x_N$. This is a witness, since

$$\phi(\langle w, a_l\rangle) = \prod_{j\in[N]} \phi(w_j)^{a_{l,j}} = \prod_{j\in[N]} z_j^{a_{l,j}} = \prod_{j\in[N]} \left(\prod_{i\in[N]} y_i^{e_{j,i}}\right)^{a_{l,j}} = \prod_{i\in[N]} y_i^{\langle a_l, e'_i\rangle} \;, \tag{34}$$

and $\langle a_l, e'_i\rangle$ equals 1 if $l = i$ and 0 otherwise. □

Corollary 3. *There exists a* $(\epsilon, \mathsf{D}(\Delta))$*-accepting basis extractor* $\mathcal{X}$ *for the batching protocol such that* $\epsilon = 2^{-n_e} + 2^{1-n_v}$*,* $\mu_{\mathsf{D}(\Delta)} \leq \frac{16\cdot N}{\Delta-\epsilon}$*, and* $t^{\mathsf{D}}(k) = e^{-N(k-1-\ln k)}$ *if* $N > 10$*.*

Proof. We set $\nu_1 = 2$ and apply Theorem 2. □

The tail bound is powerful when N is large and allows us to think of the extractor as running in strict polynomial time in most applications. The constant 16 tends to 8 for large N.

B.4 Application to Embedded Protocols

Above we consider special sound protocols in isolation, but they are almost always used as subprotocols within larger protocols. Conceptually it is natural to consider a special sound protocol as a component of a larger global protocol and think of the common input as generated somehow prior to execution.

For interactive proofs composition is straightforward since soundness is guaranteed for all instances. Indeed, if an interactive proof has soundness error ϵ and it is executed k times within the global protocol, then the probability that a false statement is accepted in any execution is bounded by $k\epsilon$.

However, when a special sound protocol is used as a proof of knowledge or computationally sound proof, it is easier to turn the global protocol into an extension of the special sound protocol and apply our theorem directly to the extended protocol than to decompose the global protocol into components with complex and artificial requirements on their composition.

B.4.1 What if the first prover message is randomly chosen?

Before we delve into the general case, we address something that the reader may have already observed in our analysis of Schnorr's protocol above.

We treat the prover as deterministic, but in most applications the first prover message α of Schnorr's protocol is chosen randomly by a probabilistic prover. Indeed, it must be randomly chosen for the protocol to be zero knowledge.

To apply our analysis we simply note that we may view the random tape of the prover as the first message of the verifier. When the protocol is viewed in this way, it is a $\big((\{0,1\}^*, \{0,1\}^{n_v}), p\big)$-special sound protocol, where is $\{0,1\}^*$ the *trivial matroid* where every subset of size at most one is a basis. Thus, our analysis applies mutatis mutandi, but the running time of the extractor is now bounded by a geometric distribution and therefore less concentrated.

B.4.2 Extending an embedded protocol.

The above is an example of a more general principle. Any global protocol with any number of parties which executes a special sound protocol as a subprotocol may be viewed as an *extended protocol* as follows. All honest, trusted parties, ideal functionalities, and the verifier of the special sound protocol are simulated by an extended verifier. All corrupted parties, including the prover of the special sound protocol are simulated by an extended prover.

In other words, we divide the parties of a multiparty protocol into two groups and consider the global protocol as an interactive protocol between the two groups of parties and entities. The prefix of the execution generates an instance on which the special sound protocol is then executed.

More precisely, the random tapes of all corrupted parties and all the randomness used in the prefix may be viewed as a single verifier message of the extended verifier. This means that there is simply an additional message from the verifier of the special sound protocol at the start of the protocol drawn from the ground set of a trivial matroid.

Then we can clearly apply the theorem directly, but there is no statement at the start of the global protocol, so what does this even mean?

In the analysis of protocols we use knowledge extractors either to reach a contradiction to a computational assumption, or to extract values needed in the simulation of a larger protocol.

B.4.3 Extract to Solve a Computational Problem.

