The preliminary version of this paper appeared in *Proceedings of the 2nd ACM ASIA Public-Key Cryptography Workshop - ASIAPKC 2014*, pp. 49-58, under the title "Attribute-Based Signatures without Pairings via the Fiat-Shamir Paradigm". This is a corrected version. We removed the proposed attribute-based identification and signature schemes because they have only one-time attribute privacy. Instead, we mentioned that our protocol serves as building blocks of cryptographic primitives for monotone predicates.

# Proofs of Knowledge on Monotone Predicates and its Application to Attribute-Based Identifications and Signatures

Hiroaki Anada[1], Seiko Arita[2], and Kouichi Sakurai[3]

[1] Department of Information Security, University of Nagasaki
W408, 1-1-1, Manabino, Nagayo-cho, Nishisonogi-gun, Nagasaki, 851-2195 Japan
anada@sun.ac.jp
[2] Institute of Information Security
509, 2-14-1, Tsuruya-cho, Kanagawa-ku, Yokohama, 221-0835 Japan
arita@iisec.ac.jp
[3] Department of Informatics, Kyushu University
W2-712, 744, Motooka, Nishi-ku, Fukuoka, 819-0395 Japan
sakurai@inf.kyushu-u.ac.jp

June 23, 2020

**Abstract.** We propose a concrete procedure of the $\Sigma$-protocol introduced by Cramer, Damgård and Schoenmakers at CRYPTO '94, which is for proving knowledge that a set of witnesses satisfies a monotone predicate in witness-indistinguishable way; that is, hiding the assignment of truth in the predicate. We provide a detailed procedure by extending the so-called OR-proof.

**Keywords:** proof of knowledge, sigma-protocol, OR-proof, witness indistinguishability

## 1 Introduction

A $\Sigma$-protocol formalized in the doctoral thesis of Cramer [Cra96] is a protocol of a 3-move public-coin interactive proof system which satisfies the three requirements of completeness, special soundness and honest-verifier zero-knowledge. It is one of the simplest protocols of zero-knowledge interactive proof systems, which have an easy but special simulator under the condition that the verifier is honest. On the other hand, it is one of the most typical proof of knowledge systems [BG92]; the knowledge-extraction property by the special soundness enables us to prove that an identification scheme derived from the $\Sigma$-protocol is secure against active and concurrent attacks reducing the security to some hardness assumptions [BP02]. For example, the Schnorr protocol [Sch89] and the Guillou-Quisquater protocol [GQ88] of identification schemes have been known as applications of the instantiated $\Sigma$-protocols. Moreover, the identification schemes are able to be converted into digital signature schemes by the Fiat-Shamir heuristic [FS86]. The signature schemes can be proved secure against chosen-message attacks in the random oracle model [PS96] based on the passive security of the identification schemes [AABN02]. By virtue of these three features, a $\Sigma$-protocol have been adopted into building blocks of various cryptographic primitives such as anonymous credential systems [CL02] and group signature schemes [BBS04]. Post-quantum study on $\Sigma$-protocols and their non-interactive versions in

the quantum random oracle model (the QROM model) were studied [Unr12,Unr15], and the soundness and proof-of-knowledge property were proved in the QROM model [DFMS19,LZ19]. Concrete constructions such as the lattice-based construction (for example, [BBC+18]) make an active area of research.

The OR-proof protocol for a $\Sigma$-protocol, which was proposed by Cramer, Damgård and Schoenmakers at CRYPTO '94 [CDS94], is a $\Sigma$-protocol derived from the original $\Sigma$-protocol [CDS94,Dam10]. It is a perfectly witness-indistinguishable protocol [FS90] by which a prover can convince a verifier that a prover knows one of the two or both witnesses while even an unbounded distinguisher cannot tell which witness is used. The OR-proof is essentially applied in, for example, the construction of a non-malleable proof of plaintext knowledge [Kat03]. In the papers [CDS94,BL88][1], a more general protocol was proposed: Suppose a prover and a verifier are given a monotone predicate $f$ over a polynomial number of boolean variables. Here a monotone predicate means a boolean-valued function which is a boolean formula without negation; that is, as a boolean formula, boolean variables of $f$ are connected by AND-gates or OR-gates, but no NOT-gate is used. As a predicate, '1' (TRUE) is assigned into every variable in $f$ at which the prover knows the corresponding witness, and '0' (FALSE) is assigned into every remaining variable. The protocol attains the perfect witness indistinguishability over all satisfying assignment patterns in the sense that the prover is able to prove that she knows one of the patterns of witnesses while even an unbounded distinguisher cannot tell which pattern is used. This protocol is an extension of the OR-proof to any monotone predicate, and in [CDS94] a high-level construction was given by using a dual access structure and a "semi-smooth" secret-sharing scheme. (As is stated in [CDS94], to remove the restriction of the monotonicity of $f$ looks difficult.)

### 1.1 Our Contribution and Related Works

In this paper, we provide a concrete procedure of the $\Sigma$-protocol for any monotone formula, which was proposed by Cramer, Damgård and Schoenmakers [CDS94], according to the secret sharing scheme of Benaloh and Leichter [BL88]. Given a $\Sigma$-protocol $\Sigma$ and a monotone predicate $f$, we construct a $\Sigma$-protocol $\Sigma^f$, concretely. Then we show that the protocol $\Sigma^f$ realized by our procedure is actually a $\Sigma$-protocol with the perfect witness indistinguishability.

Explanation for the relation to attribute-based cryptographic primitives should be in order[2]. Herranz [Her14] provided the first attribute-based identification scheme (ABID) and attribute-based signature scheme (ABS) which attain both the collusion resistance (against collecting private secret keys) and the computational attribute privacy *without pairings (pairing-free)* in the RSA setting. Recently, Herranz [Her16a] provided pairing-free ABID and ABS schemes in the discrete-logarithm setting with a constraint that the number of private secret keys is bounded in the set-up phase. In the ABID and ABS schemes [Her14,Her16a] $\Sigma$-protocols are used and described for the threshold-type predicates. Our concrete procedure of the $\Sigma$-protocol $\Sigma^f$ can serve as the building blocks of the schemes for *any monotone* predicates including the threshold-type. More generically, our concrete procedure can be used as a replacement of a $\Sigma$-protocol used in a cryptographic primitive for the purpose of treating any monotone predicates (for example, [Ana18,AA18]).

