CHAPTER 6

6 Knowledge Soundness of Compressed Σ-Protocols

6.1 Introduction

In a compressed Σ-protocol for relation $\mathfrak{R}$ a prover aims to convince a verifier to know a witness $w \in \mathfrak{R}$ for some statement $x \in \{0,1\}^*$. A dishonest prover, without knowledge of a witness w, should not be able to convince a verifier. This property is called *knowledge soundness* and is formally captured by Definition 2.27. Knowledge soundness requires the existence of an extraction algorithm, called a knowledge extractor that, on input x and given oracle access to a prover $\mathcal{P}^*$, aims to output a witness $w \in \mathfrak{R}$.

Thus far, we have only shown compressed Σ-protocols to satisfy the weaker notion *special-soundness*. It is well known that 3-round k-out-of-N special-sound interactive proofs are knowledge sound with knowledge error $(k-1)/N$, i.e., for 3-round interactive proofs, special-soundness implies knowledge soundness. Further, the t-fold parallel repetition of a 3-round 2-out-of-N special-sound interactive proof is easily seen to decrease the knowledge error from $1/N$ down to $1/N^t$. Finally, the security loss of the Fiat-Shamir transformation of a 3-round interactive proof is known to be linear in the number of random oracle queries admitted to a prover attacking the considered non-interactive proof. However, for multi-round interactive proofs, the situation is significantly more complicated. In this chapter, we discuss knowledge soundness of certain (natural variations of) multi-round special-sound interactive proofs.

In Section 6.2, we explain the difficulties that arise when generalizing existing knowledge extractors for 3-round interactive proofs to $(2\mu+1)$-round interactive proofs.

In Section 6.3, we describe an extraction algorithm for special-sound multi-round interactive proofs that runs in *strict* polynomial time. The success probability of this extractor is not large enough to prove knowledge soundness; it only shows that a subclass of special-sound interactive proofs satisfies an alternative notion of knowledge soundness (Definition 2.31). It is not known how to increase the success probability of the extractor in strict polynomial time. In fact, unless one allows for smaller success probability, strict polynomial time extraction is impossible for nontrivial constant-round zero-knowledge proofs [BL02]. This section is based on the article [AC20], co-authored by Ronald Cramer.

In Section 6.4, we show that the success probability of the extraction algorithm can be increased if it is allowed to run in *expected* polynomial time. The resulting knowledge extractor shows that, also for multi-round interactive proofs, special-soundness *tightly* implies knowledge soundness. This section is based on the article [ACK21], co-authored by Ronald Cramer and Lisa Kohl.

In Section 6.5, we consider the t-fold parallel repetition of multi-round special-sound interactive proofs. In many occasions, the knowledge error κ of an interactive proof is not small enough and parallel repetition is used to decrease it. We show that, also for multi-round protocols, the t-fold parallel repetition of a special-sound interactive proof reduces the knowledge error from κ down to κ^t. This section is based on the article [AF22], co-authored by Serge Fehr.

In Section 6.6, we show that the security loss of the Fiat-Shamir transformation of a special-sound interactive proof is independent of the number of rounds. More precisely, we show that, similar to the 3-round case, the security loss is linear in the number of random oracle queries admitted to the prover $\mathcal{P}^*$ attacking the considered non-interactive protocol. This section is based on the article [AFK22], co-authored by Serge Fehr and Michael Klooß.

Table 6.1 summarizes the main properties, i.e., the efficiency and success probability, of the different knowledge extractors described in this chapter.

Protocol	Section	Number of $\mathcal{P}^*$-queries X	Success probability P
Π	6.3	$X \leq K$	$P \geq (\epsilon - \kappa)^K$
Π	6.4	$\mathbb{E}[X] \leq K$	$P \geq \epsilon - \kappa$
Π^t	6.5	$\mathbb{E}[X] \leq t \cdot 2^\mu \cdot K \leq t \cdot K^2$	$P \geq \frac{1}{2K}(\epsilon - \kappa^t)$
$\mathsf{FS}[\Pi]$	6.6	$\mathbb{E}[X] \leq K + (K-1)Q$	$P \geq \epsilon - (Q+1)\kappa$

Table 6.1: The efficiency and success probability of different knowledge extractors for (variations of) a $(k_1, \dots, k_\mu)$-special-sound interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$. Here, Π^t and $\mathsf{FS}[\Pi]$ denote the t-fold parallel repetition and the Fiat-Shamir transformation of Π. Moreover, $\epsilon = \epsilon(x, \mathcal{P}^*)$ is the success probability of the prover $\mathcal{P}^*$ attacking the considered protocol on statement x and κ is the knowledge error of Π. Finally, Q is the number of random-oracle queries admitted to a non-interactive prover attacking $\mathsf{FS}[\Pi]$ and $K = \prod_{i=1}^{\mu} k_i$.

6.2 The Knowledge Soundness Problem for Multi-Round Special-Sound Interactive Proofs

A 3-round public-coin interactive proof is said to be 2-special-sound if there exists an efficient algorithm that, on input a *colliding* pair of accepting transcripts (a, c, z) and (a, c', z'), i.e., with common first message a and distinct challenges $c \neq c'$, outputs a witness $w \in \mathfrak{R}(x)$ for the statement x. By contrast, knowledge soundness requires the existence of an extractor that is given oracle access to a prover; it should extract a witness by interacting with a prover in a black-box manner. In particular, a knowledge extractor does not receive protocol transcripts as input. It should either generate these transcripts or extract a witness by some other means. In order to prove that 2-special-soundness implies knowledge soundness, one must show how to *efficiently* output a colliding pair of accepting transcripts given only oracle access to a prover. By special-soundness a witness can then be extracted efficiently from this pair of transcripts. Together these two steps define a knowledge extractor.

In the theory of Σ-protocols, i.e., 3-round interactive proofs, it is well known that 2-out-of-N special-soundness implies knowledge soundness with knowledge error $1/N$, where N is the size of the challenge set. This can be shown by a heavy-row type approach [Dam10; HL10]. The alternative knowledge soundness notion of Definition 2.31, requiring a *strict* polynomial time extractor that is allowed to have somewhat smaller success probability, is also implied by special-soundness. In [Cra96], it is shown how this follows by an application of Jensen's inequality.

The knowledge error $1/N$ of a 2-out-of-N special-sound interactive proof equals the probability that the prover guessed the challenge correctly before receiving it from the verifier. For this reason, it corresponds to the success probability of a trivial cheating strategy admitted by typical special-sound interactive proofs. In particular, every 3-round interactive proof that is *special honest-verifier zero-knowledge* admits such a cheating strategy. For this reason, the knowledge error $1/N$ is the best one can hope for and the implication from 2-out-of-N special-soundness to knowledge soundness is *tight*.

Recently, and in particular for compressed Σ-protocols, a natural generalization of special-soundness has become relevant: $\mathbf{k}$-*out-of*-$\mathbf{N}$ *special-soundness*, where $\mathbf{k} = (k_1, \dots, k_\mu)$ and $\mathbf{N} = (N_1, \dots, N_\mu)$. This is a generalization in two ways: (1) from requiring a colliding pair of transcripts to requiring a k-collision of k transcripts, i.e., k accepting transcripts with common first message and pairwise distinct challenges, and (2) from 3-round interactive proofs with 1 challenge, sent from the verifier to the prover, to $(2\mu + 1)$-round interactive proofs with μ challenges.

Typical $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs admit a cheating strategy that succeeds if at least one of the μ random challenges c_i, received from the verifier, hits a certain set Γ_i of size $k_i - 1$ chosen by the dishonest prover. The success probability of this cheating strategy is

$$\mathrm{Er}(k_1, \dots, k_\mu; N_1, \dots, N_\mu) := 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right). \tag{6.1}$$

This cheating strategy is a generalization of the one for 2-out-of-N Σ-protocols, where a dishonest prover succeeds if it guesses the challenge correctly before receiving it. Indeed, the latter has a success probability $\mathrm{Er}(2, N) = 1/N$, which matches the knowledge error of a 2-out-of-N special-sound Σ-protocol. It is not unnatural to expect $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs to be knowledge sound with knowledge error $\mathrm{Er}(\mathbf{k}, \mathbf{N})$.

However, in particular due to the generalization to *multi-round* interactive proofs, the mentioned extractor analyses are no longer directly applicable. Hence, it is not straightforward to show that $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness also implies knowledge soundness if $\mu > 1$. For this reason, prior works resort to alternative arguments. Bootle et al. [BCC+16] give an asymptotic analysis of $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs. Their analysis is only applicable to interactive proofs with exponentially large challenge sets and it does not give an exact knowledge error. Wikström generalizes Bootle et al.'s analysis and constructs a knowledge extractor with a *configurable* knowledge error [Wik18]. At the cost of increasing the runtime, the knowledge error of this extractor can be configured to be arbitrarily close to

$$\sum_{i=1}^{\mu} \frac{k_i - 1}{N_i} > \mathrm{Er}(\mathbf{k}, \mathbf{N}) .$$

However, as the knowledge error moves closer to $\sum_{i=1}^{\mu} \frac{k_i-1}{N_i}$, the runtime of the extractor grows indefinitely. Moreover, also Wikström's analysis only applies to interactive proofs with exponentially large challenge sets.

As a consequence of these seemingly suboptimal extractor analyses, Hoffman, Klooß and Rupp raised the question whether there even exists an efficient knowledge extractor with knowledge error $\sum_{i=1}^{\mu} \frac{k_i-1}{N_i}$ [HKR19, Question D.4.]. Hence, at that time, it was unclear whether the knowledge error $\sum_{i=1}^{\mu} \frac{k_i-1}{N_i}$ was achievable, let alone the strictly smaller knowledge error $\mathrm{Er}(\mathbf{k}, \mathbf{N})$.

Recent works, while remaining non-tight, have improved the tightness and generalized the extraction to interactive proofs with smaller, i.e., not necessarily exponentially large, challenge sets [PLS19; JT20; AL21]. A common characteristic of all aforementioned approaches for analyzing multi-round knowledge extractors is the use of tail bounds, such as Markov's inequality, to bound the success probability and/or the (expected) runtime of the knowledge extractor. Using tail bounds, non-tightness appears to be unavoidable. In particular, Albrecht and Lai deemed a knowledge extractor with knowledge error $\sum_{i=1}^{\mu} \frac{k_i-1}{N_i}$ out of reach with current techniques [AL21].

6.3 A Partial Solution: Strict Polynomial Time Extraction

This section provides a partial solution towards our goal of proving that $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness implies knowledge soundness. More precisely, we show that every $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ admits a *strict* polynomial time extractor that, given a statement x and oracle access to a prover $\mathcal{P}^*$, succeeds in extracting a witness $w \in \mathfrak{R}(x)$ with probability at least

$$\big(\epsilon(x, \mathcal{P}^*) - \mathrm{Er}(\mathbf{k}; \mathbf{N})\big)^K ,$$

where $\epsilon(x, \mathcal{P}^*)$ is the success probability of $\mathcal{P}^*$ attacking Π on public input x, $\mathrm{Er}(\mathbf{k}; \mathbf{N})$ is the knowledge error as defined in Equation (6.1) and $K = \prod_{i=1}^{\mu} k_i$.

This is only a partial solution for two reasons. First, the standard notion of knowledge soundness requires an extractor with success probability proportional in $\epsilon(x, \mathcal{P}^*) - \mathrm{Er}(\mathbf{k}; \mathbf{N})$ instead of

$$\big(\epsilon(x, \mathcal{P}^*) - \mathrm{Er}(\mathbf{k}; \mathbf{N})\big)^K .$$

Therefore, this strict polynomial time extractor only shows that, for appropriate **k**, **k**-out-of-**N** special-sound interactive proofs satisfy the alternative notion of knowledge soundness of Definition 2.31. Second, the requirements of Definition 2.31 are only satisfied if $K = \prod_{i=1}^{\mu} k_i$ is constant in the size of the input x. In fact, since the success probability of the extractor degrades exponentially in K, this result only gives a meaningful security notion if K is constant. Unfortunately, for many protocols of interest this is not the case and K even grows superlinearly in $|x|$.

Hence, the extractor presented in this section shows that, for a subclass of interactive proofs, **k**-out-of-**N** special-soundness implies a meaningful, but alternative, notion of knowledge soundness. However, in contrast to the full solution that will be presented in Section 6.4, this extractor runs in *strict* polynomial time, which is known to be impossible if one insist on the standard notion of knowledge soundness [BL02].

6.3.1 Σ-Protocols

To simplify the exposition, we start with the simpler case of Σ-protocols, i.e., 3-round interactive proofs. The general case of multi-round interactive proofs will be treated in the subsequent section.

It is well known that a 2-out-of-N special-sound Σ-protocol admits a strict polynomial time extractor that succeeds with probability at least $(\epsilon(x, \mathcal{P}^*) - 1/N)^2$ [Cra96]. This result follows from an application of Jensen's inequality to the convex function $f(X) = X(X - 1/N)$. More precisely, Cramer defined the *collision-game* described below. This is essentially the game played by the knowledge extractor and Lemma 6.1 gives a lower bound for the probability of winning the game. Both the game and the lemma presented here are almost identical to the ones found in [Cra96]. We note that Bellare and Neven use similar techniques to prove the security of non-interactive protocols in the Fiat-Shamir paradigm [BN06].

Collision-Game. Consider a 0/1-matrix H with n rows and N columns. The rows correspond to the prover's randomness and the columns to the verifier's randomness. Therefore, every entry of H corresponds to a protocol transcript. An entry of the matrix is 1 if the transcript is accepting and 0 otherwise.

The game goes as follows. Select an entry of H uniformly at random. If this entry is a 1, select another entry of the same row uniformly at random. If this entry is again a 1, the game outputs success. If any of the selected entries equals a 0, the game is lost.

To bound the probability of winning the collision-game, Jensen's inequality is used, which states that, if X is a real-valued random variable and f is a continuous convex function defined on the support of X, it holds that

$$f\left(\mathbb{E}[X]\right) \leq \mathbb{E}[f(X)] .$$

Lemma 6.1 (Lemma 2.1 of [Cra96])**.** *Let H be a $0/1$-matrix with n rows and N columns, and let ϵ denote the fraction of 1-entries in H. Then the probability of winning the collision-game is greater than or equal to $\epsilon(\epsilon - 1/N)$.*

Proof. For $1 \leq i \leq n$, let ϵ_i denote the fraction of 1-entries in the i-th row. Clearly, the probability of winning the collision-game is equal to[1]

$$\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\left(\frac{N\epsilon_i - 1}{N-1}\right) = \frac{1}{n}\frac{N}{N-1}\sum_{i=1}^{n}\epsilon_i\left(\epsilon_i - \frac{1}{N}\right) \geq \frac{1}{n}\sum_{i=1}^{n}\epsilon_i\left(\epsilon_i - \frac{1}{N}\right).$$

To complete the proof, observe that $\mathbb{E}[\epsilon_i] = \epsilon$, put $f(x) = x(x - 1/N)$ on the interval $[0,1]$ and apply Jensen's inequality (using that f is a convex function). □

Using Lemma 6.1, it is straightforward to construct a strict polynomial time knowledge extractor that succeeds with probability at least $(\epsilon(x,\mathcal{P}^*) - 1/N)^2$ for 2-out-of-N special-sound Σ-protocols.

Instead, we show that the above argument can be adapted to show that, in order to satisfy the alternative notion of knowledge soundness (Definition 2.31), it is enough to consider *deterministic* provers $\mathcal{P}^*$. This observation simplifies the extractor analysis of interactive proofs and allows us to immediately handle the more general case of k-out-of-N special-sound Σ-protocols. In particular, the first message a sent by a deterministic prover is fixed, i.e., a does not vary over different invocations of the prover. Moreover, this observation will allow us to recursively generalize the analysis to *multi-round* interactive proofs.

Recall that for the standard definition of knowledge soundness (Definition 2.27), it is straightforward to see that one only needs to consider deterministic provers (Remark 2.3). The main reason is that $\epsilon(x,\mathcal{P}^*) - \kappa(|x|)$ is *linear* in $\mathcal{P}^*$'s success probability $\epsilon(x,\mathcal{P}^*)$ and linear functions commute with the expected value operator. This reasoning does not apply to the alternative notion of knowledge soundness, where extractors have success probability $\big(\epsilon(x,\mathcal{P}^*) - \kappa(|x|)\big)^c$ for some constant $c \geq 1$. However, by an appropriate application of Jensen's inequality, the argument can be adapted.

The following lemma shows that, also for the alternative notion of knowledge soundness, it is enough to consider deterministic provers.

Lemma 6.2 (Deterministic and Probabilistic Provers)**.** *Let $\Pi = (\mathcal{P},\mathcal{V})$ be an interactive proof for relation $\mathfrak{R}$, $\kappa\colon \mathbb{N} \to [0,1]$, $c \geq 1$ a constant and q a positive polynomial. Further, let $\mathcal{E}_{\mathsf{det}}$ be a knowledge extractor for Π that, given input x and oracle access to a* deterministic *prover $\mathcal{P}^*_{\mathsf{det}}$, runs in strict polynomial time and, if $\epsilon(x,\mathcal{P}^*_{\mathsf{det}}) \geq \kappa(|x|)$, succeeds in outputting a witness $w \in \mathfrak{R}(x)$ with probability*

$$\Pr\big((x;\mathcal{E}_{\mathsf{det}}^{\mathcal{P}^*_{\mathsf{det}}}(x)) \in \mathfrak{R}\big) \geq \frac{\big(\epsilon(x,\mathcal{P}^*_{\mathsf{det}}) - \kappa(|x|)\big)^c}{q(|x|)},$$

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

[1]This is minor correction of the original proof, which incorrectly states that the success probability is equal to $\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\left(\epsilon_i - \frac{1}{N}\right)$.

Then there exists a knowledge extractor $\mathcal{E}$ that, given input x and oracle access to a (possibly probabilistic) prover $\mathcal{P}^*$, runs in strict polynomial time and, if $\epsilon(x, \mathcal{P}^*) \geq \kappa(|x|)$, succeeds in outputting a witness $w \in \mathfrak{R}(x)$ with probability

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

Proof. Let $\mathcal{P}^*$ be an arbitrary randomized dishonest prover, and let $\mathcal{P}^*[r]$ be the deterministic prover obtained by fixing $\mathcal{P}^*$'s randomness to r. Then $\epsilon(x, \mathcal{P}^*) = \mathbb{E}_r[\epsilon(x, \mathcal{P}^*[r])]$, where $\mathbb{E}_r$ denotes the expectation over the random choice of r.

Given input x and oracle access to $\mathcal{P}^*$, the knowledge extractor $\mathcal{E}$ is declared to run $\mathcal{E}_{\mathsf{det}}^{\mathcal{P}^*[r]}(x)$ for a random choice of r. Clearly, $\mathcal{E}^{\mathcal{P}^*}(x)$ runs in strict polynomial time. So let us analyze its success probability.

The extractor $\mathcal{E}^{\mathcal{P}^*}(x)$ succeeds with probability

$$\begin{aligned}
\Pr\big((x; \mathcal{E}^{\mathcal{P}^*}(x)) \in \mathfrak{R}\big) &= \mathbb{E}_r\big[\Pr\big((x; \mathcal{E}_{\mathsf{det}}^{\mathcal{P}^*[r]}(x)) \in \mathfrak{R}\big)\big] \\
&\geq \mathbb{E}_r\left[\frac{\big(\epsilon(x, \mathcal{P}^*[r]) - \kappa(|x|)\big)^c}{q(|x|)}\right] \\
&\geq \frac{\mathbb{E}_r\big[f\big(\epsilon(x, \mathcal{P}^*[r])\big)\big]}{q(|x|)} ,
\end{aligned}$$

where the function f is defined as follows

$$f : \mathbb{R} \to \mathbb{R} : \quad \alpha \mapsto \begin{cases} \big(\alpha - \kappa(|x|)\big)^c, & \text{if } \alpha \geq \kappa(|x|) , \\ 0 , & \text{otherwise.} \end{cases} \tag{6.2}$$

Note that, in the above, x is an arbitrary but *fixed* statement.

It is easily seen that f is twice-differentiable and, for all $\alpha \in \mathbb{R} \setminus \{\kappa(|x|)\}$, $f''(\alpha) \geq 0$. Moreover, for $\alpha_0 = \kappa(|x|)$ it holds that

$$\lim_{\alpha \uparrow \alpha_0} \frac{f(\alpha) - f(\alpha_0)}{\alpha - \alpha_0} = 0 \leq \lim_{\alpha \downarrow \alpha_0} \frac{f(\alpha) - f(\alpha_0)}{\alpha - \alpha_0} .$$

Hence, f is a convex function.

Therefore, by Jensen's inequality, it follows that, if $\epsilon(x, \mathcal{P}^*) \geq \kappa(|x|)$, the extractor $\mathcal{E}^{\mathcal{P}^*}(x)$ succeeds with probability

$$\begin{aligned}
\Pr\big((x; \mathcal{E}^{\mathcal{P}^*}(x)) \in \mathfrak{R}\big) &\geq \frac{\mathbb{E}_r\big[f\big(\epsilon(x, \mathcal{P}[r]^*)\big)\big]}{q(|x|)} \\
&\geq \frac{f\big(\mathbb{E}_r\big[\epsilon(x, \mathcal{P}[r]^*)\big]\big)}{q(|x|)} \\
&= \frac{f\big(\epsilon(x, \mathcal{P}^*)\big)}{q(|x|)} \\
&= \frac{\big(\epsilon(x, \mathcal{P}^*) - \kappa(|x|)\big)^c}{q(|x|)} ,
\end{aligned}$$

which completes the proof.

□

Let us now return to the extractor analysis of k-out-of-N special-sound Σ-protocols. For multiple reasons, we will state and prove our core technical results in a more abstract language. One reason is that this allows us to focus on the important aspects. Another reason is that we will actually exploit the considered abstraction, and thus generalization, of the considered problem in the subsequent sections, where we consider parallel repetitions and Fiat-Shamir transformations. In particular, it allows us to unify the notation over the different sections of this chapter. The abstraction crucially depends on Lemma 6.2, showing that it is sufficient to consider deterministic provers $\mathcal{P}^*$.

In our abstraction, we consider an arbitrary function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$, $(c,y) \mapsto V(c,y)$, and an arbitrary (possibly probabilistic) algorithm $\mathcal{A}$ that takes as input an element $c \in \mathcal{C}$ and outputs a string $y \leftarrow \mathcal{A}(c)$. The *success probability* of $\mathcal{A}$ is then naturally defined as

$$\epsilon^V(\mathcal{A}) := \Pr\bigl(V(C, \mathcal{A}(C)) = 1\bigr) ,$$

where, here and below, the probability space is defined by means of the randomness of $\mathcal{A}$ and the random variable C being uniformly random in $\mathcal{C}$.