Consider an application of an identification scheme [FS86]. Each party holds a secret key of a public key and to identify themselves they prove knowledge of their secret key. In this application we expect an illegitimate prover to fail to use another party's public key unless it is able to solve a computationally hard problem, i.e., if such a prover convinces the verifier then there should be an efficient algorithm for solving the computational problem.

For example, suppose that an honest party in the protocol chooses a secret key $w \in \mathbb{Z}_q$, computes a public key $z = \phi(w)$, and that we use Schnorr's protocol as an identification scheme. If a corrupt party executes Schnorr's protocol with an honest party this should lead to a contradiction, since it should be unable to masquerade as the owner of z.

We can construct the reduction as follows. First we modify the extended protocol such that every execution where the owner of z plays the prover is simulated using the zero-knowledge simulator. This means that the secret key is no longer necessary to sample an extended execution. Then we apply the theorem and conclude that there exists an extractor that for some common input z' appearing as a message from the extended verifier computes a witness w' such that $z' = g^{w'}$.

We note that z can be used by the extractor as long as it is randomly distributed. If the extractor uses at most Q queries and the reduction simply replaces a randomly chosen common input by z at a randomly chosen point in the extraction process, then $z' = z$ with probability $1/Q$. Furthermore, the discrete logarithm problem is random self-reducible which means that we can replace every common input by a randomization z' of z instead, and always succeed. Here the tail bound on the distribution of the running time translates into a tighter strict polynomial time reduction.

Turning this informal argument into a rigorous proof with an exact security reduction amounts to introducing notation and applying our theorem.

Computationally sound proofs are closely related to this example, but here only part of the statement is chosen randomly by the extended verifier (or it is generated jointly) and the extracted witness leads to a contradiction with non-neglible probability, which is the computational soundness error.

B.4.4 Extract to Enable Simulation.

In the global protocol special sound protocols are often used as proofs of knowledge in such a way that the overall simulator must extract the witnesses of all accepting executions to complete. The security proof of the global protocol the reduction may be based on an indistinguishability assumption, e.g., the Decision Diffie-Hellman assumption.

In a typical game hopping security reduction [Sho04] this leads to the following situation. A hypothetical experiment, parametrized by an adversary, takes an instance of the decision problem as input and outputs a single bit. The output is encoded as two random variables D_0 and D_1, respectively, depending on the type of instance of the decision problem. The contrapositive assumption that is expected to lead to a contradition is that $|\Pr[D_1 = 1] - \Pr[D_0 = 1]|$ is non-negligible.

Consider the random process leading to an outcome of D_b where the extractor is invoked whenever a prover convinces the verifier in the embedded special sound protocol. The verifier accepts with some probability Δ_b, which itself is a random variable. If D'_b is an indicator variable for the event that the verifier accepts, then we can conclude that $|\Pr[D'_1 = 1] - \Pr[D'_0 = 1]|$ is negligible, since otherwise we could ignore D_b and use D'_b directly distinguish instances of the decision problem without even invoking the extractor. Thus, for the purpose of this argument we may assume that Δ_0 and Δ_1 are identically distributed.

The expected running time is only increased by a polynomial factor when the extraction error ϵ is small, since then $\Delta_b \cdot \mathsf{poly}/(\Delta_b - \epsilon) \approx \mathsf{poly}$. The problem is that if we choose a bound t on the running time such that $\Pr[T_b \geq t] = \epsilon$, we may have $|\Pr[D_1 = 1 \,|T_1 \leq t] - \Pr[D_0 = 1 \,|T_0 \leq t]| = 0$, i.e., we may lose the entire distinguishing advantage if we truncate executions too aggressively and no amount of resampling can address this.

In other words, the tightness of the reduction is determined at this point by how well we can bound $\Pr[T_b \geq t]$. The knowledge extractor guaranteed by the definition of a proof of knowledge cannot be applied directly without loss. We would need to sample truncated executions until it completed in time and bound the total running time as a geometric distribution.

The extraction theorem provides sufficient guarantees for many protocols, and even when it does not we may often start with a more concentrated distribution which allows tighter security proofs.