### 1.2 Our Construction Idea

To construct a concrete procedure of the $\Sigma$-protocol $\Sigma^f$ with the perfect witness indistinguishability, we look into the technique employed in the OR-proof [CDS94] and expand it so that it can treat any monotone predicate, as follows. First express the boolean formula $f$ as a binary tree $\mathcal{T}^f$. That is, we put leaves each of which corresponds to each position of a variable in $f$. We connect two leaves by an $\wedge$-node or an $\vee$-node according to an AND-gate or an OR-gate which is between the two corresponding positions in $f$. Then we connect the resulting nodes by an $\wedge$-node or an $\vee$-node in the same way until we reach the root node (which is also an $\wedge$-node or an $\vee$-node). A verification equation of the given $\Sigma$-protocol $\Sigma$ is assigned to

[1] The authors would like to express their sincere apologies to the fact that they could not refer to these papers in the conference version [AAS14] of this ePrint.

[2] In the conference version [AAS14] of this ePrint, we explained that we attained the collusion resistance in the construction of ABID and ABS schemes by a naive application of the credential bundle technique [MPR11]. But instead, we partially lost the *attribute privacy* in the ABID and the ABS schemes though the attribute privacy was *wrongly* claimed in [AAS14].

every leaf. If a challenge string CHA of our $\Sigma$-protocol $\Sigma^f$ is given by the verifier, then the prover assigns the string CHA to the root node. If the root node is an $\wedge$-node, then the prover assigns the same string CHA to the two children. Else if the root node is an $\vee$-node, then the prover divides CHA into two random strings $\text{CHA}_\text{L}$ and $\text{CHA}_\text{R}$ under the constraint that $\text{CHA} = \text{CHA}_\text{L} \oplus \text{CHA}_\text{R}$, and assigns $\text{CHA}_\text{L}$ and $\text{CHA}_\text{R}$ to the left child and the right child, respectively. Here $\oplus$ means a bitwise exclusive-OR operation. Then the prover continues to apply this rule at each height, step by step, until she reaches all the leaves. Basically, the OR-proof technique assures that, at every leaf, we can either honestly execute the $\Sigma$-protocol $\Sigma$ or execute the simulator of $\Sigma$. Only when a set of witnesses satisfies the binary tree $\mathcal{T}^f$, the above procedure succeeds in satisfying verification equations at all the leaves.

### 1.3 Organization of this Paper

In Section 2, we prepare for required notions and notations. In Section 3, we describe a concrete procedure of the $\Sigma$-protocol $\Sigma^f$. In Section 4, we conclude our work re-stating our contribution.

## 2 Preliminaries

The security parameter is denoted by $\lambda$. The bit length of a string $a$ is denoted by $|a|$. The concatenation of a string $a$ with a string $b$ is denoted by $a \parallel b$. A uniform random sampling of an element $a$ from a set $S$ is denoted as $a \in_R S$. The expression $a =_? b$ returns a value 1 (TRUE) when $a = b$ and 0 (FALSE) otherwise. The expression $a \in_? S$ returns a value 1 when $a \in S$ and 0 otherwise. When an algorithm $A$ with input $a$ outputs $z$, we denote it as $z \leftarrow A(a)$, or, $A(a) \rightarrow z$. When a algorithm $A$ with input $a$ and a algorithm $B$ with input $b$ interact with each other, we denote the transcript of the messages as $\langle A(a), B(b) \rangle$.

Let $R = \{(x, w)\} \subset \{0,1\}^* \times \{0,1\}^*$ be a binary relation. We say that $R$ is polynomially bounded if there exists a polynomial $\ell(\cdot)$ such that $|w| \leq \ell(|x|)$ for any $(x, w) \in R$. We say that $R$ is an NP relation if it is polynomially bounded and there exists a polynomial-time algorithm for deciding membership of $(x, w)$ in $R$. For a pair $(x, w) \in R$ we call $x$ a statement and $w$ a witness of $x$. An NP language for an NP relation $R$ is defined as: $L \stackrel{\text{def}}{=} \{x \in \{0,1\}^*; \exists w \in \{0,1\}^*, (x, w) \in R\}$. We introduce a *relation function* $R(\cdot, \cdot)$ associated with the relation $R$ by: $R(\cdot, \cdot) : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}$, $(x, w) \mapsto 1$ if $(x, w) \in R$, and 0 otherwise. The function $R(\cdot, \cdot)$ is polynomial-time in $|x|$ as an algorithm. We denote the set of witnesses of a statement $x$ by $W(x)$.

We denote an interactive proof system for an NP relation $R$ [Bab85,GMR85] as $\Pi = (\mathtt{P}, \mathtt{V})$, where $\mathtt{P}$ and $\mathtt{V}$ are a pair of interactive Turing machines, which are called a prover and a verifier, respectively. In this paper, not only $\mathtt{V}$ but also $\mathtt{P}$ are assumed to be probabilistic polynomial-time (PPT). That is, $\Pi = (\mathtt{P}, \mathtt{V})$ is an interactive argument system.

### 2.1 $\Sigma$-protocol, Witness-Indistinguishability and OR-proof

**$\Sigma$-protocol [Cra96,Dam10]** Let $R$ be an NP relation. A *$\Sigma$-protocol* $\Sigma$ on a relation $R$ is a 3-move public-coin protocol of an interactive proof system $\Pi = (\mathtt{P}, \mathtt{V})$ [Cra96,Dam10]. $\mathtt{P}$ sends the first message called a commitment COM to $\mathtt{V}$. Then $\mathtt{V}$ sends the second message called a challenge CHA to $\mathtt{P}$, which is a public random string. Then $\mathtt{P}$ sends the third message called a response RES to $\mathtt{V}$. Then $\mathtt{V}$ applies a decision test to $(x, \text{COM}, \text{CHA}, \text{RES})$ to return 1 (accept) or 0 (reject). If $\mathtt{V}$ accepts, then the triple $(\text{COM}, \text{CHA}, \text{RES})$ is said to be an *accepting transcript on $x$*. The challenge CHA is chosen uniformly at random from the challenge space $\text{CHASP}(1^\lambda) := \{0,1\}^{l(\lambda)}$ with $l(\cdot)$ being a super-log function (i.e. $l(\lambda) = \omega(\log(\lambda))$). To state the requirements for the $\Sigma$-protocol $\Sigma$, we introduce the following notation of the six PPT algorithms of the protocol $\Sigma$: $\Sigma = (\Sigma_{\text{com}}, \Sigma_{\text{cha}}, \Sigma_{\text{res}}, \Sigma_{\text{vrf}}, \Sigma_{\text{ext}}, \Sigma_{\text{sim}})$. The first algorithm $\Sigma_{\text{com}}$ is described as $(\text{COM}, St) \leftarrow \Sigma_{\text{com}}(x, w)$. That is, on input $(x, w) \in R$, it generates a commitment message COM and outputs its inner state $St$. The second algorithm CHA, reading the size of the security parameter as $1^\lambda$ on input the statement $x$, it chooses a challenge message $\text{CHA} \in_R \text{CHASP}(1^\lambda)$ and returns as $\text{CHA} \leftarrow \Sigma_{\text{cha}}(x)$. Similarly, the third and the forth algorithms are described as $\text{RES} \leftarrow \Sigma_{\text{res}}(x, w, St, \text{COM}, \text{CHA})$ and $b \leftarrow \Sigma_{\text{vrf}}(x, \text{COM}, \text{CHA}, \text{RES})$, respectively. $\Sigma$ must satisfy the following three requirements.