The obvious instantiation of $\mathcal{A}$ is given by a *deterministic* dishonest prover $\mathcal{P}^*$ attacking the considered k-out-of-N special-sound Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ on input x. More precisely, on input c, $\mathcal{A}$ runs $\mathcal{P}^*$, sending c as the challenge, and outputs $\mathcal{P}^*$'s (fixed) first message a and its response z, and the function V is defined as the verification check that $\mathcal{V}$ performs. In this instantiation

$$\epsilon^V(\mathcal{A}) = \epsilon(x, \mathcal{P}^*) .$$

Moreover, we point out that this instantiation gives rise to a deterministic $\mathcal{A}$. However, later on, when generalizing the approach to the parallel composition of interactive proofs (Section 6.5), it will be crucial that in our abstract treatment, $\mathcal{A}$ may be an arbitrary *randomized* algorithm that decides on its output y in a randomized manner given the input c, and that V is arbitrary. Moreover, the more general treatment of probabilistic $\mathcal{A}$ does not complicate the analysis. Therefore, the abstraction will not be restricted to deterministic $\mathcal{A}$.

Motivated by the k-out-of-N special-soundness of the considered Σ-protocol, given oracle access to $\mathcal{A}$, the goal of the extractor will be to find correct responses $y_1, \dots, y_k$ for k pairwise distinct challenges $c_1, \dots, c_k \in \mathcal{C}$, i.e., such that $V(c_i, y_i) = 1$ for all i. This extractor $\mathcal{E}$ is formally described in Figure 6.1 and Lemma 6.3 shows that it runs in *strict* polynomial time and succeeds with probability at least

$$\left(\epsilon^V(\mathcal{A}) - \frac{k-1}{N}\right)^k .$$

Lemma 6.3 (Strict Polynomial Time Extraction - Σ-Protocols)**.** *Let $k \in \mathbb{N}$, $\mathcal{C}$ a finite set with cardinality $N \geq k$ and let $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$. Then there exists an oracle algorithm $\mathcal{E}$, described in Figure 6.1, with the following properties: The* *algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$, requires at most k queries to $\mathcal{A}$ and, if $\epsilon^V(\mathcal{A}) \geq (k-1)/N$, with probability at least*

$$\left(\epsilon^V(\mathcal{A}) - \frac{k-1}{N}\right)^k ,$$

it outputs k pairs $(c_1,y_1),(c_2,y_2),\dots,(c_k,y_k) \in \mathcal{C} \times \{0,1\}^*$ with $V(c_i,y_i)=1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

Proof. The extractor $\mathcal{E}^{\mathcal{A}}$ is described in Figure 6.1 and proceeds as follows. It samples $c_1 \in \mathcal{C}$ uniformly at random and evaluates $y_1 \leftarrow \mathcal{A}(c_1)$. If $V(c_1,y_1)=0$, $\mathcal{E}^{\mathcal{A}}$ aborts. Otherwise, it samples $c_2 \in \mathcal{C} \setminus \{c_1\}$ uniformly at random and evaluates $y_2 \leftarrow \mathcal{A}(c_2)$. The extractor $\mathcal{E}^{\mathcal{A}}$ continues in this manner, until either it aborts, i.e., it finds a pair (c_i,y_i) with $V(c_i,y_i)=0$, or until it has extracted k pairs $(c_1,y_1),(c_2,y_2),\dots,(c_k,y_k) \in \mathcal{C} \times \{0,1\}^*$ with $V(c_i,y_i)=1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

Clearly, $\mathcal{E}^{\mathcal{A}}$ makes at most k queries to $\mathcal{A}$. Moreover, if $\epsilon^V(\mathcal{A}) \geq (k-1)/N$, its success probability is at least

$$\prod_{j=0}^{k-1} \frac{N}{N-j}\left(\epsilon^V(\mathcal{A}) - \frac{j}{N}\right) \geq \left(\epsilon^V(\mathcal{A}) - \frac{k-1}{N}\right)^k ,$$

which completes the proof of the lemma. □

Figure 6.1: Strict Polynomial Time Extractor $\mathcal{E}$.

Parameters: $k \in \mathbb{N}$.
Oracle access to: Algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$ and verification function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$.

- For $i \in \{1,\dots,k\}$:
  - Sample $c_i \in \mathcal{C} \setminus \{c_1,\dots,c_{i-1}\}$ uniformly at random and evaluate $y_i \leftarrow \mathcal{A}(c_i)$.
  - If $V(c_i,y_i)=0$, abort. Else, continue.

Output: If $\mathcal{A}$ has not aborted, output k pairs $(c_1,y_1),\dots,(c_k,y_k) \in \mathcal{C} \times \{0,1\}^*$, with $V(c_i,y_i)=1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

The following theorem is an immediate consequence of Lemma 6.2 and Lemma 6.3. It shows that, if k is constant in $|x|$, k-out-of-N special-sound Σ-protocols satisfy the alternative knowledge soundness notion of Definition 2.31.

Theorem 6.1 (Strict Polynomial Time Extraction for Σ-Protocols)**.** *Let $\Pi = (\mathcal{P},\mathcal{V})$ be a k-out-of-N special-sound Σ-protocol for relation $\mathfrak{R}$. Then there exists an extraction algorithm $\mathcal{E}$ with the following properties: The extractor $\mathcal{E}^{\mathcal{P}^*}(x)$, given input x and oracle access to a (potentially dishonest) prover $\mathcal{P}^*$, requires at* *most k queries to $\mathcal{P}^*$ and, if $\epsilon(x,\mathcal{P}^*) \geq (k-1)/N$, outputs a witness $w \in \mathfrak{R}(x)$ with probability*

$$\Pr\big((x;\mathcal{E}^{\mathcal{P}^*}(x)) \in \mathfrak{R}\big) \geq \left(\epsilon(x,\mathcal{P}^*) - \frac{k-1}{N}\right)^k ,$$

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

6.3.2 Multi-Round Interactive Proofs

Let us now move to multi-round interactive proofs and show that $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness, for $\mathbf{k} = (k_1,\dots,k_\mu)$ and $\mathbf{N} = (N_1,\dots,N_\mu)$, implies the existence of an extraction algorithm that requires at most $K = \prod_{i=1}^{\mu} k_i$ queries to $\mathcal{P}^*$ and succeeds in extracting a witness with probability at least

$$\left(\epsilon(x,\mathcal{P}^*) - \mathrm{Er}(\mathbf{k};\mathbf{N})\right)^K ,$$

where $\mathrm{Er}(\mathbf{k};\mathbf{N})$ is as defined in Equation 6.1. Note that $\mathrm{Er}(k;N) = (k-1)/N$, i.e., this is indeed a multi-round generalization of the result of Section 6.3.1.

As a multi-round generalization of the abstraction of the previous section, we now consider a (possibly randomized) algorithm $\mathcal{A}$ that takes as input a vector $(c_1,\dots,c_\mu) \in \mathcal{C}_1 \times \dots \times \mathcal{C}_\mu$ of challenges and outputs a string y, and we consider a function

$$V\colon \mathcal{C}_1 \times \dots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\} .$$

The obvious instantiation is a deterministic prover $\mathcal{P}^*$ attacking the considered multi-round interactive proof $\Pi = (\mathcal{P},\mathcal{V})$ on input x. Formally, on input $(c_1,\dots,c_\mu)$, $\mathcal{A}$ runs $\mathcal{P}^*$, sending c_1 in the first challenge round, c_2 in the second, etc., and eventually $\mathcal{A}$ outputs all of $\mathcal{P}^*$'s messages. Then the function V captures the verification procedure of $\mathcal{V}$, i.e., $V(c_1,\dots,c_\mu,y) = 1$ if and only if the corresponding transcript is accepting for statements x. As before, this instantiation actually results in a deterministic algorithm $\mathcal{A}$. However, our analysis also allows probabilistic instantiations of $\mathcal{A}$.

Syntactically identical to the previous section, the *success probability* of $\mathcal{A}$ is defined as

$$\epsilon^V(\mathcal{A}) := \Pr\big(V(C,\mathcal{A}(C)) = 1\big) ,$$

where $C = (C_1,\dots,C_\mu)$ is uniformly random in $\mathcal{C}_1 \times \dots \times \mathcal{C}_\mu$. However, here the goal of the extractor is different: the goal is to find correct responses for a $\mathbf{k}$-tree of challenge vectors (Definition 2.33). Note that, since the prover $\mathcal{P}^*$ is deterministic, any $\mathbf{k}$-tree of challenge vectors corresponds uniquely to a $\mathbf{k}$-tree of transcripts.

Towards constructing a knowledge extractor, we make the following observation. For notational convenience, let us write $\mathbf{k}_m = (1,\dots,1,k_{m+1},\dots,k_\mu)$ for all $1 \leq m \leq \mu$. Then, a $\mathbf{k}$-tree of challenge vectors has the following recursive nature:

- A $(1,\dots,1)$-tree of challenge vectors is simply a challenge vector $(c_1,\dots,c_\mu)$;
- A $\mathbf{k}_{m-1}$-tree of challenge vectors is a set of k_m $\mathbf{k}_m$-trees, where all $\prod_{i=m}^{\mu} k_i$ challenge vectors have the first $m-1$ coordinates in common.

The following lemma exploits the recursive nature of $\mathbf{k}$-trees of challenge vectors and shows the existence of an extraction algorithm with the desired runtime and success probability.

Lemma 6.4 (Strict Polynomial Time Extraction - Multi-Round Protocols)**.** *Let $\mathbf{k} = (k_1, \dots, k_\mu), \mathbf{N} = (N_1, \dots, N_\mu) \in \mathbb{N}^\mu$, $\mathcal{C}_1, \dots, \mathcal{C}_\mu$ finite sets with cardinality $|\mathcal{C}_i| = N_i \geq k_i$ and let $V \colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\}$. Then there exists an algorithm $\mathcal{E}$ with the following properties: The algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A} \colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \to \{0,1\}^*$, requires at most $K = \prod_{i=1}^{\mu} k_i$ queries to $\mathcal{A}$ and, if $\epsilon^V(\mathcal{A}) \geq \mathrm{Er}(\mathbf{k}; \mathbf{N})$, with probability at least*

$$\left(\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k}; \mathbf{N})\right)^K ,$$

it outputs K pairs $(\mathbf{c}_1, y_1), \dots, (\mathbf{c}_K, y_K) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^*$ with $V(\mathbf{c}_i, y_i) = 1$ for all i and such that the vectors $\mathbf{c}_i \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ form a $\mathbf{k}$-tree of challenge vectors, where we recall that

$$\mathrm{Er}(\mathbf{k}; \mathbf{N}) = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right) .$$

Proof. The extraction algorithm $\mathcal{E}$ is defined recursively. To this end, we write $\mathbf{k}_m = (1, \dots, 1, k_{m+1}, \dots, k_\mu)$ and $K_m = \prod_{i=m+1}^{\mu} k_i$ for all $0 \leq m \leq \mu$, with the understanding that $\mathbf{k}_\mu = (1, \dots, 1)$ and $K_\mu = 1$.

For all m and $\vec{c}_m = (c_1, \dots, c_m) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_m$, we let $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$ be the algorithm that, given oracle access to $\mathcal{A}$, aims to output K_m pairs $(\mathbf{c}_1, y_1), \dots, (\mathbf{c}_{K_m}, y_{K_m}) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^*$ with $V\big(\mathbf{c}_i, y_i\big) = 1$ for all i and such that the vectors $\mathbf{c}_i \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ form a $\mathbf{k}_m$-tree of challenge vectors with the first m coordinates equal to $\vec{c}_m = (c_1, \dots, c_m)$.

Let us now define the extraction algorithm $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$. For $m = \mu$ and $\vec{c}_\mu = (c_1, \dots, c_\mu)$, $\mathcal{E}_\mu^{\mathcal{A}}(\vec{c}_\mu)$ simply evaluates $y \leftarrow \mathcal{A}(\vec{c}_\mu)$. If $V(\vec{c}_\mu, y) = 1$, $\mathcal{E}_\mu^{\mathcal{A}}(\vec{c}_\mu)$ successfully outputs $(\vec{c}_\mu, y)$. In this case we write $\mathcal{E}_\mu^{\mathcal{A}}(\vec{c}_\mu) \neq \bot$.

For $m < \mu$ and $\vec{c}_m = (c_1, \dots, c_m)$, $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$ runs $\mathcal{E}_{m+1}^{\mathcal{A}}(\vec{c}_m, y_\ell)$ for $1 \leq \ell \leq k_{m+1}$ and $y_\ell \in \mathcal{C}_{m+1}$ sampled uniformly at random such that $y_i \neq y_j$ for all $i \neq j$. We say $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$ aborts if any of its $\mathcal{E}_{m+1}^{\mathcal{A}}$-invocations fails, i.e., if $\mathcal{E}_{m+1}^{\mathcal{A}}(\vec{c}_m, y_\ell) = \bot$ for some ℓ. If $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$ does not abort, it is easily seen that the k_{m+1} $\mathbf{k}_{m+1}$-trees, output by its $\mathcal{E}_{m+1}^{\mathcal{A}}$-invocations, form a $\mathbf{k}_m$-tree of challenge vectors.

The extraction algorithm $\mathcal{E}^{\mathcal{A}}$ simply runs $\mathcal{E}_0^{\mathcal{A}}$. Let us now analyze the expected number of $\mathcal{A}$-queries and success probability of $\mathcal{E}^{\mathcal{A}}$.

Expected Number of $\mathcal{A}$-Queries. By induction, it immediately follows that, for all m and $\vec{c}_m = (c_1, \dots, c_m)$, $\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m)$ makes at most $K_{m+1} = k_{m+1} \cdots k_\mu$ queries to $\mathcal{A}$. Hence, $\mathcal{E}^{\mathcal{A}}$ requires at most K queries to $\mathcal{A}$, which proves the claimed number of $\mathcal{A}$-queries.

Success Probability. For all m and $\vec{c}_m = (c_1, \dots, c_m)$, let

$$\epsilon(\vec{c}_m) = \Pr\big(V(C, \mathcal{A}(C)) = 1 \mid C_1 = c_1 \wedge \cdots \wedge C_m = c_m\big) ,$$

where $C = (C_1, \dots, C_\mu)$ is uniformly random in $\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$, i.e., $\epsilon(\vec{c}_m)$ denotes the success probability of $\mathcal{A}$ conditioned on the first m challenges being equal to $\vec{c}_m$.

Moreover, similar to the convex function of Equation 6.2, we let

$$f_m \colon \mathbb{R} \to \mathbb{R} : \quad \alpha \mapsto \begin{cases} \big(\alpha - \mathrm{Er}(\mathbf{k}_m; \mathbf{N})\big)^{K_m}, & \text{if } \alpha \geq \mathrm{Er}(\mathbf{k}_m; \mathbf{N}), \\ 0, & \text{otherwise.} \end{cases}$$

By induction, for all $0 \leq m \leq \mu$ and for all $\vec{c}_m$, we will show that

$$\Pr\big(\mathcal{E}_m^{\mathcal{A}}(\vec{c}_m) \neq \bot\big) \geq f_m\big(\epsilon(\vec{c}_m)\big). \tag{6.3}$$

It holds that $\mathbf{k}_\mu = (1, \dots, 1)$, $K_\mu = 1$ and $\mathrm{Er}(\mathbf{k}_\mu; \mathbf{N}) = 0$. Therefore,

$$\Pr\big(\mathcal{E}_\mu^{\mathcal{A}}(\vec{c}_\mu) \neq \bot\big) = \epsilon(\vec{c}_\mu) = f_\mu\big(\epsilon(\vec{c}_\mu)\big),$$

which proves the base case $m = \mu$.

So let us assume that the induction hypothesis of Equation 6.3 is satisfied for m and for all $\vec{c}_m \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_m$. Then, for all $\vec{c}_{m-1}$,

$$\begin{aligned} \Pr\big(\mathcal{E}_{m-1}^{\mathcal{A}}(\vec{c}_{m-1}) \neq \bot\big) &= \mathbb{E}_{y_1, \dots, y_{k_m}}\left[\prod_{\ell=1}^{k_m} \Pr\big(\mathcal{E}_m^{\mathcal{A}}(\vec{c}_{m-1}, y_\ell) \neq \bot\big)\right] \\ &\geq \mathbb{E}_{y_1, \dots, y_{k_m}}\left[\prod_{\ell=1}^{k_m} f_m\big(\epsilon(\vec{c}_{m-1}, y_\ell)\big)\right], \end{aligned}$$

where the expected value is over the random choices of pairwise distinct $y_1, \dots, y_{k_m} \in \mathcal{C}_m$.

By basic probability theory it now follows that

$$\mathbb{E}_{y_1, \dots, y_{k_m}}\left[\prod_{\ell=1}^{k_m} f_m\big(\epsilon(\vec{c}_{m-1}, y_\ell)\big)\right] = \prod_{\ell=1}^{k_m} \mathbb{E}_{y_\ell | y_1, \dots, y_{\ell-1}}\left[f_m\big(\epsilon(\vec{c}_{m-1}, y_\ell)\big)\right],$$

where the latter expected values are over the random choices of the variables $y_\ell \in \mathcal{C}_m \setminus \{y_1, \dots, y_{\ell-1}\}$, i.e., conditioned on the first $\ell - 1$ choices to be equal to $y_1, \dots, y_{\ell-1}$.

Using the fact that f_m is convex and applying Jensen's inequality shows that

$$\Pr\big(\mathcal{E}_{m-1}^{\mathcal{A}}(\vec{c}_{m-1}) \neq \bot\big) \geq \prod_{\ell=1}^{k_m} f_m\Big(\mathbb{E}_{y_\ell | y_1, \dots, y_{\ell-1}}\big[\epsilon(\vec{c}_{m-1}, y_\ell)\big]\Big).$$

Since y_ℓ is sampled uniformly at random from $\mathcal{C}_m \setminus \{y_1, \dots, y_{\ell-1}\}$, it holds that

$$\begin{aligned} \mathbb{E}_{y_\ell | y_1, \dots, y_{\ell-1}}\big[\epsilon(\vec{c}_{m-1}, y_\ell)\big] &= \frac{N_m \cdot \epsilon(\vec{c}_{m-1}) - \sum_{j=1}^{\ell-1} \epsilon(\vec{c}_{m-1}, y_j)}{N_m - \ell + 1} \\ &\geq \frac{N_m}{N_m - \ell + 1}\left(\epsilon(\vec{c}_{m-1}) - \frac{\ell - 1}{N_m}\right) \\ &= 1 - \frac{N_m}{N_m - \ell + 1}\big(1 - \epsilon(\vec{c}_{m-1})\big). \end{aligned}$$

Hence, since f_m is monotonically increasing,

$$\begin{aligned}\Pr\bigl(\mathcal{E}^{\mathcal{A}}_{m-1}(\vec{c}_{m-1}) \neq \bot\bigr) &\geq \prod_{\ell=1}^{k_m} f_m\left(1 - \frac{N_m}{N_m - \ell + 1}\left(1 - \epsilon(\vec{c}_{m-1})\right)\right) \\ &\geq f_m\left(1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr)\right)^{k_m}.\end{aligned}$$

To complete the proof, we must express this lower bound in terms of the function f_{m-1} instead of f_m. To this end, we first consider the case $\epsilon(\vec{c}_{m-1}) < \mathrm{Er}(\mathbf{k}_{m-1}; \mathbf{N})$. In this case

$$\begin{aligned}1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr) &< 1 - \frac{N_m}{N_m - k_m + 1}\left(1 - \mathrm{Er}(\mathbf{k}_{m-1}; \mathbf{N})\right) \\ &= \mathrm{Er}(\mathbf{k}_m; \mathbf{N}).\end{aligned}$$

Hence, in this case

$$f_m\left(1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr)\right)^{k_m} = f_{m-1}\bigl(\epsilon(\vec{c}_{m-1})\bigr) = 0.$$

So let us consider the other case, i.e., $\epsilon(\vec{c}_{m-1}) \geq \mathrm{Er}(\mathbf{k}_m; \mathbf{N})$. Then

$$\begin{aligned}&f_m\left(1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr)\right)^{k_m} \\ &= \left(1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr) - \mathrm{Er}(\mathbf{k}_m; \mathbf{N})\right)^{K_{m-1}} \\ &= \left(\left(\frac{N_m}{N_m - k_m + 1}\right)\left(\epsilon(\vec{c}_{m-1}) - 1 + \frac{N_m - k_m + 1}{N_m}\bigl(1 - \mathrm{Er}(\mathbf{k}_m; \mathbf{N})\bigr)\right)\right)^{K_{m-1}} \\ &= \left(\frac{N_m}{N_m - k_m + 1}\right)^{K_{m-1}} \cdot \Bigl(\epsilon(\vec{c}_{m-1}) - \mathrm{Er}(\mathbf{k}_{m-1}; \mathbf{N})\Bigr)^{K_{m-1}} \\ &\geq \Bigl(\epsilon(\vec{c}_{m-1}) - \mathrm{Er}(\mathbf{k}_{m-1}; \mathbf{N})\Bigr)^{K_{m-1}} \\ &= f_{m-1}\bigl(\epsilon(\vec{c}_{m-1})\bigr).\end{aligned}$$

Altogether it follows that, for all $\vec{c}_{m-1} \in \mathcal{C}_1 \times \cdots \mathcal{C}_{m-1}$,

$$f_m\left(1 - \frac{N_m}{N_m - k_m + 1}\bigl(1 - \epsilon(\vec{c}_{m-1})\bigr)\right)^{k_m} \geq f_{m-1}\bigl(\epsilon(\vec{c}_{m-1})\bigr),$$

and therefore,

$$\Pr\bigl(\mathcal{E}^{\mathcal{A}}_{m-1}(\vec{c}_{m-1}) \neq \bot\bigr) \geq f_{m-1}\bigl(\epsilon(\vec{c}_{m-1})\bigr),$$

which proves the induction hypothesis of Equation 6.3.

