DOI 10.1007/s00165-008-0078-3
BCS © 2008
Formal Aspects of Computing (2008) 20: 303–348

Formal Aspects of Computing

Efficient representation of the attacker's knowledge in cryptographic protocols analysis[1]

Ivan Cibrario Bertolotti[1], Luca Durante[1], Riccardo Sisto[2] and Adriano Valenzano[1]

[1]IEIIT-CNR, C.so Duca degli Abruzzi 24, 10129 Torino, Italy. E-mail: ivan.cibrario@polito.it
[2]Politecnico di Torino, C.so Duca degli Abruzzi 24, 10129 Torino, Italy

Abstract. This paper addresses the problem of representing the intruder's knowledge in the formal verification of cryptographic protocols, whose main challenges are to represent the intruder's knowledge efficiently and without artificial limitations on the structure and size of messages. The new knowledge representation strategy proposed in this paper achieves both goals and leads to practical implementation because it is incrementally computable and is easily amenable to work with various term representation languages. In addition, it handles associative and commutative term composition operators, thus going beyond the free term algebra framework. An extensive computational complexity analysis of the proposed representation strategy is included in the paper.

Keywords: Cryptographic protocols; Knowledge representation; State space exploration

1. Introduction

The formal, automatic verification of security properties of cryptographic protocols is a difficult task, which has already been explored in previous papers. Some of them leverage automatic proof techniques, with the help of various proof systems and formalisms, e.g. [Pau98, Sch98, BDNP02]; although partial automation of proofs with theorem provers is possible, the use of such tools is usually quite time consuming and difficult.
Another approach is to use state exploration methods, such as model checking, e.g. [Low96, MS01, CJM00]. In state exploration, one of the challenges is to find a compact and efficient representation of the intruder's knowledge, which plays a central role in modelling the protocol behaviour in an hostile environment, without sacrificing the expressive power of the specification language. Previous papers only give partial solutions to this problem, the most common approach being to restrict the way in which messages can be built. For example, a common restriction is to force encryption keys to be atomic [CJM98]. Such a restriction can also be found with some theorem proving techniques, e.g. [Pau98]. It is worth noting that offering support for constructed, non-atomic keys is becoming increasingly important to analyse real-world protocols since, for example, it is common for such protocols to build a symmetric key from shared secrets and other data exchanged between parties during a run of the protocol itself.
This paper presents a novel intruder's knowledge representation strategy that achieves both the goals of compactness and implementation efficiency without artificially restricting the way in which messages can be built. In particular, the proposed representation is largely independent from the term representation language chosen, i.e. it supports all main cryptographic message construction operators, including the full term language of spi

Correspondence and offprint requests to: I. Cibrario Bertolotti, E-mail: ivan.cibrario@polito.it
[1] This work was partially supported by the Italian National Council of Research, grant number CNRC00FE45, and by the Center for Multimedia Radio Communications of Politecnico di Torino.

Table 1. Naming conventions

Symbol	Meaning
$\mathcal{P}$	is the set of term algebra operators; it contains the standard term algebra operators of Table 2, as well as the additional operators described in Sect. 4
a	ranges over names: $a \in \mathcal{A}$
$\mathcal{T}$	is the set of all terms that can be built by combining the elements of $\mathcal{A}$ with the operators of $\mathcal{P}$
σ, ρ and η	range over terms, either atomic or non-atomic, in $\mathcal{T}$
x and y	range over variables
b	ranges over fresh names
P, Q and R	range over processes
$\leq_{\mathcal{A}}$	is a total order relation on $\mathcal{A}$
$\leq_{\mathcal{P}}$	is a total order relation on $\mathcal{P}$
$\leq_{\mathcal{T}}$	is a total order relation on $\mathcal{T}$

calculus [AG99]. Moreover, it keeps the intruder's knowledge in a minimised, canonical form, and is shown to be incrementally computable.

Having a canonical intruder's knowledge representation is especially important when state exploration techniques are used to check security properties. In fact, simpler non-canonical representations, such as for example the plain set of messages the intruder has access to when eavesdropping honest agents, may lead to state proliferation, because different sets of known messages may correspond to the same intruder's knowledge.

