Special Soundness Revisited*

Douglas Wikström

KTH Royal Institute of Technology
dog@kth.se

November 27, 2018

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 concentrated running time.

1 Introduction

A three-message public-coin protocol [6,1] is defined to be *special sound* by Cramer et al. [4] if a witness can be computed efficiently from two accepting transcripts with a common first prover message, but distinct verifier messages. This notion generalizes a property of Schnorr's proof of knowledge of a discrete logarithm [8].

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 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 [8] or linear independence [2].

Recall that in proofs of knowledge [3] 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

Our work is motivated by, and addresses, a real-world need to give an exact proof of security for a complete electronic voting system. We are not aware of prior work that is sufficiently rigorous, general, flexible, and exact to be used as the toolbox we need, but given how well researched the area is the main contribution is perhaps the complete and coherent nature of our treatment with applications in both theory and practice.

3 The Extraction Problem

Given are matroids $\mathbb{M}_0, \ldots, \mathbb{M}_r$, 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 that captures both the computations performed by the prover during execution and the computations performed by the verifier to reach a verdict then has the form $\rho : \prod_{i\in[0,r]} S_i \to \{0,1\}$. An explicit description parametrized by a prover is given in Definition 13.

The goal is to find a 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.

3.1 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 $O(d/\Delta)$, where $\Delta = \Pr[\rho(v) = 1]$ for a randomly chosen $v \in \prod_{i\in[0,r]} S_i$.

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 aggressively increases the constant factor of the running time drastically.

3.2 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 quite concentrated from scratch 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 [7] for how bounds on sums of geometric distributions with different probabilities behave).

In other words, 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.

3.3 Strategy

The basic 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 has negative binomial distribution.

The general strategy is relatively natural and recursive, so 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.

4 Matroids and Trees

We denote a matroid with ground set S and independence sets I by $\mathbb{M} = (S, I)$. For completeness we provide one standard way of formalizing matroids in Section A. We denote the matroid with a singleton ground set $\{u\}$ and independence set $\{\varnothing, \{u\}\}$ by $\{u\}$. 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 (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 2 (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.

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 relax the requirement to make room for oddities introduced in cryptographic protocols.

Definition 1 (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 *$\omega_{\mathbb{M}} = \omega_{\mathbb{M},d}$.*

Note that we have $\omega_{\mathbb{M},1} = 0$ for every non-trivial matroid $\mathbb{M}$, since $\mathsf{span}(\varnothing) = \varnothing$. In Example 1 the ith subdensity is q^{i-N-1} and in Example 2 the 2nd subdensity is $1/|S|$. We introduce some additional notation that allow us to consider a list of matroids as a tree.

Definition 2 (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 S_i.*

Although a matroid tree is unordered, the children of each node are labeled uniquely, so we may abuse notation and identify a node with its label. We also use $\mathbb{M}$ to denote both the matroid tree and the list of matroids with which it is associated.

Definition 3 (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 $\mathbb{M}_i$.*

5 Predicates On Paths

As explained above, we abstract from the details of protocols by capturing the computations performed by the prover and verifier in a protocol to reach a verdict as a predicate. Definition 13 gives a concrete predicate for any given prover.

Definition 4 (Predicate). *An* $\mathbb{M}$-predicate*, where $\mathbb{M}$ is a matroid tree, is a function of the form $\rho : \prod_{i \in [0,r]} S_i \to \{0,1\}$.*

Definition 5 (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 Accepting Basis Extractors

For a matroid tree $\mathbb{M}$ we let $S = \prod_{i \in [0,r]} S_i$, where $\mathbb{M}_i = (S_i, I_i)$ and $S_0 = \{v_0\}$ for some v_0. We define $\Delta_\rho(\mathbb{M}) = |\{\rho(v) = 1 \mid v \in S\}|/|S|$, and 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 $x \in \mathbb{N}$: $\Pr[X \leq x] \geq \Pr[X' \leq x]$.