*Completeness.* A prover $\mathtt{P}(x, w)$ with a witness $w \in W(x)$ makes $\mathtt{V}(x)$ accept with the probability 1.

The fifth algorithm is described as follows.
*Special Soundness.* There is a PPT algorithm called a *knowledge extractor* $\Sigma_{\mathrm{ext}}$, which, given as input a statement $x$ and two accepting transcripts $(\mathrm{COM}, \mathrm{CHA}, \mathrm{RES})$ and $(\mathrm{COM}, \mathrm{CHA}', \mathrm{RES}')$, computes a witness $\hat{w}$ satisfying $(x, \hat{w}) \in R$ with an overwhelming probability, where the two challenges CHA and CHA′ are different $(\mathrm{CHA} \neq \mathrm{CHA}')$:

$$\hat{w} \leftarrow \Sigma_{\mathrm{ext}}(x, \mathrm{COM}, \mathrm{CHA}, \mathrm{RES}, \mathrm{CHA}', \mathrm{RES}').$$

The sixth algorithm is described as follows.
*Honest-Verifier Zero-Knowledge.* For any fixed statement $x$ there is a PPT algorithm called a *simulator* $\Sigma_{\mathrm{sim}}$ such that

$$(\tilde{\mathrm{CHA}}, \tilde{\mathrm{COM}}, \tilde{\mathrm{RES}}) \leftarrow \Sigma_{\mathrm{sim}}(x),$$

where the distribution of (simulated) transcripts $\{(\tilde{\mathrm{COM}}, \tilde{\mathrm{CHA}}, \tilde{\mathrm{RES}})\}$ is the same as the distribution of (real) accepting transcripts $\{(\mathrm{COM}, \mathrm{CHA}, \mathrm{RES})\}$ generated as $\langle \mathtt{P}(x, w), \mathtt{V}(x) \rangle$ for any fixed witness $w \in W(x)$ and for the (honest) verifier $\mathtt{V}$.

For a $\Sigma$-protocol, the above simulator $\Sigma_{\mathrm{sim}}(x)$ is modified as follows. First generate a challenge $\tilde{\mathrm{CHA}}$ by running $\Sigma_{\mathrm{cha}}(1^\lambda)$ (i.e. uniform random sampling from $\mathrm{CHASP}(1^\lambda)$), and then input the challenge $\tilde{\mathrm{CHA}}$ to the modified simulator to generate a commitment $\tilde{\mathrm{COM}}$ and a response $\tilde{\mathrm{RES}}$:

$$\tilde{\mathrm{CHA}} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda), \ (\tilde{\mathrm{COM}}, \tilde{\mathrm{RES}}) \leftarrow \Sigma_{\mathrm{sim}}(x, \tilde{\mathrm{CHA}}).$$

This modification is justified due to the fact that the challenge CHA is a public coin.

We note that an interactive proof system $\Pi = (\mathtt{P}, \mathtt{V})$ with a $\Sigma$-protocol is known to be a proof of knowledge system. (For the notion of a proof of knowledge system, see [BG92].)

**Witness-Indistinguishability [FS90,Gol01]** Let $R$ be an NP relation. Suppose that an interactive proof system $\Pi = (\mathtt{P}, \mathtt{V})$ for the relation $R$ is given. Suppose further that the proof system $\Pi$ is with a $\Sigma$-protocol $\Sigma$ for the relation $R$ so that we can set the completeness and the special soundness. We focus into the following property.
*Witness Indistinguishability.* For any PPT algorithm $\mathtt{V}^*$, any sequences $W^0 = (w_x^0)_{x \in L}$ and $W^1 = (w_x^1)_{x \in L}$ s.t. $w_x^0, w_x^1 \in W(x)$, any PPT algorithm $D$, any polynomial $\mathrm{poly}(\cdot)$, any sufficiently long string $x \in L$ and any string $z \in \{0,1\}^*$,

$$\begin{aligned} &\Pr[D\big(x, z, \langle \mathtt{P}(x, w_x^0), \mathtt{V}^*(x, z) \rangle\big) = 1] \\ &- \Pr[D\big(x, z, \langle \mathtt{P}(x, w_x^1), \mathtt{V}^*(x, z) \rangle\big) = 1] < \frac{1}{\mathrm{poly}(|x|)}. \end{aligned}$$

The interactive proof system $\Pi$ with the above property is said to be a *witness-indistinguishable proof system* (WI, for short). A stronger notion is the perfect witness indistinguishability. If for any PPT algorithm $\mathtt{V}^*$, any sequences $W^0 = (w_x^0)_{x \in L}$ and $W^1 = (w_x^1)_{x \in L}$ s.t. $w_x^0, w_x^1 \in W(x)$, any string $x \in L$ and any string $z \in \{0,1\}^*$ the two distributions $\{(x, z, \langle \mathtt{P}(x, w_x^0), \mathtt{V}^*(x, z) \rangle)\}$ and $\{(x, z, \langle \mathtt{P}(x, w_x^1), \mathtt{V}^*(x, z) \rangle)\}$ are identical, then the interactive proof system $\Pi$ is said to be a *perfectly witness-indistinguishable* proof system (in short, perfectly WIPoK).

