J Cryptol (2023) 36:36
https://doi.org/10.1007/s00145-023-09478-y

Journal of CRYPTOLOGY

Research Article

Fiat–Shamir Transformation of Multi-Round Interactive Proofs (Extended Version)*

Thomas Attema
CWI, Cryptology Group, Amsterdam, The Netherlands
Leiden University, Mathematical Institute, Leiden, The Netherlands
TNO, Cyber Security and Robustness, The Hague, The Netherlands
thomas.attema@tno.nl

Serge Fehr
CWI, Cryptology Group, Amsterdam, The Netherlands
Leiden University, Mathematical Institute, Leiden, The Netherlands
serge.fehr@cwi.nl

Michael Klooß
Karlsruhe Institute of Technology, KASTEL, Karlsruhe, Germany
michael.klooss@kit.edu

Communicated by Amit Sahai.

Received 14 October 2022 / Revised 23 June 2023 / Accepted 10 July 2023

Abstract. The celebrated Fiat–Shamir transformation turns any public-coin interactive proof into a non-interactive one, which inherits the main security properties (in the random oracle model) of the interactive version. While originally considered in the context of 3-move public-coin interactive proofs, i.e., so-called Σ-protocols, it is now applied to multi-round protocols as well. Unfortunately, the security loss for a $(2\mu+1)$-move protocol is, in general, approximately Q^μ, where Q is the number of oracle queries performed by the attacker. In general, this is the best one can hope for, as it is easy to see that this loss applies to the μ-fold sequential repetition of Σ-protocols, but it raises the question whether certain (natural) classes of interactive proofs feature a milder security loss. In this work, we give positive and negative results on this question. On the positive side, we show that for $(k_1, \ldots, k_\mu)$-special-sound protocols (which cover a broad class of use cases), the knowledge error degrades linearly in Q, instead of Q^μ. On the negative side, we show that for t-fold *parallel repetitions* of typical $(k_1, \ldots, k_\mu)$-special-sound protocols with $t \geq \mu$ (and assuming for simplicity that t and Q are integer multiples of μ), there is an attack that results in a security loss of approximately $\frac{1}{2} Q^\mu / \mu^{\mu+t}$.

*This work is an extended version of the conference paper [6]. It includes the proofs of Lemmas 1, 5 and 6. Further, a discussion on the applicability of our results to arguments, rather than proofs, is included (Sect. 2.4), the results are generalized to multi-round interactive proofs with challenge sets varying over the different rounds (Sect. 6.2), and the analysis is extended from static to adaptive security (Sects. 2.5 and 6.3). Moreover, the efficiency of sampling without replacement, a core technique used by our extractors, is argued (Appendix A). Finally, a more detailed discussion of the attack on the Fiat–Shamir transformation of multi-round interactive proofs is included (Appendix B).

Published online: 08 August 2023

1. Introduction

1.1. *Background and State of the Art*

The celebrated and broadly used Fiat–Shamir transformation turns any public-coin interactive proof into a *non-interactive* proof, which inherits the main security properties (in the random oracle model) of the interactive version. The rough idea is to replace the random challenges, which are provided by the verifier in the interactive version, by the hash of the current message (concatenated with the messages from previous rounds). By a small adjustment, where also the to-be-signed message is included in the hashes, the transformation turns any public-coin interactive proof into a signature scheme. Indeed, the latter is a commonly used design principle for constructing very efficient signature schemes.

While originally considered in the context of 3-move public-coin interactive proofs, i.e., so-called Σ-protocols, the Fiat–Shamir transformation also applies to *multi-round* protocols. However, a major drawback in the case of multi-round protocols is that, in general, the security loss obtained by applying the Fiat–Shamir transformation grows exponentially with the number of rounds. Concretely, for any $(2\mu+1)$-move interactive proof Π (where we may assume that the prover speaks first and last, so that the number of communication rounds is indeed odd) that admits a cheating probability of at most ϵ, captured by the knowledge or soundness error, the Fiat–Shamir-transformed protocol $\mathsf{FS}[\Pi]$ admits a cheating probability of (approximately) at most $Q^\mu \cdot \epsilon$, where Q denotes the number of random-oracle queries admitted to the dishonest prover. A tight reduction is due to [12] with a security loss $\binom{Q}{\mu} \approx \frac{Q^\mu}{\mu^\mu}$, where the approximation holds whenever μ is much smaller than Q, which is the typical case. More concretely, [12] introduces the notions of *state-restoration soundness (SRS)* and *state-restoration knowledge (SRK)*, and it shows that any (knowledge) sound protocol Π satisfies these notions with the claimed security loss,[1] The security of $\mathsf{FS}[\Pi]$ (with the same loss) then follows from the fact that these soundness notions imply the security of the Fiat–Shamir transformation.

Furthermore, there are (contrived) examples of multi-round protocols Π for which this Q^μ security loss is almost tight. For instance, the μ-fold sequential repetition Π of a special-sound Σ-protocol with challenge space $\mathcal{C}$ is ϵ-sound with $\epsilon = \frac{1}{|\mathcal{C}|^\mu}$, while it is easy to see that, by attacking the sequential repetitions round by round, investing Q/μ queries per round to try to find a "good" challenge, and assuming $|\mathcal{C}|$ to be much larger than Q, its Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ can be broken with probability approximately $\left(\frac{Q}{\mu}\frac{1}{|\mathcal{C}|}\right)^\mu = \frac{Q^\mu}{\mu^\mu} \cdot \epsilon$.[2]

For μ beyond 1 or 2, let alone for non-constant μ (e.g., IOP-based protocols [7,11,12] and also Bulletproofs-like protocols [9,10]), this is a very unfortunate situation when it comes to choosing concrete security parameters. If one wants to rely on the proven security reduction, one needs to choose a large security parameter for Π, in order to compensate for the order Q^μ security loss, effecting its efficiency; alternatively, one has

[1] As a matter of fact, [12] considers arbitrary *interactive oracle proofs (IOPs)* but these notions are well-defined for ordinary interactive proofs too.

[2] This is clearly a contrived example since the natural construction would be to apply the Fiat–Shamir transformation to the *parallel* repetition of the original Σ-protocol, where no such huge security loss would then occur.

to give up on proven security and simply *assume* that the security loss is much milder than what the general bound suggests. Often, the security loss is simply ignored.

This situation gives rise to the following question: *Do there exist natural classes of multi-round public-coin interactive proofs for which the security loss behaves more benign than what the general reduction suggests?* Ideally, the general Q^μ loss appears for contrived examples *only*.

So far, the only positive results, establishing a security loss linear in Q, were established in the context of *straight-line/online* extractors that do not require rewinding. These extractors either rely on the algebraic group model (AGM) [27], or are restricted to protocols using hash-based commitment schemes in the random oracle model [12]. To analyze the properties of straight-line extractors, new auxiliary soundness notions were introduced: *round-by-round (RBR) soundness* [19] and *RBR knowledge* [20]. However, it is unclear if and how these notions can be used in scenarios where straight-line extraction does not apply.

In this work, we address the above question (in the plain random-oracle model, and without restricting to schemes that involve hash-based commitments), and give both positive and negative answers, as explained in more detail below.

1.2. *Our Results*

1.2.1. *Positive Result*

We show that the Fiat–Shamir transformation of any $(k_1, \ldots, k_\mu)$-special-sound interactive proof has a security loss of at most $Q + 1$. More concretely, we consider the *knowledge error* κ as the figure of merit, i.e., informally, the maximal probability of the verifier accepting the proof when the prover does not have a witness for the claimed statement, and we prove the following result, also formalized in the theorem below. For any $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-move interactive proof Π with knowledge error κ (which is a known function of $(k_1, \ldots, k_\mu)$), the Fiat–Shamir transformed protocol $\mathsf{FS}[\Pi]$ has a knowledge error at most $(Q+1)\cdot\kappa$. This result is directly applicable to a long list of recent zero-knowledge proof systems, e.g., [1,3,9,10,13,14,25,32,38]. While all these works consider the Fiat–Shamir transformation of special-sound protocols, most of them ignore the associated security loss.

Main Theorem. (Theorem 2) *Let Π be a $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-sound interactive proof with knowledge error κ. Then, the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of Π is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q + 1) \cdot \kappa \,.$$

Since in the Fiat–Shamir transformation of any $(2\mu+1)$-move protocol Π, a dishonest prover can simulate any attack against Π and can try Q/μ times when allowed to do Q queries in total, our new upper bound $(Q + 1) \cdot \kappa$ is close to the trivial lower bound $1 - (1 - \kappa)^{Q/\mu} \approx Q\kappa/\mu$. Another, less explicit, security measure in the context of knowledge soundness is the runtime of the knowledge extractor. Our bound on the knowledge error holds by means of a knowledge extractor that makes an expected number

of $K+Q\cdot(K-1)$ queries, where $K=k_1\cdots k_\mu$. This is a natural bound: K is the number of necessary distinct "good" transcripts (which form a certain tree-like structure). The loss of $Q\cdot(K-1)$ captures the fact that a prover may finish different proofs, depending on the random oracle answers, and only one out of Q proofs may be useful for extraction, as explained below.

Our result on the *knowledge* soundness of $\mathsf{FS}[\Pi]$ for special-sound protocols Π immediately carries over to *ordinary* soundness of $\mathsf{FS}[\Pi]$, with the same security loss $Q+1$. However, proving knowledge soundness is more intricate; showing a linear-in-Q loss for ordinary soundness can be obtained via simpler arguments (e.g., there is no need to argue efficiency of the extractor).

The construction of our knowledge extractor is motivated by the extractor from [2] in the interactive case, but the analysis here in the context of a non-interactive proof is much more involved. We analyze the extractor in an inductive manner and capture the induction step (and the base case) by means of an abstract experiment. The crucial idea for the analysis (and extractor) is how to deal with accepting transcripts which are not useful.

To see the core problem, consider a Σ-protocol, i.e., a 3-move k-special-sound interactive proof, and a semi-honest prover that knows a witness and behaves as follows. It prepares, independently, Q first messages $a^1,\ldots,a^Q$ and asks for all hashes $c^i=\mathsf{RO}(a^i)$, and then decides "randomly" (e.g., using a hash over all random oracle answers) which thread to complete, i.e., for which i^* to compute the response z and then output the valid proof (a^{i^*},z). When the extractor then reprograms the random oracle at the point a^{i^*} to try to obtain another valid response but now for a different challenge, this affects i^*, and most likely the prover will then use a different thread j^* and output the proof (a^{j^*},z') with $a^{j^*}\neq a^{i^*}$. More precisely, $\Pr(j^*=i^*)=1/Q$. Hence, an overhead of Q appears in the runtime.

In case of an *arbitrary* dishonest prover with an unknown strategy for computing the a^i's above, and with an arbitrary (unknown) success probability ϵ, the intuition remains: after reprogramming, we still expect $\Pr(j^*=i^*)\geq 1/Q$ and thus a linear-in-Q overhead in the runtime of the extractor. However, providing a rigorous proof is complicated by the fact that the event $j^*=i^*$ is not necessarily independent of the prover producing a *valid* proof (again) after the reprogramming. Furthermore, conditioned on the prover having been successful in the first run and conditioned on the corresponding i^*, the success probability of the prover after the reprogramming may be skewed, i.e., may not be ϵ anymore. As a warm-up for our general multi-round result, we first give a rigorous analysis of the above case of a Σ-protocol. For that purpose, we introduce an abstract sampling game that mimics the behavior of the extractor in finding two valid proofs with $j^*=i^*$, and we bound the success probability and the "cost" (i.e., the number of samples needed) of the game, which directly translate to the success probability and the runtime of the extractor.

Perhaps surprisingly, when moving to *multi-round* protocols, dealing with the knowledge error is relatively simple by recursively composing the extractor for the Σ-protocol. However, controlling the runtime is intricate. If the extractor is recursively composed, i.e., it makes calls to a sub-extractor to obtain a sub-tree, then a naive construction and analysis gives a blow-up of Q^μ in the runtime. Intuitively, because only $1/Q$ of the sub-extractor runs produce useful sub-trees, i.e., sub-trees which extend the current a^{i^*}.

The other trees belong to some a^{j^*} with $j^* \neq i^*$ and are thus useless. This overhead of Q then accumulates per round (i.e., per sub-extractor).

The crucial observation that we exploit in order to overcome the above issue is that the very first (accepting) transcript sampled by a sub-extractor already determines whether a sub-tree will be (potentially) useful, or not. Thus, if this very first transcript already shows that the sub-tree will not be useful, there is no need to run the full-fledged sub-tree extractor, saving precious time.

To illustrate this more, we again consider the simple case of a dishonest prover that succeeds with certainty. Then, after the first run of the sub-extractor to produce the first sub-tree (which requires expected time linear in Q) and having reprogrammed the random oracle with the goal to find another sub-tree that extends the current a^{i^*}, it is cheaper to first do a single run of the prover to learn j^* and only run the full-fledged sub-extractor if $j^* = i^*$, and otherwise reprogram and re-try again. With this strategy, we expect Q tries, followed by the run of the sub-extractor, to find a second fitting sub-tree. Altogether, this amounts to linear-in-Q runs of the prover, compared to the Q^2 using the naive approach.

Again, what complicates the rigorous analysis is that the prover may succeed with bounded probability ϵ only, and the event $j^* = i^*$ may depend on the prover/sub-extractor being successful (again) after the reprogramming. Furthermore, as an additional complication, conditioned on the sub-extractor having been successful in the first run and conditioned on the corresponding i^*, *both* the success probability of the prover *and* the runtime of the sub-extractor after the reprogramming may be skewed now. Again, we deal with this by considering an abstract sampling game that mimics the behavior of the extractor, but where the cost function is now more fine-grained in order to distinguish between a single run of the prover and a run of the sub-extractor. Because of this more fine-grained way of defining the "cost", the analysis of the game also becomes substantially more intricate.

1.2.2. *Negative Result*

We also show that the general exponential security loss of the Fiat–Shamir transformation, when applied to a multi-round protocol, is *not* an artefact of contrived examples, but there exist *natural* protocols that indeed have such an exponential loss. For instance, our negative result applies to the lattice-based protocols in [2,17]. Concretely, we show that the t-fold parallel repetition Π^t of a typical $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-move interactive proof Π features this behavior when $t \geq \mu$. For simplicity, let us assume that t and Q are multiples of μ. Then, in more detail, we show that for any typical $(k_1, \ldots, k_\mu)$-special-sound protocol Π there exists a poly-time Q-query prover $\mathcal{P}^*$ against $\mathsf{FS}[\Pi^t]$ that succeeds in making the verifier accept with probability $\approx \frac{1}{2} Q^\mu \kappa^t / \mu^{\mu+t}$ for *any* statement x, where κ is the knowledge error (as well as the soundness error) of Π. Thus, with the claimed probability, $\mathcal{P}^*$ succeeds in making the verifier accept for statements x that are not in the language and/or for which $\mathcal{P}^*$ does not know a witness. Given that κ^t is the soundness error of Π^t (i.e., the soundness error of Π^t as an interactive proof), this shows that the *soundness error* of Π^t grows proportionally with Q^μ when applying the Fiat–Shamir transformation. Recent work on the knowledge error of the parallel repetition of special-sound multi-round interactive proofs [5] shows that κ^t is also the

knowledge error of Π^t, and so the above shows that the same exponential loss holds in the *knowledge error* of the Fiat–Shamir transformation of a parallel repetition.

1.3. *Related Work*

1.3.1. *Independent Concurrent Work*

In independent and to a large extent concurrent work,[3] Wikström [37] achieves a similar positive result on the Fiat–Shamir transformation, using a different approach and different techniques: [37] reduces non-interactive extraction to a form of interactive extraction and then applies a generalized version of [36], while our construction adapts the interactive extractor from [2] and offers a direct analysis. One small difference in the results, which is mainly of theoretical interest, is that our result holds and is meaningful for *any* $Q < |\mathcal{C}|$, whereas [37] requires the challenge set $\mathcal{C}$ to be large.

1.3.2. *The Forking Lemma*

Security of the Fiat–Shamir transformation of k-special-sound 3-move protocols is widely used for construction of signatures. There, unforgeability is typically proven via a forking lemma [18,33], which extracts, with probability roughly ϵ^k/Q, a witness from a signature-forging adversary with success probability ϵ, where Q is the number of queries to the random oracle. The loss ϵ^k is due to *strict* polynomial time extraction (and can be decreased, but in general not down to ϵ). Such a k-th power loss in the success probability for a constant k is fine in certain settings, e.g., for proving the security of signature schemes; however, not for proofs of knowledge (which, on the other hand, consider *expected* polynomial time extraction [16]).

A previous version of [26] generalizes the original forking lemma [18,33] to accommodate Fiat–Shamir transformations of a larger class of (multi-round) interactive proofs. However, their forking lemma only targets a subclass of the $(k_1, \ldots, k_\mu)$-special-sound interactive proofs considered in this work. Moreover, in terms of (expected) runtime and success probability, our techniques significantly outperform their generalized forking lemma. For this reason, the latest version of [26] is based on our extraction techniques instead.

A forking lemma for *interactive* multi-round proofs was presented in [10], and its analysis was improved in a line of follow-up works [8,23,28,30,36]. This forking lemma shows that multi-round special-sound interactive proofs satisfy a notion of knowledge soundness called *witness extended emulation*. Eventually, it was shown that $(k_1, \ldots, k_\mu)$-special-soundness tightly implies knowledge soundness [2].

The aforementioned techniques for interactive proofs are not directly applicable to the Fiat–Shamir mode. First, incorporating the query complexity Q of a dishonest prover $\mathcal{P}^*$ attacking the non-interactive Fiat–Shamir transformation complicates the analysis. Second, a naive adaptation of the forking lemmas for interactive proofs gives a blow-up of Q^μ in the runtime.

[3] When finalizing our write-up, we were informed by Wikström that he derived similar results a few months earlier, subsequently made available online [37].

1.4. *Structure of the Paper*

Section 2 recalls essential preliminaries. In Sect. 3, the abstract sampling game is defined and analyzed. It is used in Sect. 4 to handle the Fiat–Shamir transformation of Σ-protocols. Building on the intuition, Sect. 5 introduces the *refined* game, and Sect. 6 uses it to handle multi-round protocols. Lastly, our negative result on parallel repetitions is presented in Sect. 7.

2. Preliminaries

2.1. *Interactive Proofs*

Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be a binary relation. Following standard conventions, we call $(x; w) \in R$ a statement-witness pair, that is, x is the *statement* and w is a *witness* for x. The set of valid witnesses for a statement x is denoted by $R(x)$, i.e., $R(x) = \{w : (x; w) \in R\}$. A statement that admits a witness is said to be a *true* or *valid* statement; the set of true statements is denoted by L_R, i.e., $L_R = \{x : \exists\, w \text{ s.t. } (x; w) \in R\}$. The relation R is an NP relation if the validity of a witness w can be verified in time polynomial in the size $|x|$ of the statement x. From now on, we assume all relations to be NP relations.

In an interactive proof for a relation R, a prover $\mathcal{P}$ aims to convince a verifier $\mathcal{V}$ that a statement x admits a witness, or even that the prover *knows* a witness $w \in R(x)$.

Definition 1. (*Interactive Proof*) An *interactive proof* $\Pi = (\mathcal{P}, \mathcal{V})$ for relation R is an interactive protocol between two probabilistic machines, a prover $\mathcal{P}$ and a polynomial time verifier $\mathcal{V}$. Both $\mathcal{P}$ and $\mathcal{V}$ take as public input a statement x, and additionally, $\mathcal{P}$ takes as private input a witness $w \in R(x)$. The verifier $\mathcal{V}$ either accepts or rejects and its output is denoted as $(\mathcal{P}(w), \mathcal{V})(x)$. Accordingly, we say the corresponding transcript (i.e., the set of all messages exchanged in the protocol execution) is *accepting* or *rejecting*.