Definition 6 (Accepting Basis Extractor). *A probabilistic polynomial time oracle algorithm $\mathcal{X}_\kappa$ parametrized by $\kappa \in \{0,1\}^*$ is a* $(\epsilon_\kappa, \mathsf{D}_\kappa(\Delta))$-accepting basis extractor *with* extraction error ϵ_κ *for a matroid tree $\mathbb{M}$, where $\mathsf{D}_\kappa(\Delta)$ for fixed κ is a family of distributions on $\mathbb{N}$ parametrized by $\Delta \in [0,1]$, if for every $\mathbb{M}$-predicate $\rho : S \to \{0,1\}$ and $\Delta_\rho(\mathbb{M}) \geq \Delta_0 > \epsilon_\kappa$: $\mathcal{X}_\kappa^{\rho(\cdot)}(\mathbb{M}, \Delta_0)$ outputs a ρ-accepting basis of $\mathbb{M}$, where the distribution of the number of $\rho(\cdot)$-queries is bounded by $\mathsf{D}_\kappa(\Delta_0)$.*

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. It also allows modifying the extractor and the corresponding extraction error ϵ_κ and distribution $\mathsf{D}_\kappa(\Delta_0)$ using the parameter κ, but 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.

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.

7 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 $\mathcal{G}_\mathsf{D}(z)$.

7.1 Notation for Bounds

Consider a random variable X taking values in $\mathbb{N}$ that 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 certain lower bound λ by

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

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

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

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

$$t_s^{\mathsf{NB}}(k) = e^{-(1-1/k)^2 ks/2}$$
$$t_s^{\mathsf{CG}}(k) = e^{-(k-1-\ln k)s}$$

for the negative binomial distribution $\mathsf{NB}(s, \Delta)$ for some probability Δ, and a product of s compound geometric distributions (see Definition 29), 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 7.)

7.2 Basic Algorithms

Definition 7 (Basic Extractor). *The* basic extractor *oracle algorithm $\mathcal{B}$ takes input $(\mathbb{M}, \Delta_0)$, 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 $\Delta_1' = \Delta_0 - \omega_{\mathbb{M}_1}$.*

Proof. 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.

Remark 1. The algorithm ignores the input Δ_0 and the result could be sharpened to say that the distribution of the number of queries to the oracle is bounded by $\mathsf{NB}(d_1, \Delta_\rho - \omega_{\mathbb{M}_1})$. We choose the above exposition to keep a uniform interface and structure with the algorithms that follow below.

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

1. *Sample* $v \in S_0 \times (S_1 \setminus \mathsf{span}(B)) \times \prod_{i\in[2,r]} S_i$ *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

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

so

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

which implies

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

Markov's inequality then implies that $\Pr[\delta_U < \alpha\Delta_1'] = \Pr[1/\delta_U > 1/(\alpha\Delta_1')] \leq \alpha$ so $\Pr[\delta_U \geq \alpha\Delta_1'] \geq 1 - \alpha$.

Definition 9 (Sample Validator). *The* sample validator *oracle 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* $v \in \prod_{i\in[0,r]} S_i$ *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* $\mathcal{V}_{s,k}^{\rho(\cdot)}(\mathbb{M}, \Delta_0)$ *is bounded by* sk/Δ_0*. If* $\Delta_\rho \geq \Delta_0$*, then* $\Pr\left[\mathcal{V}_{s,k}^{\rho(\cdot)}(\mathbb{M}, \Delta_0) = 1\right] \geq h_s^{\mathsf{NB}}(k)$*, and if* $\Delta_\rho < \frac{\Delta_0}{k}$*, then* $\Pr\left[\mathcal{V}_{s,k}^{\rho(\cdot)}(\mathbb{M}, \Delta_0) = 1\right] \leq n_s^{\mathsf{NB}}(k)$*.*

Proof. Note that if the bound $c < sk/\Delta_0$ is removed, then the number of oracle calls has distribution bounded by $\mathsf{NB}(s, \Delta_\rho)$, since each sample v satisfies $\rho(v) = 1$ with probability at least Δ_ρ and we only exit the loop when $h = s$. The claims now follow directly from the tail and head bounds of the negative binomial distribution (see Theorem 8).