**OR-proof [Dam10]** Let $R$ be an NP relation. Suppose that a boolean formula $f(X_0, X_1) = X_0 \vee X_1$ is given, and fix the following relation.

$$\begin{aligned} R_{\mathrm{OR}} = &\{(x = (x_0, x_1), w = (w_0, w_1)) \in (\{0,1\}^*)^2 \times (\{0,1\}^*)^2; \\ &f(R(x_0, w_0), R(x_1, w_1)) = 1\}. \end{aligned}$$

The corresponding language is

$$L_{\mathrm{OR}} = \{x \in (\{0,1\}^*)^2; \exists w \in (\{0,1\}^*)^2, (x, w) \in R_{\mathrm{OR}}\}.$$

Suppose further that a $\Sigma$-protocol $\Sigma$ on a relation $R$ is given. Then we construct the protocol $\Sigma_{\mathrm{OR}}$ for the relation $R_{\mathrm{OR}}$ as follows. Let $(x_0, w_0)$ be in $R$, wolog. $\mathtt{P}$ computes $\mathrm{COM}_0 \leftarrow \Sigma_{\mathrm{com}}(x_0, w_0)$, $(\mathrm{COM}_1, \mathrm{CHA}_1, \mathrm{RES}_1) \leftarrow \Sigma_{\mathrm{sim}}(x_1)$ and sends $(\mathrm{COM}_0, \mathrm{COM}_1)$ to $\mathtt{V}$. Then $\mathtt{V}$ chooses $\mathrm{CHA} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda)$ and sends it to $\mathtt{P}$. Then $\mathtt{P}$ computes $\mathrm{CHA}_0 := \mathrm{CHA} \oplus \mathrm{CHA}_1, \mathrm{RES}_0 \leftarrow \Sigma_{\mathrm{res}}(x_0, w_0, \mathrm{COM}_0, \mathrm{CHA}_0)$ and sends $(\mathrm{CHA}_0, \mathrm{CHA}_1)$ and $(\mathrm{RES}_0, \mathrm{RES}_1)$ to $\mathtt{V}$. Here $\oplus$ denotes a bitwise exclusive-OR operation. Then for each $i = 0, 1$, $(\mathrm{COM}_i, \mathrm{CHA}_i, \mathrm{RES}_i)$ is an accepting transcript on $x_i$, and furthermore, the distribution of transcripts $\{(\mathrm{COM}_i, \mathrm{CHA}_i, \mathrm{RES}_i)\}$ is the same as the distribution of accepting transcripts generated as $\langle \mathtt{P}(x_i, w_i), \mathtt{V}(x_i)\rangle$ for any fixed $w_i \in W(x_i)$.

The protocol $\Sigma_{\mathrm{OR}}$ is actually a $\Sigma$-protocol [CDS94,Dam10]. We often call $\Sigma_{\mathrm{OR}}$ the *OR-proof protocol* (or simply, *OR-proof*, for short). A proof system $\Pi$ with the OR-proof protocol is, as we see, perfectly witness-indistinguishable [CDS94,Dam10]. Therefore, a proof system $\Pi$ with the OR-proof protocol is perfectly WIPoK.

### 2.2 Boolean Predicate

Let $f = f(X_{i_1}, \ldots, X_{i_a})$ be a boolean formula over boolean variables $U = \{X_1, \ldots, X_u\}$. In this paper, we consider only *monotone* boolean formulas; denoting the arity of $f$ by $a(f)$, two variables among $X_{i_1}, \ldots, X_{i_a}$ are connected by a boolean connective; an AND-gate ($\wedge$) or an OR-gate ($\vee$), and no NOT-gate ($\neg$) appears in $f$. For example, $f = X_{i_1} \wedge ((X_{i_2} \wedge X_{i_3}) \vee X_{i_4})$ for some indices $i_1, i_2, i_3, i_4$. For $f(X_{i_1}, \ldots, X_{i_a})$, we denote the set of indices of $f$ by $\mathrm{Att}(f)(= \{i_1, \ldots, i_a\})$. Hereafter we use the symbol $i_j$ to mean the following:

$$i_j \stackrel{\mathrm{def}}{=} (\text{the index of a boolean variable that is the } j\text{-th argument of } f).$$

Let $\mathcal{U} := \{1, \ldots, u\}$ denote the set of indices of $U$. Suppose that we are given an access structure as a boolean formula $f$. For $S \in 2^{\mathcal{U}}$, we evaluate the boolean value of $f$ at $S$ as follows:

$$f(S) \stackrel{\mathrm{def}}{=} f\big(X_{i_j} \leftarrow (i_j \in_? S); j = 1, \ldots, a(f)\big) \in \{0, 1\}.$$

Under this notation, a boolean formula $f$ can be seen as a map: $f : 2^{\mathcal{U}} \to \{0, 1\}$. We call a boolean formula $f$ with this map a *boolean predicate* over $\mathcal{U}$. We consider only *monotone* boolean predicates (in short, monotone predicates) as above.

**Binary-Tree Expression** A monotone boolean predicate $f$ can be represented by a finite binary tree $\mathcal{T}^f$. Each inner node represents a boolean connective, an $\wedge$-gate or an $\vee$-gate, in $f$. Each leaf corresponds to a position $X_i$ (not a variable $X_i$) in $f$ in one-to-one way. For a finite binary tree $\mathcal{T}$, we denote the set of all nodes, the root node, the set of all leaves, the set of all inner nodes (i.e. all nodes excluding the leaves) and the set of all tree-nodes (i.e. all nodes excluding the root node) as $\mathrm{Node}(\mathcal{T})$, $r(\mathcal{T})$, $\mathrm{Leaf}(\mathcal{T})$, $\mathrm{iNode}(\mathcal{T})$ and $\mathrm{tNode}(\mathcal{T})$, respectively. Then a map $\rho(\cdot)$ is defined as:

$$\rho : \mathrm{Leaf}(\mathcal{T}) \to \mathcal{U},\ \rho(l) \stackrel{\mathrm{def}}{=} (\text{the index } i \text{ where } l \text{ corresponds to the position } X_i).$$

If $\mathcal{T}$ is of height greater than 0, $\mathcal{T}$ has two subtrees whose root nodes are two children of $r(\mathcal{T})$. We denote the two subtrees by $\mathrm{Lsub}(\mathcal{T})$ and $\mathrm{Rsub}(\mathcal{T})$, which mean the left subtree and the right subtree, respectively.

## 3 Our Procedure of $\Sigma$-protocol on Monotone Predicate

In this section, we construct, given a $\Sigma$-protocol $\Sigma$ and a monotone predicate $f$, a $\Sigma$-protocol $\Sigma^f$ that is perfectly witness-indistinguishable. Our protocol $\Sigma^f$ is an extension of the OR-proof protocol $\Sigma_{\mathrm{OR}}$.

We revisit first the above notion that was introduced at a high level by Cramer, Damgård and Schoenmakers [CDS94]. Let $R$ be a binary relation. Let $f(X_{i_1}, \ldots, X_{i_{a(f)}})$ be a boolean formula over boolean variables $U = \{X_1, \ldots, X_u\}$.