Let us introduce some conventions and additional properties for interactive proof systems. We assume that the prover $\mathcal{P}$ sends the first and the last message in any interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$. Hence, the number of communication moves $2\mu + 1$ is always odd. We also say Π is a $(2\mu + 1)$-move protocol. We will refer to *multi-round* protocols as a way of emphasizing that we are not restricting to 3-move protocols.

Informally, an interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ is *complete* if for any statement-witness pair $(x; w) \in R$ the honest execution results in the verifier accepting with high probability. It is *sound* if the verifier rejects false statements, i.e., $x \notin L_R$, with high probability. We do neither require (or formally define) completeness nor soundness, as our main focus is *knowledge soundness*. Intuitively, a protocol is knowledge sound if any (potentially malicious) prover $\mathcal{P}^*$ which convinces the verifier must "know" a witness w such that $(x, w) \in R$. Informally, this means that any prover $\mathcal{P}^*$ with $\Pr((\mathcal{P}^*, \mathcal{V})(x) = \mathsf{accept})$ large enough is able to efficiently compute a witness $w \in R(x)$.

Definition 2. (*Knowledge Soundness*) An interactive proof $(\mathcal{P}, \mathcal{V})$ for relation R is *knowledge sound* with knowledge error $\kappa : \mathbb{N} \to [0, 1]$ if there exists a positive polynomial q and an algorithm $\mathcal{E}$, called a *knowledge extractor*, with the following properties.

Given input x and black-box oracle access to a (potentially dishonest) prover $\mathcal{P}^*$, the extractor $\mathcal{E}$ runs in an expected number of steps that is polynomial in $|x|$ (counting queries to $\mathcal{P}^*$ as a single step) and outputs a witness $w \in R(x)$ with probability

$$\Pr\left((x; \mathcal{E}^{\mathcal{P}^*}(x)) \in R\right) \geq \frac{\epsilon(\mathcal{P}^*, x) - \kappa(|x|)}{q(|x|)},$$

where $\epsilon(\mathcal{P}^*, x) := \Pr((\mathcal{P}^*, \mathcal{V})(x) = \mathsf{accept})$.

Remark 1. From the linearity of the expectation, it follows easily that it is sufficient to consider *deterministic* provers $\mathcal{P}^*$ in Definition 2.

An important class of protocols have particularly simple verifiers: effectively stateless verifiers which send uniformly random challenges to the prover, and run an efficient verification function on the final transcript.

Definition 3. (*Public-Coin*) An interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ is *public-coin* if all of $\mathcal{V}$'s random choices are made public. The message $c_i \leftarrow \mathcal{C}_i$ of $\mathcal{V}$ in the $2i$-th move is called the i-th *challenge*, and $\mathcal{C}_i$ is the *challenge set*. We assume every challenge set to be enumerated, i.e., encoded as $\{1, \ldots, |\mathcal{C}_i|\}$.

2.2. *Special-Sound Multi-Round Protocols*

The class of interactive proofs we are interested in are those where knowledge soundness follows from another property, namely *special-soundness*. Special-soundness is often simpler to verify, and many protocols satisfy this notion. Note that we require special-sound protocols to be public-coin.

Definition 4. (*k-out-of-N Special-Soundness*) Let $k, N \in \mathbb{N}$. A 3-move public-coin interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ for relation R, with challenge set of cardinality $N \geq k$, is k-out-of-N special-sound if there exists a polynomial time algorithm that, on input a statement x and k accepting transcripts $(a, c_1, z_1), \ldots (a, c_k, z_k)$ with common first message a and pairwise distinct challenges $c_1, \ldots, c_k$, outputs a witness $w \in R(x)$. We also say Π is k-special-sound and, if $k = 2$, it is simply said to be special-sound.

We refer to a 3-move public-coin interactive proof as a Σ- *protocol*. Note that often a Σ-protocol is required to be (perfectly) complete, special-sound and special honest-verifier zero-knowledge (SHVZK) by definition. We do not require a Σ-protocol to have these additional properties.

Definition 5. (Σ-*Protocol*) A Σ-protocol is a 3-move public-coin interactive proof.

In order to generalize k-special-soundness to multi-round protocols, we introduce the notion of a tree of transcripts. We follow the definition of [2].

Definition 6. (*Tree of Transcripts*) Let $k_1, \ldots, k_\mu \in \mathbb{N}$. A $(k_1, \ldots, k_\mu)$-tree of transcripts for a $(2\mu + 1)$-move public-coin interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ is a set of

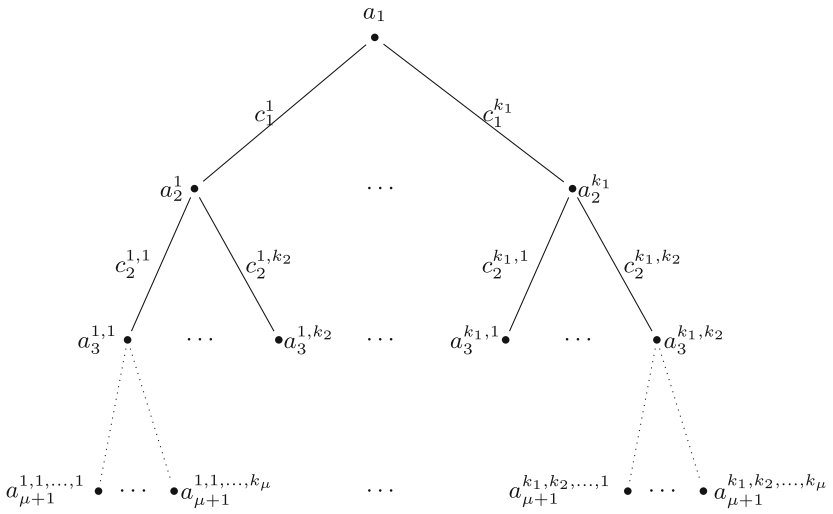

Fig. 1. $(k_1, \ldots, k_\mu)$-tree of transcripts of a $(2\mu + 1)$-move interactive proof [2].

$K = \prod_{i=1}^{\mu} k_i$ transcripts arranged in the following tree structure. The nodes in this tree correspond to the prover's messages and the edges to the verifier's challenges. Every node at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. Every transcript corresponds to exactly one path from the root node to a leaf node. See Fig. 1 for a graphical illustration. We refer to the corresponding tree of challenges as a $(k_1, \ldots, k_\mu)$-tree of challenges.

We will also write $\mathbf{k} = (k_1, \ldots, k_\mu) \in \mathbb{N}^\mu$ and refer to a $\mathbf{k}$-tree of transcripts or a $\mathbf{k}$-tree of challenges.

Definition 7. ($(k_1, \ldots, k_\mu)$-*out-of*-$(N_1, \ldots, N_\mu)$ *Special-Soundness*) Let $k_1, \ldots, k_\mu, N_1, \ldots, N_\mu \in \mathbb{N}$. A $(2\mu + 1)$-move public-coin interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ for relation R, where $\mathcal{V}$ samples the i-th challenge from a set of cardinality $N_i \geq k_i$ for $1 \leq i \leq \mu$, is $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-sound if there exists a polynomial time algorithm that, on input a statement x and a $(k_1, \ldots, k_\mu)$-tree of accepting transcripts outputs a witness $w \in R(x)$. We also say Π is $(k_1, \ldots, k_\mu)$-special-sound.

It is well known that, for 3-move protocols, k-special-soundness implies knowledge soundness, but only recently it was shown that more generally, for public-coin $(2\mu + 1)$-move protocols, $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-soundness tightly implies knowledge soundness [2], with knowledge error

$$\mathrm{Er}(k_1, \ldots, k_\mu; N_1, \ldots, N_\mu) = 1 - \prod_{i=1}^{\mu} \frac{N_i - k_i + 1}{N_i} = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right), \quad (1)$$

which matches the probability that at least one of the random challenges c_i hits a certain set Γ_i of size $k_i - 1$. Since typical protocols admit a trivial attack that succeeds if at least one of the random challenges c_i hits a certain set Γ_i of size $k_i - 1$ (we capture this by the special- *un* soundness property in Sect. 7), the soundness/knowledge error Er is tight for general special-sound protocols.

Note that $\mathrm{Er}(k; N) = (k-1)/N$ and, for all $1 \leq m \leq \mu$,

$$\begin{aligned}&\mathrm{Er}(k_m, \ldots, k_\mu; N_m, \ldots, N_\mu)\\ &\quad = 1 - \frac{N_m - k_m + 1}{N_m}\big(1 - \mathrm{Er}(k_{m+1}, \ldots, k_\mu; N_{m+1}, \ldots, N_\mu)\big),\end{aligned} \tag{2}$$

where we define $\mathrm{Er}(\emptyset; \emptyset) = 1$. If $N_1 = \cdots = N_\mu = N$, i.e., if the verifier samples all μ challenges from a set of size N, we simply write $\mathrm{Er}(k_1, \ldots, k_\mu; N)$, or $\mathrm{Er}(\mathbf{k}; N)$ for $\mathbf{k} = (k_1, \ldots, k_\mu)$.

2.3. *Non-Interactive Random Oracle Proofs (NIROP)*

In practice, interactive proofs are not typically used. Instead, transformations are used which turn them into non-interactive proofs in the random oracle model (ROM). We define non-interactive random oracle proofs (NIROP) as in [12]. Their definition is a straightforward adaption of (non-)interactive proof systems to the ROM. The same holds for their properties. Every algorithm is augmented by access to a random oracle.

In the *random oracle model*, algorithms have black-box access to an oracle $\mathsf{RO}\colon \{0,1\}^* \to \mathcal{Y}$, called the *random oracle*, which is instantiated by a uniformly random function with domain $\{0,1\}^*$ and codomain $\mathcal{Y}$. For convenience, we let the codomain $\mathcal{Y}$ be an arbitrary finite set, while typically $\mathcal{Y} = \{0,1\}^\eta$ for some $\eta \in \mathbb{N}$ related to the security parameter. Equivalently, RO is instantiated by lazy sampling, i.e., for every bitstring $x \in \{0,1\}^*$, $\mathsf{RO}(x)$ is chosen uniformly at random in $\mathcal{Y}$ (and then fixed). To avoid technical difficulties, we limit the domain from $\{0,1\}^*$ to $\{0,1\}^{\leq u}$, the finite set of all bitstrings of length at most u, for a sufficiently large $u \in \mathbb{N}$. An algorithm $\mathcal{A}^{\mathsf{RO}}$ that is given black-box access to a random oracle is called a *random-oracle algorithm*. We call $\mathcal{A}$ a *Q-query random-oracle algorithm*, if it makes at most Q queries to RO (independent of RO).

A natural extension of the random oracle model is when $\mathcal{A}$ is given access to *multiple independent* random oracles $\mathsf{RO}_1, \ldots, \mathsf{RO}_\mu$, possibly with different codomains.[4] The definitions below apply to this extension in the obvious way.

Definition 8. (*Non-Interactive Random Oracle Proof (NIROP)*) A *non-interactive random oracle proof* for relation R is a pair $(\mathcal{P}, \mathcal{V})$ of (probabilistic) random-oracle algorithms, a prover $\mathcal{P}$ and a polynomial-time verifier $\mathcal{V}$, such that: Given $(x; w) \in R$ and access to a random oracle RO, the prover $\mathcal{P}^{\mathsf{RO}}(x; w)$ outputs a proof π. Given $x \in \{0,1\}^*$, a purported proof π, and access to a random oracle RO, the verifier $\mathcal{V}^{\mathsf{RO}}(x, \pi)$ outputs 0 to reject or 1 to accept the proof.

[4] In practice, these random oracles will be instantiated by one random oracle $\mathsf{RO} : \{0,1\}^* \to \{0,1\}^\eta$ using standard techniques for domain separation and for sampling random elements from non-binary sets.

As for interactive definitions, a NIROP is complete if honestly generated proofs for $(x; w) \in R$ are accepted by $\mathcal{V}$ with high probability. It is sound if it is infeasible to produce an accepting proof for a false statement. In the non-interactive setting, the soundness error, i.e., the success probability of a cheating prover necessarily depends on the number of queries it is allowed to make to the random oracle. The same holds true for knowledge soundness of NIROPs.

Definition 9. (*Knowledge Soundness—NIROP*) A non-interactive random oracle proof $(\mathcal{P}, \mathcal{V})$ for relation R is *knowledge sound* with *knowledge error* $\kappa : \mathbb{N} \times \mathbb{N} \to [0, 1]$ if there exists a positive polynomial q and an algorithm $\mathcal{E}$, called a *knowledge extractor*, with the following properties: The extractor, given input x and oracle access to any (potentially dishonest) Q-query random oracle prover $\mathcal{P}^*$, runs in an expected number of steps that is polynomial in $|x|$ and Q and outputs a witness $w \in R(x)$, and satisfies

$$\Pr\big((x; w) \in R : w \leftarrow \mathcal{E}^{\mathcal{P}^*}(x)\big) \geq \frac{\epsilon(\mathcal{P}^*, x) - \kappa(|x|, Q)}{q(|x|)}$$

for all $x \in \{0, 1\}^*$ where $\epsilon(\mathcal{P}^*, x) = \Pr\big(\mathcal{V}^{\mathsf{RO}}(x, \mathcal{P}^{*,\mathsf{RO}}) = 1\big)$. Here, $\mathcal{E}$ implements RO for $\mathcal{P}^*$, in particular, $\mathcal{E}$ can arbitrarily program RO. Moreover, the randomness is over the randomness of $\mathcal{E}$, $\mathcal{V}$, $\mathcal{P}^*$ and RO.

Remark 2. As for the knowledge soundness of *interactive* proofs (see Remark 1), it is sufficient to consider *deterministic* provers $\mathcal{P}^*$ in Definition 9. Consequently, we will assume all dishonest provers $\mathcal{P}^*$ to be deterministic in order to simplify our analysis. Black-box access to $\mathcal{P}^*$ then simply means black-box access to the next-message function of $\mathcal{P}^*$. This in particular means that $\mathcal{E}$ can "rewind" $\mathcal{P}^*$ to any state. We stress though that $\mathcal{E}$ cannot depend on (or "know") certain properties of $\mathcal{P}^*$, such as Q or the success probability $\epsilon(\mathcal{P}^*, x)$.

Remark 3. The knowledge soundness definition for non-interactive random oracle proofs does not impose any (computational) restrictions on the prover $\mathcal{P}^*$ attacking the proof, i.e., the knowledge extractor may also be given oracle access to an unbounded prover $\mathcal{P}^*$. Therefore, it makes sense to refer to a NIROP admitting such a knowledge extractor as a *proof* of knowledge, rather than an *argument* of knowledge. However, at the same time, both the extractor's runtime and the knowledge error are allowed to depend on the query complexity Q. In fact, the knowledge error $\kappa(|x|, Q)$ typically converges to 1 as $Q \to \infty$. Therefore, in practice, one must bound the query complexity Q of a dishonest prover to derive nontrivial knowledge soundness properties. For this reason, a NIROP is sometimes also referred to as a *non-interactive random oracle argument*, even if $\mathcal{P}^*$ is allowed to be inefficient.

2.4. *(Non)-Interactive* Arguments

Interactive and non-interactive proofs for which (knowledge) soundness only holds with respect to computationally bounded provers $\mathcal{P}^*$ attacking the protocol are referred to as *arguments* (of knowledge). The computational soundness analysis of (non)-interactive

arguments can be significantly more complicated than the unconditional soundness analysis of proofs. For instance, t-fold parallel repetition is relatively easily seen to reduce the soundness error of an interactive proof from σ down to σ^t, while the same result does not hold for arguments [15].

However, when considering knowledge soundness, arguments for a relation R can typically be cast as proofs, with unconditional knowledge soundness, but now for a slightly different relation; namely for the relation R' such that $(x; w) \in R'$ if and only if $(x; w) \in R$ or w is the solution to some computational problem (depending on the (non)-interactive argument, e.g., two different openings of a commitment, or a collision in a hash function). In particular, computationally special-sound protocols (with respect to relation R) are typically unconditionally special-sound with respect to R'. Our results (and prior works) show that the unconditional special-soundness for R' implies unconditional knowledge soundness for R' (i.e., the extractor outputs a witness for the original relation R or it solves a computational problem), and thus *computational* knowledge soundness with respect to the original relation R. For this reason, our focus lies on the analysis of proofs, with the understanding that our results also apply to arguments.

Remark 4. The reason why the above does not work when considering ordinary soundness is that every statement x admits a witness with respect to relation R'; a solution to the computational problem is a witness for any x. Hence, since there do not exist statements outside the language $L_{R'} = \{0, 1\}^*$, any (non)-interactive proof is sound with respect to relation R'. In other words, the above reduction, in which an argument for relation R is cast as a proof for relation R', is only useful when considering knowledge soundness instead of ordinary soundness.

Example 1. (Bulletproofs) A typical computational problem, arising in the knowledge soundness analysis of (non)-interactive *arguments*, is breaking the binding property of some underlying commitment scheme. More precisely, in many (non)-interactive arguments the knowledge extractor either extracts a witness or it finds two distinct openings to the same commitment. For instance, Bulletproofs [9,10] use the Pedersen vector commitment scheme, which is computationally binding assuming the hardness of finding non-trivial discrete logarithm relations. Hence, Bulletproofs are arguments of knowledge with respect to some relation R, but proofs of knowledge with respect to R', where a witness $w \in R'(x)$ is either a witness $w \in R(x)$ or a non-trivial discrete logarithm relation. This observation shows that our techniques also apply to Bulletproof-like protocols.

2.5. *Adaptive Security*

Thus far, knowledge soundness has been defined with respect to *static* or *non-adaptive* provers $\mathcal{P}^*$ attacking the considered (non-)interactive proof for a *fixed* statement x. However, in many practical scenarios the dishonest provers are free to *choose* the statement x adaptively. Hence, in these cases static security is not sufficient. For *interactive* proofs, it is well-known that static knowledge soundness implies adaptive knowledge soundness. However, this does not carry over to non-interactive proofs. For instance, it is easy to see

that the static Fiat–Shamir transformation (see Definition 11) is in general not adaptively sound.

For this reason, let us formalize adaptive knowledge soundness for non-interactive random oracle proofs. An adaptive prover $\mathcal{P}^{\mathsf{a}}$ attacking the considered NIROP is given oracle access to a random oracle RO and outputs a statement x of fixed length $|x| = n$ together with a proof π. As in the static definition, adaptive knowledge soundness requires the existence of a knowledge extractor. However, formalizing the requirements of this extractor introduces some subtle issues. Namely, because $\mathcal{P}^{\mathsf{a}}$ chooses the statement x adaptively, it is not immediately clear for which statement the extractor should extract a witness. For instance, granting the extractor the same freedom of adaptively choosing the statement x, for which it needs to extract a witness w, renders knowledge extraction trivial; the extractor could simply output an arbitrary statement-witness pair $(x; w)$. For this reason, we require the extractor to output statement-witness pairs $(x; w)$ corresponding to the *valid* pairs (x, π) output by the adaptive prover $\mathcal{P}^{\mathsf{a}}$. To formalize these requirements, we also write (x, π, v), with $v \in \{0, 1\}$ indicating whether π is a valid proof for statement x. Given this notation, the extractor should output a triple (x, π, v) with the same distribution as the triples (x, π, v) produced by $\mathcal{P}^{\mathsf{a}}$; furthermore, if π is a valid proof for statement x, i.e., $v = 1$, then the extractor should additionally aim to output a witness $w \in R(x)$. As before, the success probability of the extractor is allowed to depend on the success probability of $\mathcal{P}^{\mathsf{a}}$. Finally, to ensure that the knowledge extractor can be used in compositional settings, where the NIROP is deployed as a component of a larger protocol, the prover $\mathcal{P}^{\mathsf{a}}$ is also allowed to additionally output arbitrary auxiliary information $\mathsf{aux} \in \{0, 1\}^*$ and the extractor is then required to simulate the tuple $(x, \pi, \mathsf{aux}, v)$, rather than the triple (x, π, v). The following definition formalizes adaptive knowledge soundness along these lines. For alternative definitions see, e.g., [22,34].