In particular, if $\epsilon^V(\mathcal{A}) \geq \mathrm{Er}(\mathbf{k};\mathbf{N})$,

$$\Pr\big(\mathcal{E}^{\mathcal{A}} \neq \bot\big) = \Pr\big(\mathcal{E}_0^{\mathcal{A}} \neq \bot\big) \geq \big(\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k};\mathbf{N})\big)^K \ ,$$

which completes the proof of the lemma. □

The following theorem is an immediate consequence of Lemma 6.4. It shows that, if $K = k_1 \dots k_\mu$ is constant in $|x|$, $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs satisfy the alternative knowledge soundness notion of Definition 2.31 with knowledge error $\mathrm{Er}(\mathbf{k};\mathbf{N})$. The knowledge error $\mathrm{Er}(\mathbf{k};\mathbf{N})$ is tight, since $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs typically admit a cheating strategy that succeeds with probability $\mathrm{Er}(\mathbf{k};\mathbf{N})$.

Theorem 6.2 (Strict Polynomial Time Extraction for Multi-Round Protocols)**.** *Let $\mathbf{k} = (k_1,\dots,k_\mu), \mathbf{N} = (N_1,\dots,N_\mu) \in \mathbb{N}^\mu$ and let $K = k_1 \dots k_\mu$. Let $\Pi = (\mathcal{P},\mathcal{V})$ be a $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proof for relation $\mathfrak{R}$. Then there exists an extraction algorithm $\mathcal{E}$ with the following properties: The extractor $\mathcal{E}^{\mathcal{P}^*}(x)$, given input x and oracle access to a (potentially dishonest) prover $\mathcal{P}^*$, requires at most K queries to $\mathcal{P}^*$ and, if $\epsilon(x,\mathcal{P}^*) \geq \mathrm{Er}(\mathbf{k};\mathbf{N})$, outputs a witness $w \in \mathfrak{R}(x)$ with probability*

$$\Pr\big((x;\mathcal{E}^{\mathcal{P}^*}(x)) \in \mathfrak{R}\big) \geq \big(\epsilon(x,\mathcal{P}^*) - \mathrm{Er}(\mathbf{k};\mathbf{N})\big)^K \ ,$$

where $\epsilon(x,\mathcal{P}^*) := \Pr\big((\mathcal{P}^*,\mathcal{V})(x) = \textsf{accept}\big)$ and

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1 - \prod_{i=1}^{\mu}\left(1 - \frac{k_i - 1}{N_i}\right).$$

6.4 A Complete Solution in Expected Polynomial Time

In this section, we present a complete solution to the knowledge soundness problem for $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs, i.e., we prove that $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness implies knowledge soundness with knowledge error $\mathrm{Er}(\mathbf{k},\mathbf{N})$. More precisely, towards satisfying Definition 2.27, we construct a knowledge extractor that, given a statement x and oracle access to a prover $\mathcal{P}^*$, runs in *expected* polynomial time and succeeds in extracting a witness $w \in \mathfrak{R}(x)$ with probability at least

$$\epsilon(x,\mathcal{P}^*) - \mathrm{Er}(\mathbf{k},\mathbf{N})\,,$$

where $\epsilon(x,\mathcal{P}^*)$ is the success probability of $\mathcal{P}^*$ on input x. Therefore, with respect to the partial solution of Section 6.3 we show that, at the cost of relaxing from *strict* to *expected* polynomial time extraction, the success probability can be increased from

$$\big(\epsilon(x,\mathcal{P}^*) - \mathrm{Er}(\mathbf{k},\mathbf{N})\big)^K \quad \text{to} \quad \epsilon(x,\mathcal{P}^*) - \mathrm{Er}(\mathbf{k},\mathbf{N})\,,$$

where $K = \prod_{i=1}^{\mu} k_i$. This shows that indeed $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness *tightly* implies knowledge soundness with knowledge error $\mathrm{Er}(\mathbf{k},\mathbf{N})$.

6.4.1 Σ-Protocols

As before, to simplify the exposition, we start with the simpler case of Σ-protocols, i.e., 3-round interactive proofs. Moreover, we use the same abstract notation, i.e., we consider an arbitrary algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$ and an arbitrary verification function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$. Recall that $\mathcal{A}$ has a naturally defined success probability

$$\epsilon^V(\mathcal{A}) := \Pr\big(V(C, \mathcal{A}(C)) = 1\big)\,,$$

where C is uniformly random in $\mathcal{C}$. The obvious instantiation of $\mathcal{A}$ is given by a *deterministic*[2] prover $\mathcal{P}^*$ attacking the considered k-out-of-N special-sound Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ on input x.

As before, given oracle access to $\mathcal{A}$, the goal is to find correct responses $y_1, \dots, y_k$ for k pairwise distinct challenges $c_1, \dots, c_k \in \mathcal{C}$, i.e., such that $V(c_i, y_i) = 1$ for all i. However, this time we follow a different approach. The first step of the extractor is the same as in Section 6.3.1, i.e., it samples a random challenge c_1 and evaluates $y_1 \leftarrow \mathcal{A}(c_1)$. If $V(c_1, y_1) = 0$, the extractor aborts. Otherwise, i.e., if $V(c_1, y_1) = 1$, the extractor samples challenges from $\mathcal{C} \setminus \{c_1\}$, without replacement, until either $k-1$ additional pairs $(c_2, y_2), \dots, (c_k, y_k)$, with $V(c_i, y_i) = 1$ for all i, have been found or until the entire challenge set $\mathcal{C}$ has been exhausted. This extraction algorithm is also described in Figure 6.2 and its properties are summarized in Lemma 6.5.

Recall that the strict polynomial-time extractor of Section 6.3.1 aborts if any pair (c, y) with $V(c, y) = 0$ is encountered. By contrast, if $V(c_1, y_1) = 1$, the expected polynomial time extractor described here continues searching until it has succeeded or until there are no more challenges to try. Lemma 6.5 shows that this adaptation increases the success probability from $(\epsilon^V(\mathcal{A}) - (k-1)/N)^k$ to $\epsilon^V(\mathcal{A}) - (k-1)/N$, where $N = |\mathcal{C}|$. The cost of this improvement is a degradation from strict to expected polynomial runtime. However, this is still sufficient for proving knowledge soundness.

The proof of the following lemma can be simplified by restricting to deterministic algorithms $\mathcal{A}$. This would still be sufficient for proving knowledge soundness. However, in the next section, for multi-round protocols, we will apply this lemma recursively and there it is crucial that $\mathcal{A}$ is allowed to be probabilistic.

Lemma 6.5 (Expected Polynomial Time Extraction - Σ-Protocols)**.** *Let $k \in \mathbb{N}$, $\mathcal{C}$ a finite set with cardinality $N \geq k$ and let $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$. Then there exists an oracle algorithm $\mathcal{E}$ with the following properties: The algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$, requires an expected number of at most k queries to $\mathcal{A}$ and with probability at least*

$$\frac{N}{N-k+1}\left(\epsilon^V(\mathcal{A}) - \frac{k-1}{N}\right),$$

it outputs k pairs $(c_1, y_1), (c_2, y_2), \dots, (c_k, y_k) \in \mathcal{C} \times \{0,1\}^*$ with $V(c_i, y_i) = 1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

[2]Recall that, in order to prove knowledge soundness, it is sufficient to consider deterministic provers (Remark 2.3).

Figure 6.2: Expected Polynomial Time Extractor $\mathcal{E}$.

Parameters: $k \in \mathbb{N}$.
Oracle access to: Algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$ and verification function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$.

- Sample $c_1 \in \mathcal{C}$ uniformly at random and evaluate $y_1 \leftarrow \mathcal{A}(c_1)$.
- If $V(c_1, y_1) = 0$, abort.
- Else, repeat
  - sample $c \in \mathcal{C} \setminus \{c_1\}$ uniformly at random (without replacement) and evaluate $y \leftarrow \mathcal{A}(c)$;

  until either $k-1$ additional pairs $(c_2, y_2), \dots, (c_k, y_k)$, with $V(c_i, y_i) = 1$ for all i, have been found or until all challenges $c \in \mathcal{C} \setminus \{c_1\}$ have been tried.

Output: In the former case, output k pairs $(c_1, y_1), \dots, (c_k, y_k) \in \mathcal{C} \times \{0,1\}^*$ with $V(c_i, y_i) = 1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

Proof. The extractor $\mathcal{E}^{\mathcal{A}}$, given oracle access to $\mathcal{A}$, is described in Figure 6.2 and proceeds as follows. It samples a random challenge c_1 and evaluates $y_1 \leftarrow \mathcal{A}(c_1)$. If $V(c_1, y_1) = 0$, the extractor aborts. Otherwise, if $V(c_1, y_1) = 1$, the extractor samples challenges from $\mathcal{C} \setminus \{c_1\}$, without replacement, until either $k-1$ additional pairs $(c_2, y_2), \dots, (c_k, y_k)$, with $V(c_i, y_i) = 1$ for all i, have been found or until the entire challenge set $\mathcal{C}$ has been exhausted.

We write C_1 for the random variable denoting the first challenge sampled by the extractor, i.e., C_1 is uniformly random in $\mathcal{C}$. Moreover, we write $\Gamma = 0$ and $\Gamma = 1$ for the events $V\big(C_1, \mathcal{A}(C_1)\big) = 0$ and $V\big(C_1, \mathcal{A}(C_1)\big) = 1$, respectively. In particular, note that

$$\epsilon^V(\mathcal{A}) = \Pr(\Gamma = 1)\,.$$

Let us now analyze the expected number of $\mathcal{A}$-queries and the success probability of the extractor $\mathcal{E}^{\mathcal{A}}$.

Expected Number of $\mathcal{A}$-Queries. Let T denote the number of $\mathcal{A}$-queries made by $\mathcal{E}^{\mathcal{A}}$. Moreover, let S denote the number of challenges $c \in \mathcal{C}$ for which $\mathcal{A}$ returns a correct response, i.e., $S = \big|\{c \in \mathcal{C} \mid V\big(c, \mathcal{A}(c)\big) = 1\}\big|$. Note that, since $\mathcal{A}$ is probabilistic, S is a random variable with support $\{0, \dots, N\}$.

Let us now assume that the first $\mathcal{A}$-query by $\mathcal{E}^{\mathcal{A}}$ is successful, i.e., $\Gamma = 1$. Then conditioned on $S = \ell > 0$, the remainder of the extraction algorithm can be modeled by a *negative hyper geometric distribution*; challenges are drawn (without replacement) from a set of size $N-1$ containing $\ell - 1$ correct responses.

Therefore, by Lemma 2.3,

$$\mathbb{E}[T \mid \Gamma = 1 \wedge S = \ell > 0] \leq k + (k-1)\frac{N-\ell}{\ell} = 1 + (k-1)\frac{N}{\ell}\,.$$

Moreover, $\Gamma = 0$ implies $T = 1$ and thus $\mathbb{E}[T \mid \Gamma = 0 \wedge S = \ell] = 1$. Hence, for all

$0 < \ell \leq N$,

$$\begin{aligned}
\mathbb{E}[T \mid S = \ell] &= \Pr\big(\Gamma = 0 \mid S = \ell\big) \cdot \mathbb{E}[T \mid \Gamma = 0 \wedge S = \ell] \\
&\qquad + \Pr\big(\Gamma = 1 \mid S = \ell\big) \cdot \mathbb{E}[T \mid \Gamma = 1 \wedge S = \ell] \\
&= \frac{N-\ell}{N} \cdot \mathbb{E}[T \mid \Gamma = 0 \wedge S = \ell] + \frac{\ell}{N} \cdot \mathbb{E}[T \mid \Gamma = 1 \wedge S = \ell] \\
&\leq \frac{N-\ell}{N} + \frac{\ell}{N} \cdot \left(1 + (k-1)\frac{N}{\ell}\right) \\
&= 1 + k - 1 = k\,.
\end{aligned}$$

Since $\mathbb{E}[T \mid S = 0] = 1$, it follows that $\mathbb{E}[T] \leq k$, which proves the claimed expected number of $\mathcal{A}$-queries made by $\mathcal{E}^{\mathcal{A}}$.

Success Probability. The extractor succeeds if $S \geq k$ and the first challenge returns a correct response, i.e.,

$$\begin{aligned}
\Pr\big(\mathcal{E}^{\mathcal{A}} \neq \bot\big) = \Pr\big(\Gamma = 1 \wedge S \geq k\big) &= \sum_{\ell=k}^{N} \Pr\big(S = \ell\big) \Pr\big(\Gamma = 1 \mid S = \ell\big) \\
&= \sum_{\ell=k}^{N} \Pr\big(S = \ell\big) \frac{\ell}{N}\,.
\end{aligned}$$

Now, for $\ell \leq N$, note that

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

Hence,

$$\begin{aligned}
\Pr\big(\mathcal{E}^{\mathcal{A}} \neq \bot\big) &\geq \sum_{\ell=k}^{N} \Pr\big(S = \ell\big) \frac{N}{N-k+1}\left(\frac{\ell}{N} - \frac{k-1}{N}\right) \\
&\geq \sum_{\ell=0}^{N} \Pr\big(S = \ell\big) \frac{N}{N-k+1}\left(\frac{\ell}{N} - \frac{k-1}{N}\right) \\
&= \frac{N}{N-k+1}\left(\Pr\big(\Gamma = 1\big) - \frac{k-1}{N}\right) \\
&= \frac{N}{N-k+1}\left(\epsilon^{V}(\mathcal{A}) - \frac{k-1}{N}\right),
\end{aligned}$$

which completes the proof of the lemma. □

From Lemma 6.5 it immediately follows that k-out-of-N special-soundness tightly implies knowledge soundness.

Theorem 6.3 (Knowledge Soundness of Σ-Protocols)**.** *Let* $\Pi = (\mathcal{P}, \mathcal{V})$ *be a* k*-out-of-*N *special-sound* Σ*-protocol for relation* $\mathfrak{R}$*. Then* Π *is knowledge sound with knowledge error* $\mathrm{Er}(k; N) = (k-1)/N$*.*

An alternative knowledge extractor for 2-out-of-N special-sound Σ-protocols, proving Theorem 6.3 for this special case, can be found in [HL10]. Their extractor follows a heavy-row type approach and is designed towards satisfying the equivalent, but different, knowledge soundness definition (Definition 2.28). Therefore, in order to compare the two approaches, one must perform a generic transformation [Gol04]. Concretely, towards satisfying Definition 2.28, our extractor can be repeated until it succeeds resulting in a knowledge extractor for 2-out-of-N special-sound Σ-protocols that, if $\epsilon^V(\mathcal{A}) > 1/N$, always succeeds and requires an expected number of at most

$$\frac{2}{\epsilon^V(\mathcal{A}) - 1/N}$$

queries to $\mathcal{A}$.

Our approach simplifies the extraction algorithm and its analysis. The crucial difference is that, instead of sampling challenges with replacement, our extractor samples new challenges *without* replacement. Most importantly, as we will show in the next section, our approach allows for a generalization to the multi-round case. By contrast, all known multi-round generalizations of the heavy-row approach of [HL10] result in suboptimal knowledge errors and expected runtimes.

6.4.2 Multi-Round Interactive Proofs

We are now ready to prove that $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound multi-round interactive proofs are indeed knowledge sound with knowledge error $\mathrm{Er}(\mathbf{k}; \mathbf{N})$. We use the same abstract notation as in Section 6.3.2, i.e., we consider an arbitrary probabilistic algorithm $\mathcal{A}\colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \to \{0,1\}^*$ and an arbitrary verification function

$$V\colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\}\,.$$

The obvious instantiation of $\mathcal{A}$ is given by a deterministic prover $\mathcal{P}^*$ attacking the considered interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ on input x. Recall that $\mathcal{A}$'s success probability is denoted as

$$\epsilon^V(\mathcal{A}) := \Pr\bigl(V(C, \mathcal{A}(C)) = 1\bigr)\,,$$

where $C = (C_1, \dots, C_\mu)$ is uniformly random in $\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$.

The goal of the extractor is, given oracle access to $\mathcal{A}$, to find correct responses for a $\mathbf{k}$-tree of challenge vectors (Definition 2.33). The following lemma shows the existence of an extractor with the desired properties. The extractor is a recursive application of the 3-round extractor of Lemma 6.5.

Lemma 6.6 (Expected Polynomial Time Extraction - Multi-Round Protocols)**.** *Let* $\mathbf{k} = (k_1, \dots, k_\mu), \mathbf{N} = (N_1, \dots, N_\mu) \in \mathbb{N}^\mu$*,* $\mathcal{C}_1, \dots, \mathcal{C}_\mu$ *finite sets with cardinality* $|\mathcal{C}_i| = N_i \geq k_i$ *and let* $V\colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\}$*. Then there exists an oracle algorithm* $\mathcal{E}$ *with the following properties: The algorithm* $\mathcal{E}^{\mathcal{A}}$*, given oracle* *access to a (probabilistic) algorithm* $\mathcal{A}\colon \mathcal{C}_1\times\cdots\times\mathcal{C}_\mu \to \{0,1\}^*$, *requires an expected number of at most* $K=\prod_{i=1}^{\mu} k_i$ *queries to* $\mathcal{A}$ *and with probability at least*

$$\frac{1}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})}\left(\epsilon^V(\mathcal{A})-\mathrm{Er}(\mathbf{k};\mathbf{N})\right) ,$$

it outputs K *pairs* $(\mathbf{c}_1,y_1),\dots,(\mathbf{c}_K,y_K)\in\mathcal{C}_1\times\cdots\times\mathcal{C}_\mu\times\{0,1\}^*$ *with* $V(\mathbf{c}_i,y_i)=1$ *for all* i *and such that the vectors* $\mathbf{c}_i\in\mathcal{C}_1\times\cdots\times\mathcal{C}_\mu$ *form a* $\mathbf{k}$*-tree of challenge vectors, where we recall that*

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1-\prod_{i=1}^{\mu}\left(1-\frac{k_i-1}{N_i}\right).$$

Proof. The proof goes by induction on μ. For $\mu=1$, the lemma directly follows from Lemma 6.5. So let $\mu>1$ and let us assume the lemma holds for $\mu=M$ and consider the case $\mu=M+1$.

For any $c\in\mathcal{C}_1$, let $\mathcal{A}_c$ be the algorithm that takes as input a vector $(c^2,\dots,c^\mu)\in\mathcal{C}_2\times\cdots\times\mathcal{C}_\mu$ and runs $\mathcal{A}(c,c^2,\dots,c^\mu)$. The function V_c is defined accordingly, i.e.,

$$V_c\colon \mathcal{C}_2\times\cdots\times\mathcal{C}_\mu\times\{0,1\}^*\to\{0,1\},\quad (\mathbf{c},y)\mapsto V(c,\mathbf{c},y)\,.$$

Moreover, let $\mathbf{k}'=(k_2,\dots,k_\mu),\mathbf{N}'=(N_2,\dots,N_\mu)\in\mathbb{N}^{\mu-1}$ and $K'=\prod_{i=2}^{\mu}k_i$.

By the induction hypothesis there exists an algorithm $\mathcal{E}_{\mu-1}^{\mathcal{A}_c}$ that, given oracle access to $\mathcal{A}_c$, aims to output a set $\mathcal{Y}$ of K' pairs $(\mathbf{c}_1,y_1),\dots,(\mathbf{c}_{K'},y_{K'})\in\mathcal{C}_2\times\cdots\times\mathcal{C}_\mu\times\{0,1\}^*$ with $V(c,\mathbf{c}_i,y_i)=1$ for all i such that the vectors $\mathbf{c}_i\in\mathcal{C}_2\times\cdots\times\mathcal{C}_\mu$ form a $\mathbf{k}'$-tree of challenge vectors. Moreover, $\mathcal{E}_{\mu-1}^{\mathcal{A}_c}$ requires an expected number of at most K' queries to $\mathcal{A}$ and succeeds with probability at least

$$\frac{1}{1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')}\left(\epsilon^{V_c}(\mathcal{A}_c)-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\right) .$$

We define $W\colon\mathcal{C}_1\times\{0,1\}^*\to\{0,1\}$, by setting $W(c,\mathcal{Y})=1$ if and only if $\mathcal{Y}$ is a set satisfying the above properties.

Now let $\mathcal{B}^{\mathcal{A}}\colon\mathcal{C}_1\to\{0,1\}^*$ be the algorithm that, given oracle access to $\mathcal{A}$, takes as input an element $c\in\mathcal{C}_1$ and runs $\mathcal{E}_{\mu-1}^{\mathcal{A}_c}$. By Lemma 6.5, there exists an expected polynomial time algorithm $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ that, given oracle access to $\mathcal{B}^{\mathcal{A}}$, aims to output k_1 pairs $(c_1,\mathcal{Y}_1),\dots,(c_{k_1},\mathcal{Y}_{k_1})\in\mathcal{C}_1\times\{0,1\}^*$ with $W(c_i,\mathcal{Y}_i)=1$ for all i and $c_i\neq c_j$ for all $i\neq j$. Clearly, the set of k_1 $\mathbf{k}'$-trees of challenge vectors forms a $\mathbf{k}$-tree. For this reason, the extractor $\mathcal{E}^{\mathcal{A}}$ is simply defined to run $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$. Note that, by the associativity of the composition of oracle algorithms, $\mathcal{E}^{\mathcal{A}}=\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}=(\mathcal{E}_1^{\mathcal{B}})^{\mathcal{A}}$ is indeed an oracle algorithm given oracle access to $\mathcal{A}$.

Let us now analyze the success probability and the expected number of $\mathcal{A}$-queries of the algorithm $\mathcal{E}^{\mathcal{A}}=\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$.

Success Probability. By Lemma 6.5, and the induction hypothesis, it follows that $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ succeeds with probability at least

$$\begin{aligned}
&\frac{N_1}{N_1-k_1+1}\left(\epsilon^W\big(\mathcal{B}^{\mathcal{A}}\big)-\frac{k_1-1}{N_1}\right)\\
&\quad=\frac{N_1}{N_1-k_1+1}\left(\mathbb{E}_c\big[\Pr\big(\mathcal{E}_{\mu-1}^{\mathcal{A}_c}\neq\bot\big)\big]-\frac{k_1-1}{N_1}\right)\\
&\quad\geq\frac{N_1}{N_1-k_1+1}\left(\mathbb{E}_c\left[\frac{1}{1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')}\left(\epsilon^{V_c}(\mathcal{A}_c)-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\right)\right]-\frac{k_1-1}{N_1}\right)\\
&\quad=\frac{N_1}{N_1-k_1+1}\left(\frac{1}{1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')}\left(\epsilon^{V}(\mathcal{A})-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\right)-\frac{k_1-1}{N_1}\right)\\
&\quad=\frac{1}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})}\left(\epsilon^{V}(\mathcal{A})-\mathrm{Er}(\mathbf{k}';\mathbf{N}')-\frac{k_1-1}{N_1}\big(1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\big)\right)\\
&\quad=\frac{1}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})}\left(\epsilon^{V}(\mathcal{A})-1+\frac{N_1-k_1+1}{N_1}\big(1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\big)\right)\\
&\quad=\frac{1}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})}\left(\epsilon^{V}(\mathcal{A})-\mathrm{Er}(\mathbf{k};\mathbf{N})\right)\,,
\end{aligned}$$

where we (twice) use the recursive relation

$$1-\mathrm{Er}(\mathbf{k};\mathbf{N})=\frac{N_1-k_1+1}{N_1}\left(1-\mathrm{Er}(\mathbf{k}';\mathbf{N}')\right)\,.$$

This shows that $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ has the desired success probability.