In addition, this representation goes beyond the free term algebra setting assumed by most existing work and also handles commutative and associative operators. This feature is very important to model and verify any real-world cryptographic protocol that makes use of this kind of operator, as pointed out in [BB02, CJM98, Low97, Low99, MCJ97, Mon99, RT01], and is actively being addressed by other researchers too, e.g. [CKRT03, CLS03, MN02, MS03].

The basic knowledge representation without the extension to commutative and associative operators has been used in a framework for automatic testing equivalence verification of spi calculus specifications, presented in [DSV03], where symbolic techniques are used to get finite models.

This paper presents the proposed intruder's knowledge representation with reference to the spi calculus specification language. Since the expressive power of the message specification section of such a language is greater than or equivalent to the one of most other formalisms for cryptographic protocols, the adoption of spi calculus as our reference specification language is not restrictive.

The paper assumes that the reader is familiar with basic cryptography, and is structured as follows: Sect. 2 presents the syntax of spi calculus, and informally describes its semantics. In Sects 3 and 4 we discuss our knowledge representation strategy along with its extension to handle commutative and associative term composition operators; then, we work out some examples to better illustrate our technique.

In Sect. 5 we analyse the computational complexity of the method we propose with an increasing level of accuracy, and present a comparison of the estimated asymptotic complexity against both the simulated and the actual, measured complexity of a prototype model-checking tool [DSV03]. Section 6 discusses related work and draws some conclusions.

Finally, Appendix A elaborates on some mathematical details of the computational complexity analysis whereas Appendices B through D contain the proofs of the theorems introduced in the previous sections. This work extends and completes the results first presented in the conference papers [CDSV03b, CDSV03a].

2. Spi calculus

The spi calculus is defined in [AG99] as an extension of the π calculus [MPW92] with cryptographic primitives. It is a process algebraic language designed for describing and analysing cryptographic protocols. These protocols heavily rely on cryptography and on message exchange through communication channels; accordingly, the spi calculus provides powerful primitives to express cryptography and communication.

This section summarises the syntax and describes the language semantics informally; the language used in this paper fully conforms to the original spi calculus definition [AG99], with the naming conventions outlined in Table 1.

Table 2. Term composition operators of the spi calculus

Notation	Meaning	Notation	Meaning
$(\sigma,\ \rho)$	Pair	σ^+	Public part
$\mathrm{suc}(\sigma)$	Successor	σ^-	Private part
$\mathrm{H}(\sigma)$	Hashing	$\{[\sigma]\}_\rho$	Public-key encryption
$\{\sigma\}_\rho$	Shared-key encryption	$[\{\sigma\}]_\rho$	Private-key signature

Table 3. Process syntax of the spi calculus

Notation	Meaning	Notation	Meaning
$\overline{\sigma}\langle\rho\rangle.P$	Output	$[\sigma\ is\ \rho]\ P$	Match
$\sigma(x).P$	Input	$let\ (x,\ y)\ =\ \sigma\ in\ P$	Pair splitting
$P \mid Q$	Composition	$case\ \sigma\ of\ 0 : P\ \mathrm{suc}(x) : Q$	Integer case
$(\nu b)\ P$	Restriction	$case\ \eta\ of\ \{x\}_\rho\ in\ P$	Shared-key decryption
$!P$	Replication	$case\ \eta\ of\ \{[x]\}_\rho\ in\ P$	Decryption
0	Nil	$case\ \eta\ of\ [\{x\}]_\rho\ in\ P$	Signature-check

The spi calculus has two basic language elements: terms, to represent data, and processes, to represent behaviours. Terms can be either atomic elements, i.e. names, including the special name 0 representing the integer constant zero, and variables, or compound terms built using the term composition operators listed in Table 2. Names may represent communication channels, atomic keys and key pairs, nonces (also called *fresh names*) and any other unstructured data.

Besides term specification, the spi calculus also offers a rich set of operators, shown in Table 3, to build behaviour expressions that, in turn, represent processes.