Definition 10. (*Adaptive Knowledge Soundness—NIROP*) A non-interactive random oracle proof $(\mathcal{P}, \mathcal{V})$ for relation R is *adaptively knowledge sound* with *knowledge error* $\kappa : \mathbb{N} \times \mathbb{N} \to [0, 1]$ if there exists a positive polynomial q and an algorithm $\mathcal{E}$, called a *knowledge extractor*, with the following properties: The extractor, given input $n \in \mathbb{N}$ and oracle access to any adaptive Q-query random oracle prover $\mathcal{P}^{\mathsf{a}}$ that outputs statements x with $|x| = n$, runs in an expected number of steps that is polynomial in n and Q and outputs a tuple $(x, \pi, \mathsf{aux}, v; w)$ such that $\{(x, \pi, \mathsf{aux}, v) : (x, \pi, \mathsf{aux}) \leftarrow \mathcal{P}^{\mathsf{a},\mathsf{RO}} \wedge v \leftarrow \mathcal{V}^{\mathsf{RO}}(x, \pi)\}$ and $\{(x, \pi, \mathsf{aux}, v) : (x, \pi, \mathsf{aux}, v; w) \leftarrow \mathcal{E}^{\mathcal{P}^{\mathsf{a}}}(n)\}$ are identically distributed and

$$\Pr\big(v = \mathsf{accept} \wedge (x; w) \in R : (x, \pi, \mathsf{aux}, v; w) \leftarrow \mathcal{E}^{\mathcal{P}^{\mathsf{a}}}(n)\big) \geq \frac{\epsilon(\mathcal{P}^{\mathsf{a}}) - \kappa(n, Q)}{q(n)},$$

where $\epsilon(\mathcal{P}^{\mathsf{a}}) = \Pr\big(\mathcal{V}^{\mathsf{RO}}(x, \pi) = 1 : (x, \pi) \leftarrow \mathcal{P}^{\mathsf{a},\mathsf{RO}}\big)$. Here, $\mathcal{E}$ implements RO for $\mathcal{P}^{\mathsf{a}}$, in particular, $\mathcal{E}$ can arbitrarily program RO. Moreover, the randomness is over the randomness of $\mathcal{E}$, $\mathcal{V}$, $\mathcal{P}^{\mathsf{a}}$ and RO.

Remark 5. We note that, while the tuple $(x, \pi, \mathsf{aux}, v)$ is required to have the same distribution for $\mathcal{P}^{\mathsf{a}}$ and $\mathcal{E}(n)$, by default the respective executions of $\mathcal{P}^{\mathsf{a}}$ and $\mathcal{E}(n)$ give rise

to two different probability spaces. Looking ahead though, we remark that the extractor that we eventually construct first does an honest run of $\mathcal{P}^{\mathsf{a}}$ by faithfully simulating the answers to $\mathcal{P}^{\mathsf{a}}$'s random oracle queries (this produces the tuple $(x, \pi, \mathsf{aux}, v)$ that $\mathcal{E}(n)$ eventually outputs and which so trivially has the right distribution), and then, if π is a valid proof, $\mathcal{E}(n)$ starts rewinding $\mathcal{P}^{\mathsf{a}}$ and reprogramming the random oracle to try to find enough valid proofs to compute a witness. Thus, in this sense, we can then say that $\mathcal{E}(n)$ aims to find a witness $w \in R(x)$ for *the statement x output by $\mathcal{P}^{\mathsf{a}}$*.

2.6. *Fiat–Shamir Transformations*

The Fiat–Shamir transformation [24] turns a public-coin interactive proof into a non-interactive random oracle proof (NIROP). The general idea is to compute the i-th challenge c_i as a hash of the i-th prover message a_i and (some part of) the previous communication transcript. For a Σ-protocol, the challenge c is computed as $c = H(a)$ or as $c = H(x, a)$, where the former is sufficient for *static* security, where the statement x is given as input to the dishonest prover, and the latter is necessary for *adaptive* security, where the dishonest prover can choose the statement x for which it wants to forge a proof.

For multi-round public-coin interactive proofs, there is some degree of freedom in the computation of the i-th challenge. For concreteness and simplicity, we consider a particular version where all previous prover messages are hashed along with the current message. Our techniques also apply to some other variants of the Fiat–Shamir transformation (see below), but one has to be careful, e.g., hashing only the current message is known to be not sufficient for multi-round protocols. As for Σ-protocols, we consider a static and an adaptive variant of this version of the Fiat–Shamir transformation. In contrast to the static variant, the adaptive Fiat–Shamir transformation includes the statement x in all hash function evaluations. If it is not made explicit which variant is used, the considered result holds for both variants.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a $(2\mu+1)$-move public-coin interactive proof, where the challenge from the i-th round is sampled from set $\mathcal{C}_i$. For simplicity, we consider μ random oracles $\mathsf{RO}_i : \{0,1\}^{\leq u} \to \mathcal{C}_i$ that map into the respective challenge spaces.

Definition 11. (*Fiat–Shamir Transformation*) The static Fiat–Shamir transformation $\mathsf{FS}[\Pi] = (\mathcal{P}_{\mathsf{fs}}, \mathcal{V}_{\mathsf{fs}})$ is the NIROP where $\mathcal{P}_{\mathsf{fs}}^{\mathsf{RO}_1,\ldots,\mathsf{RO}_\mu}(x; w)$ runs $\mathcal{P}(x; w)$ but instead of asking the verifier for the challenge c_i on message a_i, the challenges are computed as

$$c_i = \mathsf{RO}_i(a_1, \ldots, a_{i-1}, a_i)\,; \tag{3}$$

the output is then the proof $\pi = (a_1, \ldots, a_{\mu+1})$. On input a statement x and a proof $\pi = (a_1, \ldots, a_{\mu+1})$, $\mathcal{V}_{\mathsf{fs}}^{\mathsf{RO}_1,\ldots,\mathsf{RO}_\mu}(x, \pi)$ accepts if, for c_i as above $\mathcal{V}$ accepts the transcript $(a_1, c_1, \ldots, a_\mu, c_\mu, a_{\mu+1})$ on input x.

If the challenges are computed as

$$c_i = \mathsf{RO}_i(x, a_1, \ldots, a_{i-1}, a_i)\,; \tag{4}$$

the resulting NIROP is referred to as the *adaptive* Fiat–Shamir transformation.

By means of reducing the security of other variants of the Fiat–Shamir transformation to Definition 11, appropriately adjusted versions of our results also apply to other variants of doing the "chaining" (Eqs. 3 and 4) in the Fiat–Shamir transformation, for instance when c_i is computed as $c_i = \mathsf{RO}_i(i, c_{i-1}, a_i)$ or $c_i = \mathsf{RO}_i(x, i, c_{i-1}, a_i)$, where c_0 is the empty string.

2.7. *Negative Hypergeometric Distribution*

An important tool in our analysis is the negative hypergeometric distribution. Consider a bucket containing ℓ green balls and $N-\ell$ red balls, i.e., a total of N balls. In the negative hypergeometric experiment, balls are drawn uniformly at random from this bucket, without replacement, until k green balls have been found or until the bucket is empty. The number of red balls X drawn in this experiment is said to have a *negative hypergeometric distribution* with parameters N, ℓ, k, which is denoted by $X \sim \mathsf{NHG}(N, \ell, k)$.

Lemma 1. (Negative Hypergeometric Distribution) *Let $N, \ell, k \in \mathbb{N}$ with $\ell, k \leq N$, and let $X \sim \mathsf{NHG}(N, \ell, k)$. Then $\mathbb{E}[X] \leq k\frac{N-\ell}{\ell+1}$.*

Proof. If $\ell < k$, it clearly holds that $\Pr(X = N - \ell) = 1$. Hence, in this case, $\mathbb{E}[X] = N - \ell \leq k\frac{N-\ell}{\ell+1}$, which proves the claim.

So, let us now consider the case $\ell \geq k$. Then, for all $0 \leq x \leq N - \ell$,

$$\Pr(X = x) = \frac{\binom{x+k-1}{x}\binom{N-x-k}{N-\ell-x}}{\binom{N}{N-\ell}}.$$

Hence,

$$\begin{aligned}\mathbb{E}[X] &= \sum_{x=0}^{N-\ell} \Pr(X = x) \cdot x = \sum_{x=1}^{N-\ell} x \frac{\binom{x+k-1}{x}\binom{N-x-k}{N-\ell-x}}{\binom{N}{N-\ell}} \\ &= k\frac{N-\ell}{\ell+1} \sum_{x=1}^{N-\ell} \frac{\frac{x}{k}\binom{x+k-1}{x}\binom{N-x-k}{N-\ell-x}}{\frac{N-\ell}{\ell+1}\binom{N}{N-\ell}} = k\frac{N-\ell}{\ell+1} \sum_{x=1}^{N-\ell} \frac{\binom{x+k-1}{x-1}\binom{N-x-k}{N-\ell-x}}{\binom{N}{N-\ell-1}} \\ &= k\frac{N-\ell}{\ell+1} \sum_{x=1}^{N-\ell} \Pr(Y = x - 1) = k\frac{N-\ell}{\ell+1},\end{aligned}$$

where $Y \sim \mathsf{NHG}(N, \ell + 1, k - 1)$. This completes the proof of the lemma. □

Remark 6. Typically, negative hypergeometric experiments are restricted to the non-trivial case $\ell \geq k$. For reasons to become clear later, we also allow parameter choices with $\ell < k$ resulting in a trivial negative hypergeometric experiment in which all balls are always drawn.

Remark 7. The above has a straightforward generalization to buckets with balls of more than 2 colors: say ℓ green balls and m_i balls of color i for $1 \leq i \leq M$. The experiment proceeds as before, i.e., drawing until either k green balls have been found or the bucket is empty. Let X_i be the number of balls of color i that are drawn in this experiment. Then $X_i \sim \mathsf{NHG}(\ell + m_i, \ell, k)$ for all i. To see this, simply run the generalized negative hypergeometric experiment without counting the balls that are neither green nor of color i.

3. An Abstract Sampling Game

Towards the goal of constructing and analyzing a knowledge extractor for the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of special-sound interactive proofs Π, we define and analyze an abstract sampling game. Given access to a deterministic Q-query prover $\mathcal{P}^*$, attacking the non-interactive random oracle proof $\mathsf{FS}[\Pi]$, our extractor will essentially play this abstract game in the case Π is a Σ-protocol, and it will play this game recursively in the general case of a multi-round protocol. The abstraction allows us to focus on the crucial properties of the extraction algorithm, without unnecessarily complicating the notation.

The game considers an arbitrary but fixed U-dimensional array M, where, for all $1 \leq j_1, \ldots, j_U \leq N$, the entry $M(j_1, \ldots, j_U) = (v, i)$ contains a bit $v \in \{0, 1\}$ and an index $i \in \{1, \ldots, U\}$. Think of the bit v indicating whether this entry is "good" or "bad", and the index i points to one of the U dimensions. The goal will be to find k "good" entries with the same index i, and with all of them lying in the 1-dimensional array $M(j_1, \ldots, j_{i-1}, \cdot, j_{i+1}, \ldots, j_U)$ for some $1 \leq j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_U \leq N$.

Looking ahead, considering the case of a Σ-protocol first, this game captures the task of our extractor to find k proofs that are valid and feature the same first message but have different hash values assigned to the first message. Thus, in our application, the sequence $j_1, \ldots, j_U$ specifies the function table of the random oracle

$$\mathsf{RO}\colon \{1, \ldots, U\} \to \{1, \ldots, N\}, \quad i \mapsto j_i$$

while the entry $M(j_1, \ldots, j_U) = (v, i)$ captures the relevant properties of the proof produced by the considered prover when interacting with that particular specification of the random oracle. Concretely, the bit v indicates whether the proof is valid, and the index i is the first message a of the proof. Replacing j_i by j_i' then means to reprogram the random oracle at the point $i = a$. Note that after the reprogramming, we want to obtain another valid proof with the *same* first message, i.e., with the same index i (but now a different challenge, due to the reprogramming).

The game is formally defined in Fig. 2, and its core properties are summarized in Lemma 2. Looking ahead, we note that for efficiency reasons, the extractor will naturally not sample the entire sequence $j_1, \ldots, j_U$ (i.e., function table), but will sample its components on the fly using lazy sampling.

Parameters: $k, N, U \in \mathbb{N}$, and M a U-dimensional array with entries in $M(j_1, \ldots, j_U) \in \{0,1\} \times \{1, \ldots, U\}$ for all $1 \leq j_1, \ldots, j_U \leq N$.

- Sample $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$ uniformly at random and set $(v, i) = M(j_1, \ldots, j_U)$.
- If $v = 0$, abort.
- Else, repeat
  - sample $j' \in \{1, \ldots, N\} \setminus \{j_i\}$ (without replacement),
  - compute $(v', i') = M(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$,

  until either $k-1$ additional entries equal to $(1, i)$ have been found or until all indices j' have been tried.

Fig. 2. Abstract sampling game .

It will be useful to define, for all $1 \leq i \leq U$, the function

$$\begin{aligned} a_i &: \{1, \ldots, N\}^U \to \mathbb{N}_{\geq 0}, (j_1, \ldots, j_U) \\ &\mapsto \left| \left\{ j : M(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = (1, i) \right\} \right|. \end{aligned} \tag{5}$$

The value $a_i(j_1, \ldots, j_U)$ counts the number of entries that are "good" and have index i in the 1-dimensional array $M(j_1, \ldots, j_{i-1}, \cdot, j_{i+1}, \ldots, j_U)$. Note that a_i does not depend on the i-th entry of the input vector $(j_1, \ldots, j_U)$, and so, by a slight abuse of notation, we sometimes also write $a_i(j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_U)$.

Lemma 2. (Abstract Sampling Game) *Consider the game in Fig. 2. Let $J = (J_1, \ldots, J_U)$ be uniformly distributed in $\{1, \ldots, N\}^U$, indicating the first entry sampled, and let $(V, I) = M(J_1, \ldots, J_U)$. Further, for all $1 \leq i \leq U$, let $A_i = a_i(J)$. Moreover, let X be the number of entries of the form $(1, i)$ with $i = I$ sampled (including the first one), and let Λ be the total number of entries sampled in this game.*

Then

$$\begin{aligned} \mathbb{E}[\Lambda] &\leq 1 + (k-1)P \quad \textit{and} \\ \Pr(X = k) &\geq \frac{N}{N-k+1} \left(\Pr(V = 1) - P \cdot \frac{k-1}{N} \right), \end{aligned}$$

where $P = \sum_{i=1}^{U} \Pr(A_i > 0)$.

Remark 8. Note the abstractly defined parameter P. In our application, where the index i of $(v, i) = M(j_1, \ldots, j_U)$ is determined by the output of a prover making no more than Q queries to the random oracle with function table $j_1, \ldots, j_U$, the parameter P will be bounded by $Q + 1$. We show this formally (yet again somewhat abstractly) in Lemma 3. Intuitively, the reason is that the events $A_i > 0$ are *disjoint* for all but Q indices i (those that the considered prover does *not* query), and so their probabilities add up to at most 1.

Indeed, the output of a prover $\mathcal{P}^*$ attacking the protocol, while given oracle access to a random oracle with function table $j_1, \ldots, j_U$, that does not query index i is oblivious to the value of j_i. In particular, in this case, there exists a fixed i' such that $M(j'_1, \ldots, j'_U) \in \{(0, i'), (1, i')\}$ for all $j'_1, \ldots, j'_U$ with $j_\ell = j'_\ell$ for all indices ℓ queried by $\mathcal{P}^*$. Hence,

if $\mathcal{P}^*$ does not query i then $a_i(j_1, \ldots, j_U) > 0$ implies that $i = i'$. From this it follows that $P = \sum_{i=1}^{U} \Pr(A_i > 0) \leq Q + 1$. For a formal argument see the proof of Lemma 3.

Proof. (Proof (of Lemma 2)) **Expected Number of Samples.** Let us first derive an upper bound on the expected value of Λ. To this end, let X' denote the number of sampled entries of the form $(1, i)$ with $i = I$, but, in contrast to X, *without* counting the first one. Similarly, let Y' denote the number of sampled entries of the form (v, i) with $v = 0$ or $i \neq I$, again without counting the first one. Then $\Lambda = 1 + X' + Y'$ and

$$\Pr(X' = 0 \mid V = 0) = \Pr(Y' = 0 \mid V = 0) = 1.$$

Hence, $\mathbb{E}[X' \mid V = 0] = \mathbb{E}[Y' \mid V = 0] = 0$.

Let us now consider the expected value $\mathbb{E}[Y' \mid V = 1]$. To this end, we observe that, conditioned on the event $V = 1 \wedge I = i \wedge A_i = a$ with $a > 0$, Y' follows a negative hypergeometric distribution with parameters $N - 1$, $a - 1$ and $k - 1$. Hence, by Lemma 1,

$$\mathbb{E}[Y' \mid V = 1 \wedge I = i \wedge A_i = a] \leq (k-1)\frac{N-a}{a},$$

and thus, using that $\Pr(X' \leq k - 1 \mid V = 1) = 1$,

$$\mathbb{E}[X' + Y' \mid V = 1 \wedge I = i \wedge A_i = a] \leq (k-1) + (k-1)\frac{N-a}{a} = (k-1)\frac{N}{a}.$$

On the other hand

$$\Pr(V = 1 \wedge I = i \mid A_i = a) = \frac{a}{N}$$

and thus

$$\Pr(V = 1 \wedge I = i \wedge A_i = a) = \Pr(A_i = a)\frac{a}{N}. \tag{6}$$

Therefore, and since $\Pr(V = 1 \wedge I = i \wedge A_i = 0) = 0$,

$$\begin{aligned}
\Pr(V = 1)\,\mathbb{E}[X' + Y' \mid V = 1] &= \sum_{i=1}^{U}\sum_{a=1}^{N} \Pr(V = 1 \wedge I = i \wedge A_i = a) \\
&\qquad \mathbb{E}[X' + Y' \mid V = 1 \wedge I = i \wedge A_i = a] \\
&\leq \sum_{i=1}^{U}\sum_{a=1}^{N} \Pr(A_i = a)(k-1) \\
&= (k-1)\sum_{i=1}^{U} \Pr(A_i > 0) = (k-1)P,
\end{aligned}$$

where $P = \sum_{i=1}^{U} \Pr(A_i > 0)$. Hence,

$$\begin{aligned}\mathbb{E}[\Lambda] &= \mathbb{E}[1 + X' + Y'] \\ &= 1 + \Pr(V = 0) \cdot \mathbb{E}[X' + Y' \mid V = 0] + \Pr(V = 1) \cdot \mathbb{E}[X' + Y' \mid V = 1] \\ &\leq 1 + (k-1)P\,,\end{aligned}$$

which proves the claimed upper bound on $\mathbb{E}[\Lambda]$.