Expected Number of $\mathcal{A}$-Queries. By Lemma 6.5 it follows that $\mathcal{E}_{\mu-1}^{\mathcal{A}_c}$ requires an expected number of at most k_1 queries to $\mathcal{B}^{\mathcal{A}}$ for all c. Moreover, by the induction hypothesis, $\mathcal{B}^{\mathcal{A}}$ requires an expected number of at most K' queries to $\mathcal{A}$. Hence, $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ requires an expected number of at most K queries to $\mathcal{A}$, which completes the proof of the lemma. □

Remark 6.1. In the proof of Lemma 6.6, it is crucial that the algorithm $\mathcal{B}^{\mathcal{A}}$ is allowed to be *probabilistic*. For this reason, we did not restrict Lemma 6.5 to deterministic algorithms, even though this would have been sufficient for proving knowledge soundness of k-out-of-N special-sound Σ-protocols.

From Lemma 6.6 it immediately follows that, also for multi-round protocols, **k**-out-of-**N** special-soundness tightly implies knowledge soundness. This result is summarized in the following theorem.

Theorem 6.4 (Knowledge Soundness of Multi-Round Interactive Proofs)**.** *Let* $\Pi=(\mathcal{P},\mathcal{V})$ *be a* **k***-out-of-***N** *special-sound interactive proof for relation* $\mathfrak{R}$*, where* $\mathbf{k}=(k_1,\dots,k_\mu)$, $\mathbf{N}=(N_1,\dots,N_\mu)\in\mathbb{N}^\mu$*. Then* Π *is knowledge sound with knowledge error*

$$\mathrm{Er}(\mathbf{k};\mathbf{N})=1-\prod_{i=1}^{\mu}\left(1-\frac{k_i-1}{N_i}\right).$$

6.4.3 A Note on Witness Extended Emulation

A technical issue arises when using proofs of knowledge as sub-protocols in larger cryptographic protocols [GK96; Lin01; Lin03]. More precisely, to prove security of the compound protocol, a simulator is typically required to run the extractor of the proof of knowledge. However, the naive simulation approach does not necessarily run in polynomial time. To this end, Lindell defined the notion of *witness-extended emulation*, capturing precisely the properties required when proofs of knowledge are used as sub-protocols [Lin01; Lin03]. Moreover, he showed that any proof of knowledge, with negligible knowledge error, has witness-extended emulation, thereby solving this technical issue for all proofs of knowledge at once. Hence, from our extraction analysis it follows that any $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proof has witness-extended emulation if $\mathrm{Er}(\mathbf{k}, \mathbf{N})$ is negligible.

The first multi-round extractor analysis for $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs considered witness emulation directly [BCC+16], i.e., it did not show that $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness implies knowledge soundness, but merely that it implies witness extended emulation. In particular, their analysis does not provide a concrete knowledge error and only applies to protocols with exponentially large challenge sets.

6.5 Solving the Parallel Repetition Problem

In certain occasions, the knowledge error of a “basic” proof of knowledge (and thereby the cheating probability of a dishonest prover) is not small enough, and thus needs to be reduced. In particular, this is the case for lattice-based proofs of knowledge (PoKs), where typically challenge sets are only of polynomial size resulting in nonnegligible knowledge errors [LS18; ACX21]. Reducing the knowledge error can be done generically by repeating the PoK. Indeed, repeating a PoK t times *sequentially*, i.e., one after the other, is known to reduce the knowledge error from κ down to κ^t [Gol01]. However, this approach also increases the number of communication rounds by a factor t. This is often undesirable, and sometimes even insufficient, e.g., because the security loss of the Fiat-Shamir transformation, transforming interactive into non-interactive protocols, is oftentimes exponential in the number of rounds (see Section 6.6).

Therefore, it is much more attractive to try to reduce the knowledge error by *parallel* repetition. However, analyzing parallel repetitions is significantly more complicated than analyzing sequential repetitions, because a dishonest prover does not have to treat all t parallel instances independently, i.e., a message corresponding to a specific instance may depend on the messages and challenges of the other parallel instances. In fact, it is not true in general that the t-fold parallel repetition decreases the knowledge error from κ down to κ^t; there even exist interactive arguments for which parallel repetition does not decrease the success probability of a dishonest prover at all [BIN97; PW07].

For this reason, parallel repetition of interactive proofs has been studied extensively, but mainly in the context of decreasing the *soundness error* [HPW+10; CL10; CP15]. However, knowledge soundness is a strictly stronger requirement than soundness; there exist interactive proofs that are sound but not knowledge sound. More precisely, proving the existence of an efficient knowledge extractor is a much more delicate task than proving that the verifier is unlikely to accept a false statement.

In the special case of 2-out-of-N special-sound interactive proofs such a parallel repetition is much easier to analyze: the t-fold parallel repetition of a 2-special-sound interactive proof with challenge space of cardinality N is again 2-special-sound, but now with a challenge space of size N^t, and so knowledge-soundness with knowledge error $1/N^t$ follows immediately from the generic reduction of Theorem 6.3. Unfortunately, this reasoning does not extend to k-out-of-N special-sound interactive proofs with $k > 2$: even though we still have that the t-fold parallel repetition of a k-out-of-N special-sound interactive proof is ℓ-out-of-N^t special-sound, but now with $\ell = (k-1)^t + 1$, this large increase in the special-soundness parameter ℓ renders the extractor, obtained via the generic reduction, inefficient. More precisely, the runtime of an ℓ-out-of-N^t special-sound interactive proof scales linearly in ℓ, and therefore exponentially in t for $\ell = (k-1)^t + 1$, unless $k = 2$. In case of multi-round interactive proofs, it is not even clear that the t-fold parallel repetition of a $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound $(2\mu + 1)$-round interactive proof satisfies any meaningful notion of special-soundness.

We consider parallel repetition of interactive proofs in the context of decreasing the *knowledge error*. In Section 6.5.1, we show, based on a result from [CP15], that the t-fold parallel repetition of any *public-coin* interactive proof reduces the knowledge error from κ down to $\kappa^t + \nu$ for any noticeable term ν. This generic result is tight, since there are interactive proofs for which parallel repetition does not allow the knowledge error to be reduced down to a negligible function [DJM+12]. However, it is also suboptimal in that, when applied to a k-out-of-N special-sound protocol for instance, it does not give the knowledge error $\mathrm{Er}(k; N)^t$ that one expects (and that one should get when $k = 2$) and, worse, the knowledge error remains nonnegligible.

For this reason, in Section 6.5.2, we restrict the analysis to k-out-of-N special-sound Σ-protocols, i.e., 3-round interactive proofs, and derive a *strong* parallel repetition result. Here, as usual in the general context of parallel repetition, the term “strong” means that the figure of merit κ, here the knowledge error, drops from κ to κ^t under a t-fold parallel repetition. In Section 6.5.3, we generalize this result to $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound *multi-round* interactive proofs. Finally, in Section 6.5.4, we consider the more general case of s-out-of-t threshold parallel repetition, where the verifier accepts if s-out-of-t instantiations of the basic interactive proof are accepting. Threshold parallel repetition allows both the knowledge and the completeness error to be reduced simultaneously.

6.5.1 A Generic but Suboptimal Solution

In this section, we establish a *weak* parallel repetition theorem. We write $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ for the t-fold parallel repetition of an interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$, which runs t instances of Π in parallel and the verifier $\mathcal{V}^t$ accepts if all the parallel instances are accepted. Then we show that, if Π is public-coin and knowledge sound with knowledge error κ, Π^t has knowledge error $\kappa^t + \nu$ for any noticeable ν. The result is weak in that it does not reduce the knowledge error from κ down to κ^t. However, it is generically applicable to any *public-coin* interactive proof.

Our main building block is a result by Chung and Pass [CP15] summarized in Theorem 6.5. This theorem shows the existence of a prover $\mathfrak{P}$ that, given input x and oracle access to a dishonest prover $\mathcal{P}^*$ attacking interactive proof $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$, succeeds in convincing $\mathcal{V}$ with probability approximately $\epsilon(x, \mathcal{P}^*)^{1/t}$, where $\epsilon(x, \mathcal{P}^*)$ is the probability that the prover $\mathcal{P}^*$ successfully convinces verifier $\mathcal{V}^t$ on input x. This theorem immediately shows that the t-fold parallel repetition reduces the soundness error from σ down to approximately σ^t. Subsequently, in Theorem 6.6, we show how this result can be used to derive our parallel repetition theorem for reducing the knowledge error instead of the soundness error.

Theorem 6.5 (Theorem 2 of [CP15])**.** *Let* $\Pi = (\mathcal{P}, \mathcal{V})$ *be a public-coin interactive proof for relation* $\mathfrak{R}$*. Let* $t \in \mathbb{N}$*, and let* $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ *be the t-fold parallel repetition of* Π*. Then there exists an oracle algorithm* $\mathfrak{P}^{(\cdot)}$ *such that for every* $\xi, \delta \colon \{0,1\}^* \to (0,1)$*, every* $x \in \{0,1\}^*$*, and every PPT prover* $\mathcal{P}^*$*, it holds that if*

$$\Pr\big((\mathcal{P}^*, \mathcal{V}^t)(x) = \textsf{accept}\big) \geq \underbrace{(1 + \xi(x)) \cdot \delta(x)^t}_{\epsilon(x) :=},$$

then

$$\Pr\Big((\mathfrak{P}^{\mathcal{P}^*}, \mathcal{V})(x) = \textsf{accept}\Big) \geq \delta(x).$$

Furthermore, $\mathfrak{P}^{\mathcal{P}^*}$ *runs in time* $\mathrm{poly}\big(|x|, t, \xi(x)^{-1}, \epsilon(x)^{-1}, (1 - \delta(x))^{-1}\big)$.

Theorem 6.5 was actually established specifically in the context of decreasing the soundness error of *computationally sound* interactive proofs. Recall that computational soundness only requires the success probability of *computationally bounded* dishonest provers to be smaller than the soundness error. For this reason, in contrast to the case of unconditional soundness, analyzing the parallel repetition of computationally sound interactive proofs is significantly more complicated. More precisely, from Theorem 6.5 it follows by contraposition that parallel repetition decreases the soundness error; given a prover $\mathcal{P}^*$ attacking the parallel repetition Π^t with success probability ϵ, an oracle prover $\mathfrak{P}^{(\cdot)}$ attacking the basic interactive proof Π with success probability approximately $\epsilon^{1/t}$ is constructed. Applying this argument in the context of computational soundness requires the oracle prover $\mathfrak{P}^{(\cdot)}$ to be *efficient*. In the context of unconditional soundness the oracle prover $\mathfrak{P}^{(\cdot)}$ is not required to be efficient. In fact, it is well known that the t-fold parallel repetition of an unconditionally sound public-coin interactive proof decreases the soundness error σ down to σ^t [BGG90; Gol98], i.e., for these protocols there exists a *strong* parallel repetition result.

By contrast, both the unconditional and computational variant of *knowledge soundness* require the existence of an *efficient* extractor. Therefore, restricting to either of the two variations does not simplify the analysis. However, in the following theorem we show that, using the above oracle prover $\mathfrak{P}^{(\cdot)}$, a knowledge extractor for the parallel repetition Π^t can be constructed. The extractor invokes $\mathfrak{P}^{(\cdot)}$ a polynomial number of times and is therefore efficient as long as $\mathfrak{P}^{(\cdot)}$ is efficient. Altogether, Theorem 6.6 shows that t-fold parallel repetition decreases the knowledge error from κ down to $\kappa^t + \nu$ for any noticeable ν. However, we cannot show that Π^t has negligible knowledge error for any fixed negligible function, because the running time of $\mathfrak{P}^{\mathcal{P}^*}$ scales with $\epsilon(x, \mathcal{P}^*)^{-1}$.

While it might seem that this barrier is rather an artifact of the proof technique of [CP15] on which we build, it was shown by [DJM+12] that Theorem 6.5 is tight when considering soundness amplification of interactive proofs in general. More precisely, based on some cryptographic assumptions they showed that, for some protocols, parallel repetition does not amplify security beyond negligible, meaning that for any negligible function η one can find an instantiation that when starting with nonnegligible soundness error, the protocol can always be broken with probability $\eta(x)$, no matter how many parallel repetitions one runs.

Theorem 6.6 (Generic Parallel Repetition Theorem). *Let* $\Pi = (\mathcal{P}, \mathcal{V})$ *be a public-coin interactive proof for relation* $\mathfrak{R}$ *that is knowledge sound with knowledge error* $\kappa \colon \mathbb{N} \to [0,1]$. *Let* $\nu \colon \mathbb{N} \to (0,1)$ *be an arbitrary noticeable function. Then, the t-fold parallel repetition* $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ *of* Π *is knowledge sound with knowledge error* $\kappa' = \kappa^t + \nu$.

Proof. We construct a knowledge extractor $\mathcal{E}^t$ for $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ as follows. Let $\mathcal{P}^*$ be some (potentially dishonest) prover attacking Π with success probability $\epsilon(x, P^*)$ on input x. Let $\xi \colon \mathbb{N} \to (0,1)$ be such that $\xi(n) = \nu(n)/\kappa(n)^t$ for all $n \in \mathbb{N}$. Then, by Theorem 6.5, there exists an oracle prover $\mathfrak{P}^{(\cdot)}$ such that

$$\epsilon(x, \mathfrak{P}^{\mathcal{P}^*}) = \Pr\big((\mathfrak{P}^{\mathcal{P}^*}, \mathcal{V})(x) = \mathsf{accept}\big) \geq \delta(x)\,,$$

where

$$\delta(x) = \left(\frac{\epsilon(x, \mathcal{P}^*)}{1 + \xi(|x|)}\right)^{1/t}.$$

By assumption $\Pi = (\mathcal{P}, \mathcal{V})$ is knowledge sound with knowledge error κ and, therefore, there exists a knowledge extractor $\mathcal{E}$ for Π. We define $\mathcal{E}^t$ as the algorithm that executes the knowledge extractor $\mathcal{E}$ on the prover $\mathfrak{P}^{\mathcal{P}^*}$.

Let us now analyze the expected runtime and success probability of extractor $\mathcal{E}^t$ for interactive proof Π^t. Recall that, in order to prove knowledge soundness with knowledge error $\kappa'(|x|)$, it is enough to consider statements $x \in \{0,1\}^*$ with $\epsilon(x, \mathcal{P}^*) > \kappa'(|x|)$ (Remark 2.4). Therefore, it is left to show that the following holds:

Claim. If $\epsilon(x, \mathcal{P}^*) > \kappa'(|x|)$, then the extractor $\mathcal{E}^t$ as defined above runs in an expected polynomial number of steps and there exists a positive polynomial q such that $\mathcal{E}^t$ is successful with probability at least $(\epsilon(x, \mathcal{P}^*) - \kappa'(|x|))/q(|x|)$.

Expected Runtime. We start proving the claim by showing that $\mathfrak{P}^{\mathcal{P}^*}$ runs in an expected polynomial number of steps. By Theorem 6.5, we have that the runtime of $\mathfrak{P}^{\mathcal{P}^*}$ is in $\mathrm{poly}(|x|, t, \xi(|x|)^{-1}, \epsilon(x, \mathcal{P}^*)^{-1}, (1 - \delta(x))^{-1})$. It holds that

$$\xi(|x|) = \nu(|x|)/\kappa(|x|)^t \geq \nu(|x|)$$

and $\epsilon(x, \mathcal{P}^*) > \kappa'(|x|) \geq \nu(|x|)$ and therefore also $\xi(|x|)^{-1}, \epsilon(x, \mathcal{P}^*)^{-1} \leq \mathrm{poly}(|x|)$. It is left to show that $1 - \delta(x)$ is noticeable. Via the Taylor approximation of the function $f(a) = a^{1/t}$ in $a = 1$, we obtain

$$\delta(x) = \left(\frac{\epsilon(x,\mathcal{P}^*)}{1+\xi(|x|)}\right)^{1/t} \leq 1 - \frac{1}{t}\left(1 - \frac{\epsilon(x,\mathcal{P}^*)}{1+\xi(|x|)}\right).$$

Therefore, we also have

$$1-\delta(x) \geq \frac{1}{t}\left(1 - \frac{\epsilon(x,\mathcal{P}^*)}{1+\xi(|x|)}\right) = \frac{1}{t}\left(\frac{1+\xi(|x|)-\epsilon(x,\mathcal{P}^*)}{1+\xi(|x|)}\right) \overset{\xi,\epsilon\leq 1}{\geq} \frac{\xi(|x|)}{2t} \geq \frac{\nu(|x|)}{2t},$$

as required.

Next, note that if $\epsilon(x,\mathcal{P}^*) > \kappa'(|x|)$, then $\delta(x) > \kappa(|x|)$. This is a simple consequence of the definition of $\xi(|x|)$ and $\delta(x)$, because

$$\epsilon(x,\mathcal{P}^*) > \kappa(|x|)^t + \nu(|x|) = \kappa(|x|)^t\big(1+\xi(|x|)\big)$$

implies $\delta(x) = (\epsilon(x,\mathcal{P}^*)/(1+\xi(|x|))^{1/t} > \kappa(|x|)$ as required.

Altogether, this shows that if $\epsilon(x,\mathcal{P}^*) > \kappa'(|x|)$, then $\mathcal{E}^t$ runs in an expected polynomial number of steps.

Success Probability. Let us now consider the success probability of $\mathcal{E}^t$. By definition of the knowledge extractor $\mathcal{E}$, there exists a positive polynomial p such that $\mathcal{E}^t$ outputs a witness $w \in \mathfrak{R}(x)$ with probability at least

$$\frac{\delta(x)-\kappa(|x|)}{p(|x|)}.$$

Therefore, it is left to show that if $\epsilon(x,\mathcal{P}^*) > \kappa'(|x|)$, there exists a positive polynomial q such that

$$\frac{\delta(x)-\kappa(|x|)}{p(|x|)} \geq \frac{\epsilon(x,\mathcal{P}^*)-\kappa(|x|)^t-\nu(|x|)}{q(|x|)}.$$

To express the success probability of $\mathcal{E}^t$ in terms of $\epsilon(x,\mathcal{P}^*)$, let us define the functions $f(a) = t(a^{1/t} - b)$ and $g(a) = a - b^t$, for $b \in [0,1]$. Observe that $f(a)$ is concave for $a \geq 0$. Moreover, $f(b^t) = g(b^t) = 0$ and $f(1) = t(1-b) \geq (1-b)\sum_{i=0}^{t-1} b^i = g(1)$. Hence $\max(f(a),0) \geq g(a)$ for all $a \in [0,1]$.

From this inequality we have that whenever $\delta(x) > \kappa(|x|)$, it holds that

$$\begin{aligned}
\delta(x)-\kappa(|x|) &= \max(\delta(x)-\kappa(|x|),0)\\
&= \max\left(\left(\frac{\epsilon(x,\mathcal{P}^*)}{(1+\xi(|x|))}\right)^{1/t} - \kappa(|x|), 0\right)\\
&\geq \frac{1}{t}\left(\frac{\epsilon(x,\mathcal{P}^*)}{(1+\xi(|x|))} - \kappa(|x|)^t\right)\\
&= \frac{1}{t(1+\xi(|x|))}\big(\epsilon(x,\mathcal{P}^*) - (1+\xi(|x|))\kappa(|x|)^t\big)\\
&\geq \frac{1}{2t}\big(\epsilon(x,\mathcal{P}^*) - \kappa(|x|)^t - \nu(|x|)\big).
\end{aligned}$$

Thus, choosing $q(|x|) = 2t \cdot p(|x|)$ yields the desired result, which proves the claim and completes the proof of the theorem. □

Remark 6.2. Let M be the total size of the challenge set, i.e., $M = \prod_{i=1}^{\mu} |\mathcal{C}_i|$ where the i^{th} challenge is sampled from challenge set $\mathcal{C}_i$. If M is polynomial in the size of the input x, the analysis can be simplified significantly. In this case the knowledge extractor can query all possible challenges and still run in polynomial time. A parallel repetition theorem then follows by a simple counting argument. This is the approach in the analysis of the 5-round $(2,2)$-out-of-$(q,2)$ special-sound signature scheme MQDSS [SSH11; CHR+16]. It is much more challenging to construct efficient knowledge extractors when M is not polynomial in $|x|$.

6.5.2 Parallel Repetition of k-out-of-n Special-Sound Σ-Protocols

Let us now restrict ourselves to *special-sound* interactive proofs. To simplify the exposition, we start with the simpler case of Σ-protocols; the general case of multi-round protocols will then be treated in the subsequent section. Thus, for the remainder of this section, we consider a k-out-of-N special-sound interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ with challenge set $\mathcal{C}$ of cardinality $N \geq k$. We have seen in Section 6.4 that Π is knowledge sound with knowledge error $\mathrm{Er}(k; N) = (k-1)/N$. In this section, we prove that the t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ of Π is then again knowledge sound, but now with knowledge error $\mathrm{Er}(k; N)^t$, which is optimal. Thus, we show what is sometimes referred to as *strong* parallel repetition, meaning that the figure of merit decreases with power t under parallel repetition. This is well known to hold for special-sound Σ-protocols, i.e., for $k = 2$, but was open for general k.

The standard way to reason about parallel repetition for the special case $k = 2$ uses the fact that Π^t is ℓ-out-of-N^t special-sound with $\ell = (k-1)^t + 1$. However, this reasoning does not apply in general, because ℓ grows exponentially in t for $k > 2$. Instead, our result crucially depends on the fact that Π^t is the t-fold parallel repetition of a k-out-of-N special-sound protocol Π^t. In this section, we first construct a novel extraction algorithm for k-out-of-N special-sound interactive proofs Π, thereby reproving that k-out-of-N special-soundness implies knowledge soundness (Theorem 6.3). Subsequently, we show how this extraction algorithm can be used to deduce a strong parallel repetition result for Π^t. In Section 6.5.3, we then extend our results to multi-round interactive proofs.