C Piece-wise Special Soundness

Some protocols require multiple rounds of extraction because matroids and values that need to be extracted may depend on what have been extracted so far. This is often the case where multiple witnesses can be extracted in principle, but only one witness can be extracted without violating a computational assumption, e.g., proofs of shuffles [FS01, TW10].

This does not quite fit into the framework we have presented, since the matroids are fixed, but we can still capture the extraction properties of such protocols in a way that has the same flavor as special soundness. To this end we consider decompositions of NP relations.

Definition 19 (Decomposable NP Relation)**.** An NP relation R has a *decomposition* $(\mathsf{R}_1[\cdot], \ldots, \mathsf{R}_k[\cdot])$, where $\mathsf{R}_j[\cdot]$ is a family of NP relations, if $(x, w) \in \mathsf{R}$ if and only if there exists $y_1, \ldots, y_k$ such that $(x, y_j) \in \mathsf{R}_j[y_1, \ldots, y_{j-1}]$ for $j \in [k]$.

We need the flexibility to parametrize the independence sets of the matroids, but the ground sets remain fixed since they are defined by the sets from which the verifier draws its messages.

Definition 20 (Families of Matroids and Matroid Trees)**.** If $\mathbb{M}[z]$ is a matroid with a fixed ground set S for every $z \in Z$, then $\mathbb{M}[\cdot]$ is a *family of matroids* parametrized Z. A family of matroid trees of depth r is a list of families of matroids $(\mathbb{M}_0[\cdot], \ldots, \mathbb{M}_r[\cdot])$.

The idea is now that we can think of a protocol as special-sound if we can decompose the NP relation into a number of steps and device an extractor for each step using the approach already presented, but with different independence sets. This may seem unnecessarily complicated, but turns out to be convenient and preserves the strong properties of special soundness.

Definition 21 (Piece-wise Special Soundness)**.** A $(2r + 1)$-message public coin-protocol $(\mathcal{P}, \mathcal{V})$ is *piece-wise* $(\mathbb{M}[\cdot], p)$*-special-sound* for an NP relation R with decomposition $(\mathsf{R}_1[\cdot], \ldots, \mathsf{R}_k[\cdot])$, where $\mathbb{M}_j[\cdot]$ is a family of matroid trees of depth r if $(\mathcal{P}, \mathcal{V})$ is $(\mathbb{M}_j[z], p_j)$-special-sound for $\mathsf{R}_j[z]$ for every $z = (y_1, \ldots, y_j)$ such that $(x, y_l) \in \mathsf{R}_l[y_1, \ldots, y_{l-1}]$ for $l \in [j]$, and $p = \sum_{j\in[k]} p_j$.

It is natural to abuse notation and think of a piece-wise special sound protocol as being special sound, but the decomposition of the NP relation and the parametrized matroids must be provided along with the algorithms that compute a witnesses from accepting transcript trees.

D Matroids

We recall one set of definitions for matroids and some of their properties.

Definition 22 (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 23 (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 24 (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 25 (Rank)**.** The *rank* of a matroid (S, I) is the unique cardinality of each basis in I.

Definition 26 (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 27 (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 $\mathsf{span}(A) = A$.

E Generic Bounds on Random Variables

We use a few standard bounds in this paper depending on how much we know about a distribution and how important it is to give a tight bound. Markov's inequality can be applied to any distribution for which the expected value can be estimated. There are many variations of Chernoff's bound depending on what is most convenient and how precisely it is stated. We state the bounds in their traditional forms.

Theorem 3 (Markov's Inequality)**.** *Let X be a non-negative random variable over $\mathbb{R}$ with expected value μ and let $k \in (1, \infty)$. Then $\Pr[X \geq k\mu] \leq \frac{1}{k}$.*

Theorem 4 (Chernoff's Inequalities)**.** *Let $X_1, \ldots, X_n$ be independent binary random variables such that $\Pr[X_i = 1] = p$ and define $X = \sum_{i=1}^{n} X_i$. Then for every $\delta \in (0, 1)$:*

$$\Pr[X < (1-\delta)np] < e^{-\frac{\delta^2 p}{2}n} \quad \text{and} \tag{35}$$

$$\Pr[X > (1+\delta)np] < e^{-\frac{\delta^2 p}{3}n} \quad . \tag{36}$$

We remark that the slight asymmetry due to the factors 1/2 and 1/3 in the exponents of the bounds is necessary.