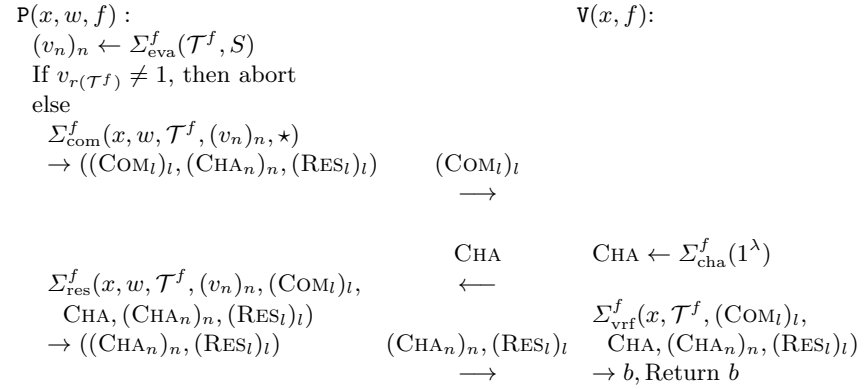


**Fig. 1.** Overview of our procedure of the $\Sigma$-protocol $\Sigma^f$ for the relation $R^f$.

**Definition 1 (Cramer, Damgård and Schoenmakers [CDS94], our Rewritten Form)** *A relation $R^f$ is defined by:*

$$R^f \stackrel{def}{=} \{(x = (x_{i_1}, \ldots, x_{i_{a(f)}}), w = (w_{i_1}, \ldots, w_{i_{a(f)}})) \in (\{0,1\}^*)^{a(f)} \times (\{0,1\}^*)^{a(f)}; \\ f(R(x_{i_1}, w_{i_1}), \ldots, R(x_{i_{a(f)}}, w_{i_{a(f)}})) = 1\}.$$

$R^f$ is a generalization of the relation $R_{\mathrm{OR}}$ [CDS94,Dam10] where $f$ was a boolean formula with a single boolean connective OR, i.e. $f = X_{i_1} \vee X_{i_2}$. Note that, if $R$ is an NP relation, then $R^f$ is also an NP relation under the assumption that the number of leaves of $\mathcal{T}^f$ is bounded by a polynomial $\ell(|x|)$. The corresponding language is

$$L_f \stackrel{\mathrm{def}}{=} \{x \in (\{0,1\}^*)^{a(f)}; \exists w \in (\{0,1\}^*)^{a(f)}, (x, w) \in R^f\}.$$

In the original paper [CDS94], a 3-move public-coin honest-verifier zero-knowledge proof of knowledge system for the language $L_f$ was defined as a perfectly witness-indistinguishable proof system on any given monotone predicate $f$. Then, in [CDS94], a $\Sigma$-protocol of the perfectly WIPoK for the relation $R^f$ was studied at a high level by using the notion of the dual access structure of the access structure determined by $f$.

### 3.1 Our Procedure

Now we construct a concrete procedure of a protocol $\Sigma^f$ of a WIPoK system for the relation $R^f$. $\Sigma^f$ is a 3-move public-coin protocol of a proof of knowledge system $\Pi = (\mathtt{P}, \mathtt{V})$ between interactive PPT algorithms $\mathtt{P}$ and $\mathtt{V}$, and it consists of seven algorithms: $\Sigma^f = (\Sigma^f_{\mathrm{eva}}, \Sigma^f_{\mathrm{com}}, \Sigma^f_{\mathrm{cha}}, \Sigma^f_{\mathrm{res}}, \Sigma^f_{\mathrm{vrf}}, \Sigma^f_{\mathrm{ext}}, \Sigma^f_{\mathrm{sim}})$. In our prover algorithm $\mathtt{P}$, there are four PPT subroutines $\Sigma^f_{\mathrm{eva}}$, $\Sigma^f_{\mathrm{com}}$, $\Sigma^f_{\mathrm{res}}$ and $\Sigma^f_{\mathrm{sim}}$. On the other hand, in our verifier algorithm $\mathtt{V}$, there are two PPT subroutines $\Sigma^f_{\mathrm{cha}}$ and $\Sigma^f_{\mathrm{vrf}}$. Moreover, $\Sigma^f_{\mathrm{vrf}}$ has two subroutines `VrfCha` and `VrfRes`. Fig. 1 shows the construction of our procedure $\Sigma^f$. (For the binary-tree expression of a boolean formula $f$, see Section 2.2.)

**Evaluation of Satisfiability.** The prover $\mathtt{P}$ begins with evaluation of whether and how $S$ satisfies $f$ by running the evaluation algorithm $\Sigma^f_{\mathrm{eva}}$. It labels each node of $\mathcal{T}^f$ with a value $v = 1$ (TRUE) or 0 (FALSE). For each leaf $l$, we label $l$ with $v_l = 1$ if $\rho(l) \in S$ and $v_l = 0$ otherwise. (For the definition of the function $\rho$, see Section 2.2.) For each inner node $n$, we label $n$ with $v_n = v_{n_{\mathrm{L}}} \wedge v_{n_{\mathrm{R}}}$ or $v_n = v_{n_{\mathrm{L}}} \vee v_{n_{\mathrm{L}}}$ according to AND/OR evaluation of two labels of its two children, $n_{\mathrm{L}}$ and $n_{\mathrm{R}}$. The computation is executed for every node from the root to each leaf, recursively, as in Fig. 2.

**Commitment.** The prover $\mathtt{P}$ computes a commitment for each leaf by running the algorithm $\Sigma^f_{\mathrm{com}}$ described in Fig. 3. Basically, $\Sigma^f_{\mathrm{com}}$ runs for every node from the root to each leaf, recursively. As a result, $\Sigma^f_{\mathrm{com}}$ generates for each leaf $l$ a value $\mathrm{COM}_l$; If $v_l = 1$, then $\mathrm{COM}_l$ is computed honestly according to $\Sigma_{\mathrm{com}}$. Else if $v_l = 0$, then $\mathrm{COM}_l$ is computed in the simulated way according to $\Sigma_{\mathrm{sim}}$. Other strings, $(\mathrm{CHA}_n)_n$ and $(\mathrm{RES}_l)_l$, are