On a high level, the crucial ingredient in our analyses is to introduce and work with a more "fine-grained" notion of success probability of a dishonest prover, as explained below.

Knowledge Soundness of a Single Invocation

Consider a dishonest deterministic prover $\mathcal{P}^*$ attacking the considered k-out-of-N special-sound interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ on public input x. The goal of the extractor is to run $\mathcal{P}^*$ sufficiently many times so as to obtain k correct answers $z_1, \dots, z_k$ for k pairwise distinct challenges $c_1, \dots, c_k \in \mathcal{C}$. By the special-soundness property, a witness $w \in \mathfrak{R}(x)$ can be computed efficiently from the resulting set of protocol transcripts. Recall that, without loss of generality, we may assume $\mathcal{P}^*$ to be deterministic and therefore its first message a to be fixed (Remark 2.3).

The crucial question is how often $\mathcal{P}^*$ needs to be invoked, and thus what is the (expected) running time of the extractor. Alternatively, towards satisfying Definition 2.27, we would like to have an extractor that runs in a fixed (expected) polynomial time, but may fail with some probability. It is quite clear that in both cases the figure of merit (i.e., the running time in the former and the success probability in the latter) depends on the *success probability* $\epsilon(x, \mathcal{P}^*)$ of $\mathcal{P}^*$ on input x; for instance, if $\epsilon(x, \mathcal{P}^*)$ is below the knowledge error $\mathrm{Er}(k; N)$ then we cannot expect extraction to work in general. However, a crucial observation is that for a given dishonest prover $\mathcal{P}^*$, its success probability $\epsilon(x, \mathcal{P}^*)$ does actually not characterize (very well) whether extraction is possible or not: if in a special-sound Σ-protocol $\mathcal{P}^*$ provides the correct response with probability $\epsilon(x, \mathcal{P}^*)$ (and fails to do so with probability $1 - \epsilon(x, \mathcal{P}^*)$) *for every* possible choice of the challenge, then extraction is still possible even when $\epsilon(x, \mathcal{P}^*) < \mathrm{Er}(k; N)$ (but not negligible), simply by trying sufficiently many times for two distinct challenges. Below, we will identify an alternative, in some sense more fine-grained, "quality measure" of $\mathcal{P}^*$, and we show that this measure does characterize when extraction is possible. This will be helpful when it comes to more complicated settings, like a *parallel repetition*, or a *multi-round* protocol, or, ultimately, a *parallel repetition* of a *multi-round* protocol.

As before, we will state and prove our core technical results in a more abstract language, i.e., we consider an arbitrary probabilistic algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$ and an arbitrary verification function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$. The success probability of $\mathcal{A}$ is denoted as

$$\epsilon(\mathcal{A}) := \Pr\bigl(V\bigl(C, \mathcal{A}(C)\bigr) = 1\bigr)\,,$$

where C is uniformly random in $\mathcal{C}$. The obvious instantiation of $\mathcal{A}$ is given by a deterministic dishonest prover $\mathcal{P}^*$ attacking the considered k-out-of-N special-sound Σ-protocol Π on input x.

Given oracle access to $\mathcal{A}$, the goal of the extractor is to find correct responses $y_1, \ldots, y_k$ for k pairwise distinct challenges $c_1, \ldots, c_k \in \mathcal{C}$, i.e., such that $V(c_i, y_i) = 1$ for all i. In Section 6.4.1, we showed how to do this in expected polynomial time and with success probability at least

$$\epsilon(\mathcal{A}) - \mathrm{Er}(k; N)\,.$$

Below we follow a different approach and show that a more fine-grained measure, capturing how well extraction can be done, is

$$\delta_k(\mathcal{A}) := \min_{S \subseteq \mathcal{C} : |S| < k} \Pr\bigl(V(C, \mathcal{A}(C)) = 1 \mid C \notin S\bigr)\,.$$

More precisely, we argue existence of an extraction algorithm $\mathcal{E}^{\mathcal{A}}$ with oracle access to $\mathcal{A}$, that runs in expected polynomial time and succeeds with probability at least $\delta_k(\mathcal{A})/k$.

Lemma 6.7 (Extraction Algorithm - Σ-protocols)**.** *Let $k \in \mathbb{N}$, $\mathcal{C}$ a finite set with cardinality $N \geq k$ and let $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$. Then there exists an oracle algorithm $\mathcal{E}$ with the following properties: The algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$, requires an expected number of at most $2k - 1$ queries to $\mathcal{A}$ and, with probability at least $\delta_k(\mathcal{A})/k$, it outputs k pairs*

$(c_1, y_1), (c_2, y_2), \ldots, (c_k, y_k) \in \mathcal{C} \times \{0,1\}^*$ *with* $V(c_i, y_i) = 1$ *for all* i *and* $c_i \neq c_j$ *for all* $i \neq j$.

Figure 6.3: Recursive Expected Polynomial Time Extractor $\mathcal{E}_k(\mathcal{D})$.

Parameters: $k \in \mathbb{N}$ and $\mathcal{D} \subseteq \mathcal{C}$.
Oracle access to: Algorithm $\mathcal{A}\colon \mathcal{C} \to \{0,1\}^*$ and verification function $V\colon \mathcal{C} \times \{0,1\}^* \to \{0,1\}$.

- Sample $c_1 \in \mathcal{D}$ uniformly at random and evaluate $y_1 \leftarrow \mathcal{A}(c_1)$.
- If $V(c_1, y_1) = 0$, abort.
- If $V(c_1, y_1) = 1$ and $k = 1$, output $(c_1, y_1) \in \mathcal{D} \times \{0,1\}^*$.
- Else, set COIN $= 0$ and repeat
  - run $\mathcal{E}_{k-1}(\mathcal{D} \setminus \{c_1\})$;
  - set COIN $\leftarrow V\big(d, \mathcal{A}(d)\big)$ for $d \in \mathcal{D}$ sampled uniformly at random;

  until either $\mathcal{E}_{k-1}(\mathcal{D} \setminus \{c_1\})$ outputs $k-1$ pairs $(c_2, y_2), \ldots, (c_k, y_k)$ with $V(c_i, y_i) = 1$ for all i have been found or until COIN $= 1$.

Output: In the former case, output k pairs $(c_1, y_1), \ldots, (c_k, y_k) \in \mathcal{D} \times \{0,1\}^*$ with $V(c_i, y_i) = 1$ for all i and $c_i \neq c_j$ for all $i \neq j$.

Proof. The extraction algorithm is defined recursively over k. For this reason, we add a subscript k and write $\mathcal{E}_k^{\mathcal{A}}$ for the extraction algorithm that, given oracle access to $\mathcal{A}$, aims to output k pairs (c_i, y_i). In this proof, we also make the set $\mathcal{D} \subseteq \mathcal{C}$, from which the extractor samples the challenges c_i, explicit by writing $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$. This allows the extractor to be deployed on subsets $\mathcal{D}$ of the full challenge set $\mathcal{C}$, i.e., extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ aims to output k pairs (c_i, y_i) with pairwise distinct challenges $c_i \in \mathcal{D}$ and $V(c_i, y_i) = 1$ for all i. When writing $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ we will always implicitly assume that $|\mathcal{D}| \geq k$. Accordingly, we also write

$$\begin{aligned} \epsilon^V(\mathcal{A}, \mathcal{D}) &:= \Pr\big(V(C, \mathcal{A}(C)) = 1\big)\,, \\ \delta_k^V(\mathcal{A}, \mathcal{D}) &:= \min_{S \subseteq \mathcal{D} : |S| < k} \Pr\big(V(C, \mathcal{A}(C)) = 1 \mid C \notin S\big)\,, \end{aligned}$$

where the probability space is defined over of the randomness of $\mathcal{A}$ and the random variable C being uniformly random in $\mathcal{D} \subseteq \mathcal{C}$. If V is clear from context we sometimes simply write $\epsilon(\mathcal{A}, \mathcal{D})$ and $\delta_k(\mathcal{A}, \mathcal{D})$. Note that for all $k \geq 1$,

$$\delta_{k+1}(\mathcal{A}, \mathcal{D}) \leq \delta_k(\mathcal{A}, \mathcal{D}) \leq \delta_1(\mathcal{A}, \mathcal{D}) = \epsilon(\mathcal{A}, \mathcal{D})\,.$$

Let us now define the extraction algorithm. The extraction algorithm is defined recursively over k and also described in Figure 6.3. Let $\mathcal{D} \subseteq \mathcal{C}$ be an arbitrary subset with cardinality at least k. For $k = 1$, the extractor $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$ simply samples a challenge $c_1 \in \mathcal{D}$ uniformly at random and computes $y_1 \leftarrow \mathcal{A}(c_1)$. If

$V(c_1, y_1) = 0$, it outputs $\perp$ and aborts. Otherwise, if $V(c_1, y_1) = 1$, it successfully outputs (c_1, y_1). This extractor queries $\mathcal{A}$ once and it succeeds with probability $\epsilon(\mathcal{A}, \mathcal{D}) = \delta_1(\mathcal{A}, \mathcal{D})$.

For $k > 1$, the extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ first runs the extractor $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$. If $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$ fails, $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ outputs $\perp$ and aborts; otherwise, if $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$ succeeds to output a pair (c_1, y_1), $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ proceeds as follows. It defines the set $\mathcal{D}' = \mathcal{D} \setminus \{c_1\}$ and runs $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$. If $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$ succeeds to output $k-1$ pairs $(c_2, y_2), \dots (c_k, y_k)$, $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ successfully outputs k pairs $(c_1, y_1), \dots, (c_k, y_k)$. On the other hand, if $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$ fails, $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ tosses a coin that returns heads with probability $\epsilon(\mathcal{A}, \mathcal{D})$. This coin can be implemented by running $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$, i.e., sampling a random challenge $d \leftarrow \mathcal{D}$ and evaluating $V\big(d, \mathcal{A}(d)\big)$. If the coin returns heads, $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ outputs $\perp$ and aborts. If the coin returns tails, $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ runs $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$ once more and performs the same steps as before. The algorithm proceeds in this manner until either it has successfully found k pairs (c_i, y_i) or until the coin returns heads.

Let us now analyze the success probability and the expected number of $\mathcal{A}$-queries of this algorithm.

Success Probability. We aim to show that, for all $k \in \mathbb{N}$ and for all $\mathcal{D} \subseteq \mathcal{C}$ with $|\mathcal{D}| \geq k$, the success probability $\Delta_k(\mathcal{D})$ of the extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ satisfies

$$\Delta_k(\mathcal{D}) \geq \delta_k(\mathcal{A}, \mathcal{D})/k\,.$$

The analysis goes by induction. Since $\Delta_1(\mathcal{D}) = \epsilon(\mathcal{A}, \mathcal{D}) = \delta_1(\mathcal{A}, \mathcal{D})/1$, the induction hypothesis is satisfied for the case $k = 1$.

Let us now consider $k > 1$ and assume that the induction hypothesis holds for $k' = k-1$ and all $\mathcal{D}'$ with $|\mathcal{D}'| \geq k-1$. We consider arbitrary $\mathcal{D} \subseteq \mathcal{C}$ with $|\mathcal{D}| \geq k$. Then, if in its first step $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ successfully runs extractor $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$ (outputting a pair (c_1, y_1) with $V(c_1, y_1) = 1$), it starts running two geometric experiments until one of them finishes. In the first geometric experiment the extractor aims to find an additional set of $k-1$ pairs (c_i, y_i) by running $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$, where $\mathcal{D}' = \mathcal{D} \setminus \{c_1\}$. By the induction hypothesis, the parameter p of this geometric distribution satisfies

$$p := \Delta_{k-1}(\mathcal{D}') \geq \delta_{k-1}(\mathcal{A}, \mathcal{D}')/(k-1) \geq \delta_k(\mathcal{A}, \mathcal{D})/(k-1)\,.$$

In the second geometric experiment the extractor tosses a coin that returns heads with probability

$$q := \epsilon(\mathcal{A}, \mathcal{D})\,.$$

The second step of the extractor succeeds if the second geometric experiment does not finish before the first, and so by Lemma 2.2 this probability is lower bounded by

$$\begin{aligned}
\Pr\big(\mathrm{Geo}(p) \leq \mathrm{Geo}(q)\big) &\geq \frac{p}{p+q} = \frac{\Delta_{k-1}(\mathcal{D}')}{\Delta_{k-1}(\mathcal{D}') + \epsilon(\mathcal{A}, \mathcal{D})} \\
&\geq \frac{\delta_k(\mathcal{A}, \mathcal{D})/(k-1)}{\delta_k(\mathcal{A}, \mathcal{D})/(k-1) + \epsilon(\mathcal{A}, \mathcal{D})} \\
&\geq \frac{\delta_k(\mathcal{A}, \mathcal{D})/(k-1)}{\epsilon(\mathcal{A}, \mathcal{D})/(k-1) + \epsilon(\mathcal{A}, \mathcal{D})} \\
&= \frac{\delta_k(\mathcal{A}, \mathcal{D})}{k \cdot \epsilon(\mathcal{A}, \mathcal{D})}\,,
\end{aligned}$$

where the second inequality follows from the monotonicity of the function $x \mapsto \frac{x}{x+q}$. Since the first step of the extractor succeeds with probability $\epsilon(\mathcal{A}, \mathcal{D})$, it follows that $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ succeeds with probability at least $\delta_k(\mathcal{A}, \mathcal{D})/k$.

Therefore, by induction it follows that for all k and $\mathcal{D}$ with $|\mathcal{D}| \geq k$, the extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ succeeds with probability at least $\delta_k(\mathcal{A}, \mathcal{D})/k$. In particular, the extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{C})$ succeeds with probability at least $\delta_k(\mathcal{A})/k$, which proves that this extractor has the desired success probability.

Expected Number of $\mathcal{A}$-Queries. For $\mathcal{D} \subseteq \mathcal{C}$ with $|\mathcal{D}| \geq k$, we let $Q_k(\mathcal{D})$ be the expected number of $\mathcal{A}$-queries made by the extractor $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$. We will prove that $Q_k(\mathcal{D}) \leq 2k - 1$ for all $k \in \mathbb{N}$ and $\mathcal{D} \subseteq \mathcal{C}$ with $|\mathcal{D}| \geq k$. The proof of this claim goes by induction. First note that, since $Q_1(\mathcal{D}) = 1$ for all $\mathcal{D} \neq \emptyset$, this claim is clearly satisfied for the base case $k = 1$.

Let us now consider $k > 1$ and assume the claim is satisfied for $k' = k - 1$. Let $\mathcal{D} \subseteq \mathcal{C}$ be arbitrary with $|\mathcal{D}| \geq k$. Then $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ first runs $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$, which requires exactly $Q_1(\mathcal{D}) = 1$ query. Then with probability $\epsilon(\mathcal{A}, \mathcal{D})$ it continues to the second step. In each iteration of the second step $\mathcal{E}_k^{\mathcal{A}}(\mathcal{D})$ runs $\mathcal{E}_{k-1}^{\mathcal{A}}(\mathcal{D}')$, for some $D' \subseteq \mathcal{C}$ with $|\mathcal{D}'| \geq k-1$, and it tosses a coin by running $\mathcal{E}_1^{\mathcal{A}}(\mathcal{D})$. Therefore, each iteration requires an expected number of at most $Q_{k-1}(\mathcal{D}') + 1 \leq 2k - 2$ queries. Moreover, the expected number of tosses until the coin returns heads is $1/\epsilon(\mathcal{A}, \mathcal{D})$. Hence, the expected number of iterations in the second step of this extraction algorithm is at most $1/\epsilon(\mathcal{A}, \mathcal{D})$. It follows that

$$Q_k(\mathcal{D}) \leq 1 + \epsilon(\mathcal{A}, \mathcal{D}) \frac{1}{\epsilon(\mathcal{A}, \mathcal{D})}(2k - 2) = 2k - 1\,.$$

Here it is crucial that the above inequality holds for all $\mathcal{D} \subseteq \mathcal{C}$. This proves the claimed upper bound on the expected number of $\mathcal{A}$-queries and completes the proof of the lemma.

□

In the context of a deterministic dishonest prover $\mathcal{P}^*$ attacking a k-out-of-N special-sound protocol, we make the following observation. First, by basic probability theory and for any $S \subseteq \mathcal{C}$,

$$\begin{aligned} \Pr\big(V(C, \mathcal{A}(C)) = 1 \mid C \notin S\big) &= \frac{\Pr\big(V(C, \mathcal{A}(C)) = 1 \wedge C \notin S\big)}{\Pr\big(C \notin S\big)} \\ &\geq \frac{\Pr\big(V(C, \mathcal{A}(C)) = 1\big) - \Pr\big(C \in S\big)}{\Pr\big(C \notin S\big)}\,. \end{aligned} \tag{6.4}$$

Thus, extractor $\mathcal{E}^{\mathcal{A}}$ succeeds with positive probability as soon as $\epsilon(\mathcal{A}) > \Pr\big(C \in S\big)$ for every $S \subseteq \mathcal{C}$ with $|S| < k$. More precisely,

$$\Pr\big(\mathcal{E}^{\mathcal{A}} \neq \bot\big) \geq \frac{\delta_k(\mathcal{A})}{k} \geq \frac{\epsilon(\mathcal{A}) - \mathrm{Er}(k; N)}{k(1 - \mathrm{Er}(k; N))}\,, \tag{6.5}$$

where $\mathrm{Er}(k; N) = (k-1)/N$.

This observation confirms that k-out-of-N special-soundness implies knowledge soundness with knowledge error $\mathrm{Er}(k; N)$, i.e., it provides an alternative proof for Theorem 6.3. Hence, in comparison to $\epsilon(\mathcal{A})$, $\delta_k(\mathcal{A})$ is indeed a more fine-grained measure capturing how well extraction can be done.

Note that the extractor of Lemma 6.7 does not strictly outperform the extractor of Lemma 6.5. Namely, it behaves somewhat worse in the (expected) polynomial runtime, and also in the success probability when the measure $\delta_k(\mathcal{A})$ is bounded by $\epsilon(\mathcal{A}) - (k-1)/N$; the expected runtime is roughly a factor two larger and the success probability is roughly a factor k smaller. However, this is still sufficient for proving that k-out-of-N special-soundness tightly implies knowledge soundness. Moreover, by exploiting the definition of δ, as we show below, we can obtain an extractor for a *parallel repetition* of the considered interactive proof by running the extractor individually on each instance of the parallel repetition. Thus, our extractor is well suited to handle parallel repetitions of k-out-of-N special-sound Σ-protocols. Nevertheless, it remains an interesting problem whether our extractor can be improved to match up with the extractor from Lemma 6.5 while still giving rise to our parallel-repetition results.

Knowledge-Soundness of the Parallel Repetition

When moving to the t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ of the k-out-of-N special-sound Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$, we consider an algorithm $\mathcal{A}$ that takes as input a row $(c^1, \dots, c^t) \in \mathcal{C}^t$ of challenges[3] and outputs a string y, and the *success probability* of $\mathcal{A}$ is then defined as

$$\epsilon(\mathcal{A}) = \Pr\big(V(C^1, \dots, C^t, \mathcal{A}(C^1, \dots, C^t)) = 1\big),$$

for some given $V\colon \mathcal{C}^t \times \{0,1\}^* \to \{0,1\}$ and where the C^j are understood to be independently and uniformly distributed over $\mathcal{C}$. We use superscripts to distinguish between the different parallel instantiations of basic Σ-protocol Π, so that later, when considering multi-round interactive proofs, the subscripts can be used to distinguish between the different rounds of the protocol.

The obvious instantiation of $\mathcal{A}$ is given by a deterministic prover $\mathcal{P}^*$ attacking the considered t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ of Π on input x. More precisely, on input $(c^1, \dots, c^t)$, $\mathcal{A}$ runs $\mathcal{P}^*$ sending $(c^1, \dots, c^t)$ as the challenges for the t repetitions of Π, and outputs $\mathcal{P}^*$'s (fixed) first messages $(a^1, \dots, a^t)$ and its responses $(z^1, \dots, z^t)$, and the function V is defined as the verification procedure of $\mathcal{V}^t$, which checks each repetition independently and accepts only if all are correct.

Such an $\mathcal{A}$ naturally induces t algorithms $\mathcal{A}_1, \dots, \mathcal{A}_t$ as considered above in the context of a single execution of a k-out-of-N special-sound protocol, taking *one* challenge as input: on input c^j, the algorithm $\mathcal{A}_j$ runs $y \leftarrow \mathcal{A}(c^1, \dots, c^t)$ with c^i chosen uniformly at random from $\mathcal{C}$ for $i \neq j$, and outputs y along with the c^i's for $i \neq j$. We can thus run the extractor from above on all of the $\mathcal{A}_j$'s individually, with the goal being that at least one of them succeeds. We know that for each $\mathcal{A}_j$ individually, the extraction succeeds with probability

$$\delta_k^V(\mathcal{A}_j) = \min_{S^j \subseteq \mathcal{C} : |S^j| < k} \Pr\big(V(C^j, \mathcal{A}_j(C^j)) = 1 \mid C^j \notin S^j\big), \tag{6.6}$$

[3]There is no rigorous meaning in the list of challenges forming a *row*; it is merely that later we will also consider a *column* of challenges, which will then play a different *contextual* role.

where V is understood to appropriately reorder its inputs. The following lemma allows us to bound the probability that at least one of the extractors $\mathcal{E}^{\mathcal{A}_j}$ succeeds to produce k challenge-response pairs $\big((c^1,\dots,c^t),y\big)$ that all verify V and for which the k choices of c^j are all distinct for the considered j.

Lemma 6.8. *Let $k,t \in \mathbb{N}$, $\mathcal{C}$ a set with $|\mathcal{C}| = N \geq k$, $V\colon \mathcal{C}^t \times \{0,1\}^* \to \{0,1\}$, and $\mathcal{A}$ a (probabilistic) algorithm that takes as input a vector $(c^1,\dots,c^t) \in \mathcal{C}^t$ and outputs a string $y \in \{0,1\}^*$. Then*

$$\sum_{j=1}^t \delta_k^V(\mathcal{A}_j) \geq \frac{\epsilon(\mathcal{A}) - \mathrm{Er}(k;N)^t}{1-\mathrm{Er}(k;N)}\,,$$

where $\mathrm{Er}(k;N) = (k-1)/N$.

Proof. Let Λ denote the event $V\big(C^1,\dots,C^t,\mathcal{A}(C^1,\dots,C^t)\big) = 1$ and, for $1 \leq j \leq t$, let S^j be such that it minimizes Equation 6.6. Moreover, let Γ_j denote the event $C^j \notin S^j$.