F Generating Functions

Probability and moment generating functions are convenient ways to describe distributions and derive bounds. They can be viewed as tools for manipulation of formal power series, but they also have an analytic meaning where they converge.

Definition 28 (Probability Generating Function)**.** The *probability generating function* of a random variable X over $\mathbb{N}$ is defined by $\mathcal{G}_X(z) = \mathrm{E}\left[z^X\right]$ for all $z \in \mathbb{R}$ for which this converges.

Theorem 5 (Properties of Probability Generating Functions)**.**

1. *If X is a random variable over $\mathbb{N}$, then $\mathsf{P}_X(k) = \left(\frac{1}{k!}\right) \mathcal{G}_X^{(k)}(0)$, i.e., the probability generating function determines P_X uniquely.*

2. *If X and Y are independent random variables over $\mathbb{N}$ with probability generating functions $\mathcal{G}_X(z)$ and $\mathcal{G}_Y(z)$, then $\mathcal{G}_X(z)\mathcal{G}_Y(z)$ is the probability generating function of the sum $X + Y$.*

Definition 29 (Moment Generating Function)**.** The *moment generating function* of a random variable X over $\mathbb{N}$ is defined by $\mathcal{M}_X(\theta) = \mathrm{E}\left[e^{\theta X}\right]$ for all $\theta \in \mathbb{R}$ for which this converges.

Theorem 6 (Properties of Moment Generating Functions)**.**

1. *If X is a random variable over $\mathbb{N}$, then $\mathrm{E}\left[X^k\right] = \mathcal{M}_X^{(k)}(0)$, i.e., the moment generating function determines all moments of P_X uniquely, which in turn determines P_X uniquely.*

2. *If X and Y are independent random variables over $\mathbb{N}$ with moment generating functions $\mathcal{M}_X(\theta)$ and $\mathcal{M}_Y(\theta)$, then $\mathcal{M}_X(\theta)\mathcal{M}_Y(\theta)$ is the moment generating function of $X+Y$.*

The following is a general form of Markov's inequality from which Chernoff's inequality follows using the right choice of θ for the binomial distribution.

Theorem 7 (Cramér's Theorem)**.** *Let $X_1, \ldots, X_n$ be identically and independently distributed random variables, and define $Y = \sum_{i=1}^n X_i$. Then for every $a > \mu$ and $\theta \in (0, \infty)$ such that $\mathcal{M}_{X_1}(\theta)$ is finite:*

$$\Pr\left[Y \geq na\right] \leq \left(\frac{\mathcal{M}_{X_1}(\theta)}{e^{\theta a}}\right)^n \quad . \tag{37}$$

Proof. We apply Markov's inequality and independence to get

$$\Pr\left[Y \geq na\right] = \Pr\left[e^{\theta Y} \geq e^{\theta na}\right] \leq \frac{\mathrm{E}\left[e^{\theta Y}\right]}{e^{\theta na}} = \frac{\mathcal{M}_Y(\theta)}{e^{\theta na}} \tag{38}$$

$$= \frac{\prod_{i\in[n]} \mathcal{M}_{X_i}(\theta)}{e^{\theta na}} = \left(\frac{\mathcal{M}_{X_1}(\theta)}{e^{\theta a}}\right)^n \quad . \tag{39}$$

□

G Distributions

We are mainly interested in two related types of distributions: geometric distributions and negative binomial distributions, where the latter appears as a sum of the former, but we exploit the exponential distribution to bound compound distributions. We write $X \sim \mathsf{D}$ if a random variable X has distribution D.

G.1 Exponential Distribution

The exponential distribution is the archetypal continuous distribution with an exponentially decreasing tail.

Definition 30 (Exponential Distribution)**.** The *exponential distribution* $\mathsf{Exp}(\lambda)$ over the non-negative reals $(0, \infty)$ is given by its cumulative distribution function $F(x, \lambda) = 1 - e^{-\lambda x}$, i.e., a random variable $X \sim \mathsf{Exp}(\lambda)$ satisfies $\Pr\left[X \leq x\right] = 1 - e^{-\lambda x}$.