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 10 (Validating Sampler). *The* validating sampler *oracle 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 $s = s_\beta^{\mathsf{NB}}(\gamma)$.*
2. *Repeat:*
   (a) *Compute $v_1 = \mathcal{BS}^{\rho(\cdot)}(\mathbb{M}, B, \Delta_0)$.*
   (b) *If $\mathcal{V}_{s,k}^{\rho(v_0,\cdot)}\big((\{v_1\}, \mathbb{M}_2, \ldots, \mathbb{M}_r), \alpha\Delta_1'\big) = 1$, 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 $\mathcal{VS}_{\alpha,\beta,\gamma}^{\rho(\cdot)}(\mathbb{M}, B, \Delta_0)$ is bounded by*

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

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 $\phi(\alpha,\beta,\gamma) = 1 - \alpha\beta n_s^{\mathsf{NB}}(1/\beta)/((1-\alpha)\gamma)$.

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 $\mathcal{V}_{s,k}^{\rho(v_0,\cdot)}$ is upper bounded by $ks/(\alpha\Delta_1') = s_\beta^{\mathsf{NB}}(\gamma)/\Delta_1$ so the distribution of the total number of queries in the ith iteration is bounded by

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

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 $h_s^{\mathsf{NB}}(k) \geq \gamma$. Thus, the distribution of the number of samples considered is bounded by $\mathsf{Geo}((1-\alpha)\gamma)$. This implies that $\mathcal{G}_{\mathsf{Geo}((1-\alpha)\gamma)}(f(z))$ 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 $\mathcal{V}_{s,k}^{\rho(v_0,\cdot)}\big((\{U_i\}, \mathbb{M}_2, \ldots, \mathbb{M}_r), \alpha\Delta_1'\big)$, i.e., it is the indicator variable for the event that $\mathcal{VS}_{\alpha,\beta,\gamma}$ returns U_i. Then Lemma 3 implies that $\Pr[A_i = 1 \,|\delta_{U_i} < \alpha\beta\Delta_1'] \leq n_s^{\mathsf{NB}}(k)$ which gives

$$\Pr[\delta_{U_i} < \alpha\beta\Delta_1' \,|A_i = 1] < \frac{\alpha\beta n_s^{\mathsf{NB}}(1/\beta)}{\Pr[A_i = 1]} \leq \frac{\alpha\beta n_s^{\mathsf{NB}}(1/\beta)}{(1-\alpha)\gamma} \ .$$

7.3 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, the distributions of queries, and bounds on their outputs we use the following notation. The sampling algorithm is defined by:

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

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

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

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 11 (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 2 (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 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 1 to get slightly better results.

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*

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

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 is bounded by $\mathsf{Geo}(\phi\lambda)$, and we need d_1 successes.

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

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

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

Proof. The expected value follows from linearity and Wald's equation [10] (or directly from Lemma 8). 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

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

and the fact that $\phi = 1 - \alpha$ when $\beta = 1$. The tail bound follows directly from Theorem 7 for this type of compound geometric distribution.

7.4 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 subdensity which is 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 1 (Extractor). *For every* $\nu_1, \ldots, \nu_{r-1} > 1$ *there exist parameters* $\kappa_i = (\alpha_i, \beta_i, \gamma_i, \lambda_i)$ *such that the oracle algorithm* $\mathcal{X}_\kappa = \mathcal{R}_{\kappa_1}[\mathcal{R}_{\kappa_2}[\cdots\mathcal{R}_{\kappa_{r-2}}[\mathcal{B}]\cdots]]$,

is a $(\epsilon_0, \mathsf{D}(\Delta_0))$*-accepting basis extractor for matroid trees of depth* r *where:*

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

$$\mu_{\mathsf{D}_0(\Delta_0)} \leq \frac{c_0 \prod_{j\in[r]} d_j}{\Delta_0 - \epsilon_0} \qquad \textit{(expected number of queries)} \tag{2}$$

$$t_{d_1}^{\mathsf{D}_0}(k) \leq t_{d_1}^{\mathsf{CG}}(k) \quad \textit{for } k > 1 \ , \qquad \textit{(tail bound)} \tag{3}$$