As an example, Fig. 1 shows the spi calculus specification of a very simple protocol inspired by [FA01]. The left-hand side of the figure shows the message exchanges involved in the protocol, using the informal, intuitive representation often encountered in the literature, whereas the right-hand side of the figure shows the corresponding spi calculus specification. In this protocol two agents A and B, represented by spi calculus processes P_A and P_B, respectively, share a secret key k and exchange two messages:

- First, A sends to B message M encrypted under key k over public channel c. This is represented, in spi calculus, by the output statement $\overline{c}\langle\{M\}_k\rangle$ in process P_A and by the corresponding input statement $c(y_1)$ in P_B; the latter statement assigns the datum just received to variable y_1 of P_B.
- Then, B tries to decrypt the message with key k, as specified by the statement $case\ y_1\ of\ \{y_2\}_k$ in P_B and, when successful, sends back to A the hashed cleartext of M on the same channel c, with the output statement $\overline{c}\langle H(y_2)\rangle$. Process A receives this message with the input statement $c(x)$.
- Finally, A checks that the hash just received is correct with the statement $[x\ is\ H(M)]$ and proceeds with further operations on message M, represented by the unspecified process $F(M)$.

The role of the spi calculus process P_{sample} in this example is twofold:

- With the restriction operator (νk), it generates a restricted, private name k, only known to P_A, P_B and itself, to be used as the encryption key.
- It instantiates both P_A and P_B to run in parallel, by means of the parallel composition operator |, so that an instance of P_{sample} represents all agents involved in a session of the protocol.

$A \rightarrow B : \{M\}_k$	$P_A(M) \stackrel{\triangle}{=} \overline{c}\langle\{M\}_k\rangle.\ c(x).\ [x\ is\ H(M)]\ F(M)$
$B \rightarrow A : H(M)$	$P_B \stackrel{\triangle}{=} c(y_1).\ case\ y_1\ of\ \{y_2\}_k\ in\ \overline{c}\langle H(y_2)\rangle.\ 0$
	$P_{\mathrm{sample}} \stackrel{\triangle}{=} (\nu k)(P_A(M) \mid P_B)$

Fig. 1. A simple spi calculus specification

3. Intruder's knowledge representation

Our approach to the representation of the knowledge that an intruder can acquire, already described in [DSV03], borrows some of the notation and concepts introduced in [CJM98], and has some similarities with [FA01] and [Hui99] as Sect. 6 points out. However, it is more sophisticated than previous representations in some respects, because encryption and decryption keys are not restricted to be atomic, the spi calculus is considered to its full extent, including public/private key cryptosystems, and the intruder's knowledge is always kept in a minimised form which both speeds up and simplifies processing.
As most other researchers do, our method relies on the well-known *perfect encryption* assumptions, and adopts the *Dolev-Yao* intruder model inspired by [DY83].

3.1. The minimal intruder's knowledge representation

Let $\mathcal{A}$ be the set of spi calculus names, including the integer constant 0, and $\mathcal{T}$ the set of all spi calculus terms that can be built by combining the elements of $\mathcal{A}$ by means of the operators defined in Table 2. For simplicity, and without loss of generality, it is assumed that distinct names and variables of the spi calculus always have lexically distinct forms, since it is always possible to construct a unique identifier for each name or variable of the spi calculus by taking scope information into account.
The closure of a set of terms $\Sigma \subseteq \mathcal{T}$ is denoted $\widehat{\Sigma}$ and is defined as the set of all spi calculus terms that can be built by combining the elements of Σ by means of the operators defined in Table 2 and their inverses. Formally, $\widehat{\Sigma}$ is the least set of terms such that, for each σ, σ_1 and $\sigma_2 \in \mathcal{T}$, the following closure rules hold:

$$\sigma \in \Sigma \;\Rightarrow\; \sigma \in \widehat{\Sigma} \tag{1}$$
$$\sigma \in \widehat{\Sigma} \;\Rightarrow\; \mathrm{suc}(\sigma) \in \widehat{\Sigma} \quad \text{(successor)} \tag{2}$$
$$\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; (\sigma_1,\ \sigma_2) \in \widehat{\Sigma} \quad \text{(pairing)} \tag{3}$$
$$\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \quad \text{(sh. key encryption)} \tag{4}$$
$$\sigma \in \widehat{\Sigma} \;\Rightarrow\; \mathrm{H}(\sigma) \in \widehat{\Sigma} \quad \text{(hashing)} \tag{5}$$
$$\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \;\Rightarrow\; \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \quad \text{(pub. key encryption)} \tag{6}$$
$$\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^- \in \widehat{\Sigma} \;\Rightarrow\; [\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma} \quad \text{(priv. key signature)} \tag{7}$$
$$\sigma \in \widehat{\Sigma} \;\Rightarrow\; \sigma^+ \in \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma} \quad \text{(key projection)} \tag{8}$$
$$\mathrm{suc}(\sigma) \in \widehat{\Sigma} \;\Rightarrow\; \sigma \in \widehat{\Sigma} \quad \text{(prec)} \tag{9}$$
$$(\sigma_1,\ \sigma_2) \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \quad \text{(projection)} \tag{10}$$
$$\{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \quad \text{(sh. key decr.)} \tag{11}$$
$$\{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \wedge \sigma_2^- \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \quad \text{(pub. key decr.)} \tag{12}$$
$$\{[\sigma_1]\}_{\sigma_2^-} \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \quad \text{(signature check)} \tag{13}$$
$$\sigma^+ \in \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma} \;\Rightarrow\; \sigma \in \widehat{\Sigma} \quad \text{(key pairing)} \tag{14}$$