Lemma 8 (Properties of the Exponential Distribution)**.** *If $X \sim \mathsf{Exp}(\lambda)$, then*

$$\mathrm{E}\left[X\right] = \lambda^{-1} \quad , \quad \mathrm{Var}\left[X\right] = \lambda^{-2} \quad , \quad \textit{and } \mathcal{M}_X(\theta) = \frac{\lambda}{\lambda - \theta} \quad \textit{for } \theta < \lambda \quad . \tag{40}$$

G.2 Geometric Distribution

Consider some experiment that succeeds with probability p, and fails with probability $1-p$. A random variable with *unshifted* geometric distribution with probability p represents the number of failures before a successful attempt. When we refer to the geometric distribution we mean the *shifted* variation, i.e., we count the total number of attempts including the successful attempt.

Definition 31 (Geometric Distribution)**.** A random variable X has *geometric distribution* over $\{x \in \mathbb{N} \mid x > 0\}$ with probability $p \in [0,1]$, denoted $\mathsf{Geo}(p)$, if $\mathsf{P}_X(x) = (1-p)^{x-1}p$.

Lemma 9 (Properties of the Geometric Distribution)**.** *If* $X \sim \mathsf{Geo}(p)$*, then*

$$\begin{aligned} \mathrm{E}[X] &= \frac{1}{p} && (41)\\ \mathrm{Var}[X] &= \frac{1-p}{p^2} && (42)\\ \mathsf{F}_X(x) &= 1-(1-p)^{x-1} && (43)\\ \mathcal{G}_X(z) &= \frac{pz}{1-(1-p)z} && (44)\\ \mathcal{M}_X(\theta) &= \frac{pe^\theta}{1-(1-p)e^\theta} \quad \textit{for } \theta < -\ln(1-p) \ . && (45) \end{aligned}$$

G.3 A Compound Geometric Distribution

Distributions formed by letting the parameters of one distribution be chosen according to another are called compound distributions. In general compound distributions can only be bounded, but in some cases they can be described concisely.

Definition 32 (Compound Geometric Distribution)**.** A random variable X has *compound geometric distribution* $\mathsf{CG}(c,p)$ where $c \in \mathbb{N}^k$, $c_1 = 0$, and $p \in (0,1]^k$, if its probability generating function $\mathcal{G}_1(z)$ is defined by the equations

$$\begin{aligned} \mathcal{G}_k(z) &= z^{c_k}\mathcal{G}_{\mathsf{Geo}(p_k)}(z) \ \text{and} && (46)\\ \mathcal{G}_{i-1}(z) &= z^{c_{i-1}}\mathcal{G}_{\mathsf{Geo}(p_{i-1})}(\mathcal{G}_i(z)) \ . && (47) \end{aligned}$$

This distribution emerges naturally in Section 5 as the running time of an algorithm that recursively identifies a sparse subtree which satisfies certain properties at each level. The geometric distribution captures the number of attempts needed and the constants represent the added work needed after successful attempts.

Lemma 10 (Properties of the Compound Geometric Distribution)**.** *If* $X \sim \mathsf{CG}(c,p)$ *and we let* $c_{k+1} = 1$*, then*

$$\mathrm{E}[X] = \sum_{i\in[k]}\prod_{j\in[i]}\frac{1}{p_j}c_{i+1} \quad \textit{and} \tag{48}$$

$$\mathcal{G}_X(z) = \frac{z^{\sum_{i\in[k+1]}c_i}\prod_{i\in[k]}p_i}{1-\sum_{i\in[k]}q_i\prod_{j\in[i+1,k]}p_j z^{\sum_{l\in[i+1,k+1]}c_l}} \ . \tag{49}$$