Success Probability. Let us now find a lower bound for the "success probability" $\Pr(X = k)$ of this game. Using (6) again, we can write

$$\Pr(X = k) = \sum_{i=1}^{U} \Pr(V = 1 \wedge I = i \wedge A_i \geq k) = \sum_{i=1}^{U} \sum_{a=k}^{N} \Pr(A_i = a) \frac{a}{N}\,.$$

Now, using $a \leq N$, note that

$$\begin{aligned}\frac{a}{N} &= 1 - \left(1 - \frac{a}{N}\right) \geq 1 - \frac{N}{N-k+1}\left(1 - \frac{a}{N}\right) \\ &= \frac{N}{N-k+1}\left(\frac{N-k+1}{N} - 1 + \frac{a}{N}\right) = \frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right).\end{aligned}$$

Therefore, combining the two, and using that the summand becomes negative for $a < k$ to argue the second inequality, and using (6) once more, we obtain

$$\begin{aligned}\Pr(X = k) &\geq \sum_{i=1}^{U} \sum_{a=k}^{N} \Pr(A_i = a) \frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right) \\ &\geq \sum_{i=1}^{U} \sum_{a=1}^{N} \Pr(A_i = a) \frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right) \\ &= \frac{N}{N-k+1} \sum_{i=1}^{U} \sum_{a=1}^{N} \left(\Pr(V = 1 \wedge I = i \wedge A_i = a) - \Pr(A_i = a) \cdot \frac{k-1}{N}\right) \\ &= \frac{N}{N-k+1}\left(\Pr(V = 1) - \frac{k-1}{N} \sum_{i=1}^{U} \Pr(A_i > 0)\right) \\ &= \frac{N}{N-k+1}\left(\Pr(V = 1) - P \cdot \frac{k-1}{N}\right),\end{aligned}$$

where, as before, we have used that $\Pr(V = 1 \wedge I = i \wedge A_i = 0) = 0$ for all $1 \leq i \leq U$ to conclude the second equality, and finally that $P = \sum_{i=1}^{U} \Pr(A_i > 0)$. This completes the proof of the lemma. □

Our knowledge extractor will instantiate the abstract sampling game via a deterministic Q-query prover $\mathcal{P}^*$ attacking the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$. The index i of

$M(v,i) = (j_1, \ldots, j_U)$ is then determined by the output of $\mathcal{P}^*$, with the random oracle being given by the function table $j_1, \ldots, j_U$. Since the index i is thus determined by Q queries to the random oracle, the following shows that the parameter P will in this case be bounded by $Q + 1$.

Lemma 3. *Consider the game in Fig. 2. Let v and be functions such that $M(j) = \big(v(j), \mathsf{idx}(j)\big)$ for all $j \in \{1, \ldots, N\}^U$. Furthermore, let $J = (J_1, \ldots, J_U)$ be uniformly distributed in $\{1, \ldots, N\}^U$, and set $A_i = a_i(J)$ for all $1 \leq i \leq U$. Let us additionally assume that for all $j \in \{1, \ldots, N\}^U$ there exists a subset $S(j) \subseteq \{1, \ldots, U\}$ of cardinality at most Q such that $(j) = (j')$ for all j' with $j'_\ell = j_\ell$ for all $\ell \in S(j)$. Then*

$$P = \sum_{i=1}^{U} \Pr(A_i > 0) \leq Q + 1.$$

Proof. By basic probability theory, it follows that[5]

$$\begin{aligned} P &= \sum_{i=1}^{U} \Pr(A_i > 0) = \sum_{j \in \{1,\ldots,N\}^U} \Pr(J = j) \sum_{i=1}^{U} \Pr(A_i > 0 \mid J = j) \\ &= \sum_{j} \Pr(J = j) \Bigg(\sum_{i \in S(j)} \Pr(A_i > 0 \mid J = j) + \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \Bigg) \end{aligned}$$

Since $|S(j)| \leq Q$ for all j, it follows that

$$\begin{aligned} P &\leq \sum_{j} \Pr(J = j) \Bigg(Q + \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \Bigg) \\ &\leq Q + \sum_{j} \Pr(J = j) \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \end{aligned}$$

Now note that, by definition of the sets $S(j)$, for all $j \in \{1, \ldots, N\}^U$, $i \notin S(j)$ and $j^* \in \{1, \ldots, N\}$, it holds that

$$\Pr\big(\mathsf{idx}(J_1, \ldots, J_{i-1}, j^*, J_{i+1}, \ldots, J_U) = \mathsf{idx}(j) \mid J = j\big) = 1.$$

Therefore, for all $i \notin S(j) \cup \{\mathsf{idx}(j)\}$,

$$\Pr(A_i > 0 \mid J = j) = 0.$$

[5] The probabilities $\Pr(A_i > 0 \mid J = j)$ are all 0 or 1; however, it's still convenient to use probability notation here.

Hence,

$$\sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \leq \Pr(A_{\mathsf{idx}(j)} > 0 \mid J = j) \leq 1.$$

Altogether, it follows that

$$P \leq Q + \sum_j \Pr(J = j) = Q + 1,$$

which completes the proof. □

4. Fiat–Shamir Transformation of Σ-Protocols

Let us first consider the Fiat–Shamir transformation of a k-special-sound Σ-protocol Π, i.e., a 3-move interactive proof, with challenge set $\mathcal{C}$; subsequently, in Sect. 6, we move to general *multi-round* interactive proofs.

Let $\mathcal{P}^*$ be a deterministic dishonest Q-query random-oracle prover, attacking the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of Π on input x. Given a statement x as input, after making Q queries to the random oracle $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$, $\mathcal{P}^*$ outputs a proof $\pi = (a, z)$. For reasons to become clear later, we re-format (and partly rename) the output and consider $I := a$ and π as $\mathcal{P}^*$'s output. We refer to the output I as the *index*. Furthermore, we extend $\mathcal{P}^*$ to an algorithm $\mathcal{A}$ that additionally checks the correctness of the proof π. Formally, $\mathcal{A}$ runs $\mathcal{P}^*$ to obtain I and π, queries RO to obtain $c := \mathsf{RO}(I)$, and then outputs

$$I = a, \quad y := (a, c, z) \qquad \text{and} \qquad v := V(y),$$

where $V(y) = 1$ if y is an accepting transcript for the interactive proof Π on input x and $V(y) = 0$ otherwise. Hence, $\mathcal{A}$ is a random-oracle algorithm making at most $Q + 1$ queries; indeed, it relays the oracle queries done by $\mathcal{P}^*$ and makes the one needed to do the verification. We may write $\mathcal{A}^{\mathsf{RO}}$ to make the dependency of $\mathcal{A}$'s output on the choice of the random oracle RO explicit. $\mathcal{A}$ has a naturally defined success probability

$$\epsilon(\mathcal{A}) := \Pr\big(v = 1 : (I, y, v) \leftarrow \mathcal{A}^{\mathsf{RO}}\big),$$

where $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$ is chosen uniformly at random. The probability $\epsilon(\mathcal{A})$ equals the success probability $\epsilon(\mathcal{P}^*, x)$ of the random-oracle prover $\mathcal{P}^*$ on input x.

Our goal is now to construct an extraction algorithm that, when given black-box access to $\mathcal{A}$, aims to output k accepting transcripts $y_1, \ldots, y_k$ with common first message a and distinct challenges. By the k-special-soundness property of Π, a witness for statement x can be computed efficiently from these transcripts.

The extractor $\mathcal{E}$ is defined in Fig. 3. We note that, after a successful first run of $\mathcal{A}$, having produced a first accepting transcript (a, c, z), we rerun $\mathcal{A}$ from the very beginning

Parameters: $k, Q \in \mathbb{N}$
Black-box access to: $\mathcal{A}$ as above

- Run $\mathcal{A}$ as follows to obtain (I, y_1, v): answer all (distinct) oracle queries with uniformly random values in $\mathcal{C}$. Let c be the response to query I.
- If $v = 0$, abort.
- Else, repeat
  - sample $c' \in \mathcal{C} \setminus \{c\}$ (without replacement);
  - run $\mathcal{A}$ as follows to obtain (I', y', v'): answer the query to I with c', while answering all other queries consistently if the query was performed by $\mathcal{A}$ already on a previous run and with a fresh random value in $\mathcal{C}$ otherwise;

  until either $k-1$ additional challenges c' with $v' = 1$ and $I' = I$ have been found or until all challenges $c' \in \mathcal{C} \setminus \{c\}$ have been tried.
- In the former case, output the k accepting transcripts $y_1, \ldots, y_k$.

Fig. 3. Extractor $\mathcal{E}$.

and answer all oracle queries consistently, except the query to a, i.e., we *only* reprogram the oracle at the point $I = a$. Note that since $\mathcal{P}^*$ and thus $\mathcal{A}$ is deterministic, and we only reprogram the oracle at the point $I = a$, in each iteration of the repeat loop $\mathcal{A}$ is ensured to make the query to I again.[6]

A crucial observation is the following. Within a run of $\mathcal{E}$, all the queries that are made by the different invocations of $\mathcal{A}$ are answered *consistently* using lazy sampling, except for the queries to the index I, where different responses $c, c', \ldots$ are given. This is indistinguishable from having them answered by a full-fledged random oracle, i.e., by means of a pre-chosen function $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$, but then replacing the output $\mathsf{RO}(I)$ at I by fresh challenges c' for the runs of $\mathcal{A}$ in the repeat loop. By enumerating the elements in the domain and codomain of RO, it is easily seen that the extractor is actually running the abstract game from Fig. 2. Thus, bounds on the success probability and the expected runtime (in terms of queries to $\mathcal{A}$) follow from Lemmas 2 and 3. Altogether, we obtain the following result.

Lemma 4. (Extractor) *The extractor $\mathcal{E}$ of Fig. 3 makes an expected number of at most $k + Q \cdot (k-1)$ queries to $\mathcal{A}$ and succeeds in outputting k transcripts $y_1, \ldots, y_k$ with common first message a and distinct challenges with probability at least*

$$\frac{N}{N-k+1}\left(\epsilon(\mathcal{A}) - (Q+1)\cdot\frac{k-1}{N}\right).$$

Proof. By enumerating all the elements in the domain and codomain of the random oracle RO, we may assume that $\mathsf{RO}\colon \{1, ..., U\} \to \{1, ..., N\}$, and thus RO can be represented by the function table $(j_1, ..., j_U) \in \{1, \ldots, N\}^U$ for which $\mathsf{RO}(i) = j_i$. Further, since $\mathcal{P}^*$ is deterministic, the outputs I, y and v of the algorithm $\mathcal{A}$ can be viewed as functions taking as input the function table $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$ of RO, and so we can consider the array $M(j_1, \ldots, j_U) = \big(I(j_1, \ldots, j_U), v(j_1, \ldots, j_U)\big)$.

[6] Of course, it would be sufficient to rewind $\mathcal{A}$ to the point where it makes the (first) query to a, but this would make the description more clumsy.

Then, a run of the extractor perfectly matches up with the abstract sampling game of Fig. 2 instantiated with array M. The only difference is that, in this sampling game, we consider full-fledged random oracles encoded by vectors $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$, while the actual extractor implements these random oracles by lazy sampling. Thus, we can apply Lemma 2 to obtain bounds on the success probability and the expected runtime. However, in order to control the parameter P, which occurs in the bound of Lemma 2, we make the following observation, so that we can apply Lemma 3 to bound $P \leq Q + 1$.

For every $(j_1, \ldots, j_U)$, let $S(j_1, \ldots, j_U) \subseteq \{1, \ldots, U\}$ be the set of points that $\mathcal{P}^*$ queries to the random oracle when $(j_1, \ldots, j_U)$ corresponds to the entire function table of the random oracle. Then, $\mathcal{P}^*$ will produce the same output when the random oracle is reprogrammed at an index $i \notin S(j_1, \ldots, j_U)$. In particular, $I(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = I(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$ for all j, j' and for all $i \notin S(j_1, \ldots, j_U)$. Furthermore, $|S(j_1, \ldots, j_U)| \leq Q$. Hence, the conditions of Lemma 3 are satisfied and $P \leq Q + 1$. The bounds on the success probability and the expected runtime now follow, completing the proof. □

The existence of the above extractor, combined with the k-special-soundness property, implies the following theorem. One subtle issue is that the sampling without replacement needs to be done efficiently, i.e., in expected polynomial time; this is not completely trivial, because in the worst case the extractor has to try all (possibly exponentially many) challenges. We discuss in Appendix A how this can be done.

Theorem 1. (Fiat–Shamir Transformation of a Σ-Protocol) *The Fiat–Shamir transformation* $\mathsf{FS}[\Pi]$ *of a k-out-of-N special-sound* Σ*-protocol* Π *is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q + 1) \cdot \kappa,$$

where $\kappa := \mathrm{Er}(k; N) = (k-1)/N$ *is the knowledge error of the (interactive)* Σ*-protocol* Π.

5. Refined Analysis of the Abstract Sampling Game

Before we prove knowledge soundness of the Fiat–Shamir transformation of *multi-round* interactive protocols, we reconsider the abstract game of Sect. 3 and consider a refined analysis of the cost of playing the game. The multi-round knowledge extractor will essentially play a recursive composition of this game; however, the analysis of Sect. 3 is insufficient for our purposes (resulting in a super-polynomial bound on the runtime of the knowledge extractor). Fortunately, it turns out that a refinement allows us to prove the required (polynomial) upper bound.

In Sect. 3, the considered cost measure is the number of entries visited during the game. For Σ-protocols, every entry corresponds to a single invocation of the dishonest prover $\mathcal{P}^*$. For multi-round protocols, every entry will correspond to a single invocation of a *sub-tree extractor*. The key observation is that some invocations of the sub-tree extractor

are *expensive* while others are *cheap*. For this reason, we introduce a cost function Γ and a constant cost γ to our abstract game, allowing us to differentiate between these two cases. Γ and γ assign a cost to every entry of the array M; Γ corresponds to the cost of an expensive invocation of the sub-tree extractor and γ corresponds to the cost of a cheap invocation. While this refinement presents a natural generalization of the abstract game of Sect. 3, its analysis becomes significantly more involved.

The following lemma provides an upper bound for the total cost of playing the abstract game in terms of these two cost functions.

Lemma 5. (Abstract Sampling Game—Weighted Version) *Consider again the game of Fig. 2, as well a cost function* $\Gamma\colon \{1,\dots,N\}^U \to \mathbb{R}_{\geq 0}$ *and a constant cost* $\gamma \in \mathbb{R}_{\geq 0}$.

Let $J = (J_1,\dots,J_U)$ *be uniformly distributed in* $\{1,\dots,N\}^U$, *indicating the first entry sampled, and let* $(V,I) = M(J_1,\dots,J_U)$. *Further, for all* $1 \leq i \leq U$, *let* $A_i = a_i(J)$, *where the function* a_i *is as defined in Eq. 5.*

We define the cost of sampling an entry $M(j_1,\dots,j_U) = (v,i)$ *with index* $i = I$ *to be* $\Gamma(j_1,\dots,j_U)$ *and the cost of sampling an entry* $M(j_1,\dots,j_U) = (v,i)$ *with index* $i \neq I$ *to be* γ. *Let* Δ *be the total cost of playing this game. Then,*

$$\mathbb{E}[\Delta] \leq k \cdot \mathbb{E}[\Gamma(J)] + (k-1)\cdot T \cdot \gamma$$

where $T = \sum_{i=1}^{U} \Pr(I \neq i \wedge A_i > 0) \leq P$.

Remark 9. Note that the parameter T in the statement here differs slightly from its counterpart $P = \sum_i \Pr(A_i > 0)$ in Lemma 2. Recall the informal discussion of P in the context of our application (Remark 8), where the array M is instantiated via a Q-query prover $\mathcal{P}^*$ attacking the Fiat–Shamir transformation of an interactive proof. We immediately see that now the defining events $I \neq i \wedge A_i > 0$ are *empty* for all $U - Q$ indices that the prover does not query, giving the bound $T \leq Q$ here, compared to the bound $P \leq Q + 1$ on P. The formal (and more abstract) statement and proof is given in Lemma 6.

Proof. Let us split up Δ into the cost measures Δ_1, Δ_2 and Δ_3, defined as follows. Δ_1 denotes the total costs of the elements $M(j_1,\dots,j_U) = (1,i)$ with $i = I$ sampled in the game, i.e., the elements with bit $v = 1$ and index $i = I$; correspondingly, X denotes the number of entries of the form $(1,i)$ with $i = I$ sampled (including the first one if $V = 1$). Second, Δ_2 denotes the total costs of the elements $M(j_1,\dots,j_U) = (0,i)$ with $i = I$ sampled, i.e., the elements with bit $v = 0$ and index $i = I$; correspondingly, Y denotes the number of entries of the form $(0,i)$ with $i = I$ sampled (including the first one if $V = 0$). Finally, Δ_3 denotes the total costs of the elements $M(j_1,\dots,j_U) = (v,i)$ with $i \neq I$ sampled; correspondingly, Z denotes the number of entries of this form sampled.

Clearly $\Delta = \Delta_1 + \Delta_2 + \Delta_3$. Moreover, since the cost γ is constant, it follows that $\mathbb{E}[\Delta_3] = \gamma \cdot \mathbb{E}[Z]$. In a similar manner, we now aim to relate $\mathbb{E}[\Delta_1]$ and $\mathbb{E}[\Delta_2]$ to $\mathbb{E}[Y]$ and $\mathbb{E}[Z]$, respectively. However, since the cost function $\Gamma\colon \{1,\dots,N\}^U \to \mathbb{R}_{\geq 0}$ is not necessarily constant, this is more involved.

For $1 \leq i \leq U$, let us write $J_i^* = (J_1,\dots,J_{i-1},J_{i+1},\dots,J_U)$, which is uniformly random with support $\{1,\dots,N\}^{U-1}$. Moreover, for all $1 \leq i \leq U$ and

$j^* = (j_1^*, \ldots, j_{i-1}^*, j_{i+1}^*, \ldots, j_U) \in \{1, \ldots, N\}^{U-1}$, let $\Lambda(i, j^*)$ denote the event

$$\Lambda(i, j^*) = [I = i \wedge J_i^* = j^*].$$

We note that conditioned on the event $\Lambda(i, j^*)$, all samples are picked from the subarray $M(j_1^*, \ldots, j_{i-1}^*, \cdot, j_{i+1}^*, \cdots, j_U^*)$; the first one uniformly at random subject to the index I being i, and the remaining ones (if $V = 1$) uniformly at random (without replacement).