In principle, if Σ is the set of messages the intruder has intercepted so far, the generation of individual elements of $\widehat{\Sigma}$ (informally, the set of all messages the intruder can generate at a given point) starting from Σ can be viewed as a derivation in a natural deduction system [Pra65].
In this respect, closure rule (1) represents the ability to derive an element from itself, closure rules (2)–(8) are equivalent to the introduction rules of the natural deduction system ($\mathcal{I}$ rules), and closure rules (9)–(14) are equivalent to its elimination rules ($\mathcal{E}$ rules). In the following, it will sometimes be necessary to consider the closure of a set Σ under $\mathcal{I}$ rules, that is, computed using only rules (1) and (2)–(8); that closure will be denoted by $\widehat{\Sigma}^I$.
For example, using the rule notation of [Pra65], closure rule (4) is equivalent to the introduction rule:

$$\frac{\sigma_1 \quad \sigma_2}{\{\sigma_1\}_{\sigma_2}} \qquad \{\}\text{-}\mathcal{I} \text{ rule.} \tag{15}$$

Informally, we can read this rule as: "when both σ_1 and σ_2 are known to the intruder, then the intruder also knows about $\{\sigma_1\}_{\sigma_2}$."
Similarly, closure rule (11) is equivalent to the elimination rule:

$$\frac{\{\sigma_1\}_{\sigma_2}\ \sigma_2}{\sigma_1} \qquad \{\}\text{-}\mathcal{E} \text{ rule.} \tag{16}$$

Informally, we can read this rule as: "when the intruder knows both $\{\sigma_1\}_{\sigma_2}$ and σ_2, then it can successfully perform a decryption and add σ_1 to its knowledge."

In an elimination rule, the premise that contains the operator removed by the rule is called the *major premise*, while all other premises are called *minor premises*. For example, in rule (16), $\{\sigma_1\}_{\sigma_2}$ is the major premise and σ_2 is the minor premise.

The theory of natural deduction [Pra65] implies that, if $\sigma \in \widehat{\Sigma}$, then σ can be deduced from Σ with a *natural* deduction in *normal* form, that is, a chain of applications of $\mathcal{E}$ rules followed by a chain of applications of $\mathcal{I}$ rules, along the rules' major premises. This is not necessarily true along minor premises, so the closure of Σ under $\mathcal{E}$ rules only is not a suitable candidate to represent the intruder's knowledge, unless some additional constraints are imposed, such as the atomicity of encryption keys; this is exactly the approach adopted, for example, in [CJM98, Pau98]. In the following, we show that this difficulty can be overcome by introducing the concept of *minimal closure seed of* Σ, denoted by $\overline{\Sigma}$, and by suitably refining the derivation rules of the deduction system.

Informally, $\overline{\Sigma}$ is a minimised, canonical representation of the intruder's knowledge, which contains no redundant elements, but includes only the minimum set of simplest terms that the intruder needs in order to generate all data it has ever known of.

For example, if the intruder eavesdrops $\{m\}_k$ but is unable to generate the decryption key k, it must add $\{m\}_k$ to its knowledge representation. If k becomes known to the intruder at a later time, the intruder can then add k and m to its knowledge representation, because k allows it to decipher $\{m\}_k$; at the same time, it can remove $\{m\}_k$ from its knowledge representation, because it is now able to re-generate this term starting from k and m.