Proof. The claim about the expected value follows immediately from Wald's equation [Wal44] (or by conditional expected values):

$$\begin{aligned} \mathrm{E}[X] &= \frac{1}{p_1}\left(\frac{1}{p_2}\left(\cdots\frac{1}{p_{k-1}}\left(\frac{1}{p_k}+c_k\right)+c_{k-1}\cdots\right)+c_2\right)+c_1 && (50)\\ &= \sum_{i\in[k]}\prod_{j\in[i]}\frac{1}{p_j}c_{i+1} \ . && (51) \end{aligned}$$

We adopt the notation from Definition 32 and set $q_t = 1-p_t$. We aim to derive $\mathcal{G}_t(z) = a_t(z)/b_t(z)$ for $t = k,\ldots,1$. Note that we have

$$\begin{aligned} a_k(z) &= z^{c_k}p_k z = p_k z^{c_k+c_{k+1}} \ \text{and} && (52)\\ b_k(z) &= 1-q_k z = 1-q_k z^{c_{k+1}} \ , && (53) \end{aligned}$$

and in general we have the relation

$$\mathcal{G}_{t-1}(z) = \frac{p_{t-1} z^{c_{t-1}} \mathcal{G}_t(z)}{1 - q_{t-1}\mathcal{G}_t(z)} \tag{54}$$

from which we conclude

$$\mathcal{G}_{t-1}(z) = \frac{p_{t-1} z^{c_{t-1}} a_t(z)}{b_t(z)} \Bigg/ \left(1 - \frac{q_{t-1} a_t(z)}{b_t(z)}\right) = \frac{p_{t-1} z^{c_{t-1}} a_t(z)}{b_t(z) - q_{t-1} a_t(z)} \quad . \tag{55}$$

This defines $a_t(z) = z^{\sum_{l\in[t,k+1]} c_l} \prod_{j\in[t,k]} p_j$. Solving the recursion gives

$$b_t(z) = 1 - \sum_{i\in[t,k]} q_i \prod_{j\in[i+1,k]} p_j z^{\sum_{l\in[i+1,k+1]} c_l} \tag{56}$$

which concludes the proof. □

Lemma 11. *If $0 < a \le b$, then $e^a < 1 + e^b a$.*

Proof. The proof follows by considering the Taylor expansion of e^a:

$$\begin{aligned} e^a - 1 &= a + \frac{a^2}{2} + \frac{a^3}{3!} + \frac{a^4}{4!} + \dots && (57)\\ &= a\left(1 + \frac{a}{2} + \frac{a^2}{3!} + \frac{a^3}{4!} + \dots\right) < ae^a \le ae^b \quad . && (58)\end{aligned}$$

□

Lemma 12 (Compound Geometric Distribution)**.** *If $X \sim \mathsf{CG}(c, p)$ and $Y \sim \mathsf{Exp}(\lambda)$, where $\mu = \mathrm{E}[X]$ and $\lambda = 1/\mu$, then $\mathcal{M}_X(\theta) < \mathcal{M}_Y(\theta)$ for $\theta < 1/\mu$.*

Proof. Set $\Lambda = \sum_{l\in[2,k+1]} c_l$. We use Lemma 11 (setting $a = \theta\sum_{l\in[i+1,k+1]} c_l$ and $b = \theta\Lambda$) to bound the statement from Lemma 14 in its form as a moment generating function

$$\begin{aligned} e^{\theta\Lambda}\mathcal{M}_X(\theta)^{-1} &= \frac{1 - \sum_{i\in[k]} q_i \prod_{j\in[i+1,k]} p_j e^{\theta\sum_{l\in[i+1,k+1]} c_l}}{\prod_{i\in[k]} p_i} && (59)\\ &> \prod_{i\in[k]} \frac{1}{p_i} - \sum_{i\in[k]} q_i \prod_{j\in[i]} \frac{1}{p_j}\left(1 + \theta e^{\theta\Lambda}\sum_{l\in[i+1,k+1]} c_l\right) && (60)\\ &= \prod_{i\in[k]} \frac{1}{p_i} - \sum_{i\in[k]} (1-p_i) \prod_{j\in[i]} \frac{1}{p_j} \;-\; \theta e^{\theta\Lambda} \sum_{i\in[k]} (1-p_i) \prod_{j\in[i]} \frac{1}{p_j} \sum_{l\in[i+1,k+1]} c_l \quad . && (61)\end{aligned}$$