We first analyze and bound $\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)]$. We observe that, for all i and j^* with $\Pr\big(\Lambda(i, j^*)\big) > 0$,

$$\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)] = \sum_{\ell=0}^{N} \Pr\big(X = \ell \mid \Lambda(i, j^*)\big) \cdot \mathbb{E}[\Delta_1 \mid \Lambda(i, j^*) \wedge X = \ell].$$

Since, conditioned on $\Lambda(i, j^*) \wedge X = \ell$ for $\ell \in \{0, \ldots, N\}$, any size-$\ell$ subset of elements with $v = 1$ and index i is equally likely to be sampled, it follows that

$$\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*) \wedge X = \ell] = \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \ell.$$

Hence,

$$\begin{aligned}\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)] &= \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \sum_{\ell} \Pr\big(X = \ell \mid \Lambda(i, j^*)\big) \cdot \ell \\ &= \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \mathbb{E}[X \mid \Lambda(i, j^*)]\,.\end{aligned}$$

Similarly,

$$\mathbb{E}[\Delta_2 \mid \Lambda(i, j^*)] = \mathbb{E}[\Gamma(J) \mid V = 0 \wedge \Lambda(i, j^*)] \cdot \mathbb{E}[Y \mid \Lambda(i, j^*)].$$

Next, we bound the expected values of X and Y conditioned on $\Lambda(i, j^*)$. The analysis is a more fine-grained version of the proof of Lemma 2. Bounding $\mathbb{E}[X \mid \Lambda(i, j^*)]$ is quite easy: since $V = 0$ implies $X = 0$ and $V = 1$ implies $X \leq k$, it immediately follows that

$$\begin{aligned}\mathbb{E}[X \mid \Lambda(i, j^*)] &= \Pr(V = 0 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[X \mid V = 0 \wedge \Lambda(i, j^*)] \\ &\quad + \Pr(V = 1 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[X \mid V = 1 \wedge \Lambda(i, j^*)] \\ &\leq \Pr(V = 1 \mid \Lambda(i, j^*)) \cdot k.\end{aligned}$$

Hence,

$$\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)] \leq k \cdot \Pr(V = 1 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)]. \quad (7)$$

Suitably bounding the expectation $\mathbb{E}[Y \mid \Lambda(i, j^*)]$, and thus $\mathbb{E}[\Delta_2 \mid \Lambda(i, j^*)]$,

is more involved. For that purpose, we introduce the following parameters. For the considered fixed choice of the index $1 \leq i \leq U$ and of $j^* = (j_1^*, \ldots, j_{i-1}^*, j_{i+1}^*, \cdots, j_U^*)$, we let,[7]

$$a := a_i(j^*) = \left|\left\{j : (v_j, i_j) = M(j_1^*, \ldots, j_{i-1}^*, j, j_{i+1}^*, \ldots, j_U^*) = (1, i)\right\}\right| \quad \text{and}$$
$$b := b_i(j^*) := \left|\left\{j : (v_j, i_j) = M(j_1^*, \ldots, j_{i-1}^*, j, j_{i+1}^*, \ldots, j_U^*) = (0, i)\right\}\right| .$$

Let us first note that

$$\Pr\big(V = 1 \mid \Lambda(i, j^*)\big) = \frac{a}{a+b} \quad \text{and} \quad \Pr\big(V = 0 \mid \Lambda(i, j^*)\big) = \frac{b}{a+b}$$

for all i and j^* with $\Pr\big(\Lambda(i, j^*)\big) > 0$. Therefore, if we condition on the event $V = 1 \wedge \Lambda(i, j^*)$ we implicitly assume that i and j^* are so that a is positive. Now, towards bounding $\mathbb{E}[Y \mid \Lambda(i, j^*)]$, we observe that conditioned on the event $V = 1 \wedge \Lambda(i, j^*)$, the random variable Y follows a negative hypergeometric distribution with parameters $a + b - 1$, $a - 1$ and $k - 1$ (see also Remark 7). Hence, by Lemma 1,

$$\mathbb{E}[Y \mid V = 1 \wedge \Lambda(i, j^*)] \leq (k-1)\frac{b}{a},$$

and thus

$$\begin{aligned}
\mathbb{E}[Y \mid \Lambda(i, j^*)] &= \Pr(V = 0 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[Y \mid V = 0 \wedge \Lambda(i, j^*)] \\
&\quad + \Pr(V = 1 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[Y \mid V = 1 \wedge \Lambda(i, j^*)] \\
&\leq \Pr\big(V = 0 \mid \Lambda(i, j^*)\big) + \Pr\big(V = 1 \mid \Lambda(i, j^*)\big) \cdot (k-1)\frac{b}{a} \\
&= \frac{b}{a+b} + \frac{a}{a+b} \cdot (k-1)\frac{b}{a} = k\frac{b}{a+b} \\
&= k \cdot \Pr(V = 0 \mid \Lambda(i, j^*)),
\end{aligned}$$

where we use that $\mathbb{E}[Y \mid V = 0 \wedge \Lambda(i, j^*)] = 1$. Hence,

$$\mathbb{E}[\Delta_2 \mid \Lambda(i, j^*)] \leq k \cdot \Pr(V = 0 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[\Gamma(J) \mid V = 0 \wedge \Lambda(i, j^*)],$$

and thus, combined with Eq. 7,

$$\mathbb{E}[\Delta_1 + \Delta_2 \mid \Lambda(i, j^*)] \leq k \cdot \mathbb{E}[\Gamma(J) \mid \Lambda(i, j^*)].$$

Since this inequality holds for all i and j^* with $\Pr\big(\Lambda(i, j^*)\big) > 0$, it follows that

$$\mathbb{E}[\Delta_1 + \Delta_2] \leq k \cdot \mathbb{E}[\Gamma(J)].$$

[7] Recall that we use $a_i(j_1, \ldots, j_U)$ and $a_i(j_1, \ldots, j_{i-1} j_{i+1}, \ldots, j_U)$ interchangeably, exploiting that $a_i(j_1, \ldots, j_U)$ does not depend on the i-th input j_i.

What remains is to show that $\mathbb{E}[Z] \leq (k-1)T$, and thus $\mathbb{E}[\Delta_3] = \gamma\,\mathbb{E}[Z] \leq (k-1)T\gamma$. The slightly weaker bound $\mathbb{E}[Z] \leq (k-1)P$ follows immediately from observing that $Z \leq Y'$ for Y' as in the proof of Lemma 2 (the number of entries counted by Z is a subset of those counted by Y'), and using that $\mathbb{E}[Y'] \leq \mathbb{E}[X'+Y'] \leq (k-1)P$ as derived in the proof of Lemma 2. In order to get the slightly better bound in terms of T, we bound $\mathbb{E}[Z]$ from scratch below. We use a similar approach as above for bounding the expectation of Y. Thus, we consider a fixed choice of i and j^* and set $a := a_i(j^*)$ and $b := b_i(j^*)$. Then, conditioned on $V = 1 \wedge \Lambda(i, j^*)$, also Z follows a negative hypergeometric distribution, but now with parameters $N-b-1$, $a-1$ and $k-1$. Therefore, for all i and j^* with $\Pr\big(V = 1 \wedge \Lambda(i, j^*)\big) > 0$,

$$\mathbb{E}[Z \mid V = 1 \wedge \Lambda(i, j^*)] \leq (k-1)\frac{N-a-b}{a}.$$

Using that $\mathbb{E}[Z \mid V = 0 \wedge \Lambda(i, j^*)] = 0$, but also recalling that $\Pr\big(V = 1 \mid \Lambda(i, j^*)\big) = a/(a+b)$ and exploiting $\Pr(I = i \mid J_i^* = j^*) = (a+b)/N$, it follows that

$$\begin{aligned}
\mathbb{E}[Z \mid \Lambda(i, j^*)] &= \Pr\big(V = 1 \mid \Lambda(i, j^*)\big) \cdot \mathbb{E}[Z \mid V = 1 \wedge \Lambda(i, j^*)] \\
&\leq \frac{a}{a+b} \cdot (k-1) \cdot \frac{N-a-b}{a} = (k-1) \cdot \frac{N-a-b}{a+b} \\
&= (k-1) \cdot \Big(\frac{1}{\Pr(I = i \mid J_i^* = j^*)} - 1\Big) \\
&= (k-1) \cdot \frac{\Pr(J_i^* = j^*) - \Pr(I = i \wedge J_i^* = j^*)}{\Pr(\Lambda(i, j^*))} \\
&= (k-1) \cdot \frac{\Pr(I \neq i \wedge J_i^* = j^*)}{\Pr(\Lambda(i, j^*))}.
\end{aligned}$$

We recall that the above holds for all i and j^* for which $a = a_i(j^*) > 0$, so that $\Pr(V = 1 \wedge \Lambda(i, j^*)) > 0$. For i and j^* with $a = a_i(j^*) = 0$, it holds that $\Lambda(i, j^*)$ implies $V = 0$, and thus $\mathbb{E}[Z \mid \Lambda(i, j^*)] = 0$. Therefore

$$\begin{aligned}
\mathbb{E}[Z] &= \sum_{i=1}^{U} \sum_{\substack{j^* \text{ s.t.} \\ a_i(j^*)>0}} \Pr[\Lambda(i, j^*)] \cdot \mathbb{E}[Z \mid \Lambda(i, j^*)] \\
&\leq (k-1) \cdot \sum_{i=1}^{U} \sum_{\substack{j^* \text{ s.t.} \\ a_i(j^*)>0}} \Pr(I \neq i \wedge J_i^* = j^*) \\
&\leq (k-1) \cdot \sum_{i=1}^{U} \Pr(I \neq i \wedge A_i > 0) = (k-1) \cdot T\,.
\end{aligned}$$

Hence $\mathbb{E}[\Delta_3] \leq (k-1) \cdot T \cdot \gamma$, as intended, and altogether it follows that

$$\mathbb{E}[\Delta] = \mathbb{E}[\Delta_1 + \Delta_2 + \Delta_3] \leq k \cdot \mathbb{E}[\Gamma(J)] + (k-1) \cdot T \cdot \gamma,$$

which completes the proof of the lemma. □

Lemma 6. *Consider the game in Fig. 2. Let v and idx be functions such that $M(j) = \big(v(j), \mathsf{idx}(j)\big)$ for all $j \in \{1, \dots, N\}^U$. Furthermore, let $J = (J_1, \dots, J_U)$ be uniformly distributed in $\{1, \dots, N\}^U$ and set $A_i = a_i(J)$ for all $1 \le i \le U$ as in Eq. 5. Let us additionally assume that for all $j \in \{1, \dots, N\}^U$ there exists a subset $S(j) \subseteq \{1, \dots, U\}$ of cardinality at most Q such that $\mathsf{idx}(j) = \mathsf{idx}(j')$ for all j, j' with $j_\ell = j'_\ell$ for all $\ell \in S(j)$. Then*

$$T = \sum_{i=1}^{U} \Pr\big(\mathsf{idx}(J) \neq i \wedge A_i > 0\big) \le Q.$$

Proof. The proof is analogous to the proof of Lemma 3. By basic probability theory, it follows that

$$\begin{aligned} T &= \sum_{i=1}^{U} \Pr\big(J) \neq i \wedge A_i > 0\big) \\ &= \sum_{j} \Pr(J = j) \Bigg(\sum_{i \in S(j)} \Pr\big(J) \neq i \wedge A_i > 0 \mid J = j\big) \\ &\qquad + \sum_{i \notin S(j)} \Pr\big(J) \neq i \wedge A_i > 0 \mid J = j\big) \Bigg) \\ &\le Q + \sum_{j} \Pr(J = j) \sum_{i \notin S(j)} \Pr\big(J) \neq i \wedge A_i > 0 \mid J = j\big), \end{aligned}$$

where the inequality follows from the fact that $|S(j)| \le Q$ for all j.

Now note that, by definition of the sets $S(j)$, for all $j \in \{1, \dots, N\}^U$, $i \notin S(j)$ and $j_i \in \{1, \dots, N\}$, it holds that

$$\Pr\big(J_1, \dots, J_{i-1}, j_i, J_{i+1}, \dots, J_U) = j) \mid J = j\big) = 1.$$

Therefore, for all $i \notin S(j) \cup \{j)\}$,

$$\Pr(A_i > 0 \mid J = j) = 0.$$

Hence,

$$\sum_{i \notin S(j)} \Pr\big(J) \neq i \wedge A_i > 0 \mid J = j\big) \le \Pr\big(J) \neq j) \wedge A_{j)} > 0 \mid J = j\big) = 0.$$

Altogether, it follows that

$$T \le Q + \sum_{j} \Pr(J = j) \sum_{i \notin S(j)} \Pr\big(J) \neq i \wedge A_i > 0 \mid J = j\big) = Q,$$

which completes the proof. □

6. Fiat–Shamir Transformation of Multi-Round Protocols

Let us now move to multi-round interactive proofs. More precisely, we consider the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of a $\mathbf{k}$-special-sound $(2\mu+1)$-move interactive proof Π, with $\mathbf{k} = (k_1, \ldots, k_\mu)$. While the multi-round extractor has a natural recursive construction, it requires a more fine-grained analysis to show that it indeed implies knowledge soundness.

To avoid a cumbersome notation, in Sect. 6.1 we first handle $(2\mu+1)$-move interactive proofs in which the verifier samples all μ challenges uniformly at random from the *same* set $\mathcal{C}$. Subsequently, in Sect. 6.2, we argue that our techniques have a straightforward generalization to interactive proofs where the verifier samples its challenges from different challenge sets. In Sect. 6.3, we also show that our results extend to *adaptive* security in a straightforward way.

6.1. *Multi-Round Protocols with a Single Challenge Set*

Consider a deterministic dishonest Q-query random-oracle prover $\mathcal{P}^*$, attacking the Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of a $\mathbf{k}$-special-sound interactive proof Π on input x. We assume all challenges to be elements in the same set $\mathcal{C}$. After making at most Q queries to the random oracle, $\mathcal{P}^*$ outputs a proof $\pi = (a_1, \ldots, a_{\mu+1})$. We re-format the output and consider

$$I_1 := a_1,\ I_2 := (a_1, a_2), \ldots,\ I_\mu := (a_1, \ldots, a_\mu) \quad \text{and} \quad \pi$$

as $\mathcal{P}^*$'s output. Sometimes it will be convenient to also consider $I_{\mu+1} := (a_1, \ldots, a_{\mu+1})$. Furthermore, we extend $\mathcal{P}^*$ to a random-oracle algorithm $\mathcal{A}$ that additionally checks the correctness of the proof π. Formally, relaying all the random oracle queries that $\mathcal{P}^*$ is making, $\mathcal{A}$ runs $\mathcal{P}^*$ to obtain $\mathbf{I} = (I_1, \ldots, I_\mu)$ and π, additionally queries the random oracle to obtain $c_1 := \mathsf{RO}(I_1), \ldots, c_\mu := \mathsf{RO}(I_\mu)$, and then outputs

$$\mathbf{I}, \quad y := (a_1, c_1, \ldots, a_\mu, c_\mu, a_{\mu+1}) \qquad \text{and} \qquad v := V(x, y),$$

where $V(x, y) = 1$ if y is an accepting transcript for the interactive proof Π on input x and $V(x, y) = 0$ otherwise. Hence, $\mathcal{A}$ makes at most $Q + \mu$ queries (the queries done by $\mathcal{P}^*$, and the queries to $I_1, \ldots, I_\mu$). Moreover, $\mathcal{A}$ has a naturally defined success probability

$$\epsilon(\mathcal{A}) := \Pr\big(v = 1 : (I, y, v) \leftarrow \mathcal{A}^{\mathsf{RO}}\big),$$

where $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$ is distributed uniformly. As before, $\epsilon(\mathcal{A}) = \epsilon(\mathcal{P}^*, x)$.

Our goal is now to construct an extraction algorithm that, when given black-box access to $\mathcal{A}$, and thus to $\mathcal{P}^*$, aims to output a $\mathbf{k}$-tree of accepting transcripts(Definition 6). By

Parameters: $k_m, Q \in \mathbb{N}$
Black-box access to: $\mathcal{E}_{m+1}$
Random oracle queries: $Q + \mu$

- Run $\mathcal{E}_{m+1}$ as follows to obtain $(\mathbf{I}, y_1, v)$: relay the $Q + \mu$ queries to the random oracle and record all query-response pairs. Let c be the response to query I_m.
- If $v = 0$, abort with output $v = 0$.
- Else, repeat
  - sample $c' \in \mathcal{C} \setminus \{c\}$ (without replacement);
  - run $\mathcal{E}_{m+1}$ as follows to obtain $(\mathbf{I}', y', v')$, aborting right after the initial run of $\mathcal{P}^*$ if $I'_m \neq I_m$: answer the query to I_m with c', while answering all other queries consistently if the query was performed by $\mathcal{E}_{m+1}$ already on a previous run and with a fresh random value in $\mathcal{C}$ otherwise;

  until either $k_m - 1$ additional challenges c' with $v' = 1$ and $I'_m = I_m$ have been found or until all challenges $c' \in \mathcal{C} \setminus \{c\}$ have been tried.
- In the former case, output $\mathbf{I}$, the k_m accepting $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-trees $y_1, \ldots, y_{k_m}$, and $v := 1$; in the latter case, output $v := 0$.

Fig. 4. Sub-extractor $\mathcal{E}_m$, as a $(Q + \mu)$-query random-oracle algorithm .

the **k**-special-soundness property of Π, a witness for statement x can then be computed efficiently from these transcripts.

To this end, we recursively introduce a sequence of "sub-extractors" $\mathcal{E}_1, \ldots, \mathcal{E}_\mu$, where $\mathcal{E}_m$ aims to find a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts. The main idea behind this recursion is that such a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts is the composition of k_m appropriate $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-trees.

For technical reasons, we define the sub-extractors $\mathcal{E}_m$ as *random-oracle* algorithms, each one making $Q+\mu$ queries to a random oracle. As we will see, the recursive definition of $\mathcal{E}_m$ is very much like the extractor from the 3-move case, but with $\mathcal{A}$ replaced by the sub-extractor $\mathcal{E}_{m+1}$; however, for this to work we need the sub-extractor to be the same kind of object as $\mathcal{A}$, thus a random-oracle algorithm making the same number of queries. As base for the recursion, we consider the algorithm $\mathcal{A}$ (which outputs a single transcript, i.e., a $(1, \ldots, 1)$-tree); thus, the sub-extractor $\mathcal{E}_\mu$ (which outputs a $(1, \ldots, 1, k_\mu)$-tree) is essentially the extractor of the 3-move case, but with $\mathcal{A}$ now outputting an index *vector* $\mathbf{I} = (I_1, \ldots, I_\mu)$, and with $\mathcal{E}_\mu$ being a *random-oracle* algorithm, so that we can recursively replace the random-oracle algorithm $\mathcal{A}$ by $\mathcal{E}_\mu$ to obtain $\mathcal{E}_{\mu-1}$, etc.

Formally, the recursive definition of $\mathcal{E}_m$ from $\mathcal{E}_{m+1}$ is given in Fig. 4, where $\mathcal{E}_{\mu+1}$ (the base case) is set to $\mathcal{E}_{\mu+1} := \mathcal{A}$, and where $\mathcal{E}_m$ exploits the following *early abort* feature of $\mathcal{E}_{m+1}$: like $\mathcal{A}$, the sub-extractor $\mathcal{E}_{m+1}$ computes the index vector it eventually outputs by running $\mathcal{P}^*$ *as its first step* (see Lemma 7). This allows the executions of $\mathcal{E}_{m+1}$ in the repeat loop in Fig. 4 to abort after a single run of $\mathcal{P}^*$ if the requirement $I'_m = I_m$ on its index vector $\mathbf{I}$ is not satisfied, without proceeding to produce the remaining parts y', v' of the output (which would invoke more calls to $\mathcal{P}^*$).

The actual extractor $\mathcal{E}$ is then given by a run of $\mathcal{E}_1$, with the $Q + \mu$ random-oracle queries made by $\mathcal{E}_1$ being answered using lazy-sampling.

Remark 10. Let us emphasize that within *one* run of $\mathcal{E}_m$, except for the query to I_m for which the response is "reprogrammed", all the queries made by the multiple runs of the sub-extractor $\mathcal{E}_{m+1}$ in the repeat loop are answered *consistently*, both with the run of $\mathcal{E}_{m+1}$ in the first step and among the runs in the repeat loop. This means, a query to a value ξ that has been answered by η in a previous run on $\mathcal{E}_{m+1}$ (within the considered run of $\mathcal{E}_m$) is again answered by η, and a query to a value ξ' that has not been queried yet

in a previous run on $\mathcal{E}_{m+1}$ (within the considered run of $\mathcal{E}_m$) is answered with a freshly chosen uniformly random $\eta' \in \mathcal{C}$. In *multiple* runs of $\mathcal{E}_m$, very naturally the random tape of $\mathcal{E}_m$ will be refreshed, and thus there is no guaranteed consistency among the answers to the query calls of $\mathcal{E}_{m+1}$ across multiple runs of $\mathcal{E}_m$.

The following lemma captures some technical property of the sub-extractors $\mathcal{E}_m$. Subsequently, Proposition 1 shows that $\mathcal{E}_m$, if successful, indeed outputs a $(1, \ldots, 1, k_m \ldots, k_\mu)$-tree of accepting transcripts. Proposition 2 bounds the success probability and expected runtime of $\mathcal{E}_m$. All statements are understood to hold for any statement x and any $m \in \{1, \ldots, \mu + 1\}$.