Given a finite set of terms Σ, we define the *minimal closure seed of* Σ, and denote it as $\overline{\Sigma}$, the largest subset of $\widehat{\Sigma}$ that satisfies the following predicates for each $a \in \mathcal{A}$, and for each $\sigma, \sigma_1, \sigma_2 \in \mathcal{T}$.

$$a \in \overline{\Sigma} \quad \Leftrightarrow \quad a \in \widehat{\Sigma} \tag{17}$$
$$\text{suc}(\sigma) \notin \overline{\Sigma} \tag{18}$$
$$(\sigma_1, \sigma_2) \notin \overline{\Sigma} \tag{19}$$
$$\{\sigma_1\}_{\sigma_2} \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma_2 \notin \widehat{\Sigma} \tag{20}$$
$$\text{H}(\sigma) \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma \notin \widehat{\Sigma} \tag{21}$$
$$\{[\sigma_1]\}_{\sigma_2^+} \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma_2^+ \notin \widehat{\Sigma} \ \vee\ \sigma_1 \notin \widehat{\Sigma} \tag{22}$$
$$[\{\sigma_1\}]_{\sigma_2^-} \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma_2^- \notin \widehat{\Sigma} \ \vee\ \sigma_1 \notin \widehat{\Sigma} \tag{23}$$
$$\sigma^+ \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma \notin \widehat{\Sigma} \tag{24}$$
$$\sigma^- \in \overline{\Sigma} \quad \Leftrightarrow \quad \sigma \notin \widehat{\Sigma} \tag{25}$$

For example, if $\Sigma = \{\{[a]\}_{k^+}, k^-\}$, then $\overline{\Sigma} = \{\{[a]\}_{k^+}, k^-, a\}$, because:

- $k^- \in \overline{\Sigma}$, by rule (25), because $k \notin \widehat{\Sigma}$. In fact, k cannot be deduced by k^- only.
- $a \in \overline{\Sigma}$: since $k^- \in \overline{\Sigma} \subset \widehat{\Sigma}$, then k^- can be used to decrypt $\{[a]\}_{k^+}$, and $a \in \widehat{\Sigma}$, by rule (12). Moreover, being a a name, by rule (17), $a \in \overline{\Sigma}$.
- $\{[a]\}_{k^+} \in \overline{\Sigma}$, by rule (22), because $k^+ \notin \widehat{\Sigma}$, so there is no way to construct $\{[a]\}_{k^+}$ starting from other members of $\overline{\Sigma}$.

Table 4. An example of reduction

i	Σ_i	$\mathcal{R}$	Σ_I	Σ_O
0	$\{c, \{\{k_1\}_{k_2}\}_{k_3}, \{[m]\}_{k_1{}^+}, k_1{}^+, k_2, k_3\}$	(33)	$\{\{\{k_1\}_{k_2}\}_{k_3}\}$	$\{\{k_1\}_{k_2}\}$
1	$\{c, \{[m]\}_{k_1{}^+}, k_1{}^+, k_2, k_3, \{k_1\}_{k_2}\}$	(33)	$\{\{k_1\}_{k_2}\}$	$\{k_1\}$
2	$\{c, \{[m]\}_{k_1{}^+}, k_1{}^+, k_2, k_3, k_1\}$	(29)	$\{k_1{}^+\}$	$\emptyset$
3	$\{c, \{[m]\}_{k_1{}^+}, k_2, k_3, k_1\}$	(34)	$\emptyset$	$\{m\}$
4	$\{c, \{[m]\}_{k_1{}^+}, k_2, k_3, k_1, m\}$	(27)	$\{\{[m]\}_{k_1{}^+}\}$	$\emptyset$
5	$\{c, k_2, k_3, k_1, m\}$			

Before discussing the basic properties of $\overline{\Sigma}$, let us define $\mathtt{r}(\sigma, \Sigma)$ as the boolean value obtained by executing the following algorithm:

```
boolean r(σ, Σ) {
  if σ ∈ Σ                then return TRUE;
  else if σ = suc(σ1)     then return r(σ1, Σ);
  else if σ = (σ1, σ2)    then return r(σ1, Σ) ∧ r(σ2, Σ);
  else if σ = {σ1}σ2      then return r(σ1, Σ) ∧ r(σ2, Σ);
  else if σ = H(σ1)       then return r(σ1, Σ);
  else if σ = {[σ1]}σ2+   then return r(σ1, Σ) ∧ r(σ2+, Σ);
  else if σ = [{σ1}]σ2−   then return r(σ1, Σ) ∧ r(σ2−, Σ);
  else if σ = σ1+         then return r(σ1, Σ);
  else if σ = σ1−         then return r(σ1, Σ);
  else (σ ∈ A\Σ)          return FALSE;
}
```

Informally, this algorithm recursively checks whether σ can be deduced from the set Σ using *introduction* rules, i.e. (2)–(7), only; in this respect, an introduction rule is a rule that builds a new term by combining one or more simpler terms.
The properties of $\overline{\Sigma}$ are then expressed by the following theorems adapted from [DSV03] and proved in Appendix C; these properties make it a good candidate as a finite and minimised representation of the term generation capabilities of an intruder that has learned the set of terms Σ.