$\Sigma^f_{\mathrm{eva}}(\mathcal{T}, S)$ :
  $\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
  If $r(\mathcal{T})$ is $\wedge$-node $n$, then $v_n := \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{L}}, S) \wedge \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{R}}, S)$,
    Return $(v_n, \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{L}}, S), \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{R}}, S))$
  else if $r(\mathcal{T})$ is $\vee$-node $n$, then $v_n := \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{L}}, S) \vee \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{R}}, S)$,
    Return $(v_n, \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{L}}, S), \Sigma^f_{\mathrm{eva}}(\mathcal{T}_{\mathrm{R}}, S))$
  else if $r(\mathcal{T})$ is a leaf $l$, then $v_l := (\rho(l) \in ? S)$
    Return $(v_l)$

**Fig. 2.** The subroutine $\Sigma^f_{\mathrm{eva}}$ of our $\Sigma^f$.

$\Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}, (v_n)_n, \mathrm{CHA})$ :
  $\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
  If $r(\mathcal{T})$ is $\wedge$-node $n$, then $\mathrm{CHA}_n := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}$
    Return $(\mathrm{CHA}_n, \Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}),$
      $\Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}))$
  else if $r(\mathcal{T})$ is $\vee$-node $n$, then $\mathrm{CHA}_n := \mathrm{CHA}$
    If $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \star$, $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \star$
    else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \star$, $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda)$
    else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \star$
    else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
    Return $(\mathrm{CHA}_n, \Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}),$
      $\Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}))$
  else if $r(\mathcal{T})$ is a leaf $l$, then $\mathrm{CHA}_l := \mathrm{CHA}$
    If $v_l = 1$, then $\mathrm{COM}_l \leftarrow \Sigma_{\mathrm{com}}(x_{\rho(l)}, w_{\rho(l)}), \mathrm{RES}_l := \star$
    else if $v_l = 0$, then $(\mathrm{COM}_l, \mathrm{RES}_l) \leftarrow \Sigma_{\mathrm{sim}}(x_{\rho(l)}, \mathrm{CHA})$
    Return$(\mathrm{COM}_l, \mathrm{CHA}_l, \mathrm{RES}_l)$

**Fig. 3.** The subroutine $\Sigma^f_{\mathrm{com}}$ of our $\Sigma^f$.

needed for the simulation. Note that the distinguished symbol $\star$ is used to indicate that the value has not been decided yet. P sends $(\mathrm{COM}_l)_l$ to V.

**Challenge.** The verifier V computes a challenge CHA by running the algorithm $\Sigma^f_{\mathrm{cha}}$ described in Fig. 4. V sends CHA to P.

$$\Sigma^f_{\mathrm{cha}}(1^\lambda) : \mathrm{CHA} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda), \mathrm{Return}(\mathrm{CHA})$$

**Fig. 4.** The subroutine $\Sigma^f_{\mathrm{cha}}$ of our $\Sigma^f$.

**Response.** The prover P computes a response for each leaf by running the algorithm $\Sigma^f_{\mathrm{res}}$ described in Fig. 5. Basically, the algorithm $\Sigma^f_{\mathrm{res}}$ runs for every node from the root to each leaf, recursively. As a result, $\Sigma^f_{\mathrm{res}}$ generates the challenge strings $(\mathrm{CHA}_n)_n$ for all the nodes $n \in \mathrm{Node}(\mathcal{T}^f)$ and the response strings $(\mathrm{RES}_l)_l$ for all the leaves $l \in \mathrm{Leaf}(\mathcal{T}^f)$. Note that the computations of all challenge strings $(\mathrm{CHA}_n)_n$ are completed (according to the "division rule" described in Section 1.2). P sends $(\mathrm{CHA}_n)_n$ and $(\mathrm{RES}_l)_l$ to V.

**Verification.** The verifier V computes a decision boolean by running the following algorithm $\Sigma^f_{\mathrm{vrf}}$ from the root to each leaf, recursively.

Now we have to check that $\Sigma^f$ is certainly a $\Sigma$-protocol for the relation $R^f$.

**Proposition 1 (Completeness)** *The completeness holds for our $\Sigma^f$.*

*Proof.* Suppose that $v_{r(\mathcal{T}^f)} = 1$. We show that, for every node in $\mathrm{Node}(\mathcal{T}^f)$, either $v_n = 1$ or $\mathrm{CHA}_n \neq *$ holds after executing $\Sigma^f_{\mathrm{com}}$. The proof is by induction on the height of $\mathcal{T}^f$. The case of height 0 follows from $v_{r(\mathcal{T}^f)} = 1$ and the completeness of $\Sigma$. Suppose that the case of height $k$ holds and consider the case of height $k+1$. The construction of $\Sigma^f_{\mathrm{com}}$ assures the case of height $k+1$. □

$\Sigma^f_{\mathrm{res}}(x, w, \mathcal{T}, (v_n)_n, (\mathrm{COM}_l)_l, \mathrm{CHA}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$ :
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is $\wedge$-node $n$, then $\mathrm{CHA}_n := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}$
Return$(\mathrm{CHA}_n, \Sigma^f_{\mathrm{res}}(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\mathrm{COM}_l)_l, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$,
$\Sigma^f_{\mathrm{res}}(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\mathrm{COM}_l)_l, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l))$
else if $r(\mathcal{T})$ is $\vee$-node $n$, then $\mathrm{CHA}_n := \mathrm{CHA}$
If $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma_{\mathrm{cha}}(1^\lambda), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then do nothing
Return$(\mathrm{CHA}_n, \Sigma^f_{\mathrm{res}}(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\mathrm{COM}_l)_l, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$,
$\Sigma^f_{\mathrm{res}}(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\mathrm{COM}_l)_l, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l))$
else if $r(\mathcal{T})$ is a leaf $l$, then $\mathrm{CHA}_l := \mathrm{CHA}$
If $v_l = 1$, then $\mathrm{RES}_l \leftarrow \Sigma_{\mathrm{res}}(x_{\rho(l)}, w_{\rho(l)}, \mathrm{COM}_l, \mathrm{CHA})$
else if $v_l = 0$, then do nothing
Return$(\mathrm{CHA}_l, \mathrm{RES}_l)$

**Fig. 5.** The subroutine $\Sigma^f_{\mathrm{res}}$ of our $\Sigma^f$.

$\Sigma^f_{\mathrm{vrf}}(x, \mathcal{T}, (\mathrm{COM}_l)_l, \mathrm{CHA}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$ :
Return$(\mathtt{VrfCha}(\mathcal{T}, \mathrm{CHA}, (\mathrm{CHA}_n)_n) \wedge \mathtt{VrfRes}(x, \mathcal{T}, (\mathrm{COM}_l)_l, (\mathrm{CHA}_l)_l, (\mathrm{RES}_l)_l))$