The constant term is essentially a telescoping sum which sums to one, i.e., we have

$$\prod_{j\in[k]} \frac{1}{p_j} + \sum_{i\in[k]} \left(\prod_{j\in[i-1]} \frac{1}{p_j} - \prod_{j\in[i]} \frac{1}{p_j}\right) = 1 \quad . \tag{62}$$

The multiple of $\theta e^{\theta\Lambda}$ can be expressed similarly

$$\begin{aligned}
&\sum_{i\in[k]}\sum_{l\in[i+1,k+1]} c_l\left(\prod_{j\in[i]}\frac{1}{p_j}-\prod_{j\in[i-1]}\frac{1}{p_j}\right) && (63)\\
&=\sum_{l\in[2,k+1]} c_l \sum_{i\in[l-1]}\left(\prod_{j\in[i]}\frac{1}{p_j}-\prod_{j\in[i-1]}\frac{1}{p_j}\right) && (64)\\
&=\sum_{l\in[2,k+1]} c_l\left(\prod_{j\in[l-1]}\frac{1}{p_j}-1\right) && (65)\\
&=\sum_{i\in[k]}\prod_{j\in[i]}\frac{1}{p_j}c_{i+1}-\sum_{l\in[2,k+1]} c_l && (66)\\
&=\mu-\Lambda\ . && (67)
\end{aligned}$$

Thus, we have

$$e^{\theta\Lambda}\mathcal{M}_X(\theta)^{-1} > 1-\theta e^{\theta\Lambda}(\mu-\Lambda) \tag{68}$$

which, using $e^\theta > 1+\theta$ for $\theta > 0$, finally gives the bound

$$\begin{aligned}
\mathcal{M}_X(\theta)^{-1} &> e^{-\theta\Lambda}-\theta(\mu-\Lambda) && (69)\\
&> 1-\theta\Lambda-\theta(\mu-\Lambda) = 1-\theta\mu && (70)
\end{aligned}$$

which can be restated as $\mathcal{M}_X(\theta) < 1/(1-\theta\mu) = \lambda/(\lambda-\theta)$ as claimed. □

Theorem 8 (Cramér's Theorem for Compound Geometric Distributions)**.** *If* $X_i \sim \mathsf{CG}(c,p)$ *for* $i \in [n]$ *are independently distributed and* $Y = \sum_{i\in[n]} X_i$ *with* $\mu = \mathrm{E}[X_1]$*, then for every* $k \in (1,\infty)$*:*

$$\Pr[Y \geq nk\mu] < e^{-n(k-1-\ln k)}\quad . \tag{71}$$

Proof. Theorem 7 implies that for every $\theta < 1/\mu$:

$$\Pr[Y \geq kn\mu] \leq \left(\frac{\mathcal{M}_{X_1}(\theta)}{e^{\theta k\mu}}\right)^n\quad . \tag{72}$$

From Lemma 12 we know that $\mathcal{M}_{X_1}(\theta) < \lambda/(\lambda-\theta)$ for all $\theta < \lambda$, where $\lambda = 1/\mu$, and if we set $\theta = (1-k^{-1})/\mu$ the claim follows. □

G.4 Negative Binomial Distribution

Consider some experiment that succeeds with probability p, and fails with probability $q = 1-p$. A random variable X with negative binomial distribution with probability p and success parameter s represents how many attempts are needed to succeed s times.

Definition 33 (Negative Binomial Distribution)**.** A random variable X has *negative binomial distribution*, denoted $\mathsf{NB}(s,p)$, over $\{x \in \mathbb{N} \mid x \geq s\}$ with probability p and success parameter s if $\mathsf{P}_X(k) = \binom{k-1}{s-1}(1-p)^{k-s}p^s$.