Without loss of generality, we may assume that $|S^j| = k-1$ for all j. Then, for all j,

$$\Pr(\Gamma_j) = \Pr(c^j \notin S^j) = 1 - \mathrm{Er}(k;N)\,.$$

Moreover, using elementary probability theory,

$$\begin{aligned}\sum_{j=1}^t \delta_k(\mathcal{A}_j) &= \sum_{j=1}^t \Pr\big(V(C^j,\mathcal{A}_j(C^j)) = 1 \mid C^j \notin S^j\big) = \sum_{j=1}^t \Pr\big(\Lambda \mid \Gamma_j\big) \\ &= \sum_{j=1}^t \frac{\Pr\big(\Lambda \wedge \Gamma_j\big)}{\Pr\big(\Gamma_j\big)} = \sum_{j=1}^t \frac{\Pr\big(\Lambda \wedge \Gamma_j\big)}{1-\mathrm{Er}(k;N)} \geq \frac{\Pr\big(\Lambda \wedge \exists j : \Gamma_j\big)}{1-\mathrm{Er}(k;N)} \\ &\geq \frac{\Pr\big(\Lambda\big) - \Pr\big(\neg\Gamma_j\ \forall j\big)}{1-\mathrm{Er}(k;N)} = \frac{\epsilon(\mathcal{A}) - \mathrm{Er}(k;N)^t}{1-\mathrm{Er}(k;N)}\,,\end{aligned}$$

which completes the proof. □

Lemma 6.8 readily provides a lower bound on $\max_i \delta_k^V(\mathcal{A}_i) \geq \sum_i \delta_k^V(\mathcal{A}_i)/t$, and thus on the success probability of the extractor. However, we can do slightly better. For this purpose, let $\Delta = \min\big(1, \sum_{i=1}^t \delta_k^V(\mathcal{A}_i)/k\big)$. Then, by the inequality of the arithmetic and the geometric mean,

$$\left(\prod_{i=1}^t \left(1 - \frac{\delta_k^V(\mathcal{A}_i)}{k}\right)\right)^{1/t} \leq \frac{1}{t}\sum_{i=1}^t \left(1 - \frac{\delta_k^V(\mathcal{A}_i)}{k}\right) \leq 1 - \frac{\Delta}{t}\,.$$

Hence, the probability that at least one extractor $\mathcal{E}^{\mathcal{A}_i}$ succeeds equals

$$1 - \prod_{i=1}^t \left(1 - \frac{\delta_k^V(\mathcal{A}_i)}{k}\right) \geq 1 - \left(1 - \frac{\Delta}{t}\right)^t \geq 1 - e^{-\Delta} \geq (1-e^{-1})\Delta \geq \frac{1}{2}\Delta\,, \quad (6.7)$$

where the third inequality uses that $(1 - e^{-x}) \geq (1 - e^{-1})x$ for all $0 \leq x \leq 1$, which is easily verified.[4] Hence, by Lemma 6.8, the probability of at least one of the extractors $\mathcal{E}^{\mathcal{A}_i}$ being successful is at least

$$\frac{\Delta}{2} \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}(k;N)^t}{2k(1 - \mathrm{Er}(k;N))} \, .$$

From this it follows that the t-fold parallel repetition Π^t of a k-out-of-N special-sound protocol Π is knowledge sound with knowledge error $\mathrm{Er}(k;N)^t$, where $\mathrm{Er}(k;N) = (k-1)/N$ is the knowledge error of a single execution of Π. This strong parallel repetition result for k-out-of-N special-sound Σ-protocols is formalized in Theorem 6.7.

Theorem 6.7 (Parallel Repetition of k-Special-Sound Σ-Protocols)**.** *Let* $\Pi = (\mathcal{P}, \mathcal{V})$ *be a k-out-of-N special-sound* Σ*-protocol. Let* $\Pi^t = (\Pi^t, \mathcal{V}^t)$ *be the t-fold parallel repetition of* Π*. Then* Π^t *is knowledge sound with knowledge error* $\mathrm{Er}(k;N)^t$*, where* $\mathrm{Er}(k;N) = (k-1)/N$*.*

Also here we have that the knowledge error $\mathrm{Er}(k;N)^t$ matches the trivial cheating probability, which succeeds if in each instance of the parallel repetition the challenge falls into a given set of size $k-1$.

Remark 6.3. The above parallel repetition result (and also the generalization of Section 6.5.3) directly generalizes to the parallel composition of t *different* protocols or to the parallel composition of t different instances of the same protocol. In this case, the knowledge error will be the product of the individual knowledge errors.

6.5.3 Parallel Repetition of Multi-Round Interactive Proofs

We now consider the general case of multi-round interactive proofs. The line of reasoning is quite similar to that of 3-round protocols, but with an appropriately adjusted definition of δ. So, for the remainder of this section, we consider a $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound $(2\mu+1)$-round interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$, where the verifier samples its i-th challenge uniformly at random from a finite set $\mathcal{C}_i$ for $1 \leq i \leq \mu$. Eventually, we want to analyze its t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$, but again we first consider a single invocation.

Knowledge Soundness of a Single Invocation

Similar to Section 6.4.2, we consider a probabilistic algorithm $\mathcal{A}$ that takes as input a vector $(c_1, \dots, c_\mu) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ of challenges and outputs a string y, and we consider a function

$$V \colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\} \, .$$

As before, the success probability of $\mathcal{A}$ is defined as

$$\epsilon^V(\mathcal{A}) := \Pr\big(V(C, \mathcal{A}(C)) = 1\big) \, ,$$

[4] For instance by observing that the two sides are equal for $x = 0$ and $x = 1$, and that the left hand side is a concave function while the right hand side is linear.

where $C = (C_1, \dots, C_\mu)$ is uniformly random in $\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$. The obvious instantiation of $\mathcal{A}$ is a deterministic prover $\mathcal{P}^*$ attacking the considered protocol. The goal of the extractor is to find correct responses for a $\mathbf{k}$-tree of challenges (Definition 2.33), where $\mathbf{k} = (k_1, \dots, k_\mu)$. Generalizing the case of ordinary Σ-protocols, i.e., 3-round interactive proofs, the figure of merit here is

$$\delta^V_{\mathbf{k}}(\mathcal{A}) := \min_{S_1, S_2(\cdot), \dots, S_\mu(\cdot)} \Pr\left(\Lambda \,\middle|\, \begin{array}{c} C_1 \notin S_1 \wedge C_2 \notin S_2(C_1) \wedge \cdots \\ \cdots \wedge C_\mu \notin S_\mu(C_1, \dots, C_{\mu-1}) \end{array} \right), \tag{6.8}$$

where Λ denotes the event $V(C, \mathcal{A}(C)) = 1$ and the minimum is over all sets $S_1 \in \mathcal{C}_1|_{<k_1}$, and over all functions $S_2 \colon \mathcal{C}_1 \to \mathcal{C}_2|_{<k_2}$, $S_3 \colon \mathcal{C}_1 \times \mathcal{C}_2 \to \mathcal{C}_3|_{<k_3}$, etc. Here for any set $\mathcal{C}$ and $k \in \mathbb{N}$, $\mathcal{C}|_{<k}$ denotes the set of subsets of $\mathcal{C}$ with cardinality smaller than k.

Indeed, the following lemma shows that there exists an expected polynomial time extractor $\mathcal{E}^{\mathcal{A}}$ with oracle access to $\mathcal{A}$ that, with probability at least $\delta^V_{\mathbf{k}}(\mathcal{A}) / \prod_{i=1}^{\mu} k_i$, succeeds to extract correct responses for a $\mathbf{k}$-tree of challenges. Exploiting the abstract notation of Lemma 6.7, the proof of this lemma follows by induction over the number of challenges μ sent by the verifier. In particular, the extractor of the following lemma follows the same recursive approach as the one in Lemma 6.6, where we also considered knowledge extraction for multi-round interactive proofs. However, instead of Lemma 6.5, here we apply Lemma 6.7 for the base case of 3-round Σ-protocols. Subsequently, we will show that this adaptation allows us to handle parallel repetitions of multi-round interactive proofs.

Lemma 6.9 (Multi-Round Extraction Algorithm)**.** *Let* $\mathbf{k} = (k_1, \dots, k_\mu)$, $\mathbf{N} = (N_1, \dots, N_\mu) \in \mathbb{N}^\mu$, $K = \prod_{i=1}^{\mu} k_i$, $\mathcal{C}_1, \dots, \mathcal{C}_\mu$ *finite sets* $\mathcal{C}_i$ *with cardinality* $N_i \geq k_i$ *and let* $V \colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\}$*. Then there exists an oracle algorithm* $\mathcal{E}$ *with the following properties: The algorithm* $\mathcal{E}^{\mathcal{A}}$*, given oracle access to a (probabilistic) algorithm* $\mathcal{A} \colon \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \to \{0,1\}^*$*, requires an expected number of at most* $2^\mu \cdot K$ *queries to* $\mathcal{A}$ *and, with probability at least* $\delta^V_{\mathbf{k}}(\mathcal{A})/K$*, outputs* K *pairs* $(\mathbf{c}_1, y_1), \dots, (\mathbf{c}_K, y_K) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^*$ *with* $V(\mathbf{c}_i, y_i) = 1$ *for all* i *and such that the vectors* $\mathbf{c}_i \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ *form a* $\mathbf{k}$*-tree.*

Proof. The proof goes by induction on μ. For the base case $\mu = 1$, the lemma directly follows from Lemma 6.7. So let us assume the lemma holds for $\mu' = \mu - 1$. Then, for any $c \in \mathcal{C}_1$, let $\mathcal{A}_c$ be the algorithm that takes as input a vector $(c_2, \dots, c_\mu) \in \mathcal{C}_2 \times \cdots \times \mathcal{C}_\mu$ and runs $\mathcal{A}(c, c_2, \dots, c_\mu)$. The function V_c is defined accordingly, i.e.,

$$V_c \colon \mathcal{C}_2 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^* \to \{0,1\}, \quad (\mathbf{c}, y) \mapsto V(c, \mathbf{c}, y)\,.$$

Moreover, let $\mathbf{k}' = (k_2, \dots, k_\mu), \mathbf{N}' = (N_2, \dots, N_\mu) \in \mathbb{N}^{\mu-1}$ and $K' = \prod_{i=2}^{\mu} k_i$. By the induction hypothesis, there exists an algorithm $\mathcal{E}^{\mathcal{A}_c}_{\mu-1}$ that aims to output a set $\mathcal{Y}$ of K' pairs $(\mathbf{c}_1, y_1), \dots, (\mathbf{c}_{K'}, y_{K'}) \in \mathcal{C}_2 \times \cdots \times \mathcal{C}_\mu \times \{0,1\}^*$ with $V(c, \mathbf{c}_i, y_i) = 1$ for all i and such that the vectors $\mathbf{c}_i \in \mathcal{C}_2 \times \cdots \times \mathcal{C}_\mu$ form a $\mathbf{k}'$-tree of challenge vectors. Moreover, $\mathcal{E}^{\mathcal{A}_c}_{\mu-1}$ requires an expected number of at most $2^{\mu-1} \cdot K'$ queries to $\mathcal{A}$ and succeeds with probability at least $\delta^{V_c}_{\mathbf{k}'}(\mathcal{A}_c)/K'$. We define $W \colon \mathcal{C}_1 \times \{0,1\}^* \to \{0,1\}$, by setting $W(c, \mathcal{Y}) = 1$ if and only if $\mathcal{Y}$ is a set satisfying the above properties.

Now let $\mathcal{B}^{\mathcal{A}} \colon \mathcal{C}_1 \to \{0,1\}^*$ be the algorithm, with oracle access to $\mathcal{A}$, that takes as input an element $c \in \mathcal{C}_1$ and runs $\mathcal{E}_{\mu-1}^{\mathcal{A}_c}$. By Lemma 6.7, there exists an expected polynomial time algorithm $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$, with oracle access to $\mathcal{B}^{\mathcal{A}}$, that aims to output k_1 pairs $(c_1, \mathcal{Y}_1), \dots, (c_{k_1}, \mathcal{Y}_{k_1}) \in \mathcal{C}_1 \times \{0,1\}^*$ with $W(c_i, \mathcal{Y}_i) = 1$ for all i and $c_i \neq c_j$ for all $i \neq j$. The extractor $\mathcal{E}^{\mathcal{A}}$ simply runs $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$. Note that, by the associativity of the composition of oracle algorithms, $\mathcal{E}^{\mathcal{A}} = \mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}} = (\mathcal{E}_1^{\mathcal{B}})^{\mathcal{A}}$ is indeed an algorithm with oracle access to $\mathcal{A}$.

Let us now analyze the success probability and the expected number of $\mathcal{A}$-queries of the algorithm $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ and therefore of $\mathcal{E}^{\mathcal{A}}$.

Success Probability. Again by Lemma 6.7 it follows that $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ succeeds with probability at least

$$\begin{aligned}
\delta_{k_1}^W(\mathcal{B}^{\mathcal{A}})/k_1 &= \min_{S_1 \subseteq \mathcal{C}_1, |S_1| < k_1} \frac{\Pr\bigl(W(C, \mathcal{B}^{\mathcal{A}}(C)) = 1 \mid C \notin S_1\bigr)}{k_1} \\
&= \min_{S_1 \subseteq \mathcal{C}_1, |S_1| < k_1} \frac{\Pr\bigl(W(C, \mathcal{B}^{\mathcal{A}}(C)) = 1 \land C \notin S_1\bigr)}{k_1 \cdot \Pr(C \notin S_1)} \\
&= \min_{S_1 \subseteq \mathcal{C}_1, |S_1| < k_1} \frac{\sum_{c \notin S_1} \Pr(C = c) \cdot \Pr\bigl(W(c, \mathcal{B}^{\mathcal{A}}(c)) = 1\bigr)}{k_1 \cdot \Pr(C \notin S_1)} ,
\end{aligned}$$

where C is uniformly random in $\mathcal{C}$. Hence, by the induction hypothesis it follows that

$$\begin{aligned}
\delta_{k_1}^W(\mathcal{B}^{\mathcal{A}})/k_1 &\geq \min_{S_1 \subseteq \mathcal{C}_1, |S_1| < k_1} \frac{\sum_{c \notin S_1} \Pr(C = c) \cdot \delta_{\mathbf{k}'}^{V_c}(\mathcal{A}_c)}{k_1 \cdot K' \cdot \Pr(C \notin S_1)} \\
&= \min_{S_1 \subseteq \mathcal{C}_1, |S_1| < k_1} \frac{\sum_{c \notin S_1} \Pr(C = c) \cdot \delta_{\mathbf{k}'}^{V_c}(\mathcal{A}_c)}{K \cdot \Pr(C \notin S_1)} . \qquad (6.9)
\end{aligned}$$

Now note that

$$\delta_{\mathbf{k}'}^{V_c}(\mathcal{A}_c) = \min_{S_2(\cdot), \dots, S_\mu(\cdot)} \Pr\left(\Lambda \,\middle|\, \begin{array}{c} C_1 = c \land C_2 \notin S_2(C_1) \land \cdots \\ \cdots \land C_\mu \notin S_\mu(C_1, \dots, C_{\mu-1}) \end{array}\right),$$

where Λ denotes the event $V(C, \mathcal{A}(C)) = 1$. Hence,

$$\sum_{c \notin S_1} \Pr(C = c) \cdot \delta_{\mathbf{k}'}^{V_c}(\mathcal{A}_c) = \\
\min_{S_2(\cdot), \dots, S_\mu(\cdot)} \Pr\left(\Lambda \land C_1 \notin S_1 \,\middle|\, \begin{array}{c} C_2 \notin S_2(C_1) \land \cdots \\ \cdots \land C_\mu \notin S_\mu(C_1, \dots, C_{\mu-1}) \end{array}\right).$$

Combining this equality with Equation 6.9, shows that

$$\delta_{k_1}^W(\mathcal{B}^{\mathcal{A}})/k_1 \geq \frac{\delta_{\mathbf{k}}^V(\mathcal{A})}{K} ,$$

which shows that $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ has the desired success probability.

Expected Number of $\mathcal{A}$-Queries. By Lemma 6.7, it follows that $\mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ requires an expected number of at most $2k_1$ queries to $\mathcal{B}^{\mathcal{A}}$. By the induction hypothesis it follows that $\mathcal{B}^{\mathcal{A}}(c)$ requires an expected number of at most $2^{\mu-1} \cdot K'$ queries to $\mathcal{A}$ for all $c \in \mathcal{C}$. Hence, $\mathcal{E}^{\mathcal{A}} = \mathcal{E}_1^{\mathcal{B}^{\mathcal{A}}}$ requires an expected number of at most $2^{\mu} \cdot K$ queries to $\mathcal{A}$, which completes the proof of the lemma.

□

Let $S_1, S_2(\cdot), \dots, S_\mu(\cdot)$ be the arguments minimizing Equation 6.8. Further, let Λ denote the event $V(C, \mathcal{A}(C)) = 1$ and let Γ denote the event

$$\Gamma = C_1 \notin S_1 \wedge C_2 \notin S_2(C_1) \wedge \cdots \wedge C_\mu \notin S_\mu(C_1, \dots, C_{\mu-1}) .$$

Then, using the same kind of reasoning as in Equation 6.4, we have

$$\delta_{\mathbf{k}}^V(\mathcal{A}) = \Pr(\Lambda \mid \Gamma) = \frac{\Pr(\Lambda \wedge \Gamma)}{\Pr(\Gamma)} \geq \frac{\Pr(\Lambda) - \Pr(\neg\Gamma)}{\Pr(\Gamma)} = \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k}; \mathbf{N})}{1 - \mathrm{Er}(\mathbf{k}; \mathbf{N})} ,$$

where

$$\mathrm{Er}(\mathbf{k}; \mathbf{N}) = \Pr(\neg\Gamma) = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right) .$$

This confirms that a **k**-out-of-**N** special-sound interactive proof is knowledge sound with knowledge error $\mathrm{Er}(\mathbf{k}; \mathbf{N})$, i.e., it provides an alternative proof for Theorem 6.4. This alternative approach, and in particular the quality measure $\delta_{\mathbf{k}}^V(\mathcal{A})$, allows us to generalize to parallel repetitions of **k**-out-of-**N** special-sound interactive proofs.

Knowledge-Soundness of the Parallel Repetition

We finally move towards stating and proving our main general parallel repetition result for multi-round protocols. Thus, consider the t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ of the given **k**-out-of-**N** special-sound $(2\mu + 1)$-round interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$.

We consider an algorithm $\mathcal{A}$ that takes as input a *row* $(\mathbf{c}^1, \dots, \mathbf{c}^t)$ of *columns* $\mathbf{c}^j = (c_1^j, \dots, c_\mu^j) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ of challenges and outputs a string y. Furthermore, we consider a verification function V, which then defines the *success probability* of $\mathcal{A}$ as

$$\epsilon^V(\mathcal{A}) = \Pr\bigl(V(C, \mathcal{A}(C)) = 1\bigr) ,$$

where $C = (C^1, \dots, C^t)$ with C^j distributed uniformly over $\mathcal{C}_1 \times \cdots \mathcal{C}_\mu$ for all $1 \leq j \leq t$.

Again, the obvious instantiation for $\mathcal{A}$ is a deterministic dishonest prover $\mathcal{P}^*$ attacking $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ on input x. More precisely, on input a row $(\mathbf{c}^1, \dots, \mathbf{c}^t)$ of columns, $\mathcal{A}$ runs $\mathcal{P}^*$ sending $(\mathbf{c}^1, \dots, \mathbf{c}^t)$ as the challenges, and outputs all of $\mathcal{P}^*$'s messages, and the function V is defined as the verification check that $\mathcal{V}^t$ performs.

Such an $\mathcal{A}$ naturally induces t algorithms $\mathcal{A}_1, \dots, \mathcal{A}_t$ as considered before in the context of a single execution of a multi-round protocol, taking one challenge-column as input and outputting one string: on input $\mathbf{c}^j$, the algorithm $\mathcal{A}_j$ runs $y \leftarrow \mathcal{A}(\mathbf{c}^1, \dots, \mathbf{c}^\mu)$ with $\mathbf{c}^i$ chosen uniformly at random from $\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ for $i \neq j$, and outputs y along with the $\mathbf{c}^i$'s for $i \neq j$. Thus, we can run the extractor from Lemma 6.9 on all of the $\mathcal{A}_j$'s individually, with the goal being that at least one of them succeeds. For each $\mathcal{A}_j$ individually, the extraction succeeds with probability at least

$$\delta^V_{\mathbf{k}}(\mathcal{A}_j)/K = \min_{S^j_1, S^j_2(\cdot),\dots,S^j_\mu(\cdot)} \Pr\left(\Lambda_j \,\middle|\, \begin{array}{c} C^j_1 \notin S^j_1 \wedge C^j_2 \notin S^j_2(C^j_1) \wedge \cdots \\ \cdots \wedge C^j_\mu \notin S^j_\mu(C^j_1,\dots,C^j_{\mu-1}) \end{array}\right)/K\,, \tag{6.10}$$

where Λ_j denotes the event $V(C^j, \mathcal{A}_j(C^j)) = 1$, V is understood to appropriately reorder its inputs and $K = \prod_{i=1}^{\mu} k_i$. The following lemma allows us to bound the probability that at least one of the extractors $\mathcal{E}^{\mathcal{A}_j}$ succeeds.