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*

$$f_{\mathsf{BS}}(d,\nu,c) = \frac{\nu^2}{(\nu-1)} \cdot \min\{k/h_d^{\mathsf{CG}}(k)\} \cdot c$$

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

$$f_{\mathsf{S}}(d,\nu,c) = \min\left\{ f_{\mathsf{BS}}(d,\nu,c), f_{\mathsf{VS}}(d,\nu,c) \right\} \ ,$$

with $D_{i,r} = \prod_{j\in[i,r]} d_j$*,* $\alpha \in (0, 1/\nu)$*,* $s \in \mathbb{N}^+$*, 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. We prove a slightly stronger result where we exploit the special properties of leaves.

Proof (Theorem 1). 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

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

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

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

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 $c_i D_{i+1,r}/\Delta_i$ for a constant c_i provided that its oracle $\rho_i(\cdot)$ and input (N_i, Δ_i) are reasonably good. More precisely, it is called with an oracle of the form $\rho_i(\cdot) = \rho(v_0, \ldots, v_{i-2}, \cdot)$ 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

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

for $i \in [r]$.

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

$$\begin{aligned}
\mu_{\mathsf{D}_{i-1}(\Delta_{i-1})} &= \frac{d_i}{(1-\alpha_i)\lambda_i} k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\Delta_i)} \\
&= \frac{d_i k^{\mathsf{D}_i}(\lambda_i)}{(1-\alpha_i)\lambda_i} \cdot \frac{c_i D_{i+1,r}}{\Delta_i - \epsilon_i} \\
&= \frac{k^{\mathsf{D}_i}(\lambda_i)}{(1-\alpha_i)\lambda_i} \cdot c_i \cdot \frac{D_{i,r}}{\alpha_i(\Delta_{i-1} - \epsilon_{i-1})} \\
&= \frac{1}{\alpha_i(1-\alpha_i)} \cdot \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i \cdot \frac{D_{i,r}}{\Delta_{i-1} - \epsilon_{i-1}} \ .
\end{aligned}$$

If we choose an optimal λ_i, then

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

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

$$\begin{aligned}
\mu_{\mathsf{D}_{i-1}(\Delta_{i-1})} &= \frac{d_i}{\phi_i\lambda_i} \left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{(1-\alpha_i)\gamma_i} \frac{1}{\Delta_i} + k^{\mathsf{D}_i}(\lambda_i)\mu_{\mathsf{D}_i(\Delta_i)} \right) \\
&= \frac{d_i}{\zeta_i\phi_i\lambda_i} \left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{(1-\alpha_i)\gamma_i} \frac{1}{\Delta_{i-1} - \epsilon_{i-1}} + k^{\mathsf{D}_i}(\lambda_i) \frac{c_i D_{i+1,r}}{\Delta_{i-1} - \epsilon_{i-1}} \right) \\
&= \frac{1}{\zeta_i\phi_i} \left(\frac{\zeta_i + s^{\mathsf{NB}}_{\beta_i}(\gamma_i)}{(1-\alpha_i)\gamma_i\lambda_i} \cdot \frac{d_i}{D_{i+1,r}} + \frac{k^{\mathsf{D}_i}(\lambda_i)}{\lambda_i} \cdot c_i \right) \frac{D_{i,r}}{\Delta_{i-1} - \epsilon_{i-1}}
\end{aligned}$$

If we choose parameters optimally, then we have

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

Leaves allow sharper bounds. Recursive calls deep in the tree are special in two ways. Firstly, the distribution of the number of queries is negative binomial, so a slightly sharper head bound $h^{\mathsf{NB}}_{d_r}(k)$, instead of $h^{\mathsf{CC}}_{d_r}(k)$, can be used to bound the probability that $\mathcal{B}$ completes in time $k\mu_{\mathsf{D}_{r-1}(\Delta_{r-1})}$.