Lemma 13 (Properties of Negative Binomial Distribution)**.** *If* $X \sim \mathsf{NB}(s,p)$*, then*

$$\begin{aligned}
\mathrm{E}[X] &= s/p && (73)\\
\mathrm{Var}[X] &= s(1-p)/p^2\ ,\quad \textit{and} && (74)\\
\mathcal{M}_X(\theta) &= \left(\frac{pe^\theta}{1-(1-p)e^\theta}\right)^s\quad \textit{for } \theta < -\ln(1-p)\ . && (75)
\end{aligned}$$

Lemma 14 (Sum of Geometric Distributions)**.** *If $X_i \sim \mathsf{Geo}(p)$ for $i \in [s]$ are independently distributed and $X = \sum_{i=1}^{s} X_i$, then $X \sim \mathsf{NB}(s,p)$.*

Proof. The multiplicative property of moment generating functions implies that

$$\mathcal{M}_X(\theta) = \prod_{i=1}^{s} \mathcal{M}_{X_i}(\theta) = \left(\frac{pe^\theta}{1-(1-p)e^\theta}\right)^s \quad , \tag{76}$$

which is the moment generating function of a binomial distribution with probability p and success parameter s. □

Chernoff's lower and upper bounds hold for negative binomial distribution similarly to the binomial distribution, but the asymmetry in the bounds is reversed since a lower bound on a random variable of the former distribution corresponds to an upper bound of one of the latter.

Theorem 9 (Chernoff's Inequalities for Negative Binomial Distribution)**.** *If $X \sim \mathsf{NB}(s,p)$ and $\mu = s/p$, then for every $k > 1$*

$$\Pr[X > k\mu] < e^{-\left(1-\frac{1}{k}\right)^2 \frac{ks}{2}} \quad \textit{and} \tag{77}$$

$$\Pr[X < \mu/k] < e^{-(k-1)^2 \frac{s}{3k}} \quad . \tag{78}$$

Proof. To prove the first inequality we set $m = k\mu = ks/p$, let $Y_1, \ldots, Y_m$ be independent binary random variables such that $\Pr[Y_i = 1] = p$, and define $Y = \sum_{i=1}^{m} Y_i$. Then

$$\Pr[X > k\mu] = \Pr[Y < s] \quad , \tag{79}$$

and the latter expression is of a convenient form to bound using Theorem 4. We have $\mathrm{E}[Y] = mp = ks$, so $s = \mu'/k$, where $\mu' = \mathrm{E}[Y]$. Thus, we set $1 - \delta = 1/k$ and conclude that

$$\Pr[Y < s] = \Pr[Y < \mu'/k] < e^{-\frac{\delta^2 \mu'}{2}} = e^{-\left(1-\frac{1}{k}\right)^2 \frac{ks}{2}} \quad . \tag{80}$$

The second inequality is proved similarly by instead setting $m = \mu/k = s/(pk)$, which gives $s = k\mu'$ with correspondingly defined random variables $Y_1, \ldots, Y_m$. Setting $1 + \delta = k$ then implies the inequality

$$\Pr[X < \mu/k] = \Pr[Y > s] = \Pr[Y > k\mu'] < e^{-\frac{\delta^2 \mu'}{3}} = e^{-(k-1)^2 \frac{s}{3k}} \quad . \tag{81}$$

□

H Stochastic Dominance and Bounding Distributions

One approach to compare distributions is to not only bound expected values, variances, or probabilities of certain events, but instead find families of distributions that are ordered stochastically. The advantage of this, when possible, is that more structural information about the original distribution can be retained. We only need first-order stochastic dominance over $\mathbb{N}$.

Definition 34 (Stochastic Dominance)**.** Let X and Y be random variables over $\mathbb{N}$. Then Y *stochastically dominates* X, denoted $X \preceq Y$, if $\mathsf{F}_X(z) \leq \mathsf{F}_Y(z)$ for every $z \in \mathbb{N}$.

This is a somewhat confusing definition when the random variables encode running times of algorithms, since providing a bound of a running time with distribution D_X amounts to defining a distribution D_Y such that Y is stochastically dominated by X. This motivates the following more natural definition.

Definition 35 (Bounding Distribution)**.** Let D_X and D_Y be distributions over $\mathbb{N}$. Then D_X *is bounded by* D_Y if $Y \preceq X$.