Lemma 6.10. *Let* $\mathbf{k} = (k_1,\dots,k_\mu), \mathbf{N} = (N_1,\dots,N_\mu) \in \mathbb{N}^\mu$, $t \in \mathbb{N}$, $\mathcal{C}_1,\dots,\mathcal{C}_\mu$ *finite sets* $\mathcal{C}_i$ *with cardinality* $N_i \geq k_i$ *and let* $V\colon \big(\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu\big)^t \times \{0,1\}^* \to \{0,1\}$. *Further, let* $\mathcal{A}$ *be a (probabilistic) algorithm that takes as input a row* $(\mathbf{c}^1,\dots,\mathbf{c}^t)$ *of columns* $\mathbf{c}^j = (c^j_1,\dots,c^j_\mu) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ *and outputs a string* $y \in \{0,1\}^*$. *Then*

$$\sum_{j=1}^{t} \delta^V_{\mathbf{k}}(\mathcal{A}_j) \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k};\mathbf{N})^t}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})}\,,$$

where

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1 - \prod_{i=1}^{\mu}\left(1 - \frac{k_i - 1}{N_i}\right).$$

Proof. Let Λ denote the event $V(C, \mathcal{A}(C)) = 1$ and, for $1 \leq j \leq t$, let S^j_1, $S^j_2(\cdot)$, $\dots$, $S^j_\mu(\cdot)$ be such that they minimize Equation 6.10. Moreover, let Γ_j denote the event

$$C^j_1 \notin S^j_1 \wedge C^j_2 \notin S^j_2(C^j_1) \wedge \cdots \wedge C^j_\mu \notin S^j_\mu(C^j_1,\dots,C^j_{\mu-1})\,.$$

Without loss of generality, we may assume that $\left|S^j_1\right| = k_1 - 1$ and

$$S^j_i : \mathcal{C}_1 \times \cdots \times \mathcal{C}_{i-1} \to \{S \subseteq \mathcal{C}_i : |S| = k_i - 1\}$$

for all $2 \leq i \leq \mu$ and $1 \leq j \leq t$. Then, for all $1 \leq j \leq t$,

$$\Pr(\Gamma_j) = \prod_{i=1}^{\mu}\left(1 - \frac{k_i - 1}{N}\right) = 1 - \mathrm{Er}(\mathbf{k};\mathbf{N})\,.$$

Moreover, using elementary probability theory,

$$\begin{aligned} \sum_{j=1}^{t} \delta^V_{\mathbf{k}}(\mathcal{A}_j) &= \sum_{j=1}^{t} \Pr\big(\Lambda \mid \Gamma_j\big) = \sum_{j=1}^{t} \frac{\Pr\big(\Lambda \wedge \Gamma_j\big)}{\Pr\big(\Gamma_j\big)} = \sum_{j=1}^{t} \frac{\Pr\big(\Lambda \wedge \Gamma_j\big)}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})} \\ &\geq \frac{\Pr\big(\Lambda \wedge \exists j : \Gamma_j\big)}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})} \geq \frac{\Pr\big(\Lambda\big) - \Pr\big(\neg\Gamma_j \ \forall j\big)}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})} = \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k};\mathbf{N})^t}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})}\,, \end{aligned}$$

which completes the proof. □

As for the parallel repetition of a 3-round protocol, it follows that the probability of at least one of the extractors $\mathcal{E}^{\mathcal{A}_j}$ being successful is at least

$$\frac{\Delta}{2} \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}(\mathbf{k};\mathbf{N})^t}{2K(1-\mathrm{Er}(\mathbf{k};\mathbf{N}))},$$

where $\Delta = \min\bigl(1, \sum_{j=1}^t \delta_{\mathbf{k}}^V(\mathcal{A}_j)/K\bigr)$ and $K = \prod_{i=1}^\mu k_i$. This gives us the following strong parallel repetition result for **k**-out-of-**N** special-sound protocols.

Theorem 6.8 (Parallel Repetition Theorem for Multi-Round Protocols)**.** *Let* $\Pi = (\mathcal{P},\mathcal{V})$ *be a* **k***-out-of-***N** *special-sound interactive proof. Then the t-fold parallel repetition* $\Pi^t = (\mathcal{P}^t,\mathcal{V}^t)$ *of* Π *is knowledge sound with knowledge error* $\mathrm{Er}(\mathbf{k};\mathbf{N})^t$*, where*

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1 - \prod_{i=1}^{\mu}\left(1 - \frac{k_i - 1}{N_i}\right),$$

is the knowledge error of Π*.*

Also here, the knowledge error $\mathrm{Er}(\mathbf{k};\mathbf{N})^t$ coincides with success probability $\prod_j \Pr(\neg\Gamma_j)$ of the trivial cheating strategy, which typical **k**-out-of-**N** special-sound interactive proofs admit.

6.5.4 Threshold Parallel Repetition

In the previous section we have shown that the knowledge error $\mathrm{Er}(\mathbf{k};\mathbf{N})^t$ of the t-fold parallel repetition $\Pi^t = (\mathcal{P}^t,\mathcal{V}^t)$ of a **k**-out-of-**N** special-sound interactive proof $\Pi = (\mathcal{P},\mathcal{V})$ decreases exponentially with t. However, the completeness error of Π^t equals $\rho' = 1-(1-\rho)^t$, where ρ is the completeness error of Π. Hence, if $\rho \notin \{0,1\}$, the completeness error of Π^t increases quickly with t. In order to decrease both the knowledge and the completeness error simultaneously, we consider a *threshold parallel repetition.* The s-out-of-t threshold parallel repetition of an interactive proof Π, denoted by $\Pi_s^t = (\mathcal{P}_s^t,\mathcal{V}_s^t)$, runs t instances of Π in parallel and $\mathcal{V}_s^t$ accepts if at least s-out-of-t instances are accepted. In particular, it holds that $\Pi_t^t = \Pi^t$. In this section, we show that if Π is **k**-out-of-**N** special-sound then Π_s^t is knowledge sound. We will immediately consider the general case of multi-round protocols.

As in Section 6.5.3, we consider an algorithm $\mathcal{A}$ that takes as input a row $\mathbf{c} = (\mathbf{c}^1,\dots,\mathbf{c}^t)$ of columns $\mathbf{c}^j = (c_1^j,\dots,c_\mu^j) \in \mathcal{C}_1\times\cdots\times\mathcal{C}_\mu$ of challenges and outputs a string y. However, this time we consider t different verification functions

$$V_j\colon \bigl(\mathcal{C}_1\times\cdots\times\mathcal{C}_\mu\bigr)^t \times \{0,1\}^* \to \{0,1\},$$

together with one additional *threshold* verification function defined as follows:

$$V(\mathbf{c},y) = \begin{cases} 1 & \text{if } \displaystyle\sum_{j=1}^t V_j(\mathbf{c},y) \geq s, \\ 0 & \text{otherwise}. \end{cases} \tag{6.11}$$

The obvious instantiation for $\mathcal{A}$ is a deterministic dishonest prover $\mathcal{P}^*$ attacking Π_s^t. This instantiation defines V_j as the verification performed by the j-th instance of $\mathcal{V}$. The verification function V then captures the verification performed by $\mathcal{V}_s^t$.

As before, such $\mathcal{A}$ induces t algorithms $\mathcal{A}_1, \dots, \mathcal{A}_t$ as considered in the context of a single execution of Π, taking one challenge-column as input and outputting one string: on input $\mathbf{c}^j$, the algorithm $\mathcal{A}_j$ runs $y \leftarrow \mathcal{A}(\mathbf{c}^1, \dots, \mathbf{c}^\mu)$ with $\mathbf{c}^i$ chosen uniformly at random from $\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ for $i \neq j$, and outputs y along with the $\mathbf{c}^i$'s for $i \neq j$. For each $\mathcal{A}_j$, we can run the extractor from Lemma 6.9, which succeeds with probability at least

$$\delta_{\mathbf{k}}^{V_j}(\mathcal{A}_j)/K = \min_{S_1^j, S_2^j(\cdot), \dots, S_\mu^j(\cdot)} \Pr\left(\Lambda_j \,\middle|\, \begin{array}{c} C_1^j \notin S_1^j \wedge C_2^j \notin S_2^j(C_1^j) \wedge \cdots \\ \cdots \wedge C_\mu^j \notin S_\mu^j(C_1^j, \dots, C_{\mu-1}^j) \end{array}\right)/K\,, \tag{6.12}$$

where Λ_j denotes the event $V_j(C_j, \mathcal{A}_j(C_j)) = 1$ and $K = \prod_{i=1}^\mu k_i$. The following lemma is a generalization of Lemma 6.10 and it allows us to bound the probability that at least one of the extractors $\mathcal{E}^{\mathcal{A}_j}$ succeeds.

Lemma 6.11. *Let $\mathbf{k} = (k_1, \dots, k_\mu)$, $\mathbf{N} = (N_1, \dots, N_\mu) \in \mathbb{N}^\mu$, $t \in \mathbb{N}$, $\mathcal{C}_1, \dots, \mathcal{C}_\mu$ finite sets $\mathcal{C}_i$ with cardinality $N_i \geq k_i$, let $V \colon (\mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu)^t \times \{0,1\}^* \to \{0,1\}$ be the threshold verification function as defined in Equation* (6.11). *Further, let $\mathcal{A}$ be a (probabilistic) algorithm that takes as input a row $(\mathbf{c}^1, \dots, \mathbf{c}^t)$ of columns $\mathbf{c}^j = (c_1^j, \dots, c_\mu^j) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_\mu$ and outputs a string $y \in \{0,1\}^*$. Then*

$$\sum_{j=1}^t \delta_{\mathbf{k}}^{V_j}(\mathcal{A}_j) \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}_s^t(\mathbf{k};\mathbf{N})}{1 - \mathrm{Er}(\mathbf{k};\mathbf{N})}\,,$$

where

$$\mathrm{Er}_s^t(\mathbf{k};\mathbf{N}) = \sum_{\ell=s}^t \binom{t}{\ell} \mathrm{Er}(\mathbf{k};\mathbf{N})^\ell (1 - \mathrm{Er}(\mathbf{k};\mathbf{N}))^{t-\ell}$$

and

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1 - \prod_{i=1}^\mu \left(1 - \frac{k_i - 1}{N_i}\right).$$

Note that $\mathrm{Er}_s^t(\mathbf{k};\mathbf{N})$ is the probability of being successful at least s times when given t trials, when each trial is successful with independent probability $\mathrm{Er}(\mathbf{k};\mathbf{N})$.

Proof. For $1 \leq j \leq t$, let Λ_j denote the event $V_j(C, \mathcal{A}_j(C)) = 1$ and let $S_1^j, S_2^j(\cdot), \dots, S_\mu^j(\cdot)$ such that they minimize Equation 6.12. Moreover, let Γ_j denote the event

$$C_1^j \notin S_1^j \wedge C_2^j \notin S_2^j(C_1^j) \wedge \cdots \wedge C_\mu^j \notin S_\mu^j(C_1^j, \dots, C_{\mu-1}^j)\,.$$

Without loss of generality, we may assume that $|S_1^j| = k_1 - 1$ and

$$S_i^j : \mathcal{C}_1 \times \cdots \mathcal{C}_{i-1} \to \{S \subset \mathcal{C}_i : |S| = k_i - 1\}$$

for all $2 \leq i \leq \mu$ and $1 \leq j \leq t$. Then, for all $1 \leq j \leq t$,

$$\Pr(\Gamma_j) = \prod_{i=1}^\mu \left(1 - \frac{k_i - 1}{N_i}\right) = 1 - \mathrm{Er}(\mathbf{k};\mathbf{N})\,.$$

Moreover, using elementary probability theory,

$$\begin{aligned}\sum_{j=1}^{t} \delta_{\mathbf{k}}^{V_j}(\mathcal{A}_j) &= \sum_{j=1}^{t} \Pr\bigl(\Lambda_j \mid \Gamma_j\bigr) = \sum_{j=1}^{t} \frac{\Pr\bigl(\Lambda_j \wedge \Gamma_j\bigr)}{\Pr\bigl(\Gamma_j\bigr)} = \sum_{j=1}^{t} \frac{\Pr\bigl(\Lambda_j \wedge \Gamma_j\bigr)}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})} \\ &\geq \frac{\Pr\bigl(\exists j : \Lambda_j \wedge \Gamma_j\bigr)}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})} \geq \frac{\Pr\bigl(|\{j : \Lambda_j\}| \geq s \wedge |\{j : \Gamma_j\}| \geq t-s+1\bigr)}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})} \\ &\geq \frac{\Pr\bigl(|\{j : \Lambda_j\}| \geq s\bigr) - \Pr\bigl(|\{j : \Gamma_j\}| \leq t-s\bigr)}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})} \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}_s^t(\mathbf{k};\mathbf{N})}{1-\mathrm{Er}(\mathbf{k};\mathbf{N})}\,,\end{aligned}$$

which completes the proof. □

As before (see Equation 6.7), it follows that the probability of at least one of the extractors $\mathcal{E}^{\mathcal{A}_j}$ being successful is at least

$$\frac{\Delta}{2} \geq \frac{\epsilon^V(\mathcal{A}) - \mathrm{Er}_s^t(\mathbf{k};\mathbf{N})}{2K(1-\mathrm{Er}(\mathbf{k};\mathbf{N}))}\,,$$

where $\Delta = \min\bigl(1, \sum_{j=1}^{t} \delta_{\mathbf{k}}^{V_j}(\mathcal{A}_j)/K\bigr)$ and $K = \prod_{i=1}^{\mu} k_i$. This proves the following threshold parallel repetition result for $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs.

Theorem 6.9 (Threshold Parallel Repetition Theorem). *Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proof. Then the s-out-of-t threshold parallel repetition $\Pi_s^t = (\mathcal{P}_s^t, \mathcal{V}_s^t)$ of Π is knowledge sound with knowledge error*

$$\mathrm{Er}_s^t(\mathbf{k};\mathbf{N}) = \sum_{\ell=s}^{t} \binom{t}{\ell} \mathrm{Er}(\mathbf{k};\mathbf{N})^{\ell} (1-\mathrm{Er}(\mathbf{k};\mathbf{N}))^{t-\ell}\,,$$

where

$$\mathrm{Er}(\mathbf{k};\mathbf{N}) = 1 - \prod_{i=1}^{\mu}\left(1 - \frac{k_i - 1}{N_i}\right),$$

is the knowledge error of Π.

As before, the knowledge error $\mathrm{Er}_s^t(\mathbf{k};\mathbf{N})$ coincides with the trivial cheating probability for Π_s^t, confirming the tightness of Theorem 6.9.

Note that the completeness error of Π_s^t equals

$$\rho_s^t = \sum_{\ell=0}^{s-1} \binom{t}{\ell} \rho^{t-\ell} (1-\rho)^{\ell}\,.$$

Hence, the completeness error ρ_s^t increases and the knowledge error $\mathrm{Er}_s^t(\mathbf{k};\mathbf{N})$ decreases in s. Moreover, it is easily seen that for t large enough and $\mathrm{Er}(\mathbf{k};\mathbf{N}) \cdot t < s < (1-\rho)t$ the threshold parallel repetition Π_s^t has a smaller knowledge and a smaller completeness error than Π, i.e., $\mathrm{Er}_s^t(\mathbf{k};\mathbf{N}) < \mathrm{Er}(\mathbf{k};\mathbf{N})$ and $\rho_s^t < \rho$. In contrast to standard parallel repetition, threshold parallel repetition therefore allows both these errors to be reduced simultaneously.

6.6 Non-Interactivity: Knowledge Extraction under the Fiat-Shamir Transformation

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 message-challenge pairs 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-round 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 $(2\mu+1)$-round interactive proof Π 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. More precisely, a tight reduction is due to [BCS16] 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, [BCS16] 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.[5] 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 $\epsilon = 1/|\mathcal{C}|^\mu$, 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 $\left(\frac{Q}{\mu}\frac{1}{|\mathcal{C}|}\right)^\mu = \frac{Q^\mu}{\mu^\mu} \cdot \epsilon$.[6]

For μ beyond 1 or 2, let alone for non-constant μ (e.g., for compressed Σ-protocols, IOP-based protocols [BCS16; AHI+17; BCR+19] and also other Bulletproofs-like protocols [BCC+16; BBB+18]), 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, affecting its efficiency; alternatively, one has to give up on proven security and simply *assume*

[5]As a matter of fact, [BCS16] considers arbitrary *interactive oracle proofs (IOPs)*, but these notions are well defined for ordinary interactive proofs too.

[6]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.

that the security loss is much milder than what the general bound suggests.

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) [GT21], or are restricted to protocols using hash-based commitment schemes in the random oracle model [BCS16]. To analyze the properties of straight-line extractors, new auxiliary soundness notions were introduced: *round-by-round (RBR) soundness* [CCH+19] and *RBR knowledge* [CMS19]. However, it is unclear if and how these notions can be used in scenarios where straight-line extraction does not apply.

In this section, 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.

6.6.1 Technical Overview

Positive Result. We show that the Fiat-Shamir transformation of any $(k_1, \dots, k_\mu)$-out-of-$(N_1, \dots, N_\mu)$ 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. For any $(k_1, \dots, k_\mu)$-out-of-$(N_1, \dots, N_\mu)$-special-sound $(2\mu + 1)$-round interactive proof Π with knowledge error $\kappa = \mathrm{Er}(k_1, \dots, k_\mu, N_1, \dots, N_\mu)$, the Fiat-Shamir transformed protocol $\mathsf{FS}[\Pi]$ has a knowledge error at most $(Q + 1) \cdot \kappa$.

Since in the Fiat-Shamir transformation of any $(2\mu + 1)$-round 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 run time 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 Section 6.4 in the interactive case, but the analysis here in the context of a non-interactive proof is 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 that are not useful.

To see the core problem, consider a Σ-protocol, i.e., a 3-round k-special-sound interactive proof, and a semi-honest prover that knows a witness and behaves as follows. It prepares, independently, Q first messages $a^1, \dots, a^Q$ 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 run-time.

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 run-time 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 run-time 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 run-time is intricate. If the extractor is recursively composed, i.e., it makes calls to a sub-extractor to obtain a subtree, then a naive construction and analysis gives a blow-up of Q^μ in the run-time. Intuitively, because only $1/Q$ of the sub-extractor runs produce useful subtrees, i.e., subtrees 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 subtree will be (potentially) useful, or not. Thus, if this very first transcript already shows that the subtree will not be useful, there is no need to run the full-fledged subtree 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 subtree (which requires expected time linear in Q) and having reprogrammed the random oracle with the goal to find another subtree 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 subtree. 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 run-time 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.

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 artifact 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 [BLN+20; ACK21]. Concretely, we show that the t-fold parallel repetition Π^t of a typical $(k_1, \dots, k_\mu)$-special-sound $(2\mu+1)$-round 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 $(k_1, \dots, k_\mu)$-special-sound protocol Π there exists a polynomial time Q-query prover $\mathcal{P}^*$ against $\mathsf{FS}[\Pi^t]$ that succeeds in making the verifier accept with probability approximately $\frac{1}{2} Q^\mu \kappa^t / \mu^{\mu+t}$ 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, by Section 6.5, κ^t is the knowledge error of Π^t (i.e., the soundness error of Π^t as an interactive proof), this shows that the knowledge error of Π^t grows proportionally with Q^μ when applying the Fiat-Shamir transformation.

6.6.2 Related Work

Independent Concurrent Work. In independent and to a large extent concurrent work,[7] Wikström [Wik21] achieves a similar positive result on the Fiat-Shamir transformation, using a different approach and different techniques: [Wik21] reduces non-interactive extraction to a form of interactive extraction and then applies a generalized version of [Wik18], while our construction adapts the interactive extractor from Section 6.4 and offers a direct analysis. One difference in the results, which is mainly of theoretical interest, is that our result holds and is meaningful for *any* $Q < |\mathcal{C}|$, whereas [Wik21] requires the challenge set $\mathcal{C}$ to be large.

[7]When finalizing our write-up [AFK22], we were informed by Wikström that he derived similar results a few months earlier, subsequently made available online [Wik21].

The Forking Lemma. The security of the Fiat–Shamir transformation of k-out-of-N special-sound Σ-protocols is widely used for construction of signatures. There, unforgeability is typically proven via a forking lemma [PS96; BN06], 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 [BL02]).

We are not aware of forking lemmas being used in the context of the Fiat–Shamir transformation for multi-round interactive proofs, i.e., for $(2\mu+1)$-round interactive proofs with $\mu > 1$. The 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 run-time.

6.6.3 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 $1 \le j_1, \dots, j_U \le N$, the entry $M(j_1, \dots, j_U) = (v, i)$ contains a bit $v \in \{0,1\}$ and an index $i \in \{1, \dots, U\}$. Think of the bit v indicating whether this entry is "good" or "bad," and the index i pointing 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 $M(j_1, \dots, j_{i-1}, \cdot, j_{i+1}, \dots, j_U)$ for some $1 \le j_1, \dots, j_{i-1}, j_{i+1}, \dots, j_U \le N$.

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 $j_1, \dots, j_U$ specifies the function table of the random oracle

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

while the entry $M(j_1, \dots, j_U) = (v, i)$ 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 Figure 6.4 and its core properties are summarized in Lemma 6.12 below. Looking ahead, we note that for efficiency reasons, the extractor will naturally not sample the entire sequence $j_1, \dots, j_U$ (i.e., function table), but will sample the relevant components on the fly using lazy sampling.

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

$$\begin{aligned} a_i \colon \{1, \dots, N\}^U &\to \mathbb{N}_{\ge 0}, \\ (j_1, \dots, j_U) &\mapsto \left|\{j : \; M(j_1, \dots, j_{i-1}, j, j_{i+1}, \dots, j_U) = (1, i) \;\}\right| . \end{aligned} \tag{6.13}$$

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

Lemma 6.12 (Abstract Sampling Game). *Consider the game in Figure 6.4. 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 $1 \le i \le U$, 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*

$$\mathbb{E}[\Lambda] \le 1 + (k-1)P \quad \textit{and}$$
$$\Pr(X = k) \ge \frac{N}{N-k+1}\left(\Pr(V = 1) - P \cdot \frac{k-1}{N}\right),$$

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

Remark 6.4. Note the abstractly defined parameter P. In our application, where the index i of $(v, i) = M(j_1, \dots, j_U)$ is determined by the output of a prover making no more than Q queries to the random oracle with function table $j_1, \dots, j_U$, the parameter P will be bounded by $Q+1$. We show this formally (yet again somewhat abstractly) in Lemma 6.13. 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, if $a_i(j_1, \dots, j_U) > 0$ for an index i that the algorithm did *not* query, then $M(j_1, \dots, j_U) \in \{(0, i), (1, i)\}$; namely, since i has not been queried, the index i output by the algorithm is oblivious to the value of j_i. Therefore, given $j_1, \dots, j_U$, there is at most one *unqueried* index i with $a_i(j_1, \dots, j_U) > 0$.

Proof (of Lemma 6.12). **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

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

Figure 6.4: Abstract Sampling Game.

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.