$\mathtt{VrfCha}(\mathcal{T}, \mathrm{CHA}, (\mathrm{CHA}_n)_n)$ :
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is $\wedge$-node $n$, then
Return $((\mathrm{CHA} =_? \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}) \wedge (\mathrm{CHA} =_? \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge \mathtt{VrfCha}(\mathcal{T}_{\mathrm{L}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_n)_n) \wedge \mathtt{VrfCha}(\mathcal{T}_{\mathrm{R}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_n)_n))$
else if $r(\mathcal{T})$ is $\vee$-node $n$, then
Return $((\mathrm{CHA} =_? \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge \mathtt{VrfCha}(\mathcal{T}_{\mathrm{L}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_n)_n) \wedge \mathtt{VrfCha}(\mathcal{T}_{\mathrm{R}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_n)_n))$
else if $r(\mathcal{T})$ is a leaf $l$, then
Return $(\mathrm{CHA} \in_? \mathrm{CHASP}(1^\lambda))$

$\mathtt{VrfRes}(x, \mathcal{T}, (\mathrm{COM}_l)_l, (\mathrm{CHA}_l)_l, (\mathrm{RES}_l)_l)$ :
For $l \in \mathrm{Leaf}(\mathcal{T})$ : If $\Sigma_{\mathrm{vrf}}(x_{\rho(l)}, \mathrm{COM}_l, \mathrm{CHA}_l, \mathrm{RES}_l) = 0$, then Return $(0)$
Return $(1)$

**Fig. 6.** The subroutine $\Sigma^f_{\mathrm{vrf}}$ of our $\Sigma^f$.

**Proposition 2 (Special Soundness)** *The special soundness holds for our $\Sigma^f$.*

We construct a knowledge extractor $\Sigma^f_{\mathrm{ext}}$ by employing the knowledge extractor $\Sigma_{\mathrm{ext}}$ of the underlying $\Sigma$-protocol $\Sigma$ as in Fig. 7. Then Lemma 1 assures the above proposition.

$\Sigma^f_{\mathrm{ext}}(x, f, (\mathrm{COM}_l)_l,\ \mathrm{CHA}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l,\ \mathrm{CHA}', (\mathrm{CHA}'_n)_n, (\mathrm{RES}'_l)_l)$ :
If $\mathrm{CHA} = \mathrm{CHA}'$ then Return TheSameCha
else if $\Sigma^f_{\mathrm{vrf}}(x, \mathcal{T}^f, \mathrm{CHA}, (\mathrm{COM}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l) = 0$
or $\Sigma^f_{\mathrm{vrf}}(x, \mathcal{T}^f, \mathrm{CHA}', (\mathrm{COM}_l)_l, (\mathrm{CHA}'_n)_n, (\mathrm{RES}'_l)_l) = 0$, then Return $\perp$
else
For $l \in \mathrm{Leaf}(\mathcal{T}^f)$:
If $\mathrm{CHA}_l = \mathrm{CHA}'_l$, then $\hat{w}_{\rho(l)} \in_R \{0,1\}^{\ell(|x_{\rho(l)}|)}$
else $\hat{w}_{\rho(l)} \leftarrow \Sigma_{\mathrm{ext}}(x_{\rho(l)}, \mathrm{COM}_l, \mathrm{CHA}_l, \mathrm{RES}_l, \mathrm{CHA}'_l, \mathrm{RES}'_l)$
Return $(\hat{w} := (\hat{w}_{i_j})_{1 \le j \le a(f)})$

**Fig. 7.** The knowledge-extractor $\Sigma^f_{\mathrm{ext}}$ of our $\Sigma^f$.

**Lemma 1 (Knowledge Extraction)** *The string $\hat{w}$ output by $\Sigma^f_{ext}$ satisfies $(x, \hat{w}) \in R^f$.*

*Proof.* We prove the lemma by induction on the number of all $\vee$-nodes in $\mathrm{iNode}(T_f)$. First remark that $\mathrm{CHA} \neq \mathrm{CHA}'$.

Suppose that all nodes in $\mathrm{iNode}(T_f)$ are $\wedge$-nodes. Then the above claim follows immediately because $\mathrm{CHA}_l \neq \mathrm{CHA}'_l$ holds for all leaves.

Suppose that the case of $k$ $\vee$-nodes holds and consider the case of $k+1$ $\vee$-nodes. Look at one of the lowest height $\vee$-node and name the height and the node as $h^*$ and $n^*$, respectively. Then $\mathrm{CHA}_{n^*} \neq \mathrm{CHA}'_{n^*}$ because all nodes with their heights less than $h^*$ are $\wedge$-nodes. So at least one of children of $n^*$, say $n^*_L$, satisfies $\mathrm{CHA}_{n^*_L} \neq \mathrm{CHA}'_{n^*_L}$. Divide the tree $\mathcal{T}^f$ into two subtrees by cutting the branch right above $n^*$, and the induction hypothesis assures the claim. □

**Proposition 3 (HVZK)** *The honest-verifier zero-knowledge property holds for our $\Sigma^f$.*

*Proof.* We construct a polynomial-time simulator $\Sigma^f_{\mathrm{sim}}$, which on input a statement $x \in L_f$ and a predicate $f$ returns an accepting transcript $((\mathrm{COM}_l)_l, \mathrm{CHA}, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$, as in Fig. 8.

$\Sigma^f_{\mathrm{sim}}(x, f)$ :
$\tilde{\mathrm{CHA}} \leftarrow \Sigma^f_{\mathrm{cha}}(1^\lambda)$, $w \in_R \{0,1\}^{\ell(|x_{\rho(l)}|)}$, For $n \in \mathrm{Node}(\mathcal{T}^f) : v_n := 0$
$((\tilde{\mathrm{COM}}_l)_l, (\tilde{\mathrm{CHA}}_n)_n, (\tilde{\mathrm{RES}}_l)_l) \leftarrow \Sigma^f_{\mathrm{com}}(x, w, \mathcal{T}^f, (v_n)_n, \tilde{\mathrm{CHA}})$
Return$((\tilde{\mathrm{COM}}_l)_l, \tilde{\mathrm{CHA}}, (\tilde{\mathrm{CHA}}_n)_n, (\tilde{\mathrm{RES}}_l)_l)$

**Fig. 8.** The simulator $\Sigma^f_{\mathrm{sim}}$ of our $\Sigma^f$.

□

We summarize the above results into the following theorem and corollary.