Lemma 7. (Consistency of $\mathcal{P}^*$ and $\mathcal{E}_m$) *$\mathcal{E}_m$ obtains the index vector $\mathbf{I}$, which it eventually outputs, by running $(\mathbf{I}, \pi) \leftarrow \mathcal{P}^*$ as its first step. In particular, for any fixed choice of the random oracle RO, the index vector $\mathbf{I}$ output by $\mathcal{E}_m^{\mathsf{RO}}$ matches the one output by $\mathcal{P}^{*,\mathsf{RO}}$.*

Proof. The first claim holds for $\mathcal{E}_{\mu+1} = \mathcal{A}$ by definition of $\mathcal{A}$, and it holds for $\mathcal{E}_m$ with $m \leq \mu$ by induction, given that $\mathcal{E}_m$ runs $\mathcal{E}_{m+1}$ as a first step. The claim on the matching index vectors then follows trivially. □

Proposition 1. (Correctness) *For any fixed choice of the random oracle let $(\mathbf{I}, y_1, \ldots, y_{k_m}, v) \leftarrow \mathcal{E}_m^{\mathsf{RO}}(x)$. If $v = 1$ then $(y_1, \ldots, y_{k_m})$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts.*

Proof. All $k_{m+1} \cdots k_\mu$ transcripts in a $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree contain the same partial transcript $(a_1, c_1, \ldots, c_m, a_{m+1})$, i.e., the first $2m - 1$ messages in all these transcripts coincide. Hence, any $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree of transcripts has a well-defined *trunk* $(a_1, c_1, \ldots, c_m, a_{m+1})$.

By induction on m, we will prove that if $v = 1$ then $(y_1, \ldots, y_{k_m})$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, \mathsf{RO}(I_{m-1}), a_m)$, where $I_{m+1} = (a_1, \ldots, a_{m+1})$. This obviously implies the correctness claim.

For the base case $m = \mu + 1$, recall that $\mathcal{E}_{\mu+1} = \mathcal{A}$, and that by definition of $\mathcal{A}$ and its output $(\mathbf{I}, y, v)$, if $v = 1$ then y is an accepting transcript, and thus a $(1, \ldots, 1)$-tree of accepting transcripts with $(a_1, \mathsf{RO}(I_1), \ldots, \mathsf{RO}(I_\mu), a_{\mu+1})$ as trunk where $I_{\mu+1} = (a_1, \ldots, a_{\mu+1})$, by definition of $\mathbf{I} = (I_1, \ldots, I_\mu)$.

For the induction step, by the induction hypothesis on $\mathcal{E}_{m+1}$ and its output $(\mathbf{I}, y, v)$, if $v = 1$ then y is a $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, a_m, \mathsf{RO}(I_m), a_{m+1})$, where $I_{m+1} = (a_1, \ldots, a_{m+1})$. This holds for $(\mathbf{I}, y_1, v)$ output by $\mathcal{E}_{m+1}$ in the first step of $\mathcal{E}_m$, but also for any invocation of $\mathcal{E}_{m+1}$ in the repeat loop with output $(\mathbf{I}', y', v')$, here with trunk $(a'_1, \mathsf{RO}'(I'_1), \ldots, a'_m, \mathsf{RO}'(I'_m), a'_{m+1})$, where $I'_{m+1} = (a'_1, \ldots, a'_{m+1})$ and RO' is such that $\mathsf{RO}'(I_j) = \mathsf{RO}(I_j)$ for all $j \neq m$, while $\mathsf{RO}(I_m) = c_i$ and $\mathsf{RO}'(I_m) = c'_i$. By definition of the output of $\mathcal{E}_m$, for y_1 and y' occurring in the output of $\mathcal{E}_m$, it is ensured that $I_m = I'_m$.

Now note that, by Lemma 7, for the purpose of the argument, $\mathcal{E}_m$ could have run $\mathcal{P}^*$ instead of $\mathcal{E}_{m+1}$ to obtain $\mathbf{I}$ and $\mathbf{I}'$. Therefore, by definition of the index vectors output by $\mathcal{P}^*$, which is such that I_j is a (fixed-size) prefix of I_m for $j < m$, it follows that also $I_j = I'_j$ for all $j < m$.

Therefore, the output $y_1, \ldots, y_{k_m}$ of $\mathcal{E}_m$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, a_{m-1}, \mathsf{RO}(I_{m-1}), a_m)$, where $I_m = (a_1, \ldots, a_m)$. This completes the proof. □

Proposition 2. (Run Time and Success Probability) *Let $K_m = k_m \cdots k_\mu$. The extractor $\mathcal{E}_m$ makes an expected number of at most $K_m + Q \cdot (K_m - 1)$ queries to $\mathcal{A}$ (and thus to $\mathcal{P}^*$) and successfully outputs $v = 1$ with probability at least*

$$\frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_m}{1 - \kappa_m}$$

where $\kappa_m := \mathrm{Er}(k_m, \ldots, k_\mu; N)$ is as defined in Eq. 1.

Proof. The proof goes by induction on m. The base case $m = \mu + 1$ holds trivially, understanding that $K_{\mu+1} = 1$ and $\mathrm{Er}(\emptyset, N) = 0$. Indeed, $\mathcal{E}_{\mu+1}$ makes 1 call to $\mathcal{A}$ and outputs $v = 1$ with probability $\epsilon(\mathcal{A})$. Alternatively, we can take $m = \mu$ as base case, which follows immediately from Lemma 4.

For the induction step, we assume now that the lemma is true for $m' = m + 1$ and consider the extractor $\mathcal{E}_m$. As in the 3-move case, we observe that, within a run of $\mathcal{E}_m$, all the queries that are made by the different invocations of $\mathcal{E}_{m+1}$ are answered *consistently* using lazy sampling, except for the queries to the index I_m, which is answered with different responses c'. This is indistinguishable from having them answered by a full-fledged random oracle $\mathsf{RO}\colon \{1, \ldots, U\} \to \{1, \ldots, N\}$, where we have enumerated the domain and codomain of RO as before. This enumeration allows RO to be identified with its function table $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$. Thus, the extractor is actually running the abstract sampling game from Fig. 2.

However, in contrast to the instantiation of Sect. 4, the entries of the array M are now *probabilistic*. Namely, while $\mathcal{A}$ is deterministic, the extractor $\mathcal{E}_{m+1}$ is a probabilistic algorithm. Fortunately, this does not influence the key properties of the abstract sampling game. For the purpose of the analysis, we may namely fix the randomness of the extractor $\mathcal{E}_{m+1}$. By linearity of the success probability and the expected runtime, the bounds that hold for any fixed choice of randomness also hold when averaged over the randomness. Thus, we can apply Lemmas 2 and 5 to bound the success probability and the expected runtime.[8]

To control the parameters P and T, which occur in the bounds of these lemmas, we make the following observation. A similar observation was required in the proof of Lemma 4.

[8] To be more precise, to allow for fresh randomness in the different runs of $\mathcal{E}_{m+1}$ within $\mathcal{E}_m$, we first replace the randomness of $\mathcal{E}_{m+1}$ by $F(j_1, \ldots, j_U)$ for a random function F, where $(j_1, \ldots, j_U)$ is the function table of the random oracle providing the answers to $\mathcal{E}_{m+1}$'s queries, and then we fix the choice of F and average over F after having applied Lemmas 2 and 5.

First, by Lemma 7, the index vector $\mathbf{I}$ output by $\mathcal{E}_{m+1}$ matches the index vector output by $\mathcal{P}^*$, when given the same random oracle RO. Second, since $\mathcal{P}^*$ is deterministic, its output can only change when the random oracle is reprogrammed at one of the indices $i \in \{1, \ldots, U\}$ queried by $\mathcal{P}^*$. Therefore, for every $(j_1, \ldots, j_U)$, let $S(j_1, \ldots, j_U) \subseteq \{1, \ldots, U\}$ be the set of points that $\mathcal{P}^*$ queries to the random oracle when $(j_1, \ldots, j_U)$ corresponds to the entire function table of the random oracle. Then, $\mathcal{P}^*$ will produce the same output when the random oracle is reprogrammed at an index $i \notin S(j_1, \ldots, j_U)$. In particular, $\mathbf{I}(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = \mathbf{I}(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$ for all j, j' and for all $i \notin S(j_1, \ldots, j_U)$. Furthermore, $|S(j_1, \ldots, j_U)| \leq Q$. Hence, the conditions of Lemmas 3 and 6 are satisfied, and it follows that $P \leq Q + 1$ and $T \leq Q$. We are now ready to analyze the success probability and the expected number of $\mathcal{A}$ queries of $\mathcal{E}_m$.

Success Probability. By the induction hypothesis, the success probability p_{m+1} of $\mathcal{E}_{m+1}$ is bounded by

$$p_{m+1} \geq \frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_{m+1}}{1 - \kappa_{m+1}}.$$

Then, by Lemmas 2 and 3, the success probability p_m of $\mathcal{E}_m$ is bounded by

$$\begin{aligned} p_m &\geq \frac{N}{N - k_m + 1} \left(p_{m+1} - (Q+1)\frac{k_m - 1}{N} \right) \\ &\geq \frac{N}{N - k_m + 1} \left(\frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_{m+1}}{1 - \kappa_{m+1}} - (Q+1)\frac{k_m - 1}{N} \right). \end{aligned}$$

By the recursive property (2) of $\kappa_m = \mathrm{Er}\big(k_m, \ldots, k_\mu; N, \ldots, N\big)$, it follows that

$$\frac{N - k_m + 1}{N}(1 - \kappa_{m+1}) = 1 - \kappa_m.$$

Hence,

$$\begin{aligned} p_m &\geq \frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_{m+1}}{1 - \kappa_m} - (Q+1)\frac{k_m - 1}{N - k_m + 1} \\ &= \frac{1}{1 - \kappa_m} \left(\epsilon(\mathcal{A}) - (Q+1) \cdot \left(\kappa_{m+1} + (1 - \kappa_m)\frac{k_m - 1}{N - k_m + 1} \right) \right) \\ &= \frac{1}{1 - \kappa_m} \left(\epsilon(\mathcal{A}) - (Q+1) \cdot \left(1 - (1 - \kappa_m) \cdot \frac{N}{N - k_m + 1} + (1 - \kappa_m)\frac{k_m - 1}{N - k_m + 1} \right) \right) \\ &= \frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_m}{1 - \kappa_m}, \end{aligned}$$

which proves the claimed success probability.

Expected Number of $\mathcal{A}$-Queries. Let the random variable T_m denote the number of $\mathcal{A}$-queries made by extractor $\mathcal{E}_m$. By the induction hypothesis, it holds that

$$\mathbb{E}[T_{m+1}] \leq K_{m+1} + Q \cdot (K_{m+1} - 1).$$

We make one crucial observation, allowing us to achieve the claimed query complexity, linear in Q. Namely, we can view the run of a (sub)extractor as a *two-stage* algorithm that allows an *early abort*. By Lemma 7, after only one $\mathcal{A}$-query $\mathcal{E}_{m+1}$ already returns the index I_m. At this stage, $\mathcal{E}_m$ can decide whether to continue the execution of $\mathcal{E}_{m+1}$ or to *early abort* this execution. If the index is incorrect, i.e., it does not match the one obtained in the first invocation of $\mathcal{E}_{m+1}$, then $\mathcal{E}_m$ early aborts the execution of $\mathcal{E}_{m+1}$. Only if the index is correct, the $\mathcal{E}_{m+1}$ execution has to be finished.

For this reason, we define the function $(j_1, \ldots, j_U) \mapsto \Gamma(j_1, \ldots, j_U)$, where $\Gamma(j_1, \ldots, j_U)$ is the (expected) costs of running $\mathcal{E}_{m+1}$ (completely) with random oracle $(j_1, \ldots, j_U)$. Moreover, we set $\gamma = 1$ indicating the cost of an early abort invocation of $\mathcal{E}_{m+1}$. These cost functions measure the expected number of calls to $\mathcal{A}$.

Hence, by Lemmas 5 and 6, the expected cost of running $\mathcal{E}_m$ is at most

$$\begin{aligned}\mathbb{E}[T_m] &\leq k_m \cdot \mathbb{E}[\Gamma(C)] + \gamma \cdot Q \cdot (k_m - 1) = k_m \cdot \mathbb{E}[T_{m+1}] + Q \cdot (k_m - 1) \\ &\leq K_m + Q \cdot (K_m - k_m) + Q \cdot (k_m - 1) = K_m + Q \cdot (K_m - 1),\end{aligned}$$

where C is distributed uniformly at random in $\mathcal{C}^U$. This completes the proof. □

The existence of extractor $\mathcal{E}_1$, combined with the **k**-special-soundness property, implies the following. This theorem shows that the Fiat–Shamir security loss for **k**-out-of-**N** special-sound $(2\mu+1)$-round interactive proofs is $Q+1$, i.e., the security loss is linear in the query complexity Q of provers $\mathcal{P}^*$ attacking the considered non-interactive random oracle proof $\mathsf{FS}[\Pi]$. In particular, the Fiat–Shamir security loss is independent of the number of rounds $(2\mu + 1)$ of the interactive proof Π. As before, a subtle issue is that the extractor needs to do the sampling without replacement efficiently. In Appendix A we discuss how this can be done.

Theorem 2. (FS Transformation of a $(k_1, \ldots, k_\mu)$-Special-Sound Protocol) *The Fiat–Shamir transformation* $\mathsf{FS}[\Pi]$ *of a* $\mathbf{k} = (k_1, \ldots, k_\mu)$*-special-sound interactive proof* Π*, in which all challenges are sampled from a set* $\mathcal{C}$ *of size* N*, is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q + 1)\kappa,$$

where $\kappa := \mathrm{Er}(\mathbf{k}; N)$ *is the knowledge error of the interactive proof* Π*.*

6.2. *Multi-Round Protocols with Arbitrary Challenge Sets*

Thus far, we considered and analyzed multi-round interactive proofs in which all challenges are sampled uniformly at random from the *same* set $\mathcal{C}$ of cardinality N. However, it is straightforward to verify that our techniques also apply to multi-round interactive proofs with different challenge sets, i.e., where the i-th challenge is sampled from a set $\mathcal{C}_i$ of cardinality N_i.

A natural first step in this generalization is to consider μ random oracles $\mathsf{RO}_i \colon \{0,1\}^{\leq u} \to \mathcal{C}_i$ instead of one. Besides some additional bookkeeping, all the reasoning goes through unchanged. Indeed, everything works as is when the prover $\mathcal{P}^*$

has the additional freedom to choose which random oracle it queries. Thus, we obtain the following generalization of Theorem 2.

Theorem 3. (FS Transformation of a **k**-out-of-**N** Special-Sound Interactive Proof) *The Fiat–Shamir transformation of a* **k***-out-of-***N** *special-sound interactive proof Π is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q+1)\kappa,$$

where $\kappa := \mathrm{Er}(\mathbf{k}; \mathbf{N})$ *is the knowledge error of the interactive proof Π.*

Remark 11. Alternatively, one could fix μ mappings $f_i : \{0,1\}^* \to \mathcal{C}_i$ and define the random oracle to output sufficiently long bitstrings. As before, this allows the prover $\mathcal{P}^*$ to take as input a single random oracle. Of course, this approach closely resembles practice, where the random oracles are replaced hash functions. However, one must be careful, since distinct bitstrings do not necessarily map to distinct challenges and uniformly random bitstrings do not necessary correspond to uniformly random challenges.

6.3. *Adaptive Security*

Thus far, we restricted our extractor analysis to the *static* or *non-adaptive* knowledge soundness notion of Definition 9. More precisely, our knowledge extractor takes as input a fixed statement x, is given oracle access to a *static* dishonest prover $\mathcal{P}^*$ attacking the considered protocol on input x, and aims to output a witness w for x.

However, our approach is easily modified towards proving *adaptive* knowledge soundness (Definition 10). To this end, let $\mathcal{P}^{\mathsf{a}}$ be an adaptive Q-query prover attacking the adaptive Fiat–Shamir transformation $\mathsf{FS}[\Pi]$ of a **k**-out-of-**N** special-sound interactive proof, i.e., $\mathcal{P}^{\mathsf{a}}$ takes no input and outputs a statement-proof pair (x, π), with $|x| = n$ for some fixed n, together with some auxiliary information aux. The random oracle algorithm $\mathcal{A}$ is defined to run $(x, \pi, \mathsf{aux}) \leftarrow \mathcal{P}^{\mathsf{a}}$ and verify that π is a valid proof for statement x. The main difference with the static case is that the indices are now defined as

$$I_1 := (x, a_1),\ I_2 := (x, a_1, a_2), \ldots,\ I_\mu := (x, a_1, \ldots, a_\mu),$$

so as to match up with(4), i.e., with the adaptive Fiat–Shamir transformation.

The statement x can thus be considered as part of the first message (x, a_1). Since all transcripts in a tree of transcripts have a common first message, it is easily seen that the extractor of Sect. 6, if successfully applied to this adaptive instantiation of $\mathcal{A}$, outputs a well-defined statement x together with a tree of accepting transcripts for this statement x. Moreover, x is the statement output by the extractor's first invocation of $\mathcal{P}^{\mathsf{a}}$.

For this reason, it immediately follows that our knowledge extractor, when applied to adaptive Q-query provers $\mathcal{P}^{\mathsf{a}}$, has the required properties. This proves the following theorem, showing that the adaptive Fiat–Shamir transformation of a special-sound interactive proof is adaptively knowledge sound.

Theorem 4. (FS Transformation—Adaptive Knowledge Soundness) *The* adaptive *Fiat–Shamir transformation of a* $\mathbf{k}$*-out-of-*$\mathbf{N}$ *special-sound interactive proof* Π *is* adaptively *knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q+1)\kappa,$$

where $\kappa := \mathrm{Er}(\mathbf{k}; \mathbf{N})$ *is the knowledge error of the interactive proof* Π.

7. The Fiat–Shamir Transformation of Parallel Repetitions

In the previous sections, we have established a positive result; for a broad class of interactive proofs the Fiat–Shamir security loss is only linear in the query complexity Q and independent of the number of rounds. One might therefore wonder whether the generic $(Q+1)^{\mu}$ security loss, for $(2\mu+1)$-move protocols, is only tight for contrived examples. In this section, we show that this is *not* the case. We demonstrate a non-trivial attack on the Fiat–Shamir transformation of the *parallel repetition* of $\mathbf{k}$-special-sound protocols.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a $(2\mu+1)$-move $\mathbf{k}$-special-sound interactive proof. We write $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ for its t-fold parallel repetition. That is, the prover $\mathcal{P}^t(x; w)$ runs t instances of $\mathcal{P}(x; w)$, i.e., each message is a tuple $(a^1, \ldots, a^t)$ of messages, one for each parallel thread of execution. Likewise, the verifier $\mathcal{V}^t(x)$ runs t instances of $\mathcal{V}(x)$ in parallel, i.e., each challenge is a tuple $(c^1, \ldots, c^t)$ of challenges, one for each parallel thread of the execution. Finally, the verifier accepts if all parallel instances are accepting.

Assuming certain natural properties on Π, which are satisfied by typical examples, and assuming again for simplicity that the challenge spaces $\mathcal{C}_i$ all have the same cardinality N, we show that, when $t \geq \mu$, there exists a malicious Q-query prover $\mathcal{P}^*$, attacking $\mathsf{FS}[\Pi^t]$, that, for any statement x, succeeds in convincing the verifier with probability at least

$$\frac{1}{2}\frac{Q^{\mu}}{\mu^{t+\mu}}\,\mathrm{Er}(\mathbf{k}; N)^t,$$

assuming some mild conditions on the parameters. Given that $\mathrm{Er}(\mathbf{k}; N)^t$ equals the soundness as well as the knowledge error of Π^t,[9] our attack shows that the security loss of the Fiat–Shamir transformation, when applied to the t-fold parallel repetition of Π, is at least $\frac{1}{2}Q^{\mu}/\mu^{t+\mu}$ (both, as a proof of language membership as well as a proof of knowledge). This stands in stark contrast to a single execution of a $\mathbf{k}$-special-sound protocol, where the loss is linear in Q and independent of μ.