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 2.3,

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

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

$$\Pr(V = 1 \wedge I = i \wedge A_i = a) = \Pr(A_i = a)\frac{a}{N}\,. \tag{6.14}$$

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

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

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 (6.14) 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 (6.14) once more, we obtain

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

Hence,

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

where, as before, we have used that $\Pr(V = 1 \wedge I = i \wedge A_i = 0) = 0$ for all $1 \leq i \leq U$, 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 6.13. *Consider the game in Figure 6.4. Let v and* idx *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 $1 \le i \le U$. 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 $\mathsf{idx}(j) = \mathsf{idx}(j')$ for all j' with $j'_\ell = j_\ell$ for all $\ell \in S(j)$. Then*

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

Proof. By basic probability theory, it follows that[8]

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

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 $j \in \{1,\ldots,N\}^U$, $i \notin S(j)$ 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 $i \notin S(j) \cup \{\mathsf{idx}(j)\}$,

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

Hence,

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

Altogether, it follows that

$$P \le Q + \sum_{j} \Pr(J = j) = Q + 1\,,$$

which completes the proof. □

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

6.6.4 The Fiat-Shamir Transformation of Σ-Protocols

Let us first consider the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a k-out-of-N special-sound Σ-protocol Π, i.e., a 3-round interactive proof with challenge set $\mathcal{C}$ of cardinality N. Subsequently, in Section 6.6.6, 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. The random-oracle algorithm $\mathcal{A}$ 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 $\epsilon(\mathcal{A})$ corresponds to the success probability $\epsilon(x, \mathcal{P}^*)$ of the random-oracle prover $\mathcal{P}^*$ on input x.

Our goal is now to construct an extraction algorithm that, when given oracle access to $\mathcal{A}$, aims to output k accepting transcripts $y_1, \dots, y_k$ with common first message a and distinct challenges. By the k-out-of-N special-soundness of Π, a witness for statement x can be computed efficiently from these transcripts. Recall that an extractor with oracle access to a random oracle algorithm is free to choice the answers to the random oracle queries made by the algorithm. However, the answers provided by the extractor must be indistinguishable from those provided by a true random oracle algorithm.

The extractor $\mathcal{E}$ is defined in Figure 6.5. 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 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.[9]

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 $c, c', \dots$

[9]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.

Figure 6.5: Extractor $\mathcal{E}$ for Random Oracle Algorithms.

Parameters: $k, Q \in \mathbb{N}$.
Oracle access to: The $(Q+1)$-query random oracle algorithm $\mathcal{A}$ 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 $y_1, \ldots, y_k$. In the latter case, the algorithm aborts.

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 Figure 6.4. Thus, bounds on the success probability and the expected run time (in terms of queries to $\mathcal{A}$) follow from Lemma 6.12 and Lemma 6.13. Altogether we obtain the following result.

Lemma 6.14 (Extractor for Random Oracle Algorithms)**.** *The extractor $\mathcal{E}$ of Figure 6.5 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 $M(j_1,\ldots,j_U) = \big(I(j_1,\ldots,j_U), v(j_1,\ldots,j_U)\big)$.

Then, a run of the extractor perfectly matches up with the abstract sampling game of Figure 6.4 instantiated with array M. The only difference is that, in this sampling game, we consider full-fledged random oracles encoded by vectors $(j_1, \dots, j_U) \in \{1, \dots, N\}^U$, while the actual extractor implements these random oracles by lazy sampling. Thus, we can apply Lemma 6.12 to obtain bounds on the success probability and the expected run time. However, in order to control the parameter P, which occurs in the bound of Lemma 6.12, we make the following observation, so that we can apply Lemma 6.13 to bound $P \leq Q + 1$.

For every $(j_1, \dots, j_U)$, let $S(j_1, \dots, j_U) \subseteq \{1, \dots, U\}$ be the set of points that $\mathcal{P}^*$ queries to the random oracle when $(j_1, \dots, j_U)$ 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 $i \notin S(j_1, \dots, j_U)$. In particular, $I(j_1, \dots, j_{i-1}, j, j_{i+1}, \dots, j_U) = I(j_1, \dots, j_{i-1}, j', j_{i+1}, \dots, j_U)$ for all j, j' and for all $i \notin S(j_1, \dots, j_U)$. Furthermore, $|S(j_1, \dots, j_U)| \leq Q$. Hence, the conditions of Lemma 6.13 are satisfied and $P \leq Q+1$. The bounds on the success probability and the expected run time now follow, completing the proof. □

The existence of the above extractor, combined with the k-out-of-N special-soundness property, implies the following theorem. This theorem shows that the security loss of the Fiat-Shamir transformation for k-out-of-N Σ-protocols is $Q+1$, i.e., the security loss is linear in the query complexity Q of a prover $\mathcal{P}^*$ attacking the Fiat-Shamir transformation.

Theorem 6.10 (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.

6.6.5 A 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 Section 6.6.3, and present 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 Section 6.6.3 is insufficient for our purposes (resulting in a super-polynomial bound on the run-time of the knowledge extractor). Fortunately, it turns out that a refinement allows us to prove the required (polynomial) upper bound.

In Section 6.6.3, 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 Section 6.6.3, 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 6.15 (Abstract Sampling Game - Weighted Version)**.** *Consider again the game of Figure 6.4, 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 Equation 6.13.*

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 6.5. Note that the parameter T in the statement here differs slightly from its counterpart $P = \sum_i \Pr(A_i > 0)$ in Lemma 6.12. Recall the informal discussion of P in the context of our application (Remark 6.4), 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 6.16.

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 $j^* = (j_1^*,\dots,j_{i-1}^*,j_{i+1}^*,\cdots,j_U) \in \{1,\dots,N\}^{U-1}$, 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 $M(j_1^*,\dots,j_{i-1}^*,\cdot,j_{i+1}^*,\cdots,j_U^*)$; 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 $\Pr(\Lambda(i,j^*)) > 0$,

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

Since, conditioned on $\Lambda(i,j^*) \wedge X = \ell$ for $\ell \in \{0,\dots,N\}$, any size-ℓ 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 6.12. Bounding $\mathbb{E}[X \mid \Lambda(i,j^*)]$ is quite easy: since $V = 0$ implies $X = 0$ and $V = 1$ implies $X \le k$, it immediately follows that

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

Hence,

$$\mathbb{E}[\Delta_1 \mid \Lambda(i,j^*)] \le k \cdot \Pr(V = 1 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i,j^*)] \,. \tag{6.15}$$

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 $1 \le i \le U$ and of $j^* = (j_1^*,\dots,j_{i-1}^*,j_{i+1}^*,\cdots,j_U^*)$, we let[10]

$$\begin{aligned}a &:= a_i(j^*) = \big|\{j : \ (v_j,i_j) = M(j_1^*,\dots,j_{i-1}^*,j,j_{i+1}^*,\dots,j_U^*) = (1,i) \ \}\big| \quad \text{and} \\ b &:= b_i(j^*) := \big|\{j : \ (v_j,i_j) = M(j_1^*,\dots,j_{i-1}^*,j,j_{i+1}^*,\dots,j_U^*) = (0,i) \ \}\big| \,.\end{aligned}$$

Let us first note that

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

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

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 2.2). Hence, by Lemma 2.3,

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

and thus

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

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

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

and thus, combined with Equation 6.15,

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

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

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

What remains is to show that $\mathbb{E}[Z] \le (k-1)T$, from which it follows that $\mathbb{E}[\Delta_3] = \gamma\mathbb{E}[Z] \le (k-1)T\gamma$. The slightly weaker bound $\mathbb{E}[Z] \le (k-1)P$ follows immediately from observing that $Z \le Y'$ for Y' as in the proof of Lemma 6.12 (the number of entries counted by Z is a subset of those counted by Y'), and using that $\mathbb{E}[Y'] \le \mathbb{E}[X' + Y'] \le (k-1)P$ as derived in the proof of Lemma 6.12. 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 $\Pr\big(V = 1 \wedge \Lambda(i,j^*)\big) > 0$,

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

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

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

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 6.16. *Consider the game in Figure 6.4. 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 $1 \leq i \leq U$ as in Equation 6.13. 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*

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

Proof. The proof is analogous to the proof of Lemma 6.13. By basic probability theory, it follows that

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

where the inequality follows from the fact that $|S(j)| \leq Q$ 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

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

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

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

Hence,

$$\begin{aligned}
&\sum_{i \notin S(j)} \Pr\big(\mathsf{idx}(J) \neq i \wedge A_i > 0 \mid J = j\big) \\
&\qquad\qquad \leq \Pr\big(\mathsf{idx}(J) \neq \mathsf{idx}(j) \wedge A_{\mathsf{idx}(j)} > 0 \mid J = j\big) = 0 \,.
\end{aligned}$$

Altogether, it follows that

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

which completes the proof. □

6.6.6 The 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 $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound $(2\mu + 1)$-round interactive proof Π, with $\mathbf{k} = (k_1, \dots, k_\mu)$. 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, we first handle $(2\mu + 1)$-round interactive proofs in which the verifier samples all μ challenges uniformly at random from the *same* set $\mathcal{C}$. Subsequently, we argue that our techniques have a straightforward generalization to interactive proofs where the verifier samples its challenges from different challenge sets.

Multi-Round Interactive Proofs 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 $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proof Π on input x. We assume all challenges to be elements of 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, $\epsilon(\mathcal{A}) = \epsilon(x, \mathcal{P}^*)$.

Our goal is now to construct an extraction algorithm that, when given oracle access to $\mathcal{A}$, and thus to $\mathcal{P}^*$, aims to output a $\mathbf{k}$-tree of accepting transcripts (Definition 2.33). By the $\mathbf{k}$-out-of-$\mathbf{N}$ special-soundness 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-round 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-round 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.

Figure 6.6: Sub-extractor $\mathcal{E}_m$, as a $(Q+\mu)$-query random-oracle algorithm.

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

- 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 $v := 0$.

Formally, the recursive definition of $\mathcal{E}_m$ from $\mathcal{E}_{m+1}$ is given in Figure 6.6, 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 6.17 below). This allows the executions of $\mathcal{E}_{m+1}$ in the repeat loop in Fig. 6.6 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 6.6. 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 that 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 6.1 shows that $\mathcal{E}_m$, if successful, indeed outputs a $(1,\dots,1,k_m\dots,k_\mu)$-tree of accepting transcripts. Proposition 6.2 bounds the success probability and expected run time of $\mathcal{E}_m$. All statements are understood to hold for any statement x and any $m \in \{1,\dots,\mu+1\}$.

Lemma 6.17 (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 6.1 (Correctness)**.** *For any fixed choice of the random oracle let $(\mathbf{I}, y_1,\dots,y_{k_m}, v) \leftarrow \mathcal{E}_m^{\mathsf{RO}}(x)$. If $v = 1$ then $(y_1,\dots,y_{k_m})$ forms a $(1,\dots,1,k_m,\dots,k_\mu)$-tree of accepting transcripts.*

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

By induction on m, we will prove that if $v = 1$ then $(y_1,\dots,y_{k_m})$ forms a $(1,\dots,1,k_m,\dots,k_\mu)$-tree of accepting transcripts with trunk $(a_1,\mathsf{RO}(I_1),\dots,\mathsf{RO}(I_{m-1}),a_m)$, where $I_j = (a_1,\dots,a_j)$. 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 $(1,\dots,1)$-tree of accepting transcripts with $(a_1,\mathsf{RO}(I_1),\dots,\mathsf{RO}(I_\mu),a_{\mu+1})$ as trunk by definition of $\mathbf{I} = (I_1,\dots,I_\mu)$.

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 $(1,\dots,1,k_{m+1},\dots,k_\mu)$-tree of accepting transcripts with trunk $(a_1,\mathsf{RO}(I_1),\dots,a_m,\mathsf{RO}(I_m),a_{m+1})$, where $I_{m+1} = (a_1,\dots,a_{m+1})$. 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 $(a_1',\mathsf{RO}'(I_1'),\dots,a_m',\mathsf{RO}'(I_m'),a_{m+1}')$, where 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 6.17, 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 $y_1,\dots,y_{k_m}$ of $\mathcal{E}_m$ forms a $(1,\dots,1,k_m,\dots,k_\mu)$-tree of accepting transcripts with trunk $(a_1,\mathsf{RO}(I_1),\dots,a_{m-1},\mathsf{RO}(I_{m-1}),a_m)$, where $I_m = (a_1,\dots,a_m)$. This completes the proof. □

Proposition 6.2 (Run Time and Success Probability)**.** *Let $K_m = \prod_{j=m}^{\mu} k_j$. 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

$$\kappa_m := \mathrm{Er}(k_m, \ldots, k_\mu; N, \ldots, N) = 1 - \prod_{i=m+1}^{\mu} \left(1 - \frac{k_i - 1}{N}\right).$$

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 one 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 6.14.

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-round 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 are 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 Figure 6.4.

However, in contrast to the instantiation of Section 6.6.4, 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. Namely, for the purpose of the analysis, we may fix the randomness of the extractor $\mathcal{E}_{m+1}$. By linearity of the success probability and the expected run time, the bounds that hold for any fixed choice of randomness also hold when averaged over the randomness. Thus, we can apply Lemma 6.12 and Lemma 6.15 to bound the success probability and the expected run time.[11]

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 6.14.

First, by Lemma 6.17, 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

[11] 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 Lemma 6.12 and Lemma 6.15.

the random oracle is reprogrammed at an index $i \notin S(j_1, \dots, j_U)$. In particular, $\mathbf{I}(j_1, \dots, j_{i-1}, j, j_{i+1}, \dots, j_U) = \mathbf{I}(j_1, \dots, j_{i-1}, j', j_{i+1}, \dots, , j_U)$ for all j, j' and for all $i \notin S(j_1, \dots, j_U)$. Furthermore, $|S(j_1, \dots, j_U)| \leq Q$. Hence, the conditions of Lemma 6.13 and Lemma 6.16 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 Lemma 6.12 and Lemma 6.13, the success probability of $\mathcal{E}_m$ is bounded by

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

Now observe that, for $\kappa_m = \mathrm{Er}\big(k_m, \dots, k_\mu; N, \dots, N\big)$, the following recursive property is easily derived:

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

Hence,

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

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 6.17, 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 $(j_1, \dots, j_U) \mapsto \Gamma(j_1, \dots, j_U)$, where $\Gamma(j_1, \dots, j_U)$ is the (expected) costs of running $\mathcal{E}_{m+1}$ (completely) with random oracle $(j_1, \dots, j_U)$. 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 Lemma 6.15 and Lemma 6.16, 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 $\mathbf{k}$-special-soundness property, implies Theorem 6.11. This theorem shows that the Fiat-Shamir security loss for $\mathbf{k}$-out-of-$\mathbf{N}$ 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.

Theorem 6.11 (Fiat-Shamir Transformation of a Multi-Round Interactive Proof with a Single Challenge Set)**.** *Let* $\mathbf{k} = (k_1, \dots, k_\mu)$, $\mathbf{N} = (N, \dots, N) \in \mathbb{N}^\mu$. *The Fiat-Shamir transformation* $\mathsf{FS}[\Pi]$ *of a* $\mathbf{k}$*-out-of-*$\mathbf{N}$ *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*

$$(Q + 1) \cdot \mathrm{Er}(\mathbf{k}; \mathbf{N})\,,$$

where

$$\mathrm{Er}(\mathbf{k}; \mathbf{N}) = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N}\right)$$

is the knowledge error of the interactive proof Π.

Multi-Round Interactive Proofs 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 6.11.

Theorem 6.12 (Fiat-Shamir Transformation of a Multi-Round Interactive Proof)**.** *Let* $\mathbf{k} = (k_1, \dots, k_\mu) \in \mathbb{N}^\mu$ *and* $\mathbf{N} = (N_1, \dots, N_\mu) \in \mathbb{N}^\mu$*. The Fiat-Shamir transformation of a* $\mathbf{k}$*-out-of-*$\mathbf{N}$ *special-sound interactive proof* Π *is knowledge sound with knowledge error* $(Q+1) \cdot \mathrm{Er}(\mathbf{k}; \mathbf{N})$*, where*

$$\mathrm{Er}(\mathbf{k}; \mathbf{N}) := 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right)$$

is the knowledge error of the interactive proof Π*.*

6.6.7 An Attack on the Fiat-Shamir Transformation of a Parallel Repetition

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 number of queries Q admitted to a prover $\mathcal{P}^*$ attacking the considered non-interactive random oracle proof. One might therefore wonder whether the generic security loss for $(2\mu + 1)$-round interactive proofs, roughly equal to Q^μ, is only tight for contrived examples. In this section, we show that this is not the case. We demonstrate a nontrivial attack on the Fiat–Shamir transformation of the *parallel repetition* of $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs.

Recall that typical $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs Π admit a cheating strategy that succeeds if at least one of the μ random challenges c_i, received from the verifier, hits a certain set Γ_i of size $k_i - 1$ chosen by the dishonest prover. The success probability of this cheating strategy matches the knowledge error

$$\mathrm{Er}(\mathbf{k}; \mathbf{N}) = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right).$$

A straightforward analysis shows that this approach generalizes to a cheating strategy for the t-fold parallel repetition $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ of Π, with success probability $\mathrm{Er}(\mathbf{k}; \mathbf{N})^t$ again matching the knowledge error (now of Π^t).

The following (informal) theorem shows the existence of an attack strategy for the Fiat-Shamir transformation of Π^t that succeeds with probability roughly $Q^\mu / \mu^{t+\mu} \cdot \mathrm{Er}(\mathbf{k}; \mathbf{N})^t$. In particular, the security loss of the Fiat-Shamir transformation, when applied to the t-fold parallel repetition Π^t, is roughly $Q^\mu / \mu^{t+\mu}$. This stands in stark contrast to a single execution of a $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound protocol, where the loss is linear in Q and independent of μ. The main idea of this attack is that a dishonest prover $\mathcal{P}^*$ can attack different groups of parallel instances in different rounds of the protocol *independently*. More precisely, in every round the dishonest prover $\mathcal{P}^*$ attacks t/μ parallel instances.

In order to focus on the crucial aspects of the attack, the theorem is stated informally, allowing us to avoid certain cumbersome details. First, we do not formalize the properties required by the basic interactive proof Π and merely state that this attacks applies to "typical" $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs. Informally, our attack applies to interactive proofs where:

1. the aforementioned cheating strategy, with success probability $\mathrm{Er}(\mathbf{k},\mathbf{N})$, applies;
2. in the Fiat-Shamir mode the prover $\mathcal{P}^*$ can try sufficiently many message-challenge pairs in every round of the protocol.

The second property ensures that if, at some point during the attack, the random oracle returns a challenges c that does not hit the subset specified by the dishonest prover $\mathcal{P}^*$, i.e., this phase of the attack fails, then $\mathcal{P}^*$ can simply try again by querying the random oracle with a different input value. Typical $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs admit both properties. However, there exist (artificial) counterexamples. Moreover, we only give an approximation of the success probability, and the accuracy of this approximation is not discussed. For a more formal treatment of this attack we refer the reader to the article [AFK22], co-authored by Serge Fehr and Michael Klooß, on which this section is based.

Theorem 6.13 (Informal - Attack on the Fiat-Shamir Transformation of a Parallel Repetition)**.** *The Fiat-Shamir transformation of* $\mathsf{FS}[\Pi^t]$ *of the t-fold parallel repetition* Π^t *of a "typical"* $\mathbf{k}$*-out-of-*$\mathbf{N}$ *special-sound interactive proof* Π *admits a Q-query cheating strategy that succeeds with probability "roughly"*

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

where $\mathrm{Er}(\mathbf{k};\mathbf{N})$ *is the knowledge error of* Π *and, thus,* $\mathrm{Er}(\mathbf{k};\mathbf{N})^t$ *is the knowledge error of* Π^t*.*

Proof. For simplicity, let us assume $\mathbf{k} = (k,\dots,k)$ and $\mathbf{N} = (N,\dots,N)$ for some $k, N \in \mathbb{N}$, and assume t and Q to be multiples of μ, i.e., $t = \mu \cdot t'$ and $Q = \mu \cdot Q'$ for some $t', \mu' \in \mathbb{N}$. For a more general treatment we refer to [AFK22].

The main idea of the cheating strategy is that a cheating prover $\mathcal{P}^*$ attacks t' parallel instances of Π in every round of the protocol. The attacks in the different rounds can be executed independently.

More precisely, the cheating strategy proceeds as follows. In the first round, the cheating prover $\mathcal{P}^*$ chooses random first messages $a_1^1,\dots,a_1^{t'}$ together with subsets $\Gamma_1,\dots,\Gamma_{t'} \subseteq \mathcal{C}_1$ of cardinality $k-1$, such that the following holds. If the first challenge c_1^j for instance $1 \le j \le t'$ lands in Γ_j, then $\mathcal{P}^*$ is able to honestly complete the execution of instance j and have the verifier accept that instance. Recall that typical $\mathbf{k}$-out-of-$\mathbf{N}$ special-sound interactive proofs admit a cheating strategy following precisely this approach. The first messages $a_1^{t'+1},\dots,a_1^{\mu}$ for the remaining $t-t'$ parallel instances are chosen at random. Then, the prover $\mathcal{P}^*$ queries the random oracle to receive the first round challenges $c_1^1,\dots,c_1^t \in \mathcal{C}_1$ for all parallel instances. This step of the attack succeeds if $c_1^j \in \Gamma_j$ for all $1 \le j \le t'$, i.e., if the first t' challenges land in the previously specified subsets Γ_j, which happens with probability $(k-1)^{t'}/N^{t'}$. If this step of the attack has not succeeded, $\mathcal{P}^*$ rewinds to the start of the first round, chooses new first messages and proceeds as before. The cheating prover $\mathcal{P}^*$ tries to attack this round at most Q' times and therefore succeeds in doing so with probability

$$1 - \left(1 - \left(\frac{k-1}{N}\right)^{t'}\right)^{Q'} \approx Q' \cdot \left(\frac{k-1}{N}\right)^{t'},$$

where the approximation holds if $Q' \ll N^{t'}/(k-1)^{t'}$.

If the attack of the first round has succeeded, $\mathcal{P}^*$ moves to the second round and tries to attack parallel instances $t'+1, \dots, 2t'$ in a similar manner, again succeeding with probability roughly $Q' \cdot (k-1)^{t'}/N^{t'}$. While doing so $\mathcal{P}^*$ generates the messages for parallel instances $1, \dots, t'$ honestly and samples the messages for instances $2t'+1, \dots, t$ randomly. The cheating prover $\mathcal{P}^*$ continues until it has either aborted or successfully attacked all t parallel instances.

In every round, $\mathcal{P}^*$ makes at most Q' random oracle queries. Therefore, $\mathcal{P}^*$ is a $Q' \cdot \mu = Q$-query random oracle algorithm. Moreover, this attack strategy succeeds with probability roughly

$$\left(Q' \cdot \left(\frac{k-1}{N}\right)^{t'}\right)^{\mu} = \left(\frac{Q}{\mu}\right)^{\mu} \cdot \left(\frac{k-1}{N}\right)^{t}.$$

The observation that

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

completes the proof of this informal theorem. □