**Theorem 1 ($\Sigma^f$ is a $\Sigma$-protocol)** *If a given protocol $\Sigma$ on a relation $R$ is a $\Sigma$-protocol, and if a boolean predicate $f$ is monotone and the number of leaves of $\mathcal{T}^f$ is bounded by a polynomial $\ell(|x|)$, then the protocol $\Sigma^f$ with our procedure is a $\Sigma$-protocol for the relation $R^f$.*

**Theorem 2 ($\Sigma^f$ is a perfectly WIPoK)** *If a given protocol $\Sigma$ on a relation $R$ is a $\Sigma$-protocol, and if a boolean predicate $f$ is monotone and the number of leaves of $\mathcal{T}^f$ is bounded by a polynomial $\ell(|x|)$, then the protocol $\Sigma^f$ with our procedure is a protocol of a perfectly witness-indistinguishable proof of knowledge system for the relation $R^f$.*

*Proof.* For any statement $x$ and any two witnesses $w_1$ and $w_2$ satisfying $R^f(x, w_1) = R^f(x, w_2) = 1$, the distribution of the transcript $\mathtt{P}(x, w_1)$ and $\mathtt{V}(x)$ of $\Sigma^f$ and the distribution of the transcript $\mathtt{P}(x, w_2)$ and $\mathtt{V}(x)$ of $\Sigma^f$ are identical. □

### 3.2 Non-interactive Version

The Fiat-Shamir transform FS($\cdot$) can be applied to any $\Sigma$-protocol $\Sigma$ ([FS86,AABN02]). Therefore, the non-interactive version of our procedure $\Sigma^f$ is obtained.

**Theorem 3 (FS($\Sigma^f$) is a non-interactive perfectly WIPoK)** *If a given protocol $\Sigma$ on a relation $R$ is a $\Sigma$-protocol, and if a boolean predicate $f$ is monotone and the number of leaves of $\mathcal{T}^f$ is bounded by a polynomial $\ell(|x|)$, then the protocol FS($\Sigma^f$) is a protocol of a non-interactive perfectly witness-indistinguishable proof of knowledge system for the relation $R^f$. A knowledge extractor is constructed in the random oracle model.*

### 3.3 Discussion

As is mentioned in [CDS94], the $\Sigma$-protocol $\Sigma^f$ can be considered as a proto-type of an attribute-based identification scheme [AAHI13]. Also, the non-interactive version FS($\Sigma^f$) can be considered a proto-type of an attribute-based signature scheme [MPR11]. That is, $\Sigma^f$ and FS($\Sigma^f$) are an attribute-based identification scheme and an attribute-based signature scheme *without the collusion resistance against collecting private secret keys*, respectively.

## 4 Conclusion

We provided a concrete procedure of a $\Sigma$-protocol $\Sigma^f$, which is of a perfectly witness-indistinguishable proof of knowledge system for an NP relation $R^f$, where $f$ is an input monotone predicate. Our concrete procedure is for any monotone predicate $f$ on condition that the number of leaves of $\mathcal{T}^f$ is bounded by a polynomial $\ell(|x|)$. It serves as building blocks of cryptographic primitives which use $\Sigma$-protocols; for example, the pairing-free ABID and ABS schemes [Her14,Her16a].

**Acknowledgements** We appreciate sincere comments from Javier Herranz via e-mail communication [Her16b] on the topic in this paper. We would like to thank to Keita Emura and Takahiro Matsuda for their sincere comments and encouragements on the construction of attribute-based signature schemes. We would like to thank to Shingo Hasegawa and Masayuki Fukumitsu for their sincere comments on the construction of the $\Sigma$-protocol on monotone predicates.

## References

[AA18] Hiroaki Anada and Seiko Arita. Witness-indistinguishable arguments with $\Sigma$-protocols for bundled witness spaces and its application to global identities. *IACR Cryptology ePrint Archive*, 2018/742, 2018.

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, pages 418–433, 2002.

[AAHI13] Hiroaki Anada, Seiko Arita, Sari Handa, and Yosuke Iwabuchi. Attribute-based identification: Definitions and efficient constructions. In *Information Security and Privacy - 18th Australasian Conference, ACISP 2013, Brisbane, Australia, July 1-3, 2013. Proceedings*, pages 168–186, 2013.

[AAS14] Hiroaki Anada, Seiko Arita, and Kouichi Sakurai. Attribute-based signatures without pairings via the fiat-shamir paradigm. In *ASIAPKC'14, Proceedings of the 2nd ACM Wookshop on ASIA Public-Key Cryptography, June 3, 2014, Kyoto, Japan*, pages 49–58, 2014.

[Ana18] Hiroaki Anada. Detailed instantiation of the decentralized multi-authority anonymous authentication scheme and tighter reduction for security. In *Sixth International Symposium on Computing and Networking, CANDAR 2018, Takayama, Japan, November 23-27, 2018*, pages 85–91, 2018.

[Bab85] László Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 421–429, 1985.

[BBC+18] Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 669–699. Springer, 2018.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 41–55, 2004.

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, pages 390–420, 1992.

[BL88] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35. Springer, 1988.

[BP02] Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 162–177, 2002.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, pages 174–187, 1994.

[CL02] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, pages 268–289, 2002.

[Cra96] Ronald Cramer. *Modular Designs of Secure, yet Practical Cyptographic Protocols*. PhD thesis, University of Amsterdam, Amsterdam, the Netherlands, 1996.

[Dam10] Ivan Damgård. On $\sigma$-protocols. In Course Notes, http://cs.au.dk/ ivan/CPT.html, 2010.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 356–383. Springer, 2019.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426, 1990.

[GMR85] S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1, Basic Techniques*. Cambridge University Press, 2001.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 216–231, 1988.

[Her14] Javier Herranz. Attribute-based signatures from rsa. *Theoretical Computer Science*, 527:73–82, 2014.

[Her16a] Javier Herranz. Attribute-based versions of schnorr and elgamal. *Appl. Algebra Eng. Commun. Comput.*, 27(1):17–57, 2016.

[Her16b] Javier Herranz. Private communication via e-mail, dept. matemàtica aplicada iv, universitat politècnica de catalunya, July 2014, Sept 2015 and May 2016.

[Kat03] Jonathan Katz. Efficient and non-malleable proofs of plaintext knowledge and applications. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, pages 211–228, 2003.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 326–355. Springer, 2019.

[MPR11] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, pages 376–392, 2011.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 387–398, 1996.

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, pages 239–252, 1989.

[Unr12] Dominique Unruh. Quantum proofs of knowledge. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 135–152, 2012.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 755–784, 2015.