We go on to discuss the kind of $\mathbf{k}$-special-sound protocols Π for which our attack applies. For simplicity, we restrict our attention here to $\mathbf{k} = (k, \ldots, k)$ and assume t and Q to be multiples of μ. In Appendix B.3, we consider the case of arbitrary $\mathbf{k}$, and the restrictions on t and Q can be easily avoided with some adjustments to the bound

[9] The soundness and knowledge error of a single invocation of Π are both equal to $\mathrm{Er}(\mathbf{k}; N)$. Therefore, it immediately follows that the soundness error of the parallel repetition Π^t is $\mathrm{Er}(\mathbf{k}; N)^t$. The fact that the knowledge error of Π^t also equals $\mathrm{Er}(\mathbf{k}; N)^t$ follows from the recent work [5].

and the reasoning. Let $\boldsymbol{\ell} = (\ell, \ldots, \ell)$ where $\ell \leq k - 1$. The attack on $\mathsf{FS}[\Pi^t]$ uses a property most $\mathbf{k}$-special-sound protocols Π satisfy, namely that there exists an efficient attack strategy $\mathcal{A}$ against Π which tries to guess challenges up front so that:

1. In any round, $\mathcal{A}$ can prepare and send a message so that if he is lucky and the next challenge falls in a certain set Γ of cardinality ℓ, $\mathcal{A}$ will be able to complete the protocol and have the verifier accept (no matter what challenges $\mathcal{A}$ encounters in the remaining rounds), and
2. until $\mathcal{A}$ is lucky in the above sense, in any round $\mathcal{A}$ can actually prepare B distinct messages as above, for a given parameter B.

We call protocols which admit such an attack strategy $\boldsymbol{\ell}$-special-unsound with B potential responses per round (see Appendix B.1 for a formal definition). The first point in particular implies an attack strategy for the interactive proof Π that succeeds with probability $\mathrm{Er}(\boldsymbol{\ell} + 1, N)$. Since many $\mathbf{k}$-special-sound interactive proofs Π are $\boldsymbol{\ell}$-special-unsound with $\boldsymbol{\ell} = \mathbf{k} - 1$, this confirms the tightness of the knowledge error $\mathrm{Er}(\mathbf{k}, N)$, as already mentioned at the end of Sect. 2.2. The second point implies that in the context of the Fiat–Shamir transformation, an attacker can produce and try multiple message-challenge pairs in any round.

These requirements are very common (for non-trivial $\boldsymbol{\ell}$ and large B). For example, the folding technique of [10], when used to fold two parts into one, satisfies $(3, \ldots, 3)$-special-soundness and $(2, \ldots, 2)$-special-unsoundness with an exponential parameter B; we discuss this in detail in Appendix B.2. Note that, while the *honest* prover is *deterministic*, a dishonest prover can produce different messages (and hope to be lucky with one of the corresponding challenges).

The following theorem gives a lower bound for the success probability of our attack on the Fiat–Shamir transformation $\mathsf{FS}[\Pi^t]$ of the t-fold parallel repetition Π^t of an interactive proof Π with certain common soundness and unsoundness properties.

Theorem 5. *Let Π be a $(2\mu + 1)$-move $(k, \ldots, k)$-out-of-$(N, \ldots, N)$ special-sound interactive proof that is $(\ell, \ldots, \ell)$-special-unsound with B responses per round for $\ell = k-1$. Furthermore, let t, $Q \in \mathbb{N}$ be integer multiples of μ such that $Q \cdot \left(\frac{\ell}{N}\right)^{t/\mu} \leq 1/4$ and $B \geq Q$. Then there exists a Q-query dishonest prover $\mathcal{P}^*$ against $(\mathcal{P}, \mathcal{V}) = \mathsf{FS}[\Pi^t]$ such that, for any statement $x \in \{0,1\}^*$,*

$$\epsilon(\mathcal{P}^*, x) = \Pr\left(\mathcal{V}^{\mathsf{RO}}(x, \mathcal{P}^{*,\mathsf{RO}}) = 1\right) \geq \left(1 - \left(1 - \left(\frac{k-1}{N}\right)^{t/\mu}\right)^{Q/\mu}\right)^{\mu} \geq \frac{1}{2}\frac{Q^{\mu}}{\mu^{t+\mu}}\,\mathrm{Er}(\mathbf{k}; N)^t.$$

The runtime of $\mathcal{P}^*$ is at most $t\,Q$ times the runtime of attack strategy $\mathcal{A}$ against Π.

Proof. The basic idea of the attack is that (groups of) parallel threads can be attacked individually and independently from each other over the different rounds of the protocol. Concretely, the attack is given by the adversary $\mathcal{P}^*$ against $\mathsf{FS}[\Pi^t]$, which makes up to $Q = \mu \cdot Q'$ queries, defined as follows: $\mathcal{P}^*$ runs attack strategy $\mathcal{A}$ in parallel against all $t = \mu \cdot t'$ threads. Let us call a thread *green* if strategy $\mathcal{A}$ succeeds in guessing the challenge for that thread (and hence, $\mathcal{V}$ will eventually accept for that thread). Otherwise, a thread is *red*. All threads start out red, and the goal of $\mathcal{P}^*$ is to turn all threads green.

To do so, in every round $\mathcal{P}^*$ tries to turn at least $t' = t/\mu$ red threads into green threads (or all red threads into green threads if fewer than t/μ remain). For this, $\mathcal{P}^*$ uses $\mathcal{A}$ to get the messages which it feeds to the random oracle. If $\mathcal{P}^*$ was lucky with the received challenges for at least $t' = t/\mu$ threads, then enough red threads turn green. Else, $\mathcal{P}^*$ tries the considered round again, exploiting that $\mathcal{A}$ can produce up to B distinct messages that give him a chance, each one giving a fresh challenge from the random oracle. The dishonest prover $\mathcal{P}^*$ tries up to $Q' = Q/\mu$ times per round until it gives up (and fails).

The number of queries $\mathcal{P}^*$ makes to the random oracle is at most Q; hence, $\mathcal{P}^*$ is a Q-query adversary. The probability that $\mathcal{P}^*$ succeeds for any try in any round to turn at least $t' = t/\mu$ red threads into green threads is at least $(\frac{\ell}{N})^{t'} = \lambda^{t'}$, where we introduce $\lambda = \frac{\ell}{N}$ to simplify the upcoming expressions. Therefore, since $\mathcal{P}^*$ makes at most $Q' = Q/\mu$ queries in every round, the success probability for any fixed round is at least

$$1 - \left(1 - \lambda^{t'}\right)^{Q'} \;\geq\; Q'\lambda^{t'} - 2\,Q'^2\lambda^{2t'} \;=\; Q'\lambda^{t'}\left(1 - 2\,Q'\lambda^{t'}\right). \tag{8}$$

where the inequality follows from the fact that $1 - (1-x)^n \geq nx - 2n^2x^2$, which can be shown to hold when $nx \leq 1/2$ (see Appendix B), which is (more than) satisfied for $x = \lambda^{t'}$ and $n = Q'$ by assumption. Hence, $\mathcal{P}^*$ succeeds (in all μ rounds) with probability at least

$$Q'^{\mu}\lambda^{t}\left(1 - 2\,Q'\lambda^{t'}\right)^{\mu} \;\geq\; Q'^{\mu}\lambda^{t}\left(1 - 2Q\lambda^{t'}\right) \;\geq\; \frac{1}{2}Q'^{\mu}\lambda^{t},$$

where we use that $(1-z)^n \geq 1 - nz$ for $n \in \mathbb{N}$ and $z \in [0,1]$ to argue the first inequality, and $Q \cdot \left(\frac{\ell}{N}\right)^{t'} \leq 1/4$ for the second. To complete the analysis of $\mathcal{P}^*$'s success probability, we observe that

$$\mathrm{Er}(\mathbf{k}; N) = 1 - \left(1 - \frac{k-1}{N}\right)^{\mu} \leq \mu \cdot \frac{k-1}{N} = \mu \cdot \frac{\ell}{N} = \mu \cdot \lambda.$$

Hence, the success probability of $\mathcal{P}^*$ is at least $\frac{1}{2}Q'^{\mu}\left(\frac{\mathrm{Er}(\mathbf{k};N)}{\mu}\right)^{t}$, as claimed. □

Recall that we assume t and Q to be divisible by μ; this is mainly for simplicity. In general, i.e., when dropping this assumption, the success probability has lower bound $1/2 \cdot \lfloor Q/\mu \rfloor^{\mu} \cdot \left(\mathrm{Er}(\mathbf{k}; N)/\mu\right)^{\lceil t/\mu \rceil \mu}$.

Acknowledgements

The first author was supported by EU H2020 Project No. 780701 (PROMETHEUS) and the Vraaggestuurd Programma Cyber Security & Resilience, part of the Dutch Top Sector High Tech Systems and Materials program. The third author was supported by the topic Engineering Secure Systems (46.23.01) of the Helmholtz Association (HGF) and by KASTEL Security Research Labs.

Appendix

A. A Note on Sampling Without Replacement

Our extractor constructions require to *efficiently* sample from the challenge set $\mathcal{C}$ *without replacement*, until a certain number of good elements are found or $\mathcal{C}$ is exhausted. Recall that we may assume $\mathcal{C}$ to be encoded as $\{1, \ldots, N\}$ where $N = |\mathcal{C}|$ (see Definition 3). While we formally argue that the expected number of samples without replacement is bounded, it is a priori not clear that these samples can be efficiently prepared. Indeed, the naive way of re-sampling until a new element is obtained becomes very ineffective if one is close to having exhausted $\mathcal{C}$ (which may well happen with probability $\approx 1/N$). For this reason, we recall here how to sample sequentially without replacement in an efficient manner, thereby justifying the expected polynomial runtime of our knowledge extractors.

Sequential sampling for $\{1, \ldots, N\}$ without replacement is a fundamental statistical problem that has been studied extensively, e.g., [4,21,29,31,35]. However, the first sampling algorithms have an expected runtime that scales linearly in the size N of the population [29,31]. In our setting, the population (i.e., challenge set) is oftentimes exponentially large, i.e., we require a more advanced sampling algorithm to argue knowledge soundness. Vitter's sampling algorithm [35] improved upon prior work and has an expected runtime that is linear in the number ℓ of samples that have to be drawn. While Vitter's approach is sufficiently efficient for our purposes, it does require prior knowledge of the sample size ℓ. Our knowledge extractors do not know how many samples they will need to draw before sufficiently many good ones have been found, i.e., also Vitter's algorithm does not provide the required sampling strategy.

So let us now sketch a sequential sampling approach that does render our knowledge extractors efficient. The population is (implicitly) initialized as $\mathcal{C}_0 = \{1, \ldots, N\}$ and the sampled set as $\mathcal{S}_0 = \emptyset$. The main idea is that the sampler samples c uniformly at random from $\mathcal{C}_0$ and, if $c \notin \mathcal{S}_0$, c is added to $\mathcal{S}_0$. Since $\Pr(c \notin \mathcal{S}_0) = 1 - |\mathcal{S}_0|/N$, it holds that, as long as $|\mathcal{S}_0| \leq N/2$, the expected number of samples before a fresh c is found, and added to $\mathcal{S}_0$, is at most 2.

Unfortunately, this sampling strategy becomes less efficient as $\mathcal{S}_0$ grows and the extractor might very well require more than $N/2$ samples. For this reason, once $|S_0| = N/2$ (for simplicity we assume N to be a power of two), the sampler defines a new population $\mathcal{C}_1 = \mathcal{C}_0 \setminus \mathcal{S}_0$ of size $N/2$, stored in an array $\mathcal{C}_1[1], \ldots, \mathcal{C}_1[N/2]$, and initializes $\mathcal{S}_1 = \emptyset$. The sampler now proceeds as before, but now using $\mathcal{C}_1$ and $\mathcal{S}_1$ until $|\mathcal{S}_1| = |C_1|/2 = N/4$, at which point population $\mathcal{C}_2 = \mathcal{C}_1 \setminus \mathcal{S}_1$ and set $\mathcal{S}_2 = \emptyset$ are defined. In every iteration, it requires at most 2 samples from $\mathcal{C}_i$ to find a new element that can be added to $\mathcal{S}_i$ (in expectation). However, the set-difference operation to create $\mathcal{C}_i = \mathcal{C}_{i-1} \setminus \mathcal{S}_{i-1}$ takes time proportional in $|\mathcal{C}_{i-1}| = N/2^{i-1}$. Fortunately, these costs can be amortized. More precisely, suppose the extractor ends up sampling ℓ challenges, with $N - N/2^i < \ell \leq N - N/2^{i+1}$ for some $0 \leq i \leq \log_2 N$. Then for each of the ℓ samples, the sampler requires 2 trials. Further, the expected cost of creating the arrays $\mathcal{C}_1, \ldots, \mathcal{C}_i$ is proportional to

$$N + \frac{N}{2} + \cdots + \frac{N}{2^{i-1}} = 2N - \frac{N}{2^{i-1}} < 2\ell.$$

Hence, the expected computational cost of sampling ℓ elements linear in ℓ. Furthermore, the sampling strategy (and the analysis) does not require the sample size to be known in advance, but also applies if the sample size is determined by the sampled elements, i.e., if we keep sampling until k "good" elements are found, for a given

k. The expected computational costs is then proportional to the expected number of samples. Altogether, this shows that the extractor can indeed efficiently sample without replacement.

B. Detailed Discussion of the Attack

We discuss our attack in more detail, but also in more generality, here. Let Π be a $(2\mu+1)$-move public-coin interactive proof for relation R with challenge sets $\mathcal{C}_1, \ldots, \mathcal{C}_\mu$.
In Appendix B.1, we give a formal definition of the unsoundness property required for our attack to succeed. In Appendix B.2, we give an example of a multi-round protocol satisfying this unsoundness property, i.e., a protocol to which our attack applies. In Appendix B.3, we generalize the attack to a broader class of protocols and describe the properties of this generalization.

B.1. Special-Unsoundness with Multiple Potential Responses

For simplicity, we assume perfect correctness. The attack applies when Π satisfies the following property.

Definition 12. (*$\boldsymbol{\ell}$-special-unsoundness with B potential responses per round*) We say that Π has *$\boldsymbol{\ell}$-special-unsoundness* for $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_\mu) \in \mathbb{N}^\mu$ if there exists a dishonest prover $\mathcal{A}$ of the following form, and so that in the execution with $\mathcal{V}$ and input x the following holds:

- $\mathcal{A}$ starts off in *active mode*, which is so that in every round $2i-1$ when $\mathcal{A}$ sends the response a_i to $\mathcal{V}$, there exists a subset $\Gamma_i \subseteq \mathcal{C}_i$ of cardinality ℓ_i (defined as a function of the state of $\mathcal{A}$ at that point) such that if the subsequent challenge c_i is in Γ_i then $\mathcal{A}$ switches into *passive mode*.
- If $\mathcal{A}$ switches to *passive mode* then it remains in *passive mode*, and $\mathcal{V}$ will accept at the end of the protocol execution.

We say Π is $\boldsymbol{\ell}$-special-unsound with B *potential responses per round* if additionally the following holds:
As long as $\mathcal{A}$ is in active mode, the computation of a_i involves a designated *seed* s_i, which is chosen arbitrarily from some set $\mathcal{S}_i$. By redoing the computation with different seeds (but fixed randomness), $\mathcal{A}$ can obtain at least B *distinct* a_i's satisfying the properties specified above (i.e., existence of Γ_i such that $\mathcal{A}$ switches to passive mode if $c_i \in \Gamma_i$). Moreover, we require that distinct seeds produce distinct a_i, that is, the mapping from seed s_i to message a_i is injective.

The "B potential responses per round", which can be derived by changing the seed s, are used so that $\mathcal{A}$ can "retry" each round at least B times. This will be used to obtain fresh challenges from the random oracles. The requirement that different seeds produce different a_i will simplify statements and proofs, but it will be evident that it is stronger than necessary. One can relax the definition as long as $\mathcal{A}$ can produce distinct a_i efficiently, say by picking a few seeds at random; the (runtime) analysis must then take these additional tries into account.

Protocols that are $(k_1, \ldots, k_\mu)$-special-sound are often $(k_1-1, \ldots, k_\mu-1)$-special-unsound. For example, k-special-sound Σ-protocols are typically $(k-1)$-special-unsound in that a dishonest prover can first pick an arbitrary subset $\Gamma \subset \mathcal{C}$ of cardinality $k-1$, then choose an arbitrary response z, and finally compute a first message a (as a function of Γ and z), so that (a, c, z) is an accepting conversation whenever $c \in \Gamma$.[10] Furthermore, different choices (possibly of a certain form) of the pre-chosen response z typically lead to different values of a, thus satisfying the multiple responses per round property as well.

For multi-round $\mathbf{k}$-special-sound protocols, this kind of attack often extends in such a way that once $\mathcal{A}$ is successful in one round, it has all the information needed to continue as an honest prover, and the verifier will accept. In other words, the *passive mode* of $\mathcal{A}$ usually corresponds to following the remainder of the protocol honestly. Below, we discuss in detail that in particular Bulletproofs-like protocols satisfy the Definition 12.

[10] For 2-special-sound Σ-protocols, this is very much in line with being *special honest-verifier zero-knowledge*

B.2. An Example Protocol

Bulletproofs-like protocols are typically $(2, \ldots, 2)$-special-unsound (or worse), as we explain here. At the core of these protocols is the folding technique of [10]. Here, we describe the adaptation considered [1], which, in the plain DL-setting, is the following 3-special-sound Σ-protocol for proving knowledge of $\mathbf{x} \in \mathbb{F}_q^n$ with $\mathbf{g}^{\mathbf{x}} = h$; we refer to this protocol as the *folding protocol* in the remainder. Here, $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n$ and $h \in \mathbb{G}$ are publicly known, where $\mathbb{G}$ is a group with prime order q, and $\mathbf{g}^{\mathbf{x}} := g^{x_1} \cdots g_n^{x_n}$; furthermore, it is assumed (w.l.o.g.) that n is a power of 2, and so in particular n is even and we can write $\mathbf{g}_L = (g_1, \ldots, g_{n/2})$ and $\mathbf{g}_R = (g_{n/2+1}, \ldots, g_{n/2})$, and similarly for $\mathbf{x}$, so that $\mathbf{g}^{\mathbf{x}}$ is the component-wise product $\mathbf{g}^{\mathbf{x}} = \mathbf{g}_L^{\mathbf{x}_L} \mathbf{g}_R^{\mathbf{x}_R}$ of $\mathbf{g}_L^{\mathbf{x}_L}$ and $\mathbf{g}_R^{\mathbf{x}_R}$. Lastly, we may assume that $\mathbf{g} \neq (1, \ldots, 1)$, since in this case the relation $\mathbf{g}^{\mathbf{x}} = h$ is trivial and so there is nothing for the prover to prove.

$$\left(\mathbf{g}_L^c \odot \mathbf{g}_R\right)^{\mathbf{z}} = a h^c b^{c^2}, \tag{9}$$

where $\mathbf{g}_L^c \odot \mathbf{g}_R$ denotes the component-wise product of $\mathbf{g}_L^c$ and $\mathbf{g}_R$. The communication complexity of this Σ-protocol can now be improved by not sending the answer $\mathbf{z}$, but instead proving knowledge of $\tilde{\mathbf{x}}$ with $\tilde{\mathbf{g}}^{\tilde{\mathbf{x}}} = \tilde{h} := a h^c b^{c^2}$ for $\tilde{\mathbf{g}} := \mathbf{g}_L^c \odot \mathbf{g}_R$; the latter is done by means of running another instance of the protocol but now with the smaller witness $\tilde{\mathbf{x}} = \mathbf{z}$. But then, also in this instance, instead of sending the answer one can run yet another instance of the protocol to prove knowledge of the answer, etc. This results in a *compressed* multi-round interactive proof of knowledge with communication complexity logarithmically in n (instead of linear).