Theorem 3.1 (Finiteness and uniqueness) *For each finite set of terms $\Sigma \subseteq \mathcal{T}$, $\overline{\Sigma}$ is finite and unique.*

Theorem 3.2 (Minimality under introduction rules) *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, and $\sigma \in \overline{\Sigma}$. Then $\widehat{(\overline{\Sigma}\setminus\{\sigma\})}^{I} \subset \widehat{\overline{\Sigma}}$.*

Theorem 3.3 (Closure preservation) *For each finite set of terms $\Sigma \subseteq \mathcal{T}$, $\widehat{\overline{\Sigma}} = \widehat{\Sigma}$.*

Theorem 3.4 (Computability) *For each finite set of terms $\Sigma \subseteq \mathcal{T}$, $\overline{\Sigma}$ can be computed in a finite number of steps.*

Theorem 3.5 (Decidability) *Let $\sigma \in \mathcal{T}$ be any finite term and $\Sigma \subseteq \mathcal{T}$ be a finite set of terms. Then, the question if $\sigma \in \widehat{\Sigma}$ is decidable.*

The computation of $\overline{\Sigma}$ from Σ can be carried out by repeatedly applying closure rules (1)–(14). More precisely, let us define a *reduction rule* as a triple $R = \langle \Sigma_I,\ C,\ \Sigma_O \rangle$, where Σ_I and Σ_O are sets of terms representing, respectively premises and conclusions of closure rule C.
Applying reduction step R to a finite set of terms Σ means eliminating the premises from and adding the conclusions to Σ. This is written $\Sigma \xrightarrow{R} \Sigma'$, where $\Sigma' = (\Sigma \setminus \Sigma_I) \cup \Sigma_O$ is the resulting set.
Given a finite set of terms Σ, a *reduction of* Σ is a finite sequence of application of reduction rules R_i to finite sets of terms Σ_i, denoted:

$$\Sigma_0 \xrightarrow{R_0} \Sigma_1 \xrightarrow{R_1} \Sigma_2 \cdots \Sigma_{k-1} \xrightarrow{R_{k-1}} \Sigma_k$$

such that $\Sigma_0 = \Sigma$ and $R_i \in \mathcal{R}(\Sigma_i)$, where $\mathcal{R}(\Sigma_i)$ is the set of reduction rules whose pre-conditions are satisfied by Σ_i. Below, the notation $a \rightarrow b$ means that if the pre-condition a is true, then the reduction rule b can be applied

in Σ_i, that is, $b \in \mathcal{R}(\Sigma_i)$. The set $\mathcal{R}(\Sigma_i)$ is the least set such that the following relations hold:

$$\mathrm{H}(\sigma) \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma, \Sigma_i) \ \rightarrow\ \langle\{\mathrm{H}(\sigma)\}, (5), \emptyset\rangle \tag{26}$$

$$\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma_1, \Sigma_i) \ \wedge\ \mathtt{r}(\sigma_2^+, \Sigma_i) \ \rightarrow\ \langle\{\{[\sigma_1]\}_{\sigma_2^+}\}, (6), \emptyset\rangle \tag{27}$$

$$[\{\sigma_1\}]_{\sigma_2^-} \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma_1, \Sigma_i) \ \wedge\ \mathtt{r}(\sigma_2^-, \Sigma_i) \ \rightarrow\ \langle\{[\{\sigma_1\}]_{\sigma_2^-}\}, (7), \emptyset\rangle \tag{28}$$

$$\sigma^+ \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma, \Sigma_i) \ \rightarrow\ \langle\{\sigma^+\}, (8), \emptyset\rangle \tag{29}$$

$$\sigma^- \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma, \Sigma_i) \ \rightarrow\ \langle\{\sigma^-\}, (8), \emptyset\rangle \tag{30}$$