Secondly, and more importantly for the second strategy it is worthwile to re-use paths from the sampler in the recursive call if d_r is large. This gives a mutual dependency between the parameters of recursive calls at depth $r-2$ and $r-1$, but fortunately the advantage of re-use diminishes quickly, so it suffices to consider this for $\mathcal{X}_{\nu,r-2}$. In this case we have $c_{r-1} = \max\{1, d_r - 1 - s^{\mathsf{NB}}_{\beta_{r-2}}(\gamma_{r-2})\}/d_r$.

7.5 Interpretation

In the following we assume that Δ_0 is a tight lower bound of Δ_ρ, since this is a setup assumption in our approach. 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 [3] 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 $\epsilon_0 < \Delta_0$, 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. 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.

7.6 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 12 (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 that is a multiplication in a field or similar.

8 Special Soundness

We can now express a general form of special soundness using matroid trees. Note that the message spaces and what usually appears as the knowledge error in concrete presentations are captured by the ground sets, ranks, and the subdensities of the matroids.

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. The following definition instantiates our abstract predicate.

Definition 13 (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 $c = (v_1, \ldots, v_r)$.*

Definition 14 (Accepting Transcript Tree). *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: $(v_{u_0}, a_{u_0}, \ldots, v_{u_r}, a_{u_r})$ is accepting, and $\ell(u_i) = (v_{u_i}, a_{u_i})$.*

Note that v_{u_0} corresponds to the instance of the execution. 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. We need a convenient notation to project the labels of the tree to their verifier message parts to allow us to state conditions on accepting transcript trees.

Definition 15 (Challenge Tree). *The* challenge tree $\mathcal{C}(T)$ *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 $\ell(u) = (v, a)$.*

Definition 16 (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 $\mathcal{C}(T)$ 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.*

9 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 [5,9].

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 decompose an NP relation.

Definition 17 (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 $j \in [k]$.*

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. This may seem complicated, but turns out to be convenient and preserves the strong properties of special soundness.

Definition 18 (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 $p = \sum_{j \in [k]} p_j$.*

Although the parametrized matroids have the same ground sets, the sets of independence sets may differ. 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 parametrized matroids must be provided along with the algorithms that compute a witnesses from accepting transcript trees.

References

1. L. Babai. Trading group theory for randomness. In R. 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.

2. M. Bellare, J. A. Garay, and T. Rabin. Fast batch verification for modular exponentiation and digital signatures. In K. 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.
3. M. Bellare and O. Goldreich. On defining proofs of knowledge. In E. 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.
4. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. 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.
5. J. Furukawa and K. Sako. An efficient scheme for proving a shuffle. In J. 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.
6. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
7. S. Janson. Tail bounds for sums of geometric and exponential variables, 2017.
8. C. Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.
9. B. Terelius and D. Wikström. Proofs of restricted shuffles. In D. J. Bernstein and T. 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.
10. A. Wald. On cumulative sums of random variables. *Ann. Math. Statist.*, 15(3):283–296, 09 1944.

A Matroids

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

Definition 19 (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 20 (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 21 (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 22 (Rank). *The* rank *of a matroid (S, I) is the unique cardinality of each basis in I.*

Definition 23 (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 24 (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 $\mathsf{span}(A) = A$.*

B Generic Bounds on Random Variables

We use two 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 2 (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 3 (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 $\delta \in (0, 1)$:*

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

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

C 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 25 (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 4 (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 $\mathcal{G}_Y(\cdot)$, then $\mathcal{G}_X(z)\mathcal{G}_Y(z)$ is the probability generating function of the sum $X+Y$.*

Definition 26 (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 5 (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 6 (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:*

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

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

$$\begin{aligned}\Pr\left[Y \geq na\right] = \Pr\left[e^{\theta Y} \geq e^{\theta na}\right] &\leq \frac{\mathrm{E}\left[e^{\theta Y}\right]}{e^{\theta na}} = \frac{\mathcal{M}_Y(\theta)}{e^{\theta na}} \\ &= \frac{\prod_{i\in[n]} \mathcal{M}_{X_i}(\theta)}{e^{\theta na}} = \left(\frac{\mathcal{M}_{X_1}(\theta)}{e^{\theta a}}\right)^n .\end{aligned}$$

D 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.