Towards arguing unsoundness, i.e., Definition 12, it follows directly from the construction design for the compression protocol that once a dishonest prover holds the correct answer, then he can honestly follow the remainder of the protocol and the verifier will accept. Thus, this defines the *passive mode*. For the *active mode*, we observe that the following holds for the folding protocol. For any two $c_1, c_2 \in \mathbb{F}_q$ with $c_1^2 \neq c_2^2$ and with $\tilde{\mathbf{g}}_1 := \mathbf{g}_L^{c_1} \odot \mathbf{g}_R$ and $\tilde{\mathbf{g}}_2 := \mathbf{g}_L^{c_2} \odot \mathbf{g}_R$ both not being equal to $(1, \ldots, 1)$, $\mathcal{A}$ can pick an arbitrary answer $\mathbf{z} \in \mathbb{F}_q^{n/2}$ and solve the equation system

$$\begin{aligned} a b^{c_1^2} &= \tilde{\mathbf{g}}_1^{\mathbf{z}} h^{-c_1} \\ a b^{c_2^2} &= \tilde{\mathbf{g}}_2^{\mathbf{z}} h^{-c_2} \end{aligned}$$

for a and b by applying the inverse of the matrix $\begin{pmatrix} 1 & c_1^2 \\ 1 & c_2^2 \end{pmatrix}$ to both sides "in the exponent".

Then, by construction, the pre-chosen answer $\mathbf{z}$ satisfies (9) if the verifier's challenge c happens to be in $\Gamma = \{c_1, c_2\}$, and thus by switching to the *passive mode* now if this happens to be the case and following the remainder of the protocol honestly, $\mathcal{A}$ will make the verifier accept, as required by Definition 12. Also, if desired, $\mathcal{A}$ can solve the above equation system with a different choice of $\mathbf{z}$ to obtain a new pair (a, b). More precisely, let $i \in \{1, \ldots, n/2\}$ be so that the i-th coordinate of, say, $\tilde{\mathbf{g}}_1$ is not 1 (which exists by choice of c_1), then the q possible choices of the i-th coordinate $z_i \in \mathbb{F}_q$ of $\mathbf{z}$ lead to q *distinct* right-hand sides in the above equation system, and thus to q *distinct* pairs (a, b), when keeping the remaining coordinates of $\mathbf{z}$ fixed.

B.3. Generalizing Theorem 5 to Arbitrary Special-Unsound Protocols

Theorem 6 removes the restriction $\ell_1 = \cdots = \ell_\mu$ in Theorem 5. Indeed, it considers a $(\ell_1, \ldots, \ell_\mu)$-special-unsound protocol with arbitrary ℓ_i's.

Theorem 6. *Let Π be a $(2\mu + 1)$-move public-coin interactive proof with challenge spaces $\mathcal{C}_1, \ldots, \mathcal{C}_\mu$. Suppose Π has $(\ell_1, \ldots, \ell_\mu)$-special-unsoundness with B responses per round. Let Π^t be the t-fold parallel repetition of Π. Let $m_1, \ldots, m_\mu \in \mathbb{N}$ such that $\sum_{i=1}^{\mu} m_i = t$, and set $\alpha_i = (\frac{\ell_i}{|\mathcal{C}_i|})^{m_i}$. Let $Q = \mu Q'$ for $Q' \in \mathbb{N}$ with $Q' \sum_{i=1}^{\mu} \alpha_i < 1/4$ and $Q' \leq B$. Then there is a Q-query dishonest prover $\mathcal{P}^*$ against*

$(\mathcal{P}, \mathcal{V}) = \mathsf{FS}[\Pi^t]$ *so that for every statement* x

$$\epsilon(\mathcal{P}^*, x) = \Pr\big(\mathcal{V}^{\mathsf{RO}}(x, \mathcal{P}^{*,\mathsf{RO}}) = 1\big) \geq \frac{1}{2}\left(\frac{Q}{\mu}\right)^{\mu} \cdot \prod_{i=1}^{\mu} \alpha_i .$$

The run time of $\mathcal{P}^*$ *is* $O(t \cdot Q \cdot T_{\mathcal{A}})$*, where* $T_{\mathcal{A}}$ *is an upper-bound on the runtime of some* $\boldsymbol{\ell}$*-special-unsoundness dishonest prover* $\mathcal{A}$ *attacking the (interactive) proof system* Π *(i.e., when* $\mathcal{A}$ *computes* one *message* a_i *per round).*

We recover the statement of Theorem 5 by considering Π that is $(\ell, \ldots, \ell)$-special-unsound and has challenge spaces of fixed size $|\mathcal{C}_i| = N$, and by setting $m_1 = \cdots = m_\mu = t/\mu$ and thus $\alpha_1 = \cdots = \alpha_\mu = (\frac{k-1}{N})^{t/\mu}$ then. In general, the freedom in choosing $m_1, \ldots, m_\mu$ allows to adapt the number of threads that should be successfully attacked in each round, which is useful when the $|\mathcal{C}_i|$'s and/or the ℓ_i's vary over the different rounds.

Proof. The proof is analogous to the proof of Theorem 5, with the attack working in a thread-by-thread manner. Let $\mathcal{A}$ be a $(\ell_1, \ldots, \ell_\mu)$-special-unsoundness malicious prover for Π with B potential responses per round. Let $\mathcal{P}^*$ be the adversary defined as follows:

1. Start t parallel instances $\mathcal{A}$ with input x, denoted as $\mathcal{A}_1, \ldots, \mathcal{A}_t$. We write a_i^j for the $(2i-1)$-th move message of the j-th instance, and similarly for the corresponding unsoundness set Γ_i^j.
2. From $i = 1$, repeat until $i = \mu$
   - Try up to Q/μ times:
     - For all $\mathcal{A}_j$ which are in active mode, pick a seed $s_i^j \in \mathcal{S}$, distinct from all seeds previously chosen for $\mathcal{A}_j$ in this round.
     - Run all $\mathcal{A}_j$ to obtain the $(2i-1)$-th move message a_i^j for all i, j. Moreover, compute the challenges c_i^j for all threads j.
     - If, after receiving the challenges c_i^j, at least $\sum_{k=1}^{i} m_k$ of the $\mathcal{A}_j$'s would be in *passive mode*, send the challenges and increase i (i.e., move on to attacking the next round). In particular, if at least m_i $\mathcal{A}_j$'s switch from active to passive, i will increase.
3. $\mathcal{P}^*$ receives $a_{\mu+1}^j$ from $\mathcal{A}_j$ (for $j = 1, \ldots, t$) and completes the fake proof.

First, let us analyze the efficiency of $\mathcal{P}^*$. Clearly, $\mathcal{P}^*$ is a Q-query random-oracle algorithm. Moreover, $\mathcal{P}^*$ emulates at most $t \cdot Q$ (partial) runs of the $\boldsymbol{\ell}$-special-unsoundness malicious prover instances $\mathcal{A}_j$.
Now, let us analyze the success probability. For the i-th challenge, $\mathcal{P}^*$ will execute the inner loop body at most $Q' \in \mathbb{N}$ times, where $Q = \mu Q'$ by definition. Moreover, for each retry, at least one message a_i^j is different from its previous choices. (Because different seeds s_i^j lead to different a_i^j for any $\mathcal{A}_j$ which is in active mode, and at least one $\mathcal{A}_j$ is still in active mode, because otherwise the inner loop increases i). Thus, the random oracle queries provide fresh random challenges. The probability that an inner iteration in move $2i-1$ succeeds for a uniformly random challenge choice is at least $\alpha_i = (\ell_i/|\mathcal{C}_i|)^{m_i}$, since by construction at most m_i threads/instances $\mathcal{A}_j$ need to be lucky in this round. Consequently, the probability that Q' tries are sufficient to switch enough $\mathcal{A}_j$ from active to passive mode in the i-the iteration is

$$1 - (1-\alpha_i)^{Q'} \geq Q'\alpha_i - 2(Q'\alpha_i)^2 . \tag{10}$$

For the inequality we used that $1 - (1-x)^n \geq nx - 2n^2x^2$ for any $x \geq 0$ and $n \in \mathbb{N}$ with $0 \leq nx \leq 1/2$, where the latter follows from

$$|1 - (1-x)^n - nx| \leq \sum_{i=2}^{n} \binom{n}{i} x^i \leq \sum_{i=2}^{n} (nx)^i \leq \sum_{i=2}^{\infty} (nx)^i = \frac{n^2x^2}{1-nx} \leq 2n^2x^2 .$$

Noting that $Q'\alpha_i \leq Q' \sum_i \alpha_i \leq 1/4 \leq 1/2$ by assumption, we can thus indeed conclude (10).

As every round must be successful for $\mathcal{P}^*$ to succeed, the overall success probability of $\mathcal{P}^*$ is thus at least

$$\prod_{i=1}^{\mu}\left(1-(1-\alpha_i)^{Q'}\right) \geq \prod_{i=1}^{\mu}\left(Q'\alpha_i - 2(Q'\alpha_i)^2\right) = Q'^{\mu}\prod_{i=1}^{\mu}\alpha_i \cdot \prod_{i=1}^{\mu}(1-2Q'\alpha_i)\,.$$

Using that $\prod_{i=1}^{\mu}(1-z_i) \geq 1-\sum_{i=1}^{\mu} z_i$ for $z_i \geq 0$, we can further bound the right-hand side as

$$Q'^{\mu}\prod_{i=1}^{\mu}\alpha_i \cdot \prod_{i=1}^{\mu}(1-2Q'\alpha_i) \geq Q'^{\mu}\prod_{i=1}^{\mu}\alpha_i \cdot \left(1-2\sum_{i=1}^{\mu}Q'\alpha_i\right) \geq \frac{1}{2}\cdot Q'^{\mu}\prod_{i=1}^{\mu}\alpha_i\,,$$

where we used again $Q'\sum_i \alpha_i \leq 1/4$ for the final inequality. This proves the claim.

□

References

[1] T. Attema, R. Cramer, Compressed Σ-protocol theory and practical application to plug & play secure algorithmics, in D. Micciancio, T. Ristenpart, editor, *CRYPTO 2020, Part III*. LNCS, vol. 12172 (Springer, Heidelberg, 2020), pp. 513–543

[2] T. Attema, R. Cramer, L. Kohl, A compressed Σ-protocol theory for lattices, in T. Malkin, C. Peikert, editors, *CRYPTO 2021, Part II, Virtual Event, August 2021*. LNCS, vol. 12826 (Springer, Heidelberg, 2021), pp. 549–579

[3] T. Attema, R. Cramer, M. Rambaud, Compressed Σ-protocols for bilinear group arithmetic circuits and application to logarithmic transparent threshold signatures, in M. Tibouchi, H. Wang, editors *ASIACRYPT 2021, Part IV*. LNCS, vol. 13093 (Springer, Heidelberg, 2021), pp. 526–556

[4] J.H. Ahrens and U. Dieter. Sequential random sampling. *ACM Trans. Math. Softw.*, 11(2):157–169, 1985

[5] T. Attema, S. Fehr, Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs, in Y. Dodis, T. Shrimpton, editors, *CRYPTO*. Lecture Notes in Computer Science, vol. 13507(Springer, 2022), pp. 415–443

[6] T. Attema, S. Fehr, M. Klooß, Fiat-Shamir transformation of multi-round interactive proofs, in *Theory of Cryptography Conference (TCC)* (2022)

[7] S. Ames, C. Hazay, Y. Ishai, M. Venkitasubramaniam, Ligero: lightweight sublinear arguments without a trusted setup, in B.M. Thuraisingham, D. Evans, T. Malkin, D. Xu, editors, *ACM CCS 2017*. (ACM Press, October/November 2017), pp. 2087–2104

[8] M.R. Albrecht, R.W.F. Lai, Subtractive sets over cyclotomic rings - limits of Schnorr-like arguments over lattices, in T. Malkin, C. Peikert, editors, *CRYPTO 2021, Part II, Virtual Event, August 2021*. LNCS, vol. 12826 (Springer, Heidelberg, 2021), pp. 519–548

[9] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, G. Maxwell, Bulletproofs: Short proofs for confidential transactions and more, in *2018 IEEE Symposium on Security and Privacy*. (IEEE Computer Society Press, May 2018), pp. 315–334

[10] J. Bootle, A. Cerulli, P. Chaidos, J. Groth, C. Petit, Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting, in M. Fischlin, J.-S. Coron, editors, *EUROCRYPT 2016, Part II. LNCS*, vol. 9666 (Springer, Heidelberg, 2016), pp. 327–357

[11] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, N.P. Ward, Aurora: transparent succinct arguments for R1CS, in Y. Ishai, V. Rijmen, editors, *EUROCRYPT 2019, Part I*. LNCS, vol. 11476 (Springer, Heidelberg, 2019), pp. 103–128

[12] E. Ben-Sasson, A. Chiesa, N. Spooner, Interactive oracle proofs. in M. Hirt, A.D. Smith, editors, *TCC 2016-B, Part II*. LNCS, vol. 9986 (Springer, Heidelberg, October/November 2016), pp. 31–60

[13] B. Bünz, B. Fisch, A. Szepieniec, Transparent SNARKs from DARK compilers, in A. Canteaut, Y. Ishai, editors, *EUROCRYPT 2020, Part I*. LNCS, vol. 12105 (Springer, Heidelberg, 2020), pp. 677–706

[14] A.R. Block, J. Holmgren, A. Rosen, R.D. Rothblum, P. Soni, Time- and space-efficient arguments from groups of unknown order, in T. Malkin, C, Peikert, editors, *CRYPTO 2021, Part IV, Virtual Event, August 2021*. LNCS, vol. 12828 (Springer, Heidelberg, 2021), pp.123–152

[15] M. Bellare, R. Impagliazzo, M. Naor, Does parallel repetition lower the error in computationally sound protocols? in *38th FOCS, October 1997*. (IEEE Computer Society Press, 1997), pp. 374–383
[16] B. Barak, Y. Lindell, Strict polynomial-time in simulation and extraction, in *34th ACM STOC, May 2002*. (ACM Press, 2002), pp. 484–493
[17] J. Bootle, V. Lyubashevsky, N.K. Nguyen, G. Seiler, A non-PCP approach to succinct quantum-safe zero-knowledge, in D. Micciancio, T. Ristenpart, editors, *CRYPTO 2020, Part II*. LNCS, vol. 12171 (Springer, Heidelberg, 2020), pp. 441–469
[18] M. Bellare, G. Neven, Multi-signatures in the plain public-key model and a general forking lemma, in A. Juels, R.N. Wright, S. De Capitani di Vimercati, editors, *ACM CCS 2006, October/November 2006*, (ACM Press, 2006), pp. 390–399
[19] R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G.N. Rothblum, R.D. Rothblum, D. Wichs, Fiat–Shamir: from practice to theory, in M. Charikar, E. Cohen, editors, *51st ACM STOC, June 2019*. (ACM Press, 2019), pp. 1082–1090
[20] A. Chiesa, P. Manohar, N. Spooner, Succinct arguments in the quantum random oracle model, in D. Hofheinz, A. Rosen, editors, *TCC 2019, Part II*. LNCS, vol. 11892 (Springer, Heidelberg, 2019), pp. 1–29
[21] L. Devroye. *Non-Uniform Random Variate Generation*. (Springer, Berlin, 1986)
[22] J. Don, S. Fehr, C. Majenz, C. Schaffner, Security of the Fiat–Shamir transformation in the quantum random-oracle model, in A. Boldyreva, D. Micciancio, editors, *CRYPTO 2019, Part II*. LNCS, vol. 11693 (Springer, Heidelberg, 2019), pp. 356–383
[23] R. del Pino, V. Lyubashevsky, G. Seiler, Short discrete log proofs for FHE and ring-LWE ciphertexts, in D. Lin, K. Sako, editors, *PKC 2019, Part I*. LNCS, vol. 11442 (Springer, Heidelberg, 2019), pp. 344–373
[24] A. Fiat, A. Shamir, How to prove yourself: pactical solutions to identification and signature problems, in A.M. Odlyzko, editor, *CRYPTO'86*, LNCS, vol. 263. (Springer, Heidelberg, 1987), pp. 186–194
[25] C. Gentry, S. Halevi, V. Lyubashevsky, Practical non-interactive publicly verifiable secret sharing with thousands of parties, in O. Dunkelman, S. Dziembowski, editors, *EUROCRYPT 2022, Part I, May/June 2022*. LNCS, vol. 13275 (Springer, Heidelberg, 2022), pp. 458–487.
[26] C. Ganesh, H. Khoshakhlagh, M. Kohlweiss, A. Nitulescu, M. Zajac, What makes Fiat–Shamir zk-SNARKs (updatable SRS) simulation extractable? in *SCN*. Lecture Notes in Computer Science, vol. 13409 (Springer, Berlin, 2022), pp. 735–760
[27] A. Ghoshal, S. Tessaro, Tight state-restoration soundness in the algebraic group model, in T. Malkin, C. Peikert, editors, *CRYPTO 2021, Part III, Virtual Event, August 2021*. LNCS, vol. 12827 (Springer, Heidelberg, 2021), pp. 64–93
[28] M. Hoffmann, M. Klooß, A. Rupp, Efficient zero-knowledge arguments in the discrete log setting, revisited. in L. Cavallaro, J. Kinder, X. Wang, J. Katz, editors, *ACM CCS 2019, November 2019* (ACM Press, 2019), pp. 2093–2110
[29] T.G. Jones. *A Note on Sampling a Tape-File*. *Commun. ACM*, 5(6):343, 1962.
[30] J. Jaeger, S. Tessaro. Expected-time cryptography: generic techniques and applications to concrete soundness, in R. Pass, K. Pietrzak, editors, *TCC 2020, Part III*, *LNCS*, vol. 12552 (Springer, Heidelberg, 2020), pp. 414–443
[31] D.E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison-Wesley, Boston 1969.
[32] M. Maller, S. Bowe, M. Kohlweiss, S. Meiklejohn, Sonic: zero-knowledge SNARKs from linear-size universal and updatable structured reference strings, in L. Cavallaro, J. Kinder, X. Wang, J. Katz, editors, *ACM CCS, November 2019*, (ACM Press, 2019), pp. 2111–2128
[33] D. Pointcheval, J. Stern, Security proofs for signature schemes, in U.M. Maurer, editor, *EUROCRYPT'96*. vol. 1070, LNCS. (Springer, Heidelberg, 1996), pp. 387–398
[34] D. Unruh, Post-quantum security of Fiat–Shamir, in T. Takagi. T. Peyrin, editors, *ASIACRYPT 2017, Part I, December 2017*. LNCS, vol. 10624. (Springer, Heidelberg, 2017).
[35] J.S. Vitter, An efficient algorithm for sequential random sampling. *ACM Trans. Math. Softw.*, 13(1):58–67, 1987.
[36] D. Wikström, Special soundness revisited. Cryptology ePrint Archive, Report 2018/1157, 2018. https://eprint.iacr.org/2018/1157
[37] D. Wikström, Special soundness in the random oracle model. Cryptology ePrint Archive, Report 2021/1265, 2021. https://eprint.iacr.org/2021/1265.

[38] R.S. Wahby, I. Tzialla, A. Shelat, J. Thaler, M. Walfish, Doubly-efficient zkSNARKs without trusted setup, in *2018 IEEE Symposium on Security and Privacy, May 2018* (IEEE Computer Society Press, 2018), pp. 926–943