D.1 Exponential Distribution

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

Definition 27 (Exponential Distribution). *The* exponential distribution $\mathsf{Exp}(\lambda)$ *over* $(0, \infty)$ *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* $\Pr[X \leq x] = 1 - e^{-\lambda x}$.

Lemma 6 (Properties of the Exponential Distribution). *If* $X \sim \mathsf{Exp}(\lambda)$, *then*

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

D.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 28 (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 7 (Properties of the Geometric Distribution). *If* $X \sim \mathsf{Geo}(p)$, *then*

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

D.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 29 (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* $g_1(z)$ *is defined by the equations*

$$\begin{aligned}
g_k(z) &= z^{c_k} f_k(z) \\
g_{i-1}(z) &= z^{c_{i-1}} f_{i-1}(g_i(z))
\end{aligned}$$

where $f_i(z) = \mathcal{G}_{\mathsf{Geo}(p_i)}(z)$.

This distribution emerges naturally in Section 8 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 8 (Properties of Compound Geometric Distribution). *If* $X \sim \mathsf{CG}(c, p)$ *and we let* $c_{k+1} = 1$*, then*

$$\mathrm{E}[X] = \sum_{i\in[k]} \prod_{j\in[i]} \frac{1}{p_j} c_{i+1} \quad \textit{and}$$
$$\mathcal{G}_X(z) = \frac{z^{\sum_{i\in[k+1]} c_i} \prod_{i\in[k]} p_i}{1 - \sum_{i\in[k]} q_i \prod_{j\in[i+1,k]} p_j z^{\sum_{l\in[i+1,k+1]} c_l}} \ .$$

Proof. The claim about the expected value follows immediately from Wald's equation [10] (or by conditional expected values):

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

We adopt the notation from Definition 29 and set $q_t = 1 - p_t$. We aim to derive $g_t(z) = a_t(z)/b_t(z)$ for $t = k, \ldots, 1$. Note that we have

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

and in general we have the relation

$$g_{t-1}(z) = \frac{p_{t-1} z^{c_{t-1}} g_t(z)}{1 - q_{t-1} g_t(z)}$$

from which we conclude

$$g_{t-1}(z) = \frac{p_{t-1} z^{c_{t-1}} a_t(z)}{b_t(z)} \Big/ \left(1 - \frac{q_{t-1} a_t(z)}{b_t(z)}\right) = \frac{p_{t-1} z^{c_{t-1}} a_t(z)}{b_t(z) - q_{t-1} a_t(z)} \ .$$

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

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

which concludes the proof.

Lemma 9. *If* $0 < a \le b$*, then* $e^a < 1 + e^b a$.

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

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

Lemma 10 (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 9 (setting $a = \theta \sum_{l\in[i+1,k+1]} c_l$ and $b = \theta\Lambda$) to bound the statement from Lemma 12 in its form as a moment generating function

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

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

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

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

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

Thus, we have

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

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

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

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

Theorem 7 (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 $k \in (1, \infty)$:*

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

Proof. Theorem 6 implies that for every $\theta < 1/\mu$:

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

From Lemma 10 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.

D.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 30 (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 11 (Properties of Negative Binomial Distribution). *If $X \sim \mathsf{NB}(s, p)$, then*

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

Lemma 12 (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

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

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

$$\Pr[X > k\mu] < e^{-\left(1-\frac{1}{k}\right)^2 \frac{ks}{2}} \quad \textit{and} \tag{4}$$

$$\Pr[X < \mu/k] < e^{-(k-1)^2 \frac{s}{3k}} \quad . \tag{5}$$

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

$$\Pr[X > k\mu] = \Pr[Y < s] \quad ,$$

and the latter expression is of a convenient form to bound using Theorem 3. 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

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

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

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

E 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 31 (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 32 (Bounding Distribution). *Let* D_X *and* D_Y *be distributions over* $\mathbb{N}$. *Then* D_X is bounded by D_Y *if* $Y \preceq X$.