$$\mathrm{suc}(\sigma) \in \Sigma_i \ \rightarrow\ \langle\{\mathrm{suc}(\sigma)\}, (9), \{\sigma\}\rangle \tag{31}$$

$$(\sigma_1, \sigma_2) \in \Sigma_i \ \rightarrow\ \langle\{(\sigma_1, \sigma_2)\}, (10), \{\sigma_1, \sigma_2\}\rangle \tag{32}$$

$$\{\sigma_1\}_{\sigma_2} \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma_2, \Sigma_i) \ \rightarrow\ \langle\{\{\sigma_1\}_{\sigma_2}\}, (11), \{\sigma_1\}\rangle \tag{33}$$

$$\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma_2^-, \Sigma_i) \ \wedge\ \neg\mathtt{r}(\sigma_1, \Sigma_i) \ \rightarrow\ \langle\{\{[\sigma_1]\}_{\sigma_2^+}\}, (12), \{\sigma_1, \{[\sigma_1]\}_{\sigma_2^+}\}\rangle \tag{34}$$

$$[\{\sigma_1\}]_{\sigma_2^-} \in \Sigma_i \ \wedge\ \mathtt{r}(\sigma_2^+, \Sigma_i) \ \wedge\ \neg\mathtt{r}(\sigma_1, \Sigma_i) \ \rightarrow\ \langle\{[\{\sigma_1\}]_{\sigma_2^-}\}, (13), \{\sigma_1, [\{\sigma_1\}]_{\sigma_2^-}\}\rangle \tag{35}$$

$$\sigma^+ \in \Sigma_i \ \wedge\ \sigma^- \in \Sigma_i \ \rightarrow\ \langle\{\sigma^+, \sigma^-\}, (14), \{\sigma\}\rangle \tag{36}$$

It can be shown that the reduction process preserves closures and leads to $\overline{\Sigma}$ in a finite number of steps, i.e. that the following propositions hold, as shown in Appendix C:

Proposition 3.1 *If* $\Sigma \xrightarrow{R} \Sigma'$ *is a one-step reduction, then* $\widehat{\Sigma} = \widehat{\Sigma'}$.

Proposition 3.2 *Given a finite set of terms* Σ*, all of its reductions* $\Sigma = \Sigma_0 \xrightarrow{R_0} \Sigma_1 \cdots \Sigma_{k-1} \xrightarrow{R_{k-1}} \Sigma_k$ *have a finite number* k *of steps and end in* $\overline{\Sigma}$*, that is,* $\Sigma_k = \overline{\Sigma}$.

In analogy with the natural deduction system, and unlike [CJM98], we allow *both* $\mathcal{I}$ and $\mathcal{E}$ rules to be applied in the computation of $\overline{\Sigma}$ from Σ, under the constraint of their pre-condition, and at the expense of a greater computational complexity, which will be analysed in Sect. 5. However, as entailed by these theorems, this approach does neither sacrifice decidability nor computability.
The discussion above entails that if a new term ρ is added to a minimal closure seed $\overline{\Sigma}$, e.g. as a consequence of an output process, the new minimal closure seed $\overline{\overline{\Sigma} \cup \{\rho\}}$ can be *incrementally computed* by a reduction that starts from $\overline{\Sigma} \cup \{\rho\}$, without restarting from scratch; it can be expected that the incremental computation is far less expensive in terms of computing power with respect to the full one.

3.2. Examples of reduction

As an example, let us start with the minimal closure seed

$$\overline{\Sigma} = \{c, \{\{k_1\}_{k_2}\}_{k_3}, \{[m]\}_{k_1{}^+}, k_1{}^+, k_2\}$$

and let us observe the reduction process described above when the new term $\rho = k_3$ is added; in Table 4 the second column lists the contents of the partially reduced sets Σ_i at each reduction step, the next column recalls the reduction rule applied in that step, and the rightmost two columns list the set of elements removed from and added to Σ_i by the application of the rule, denoted Σ_I and Σ_O, respectively.
In the table, rule applications are serialised, i.e. only one rule is applied at each step for clarity; in an actual implementation, all independent rules can be applied simultaneously, as outlined in the complexity analysis carried out in Sect. 5.
Table 5 presents a reduction involving a non-atomic symmetric key; the initial intruder's knowledge is $\overline{\Sigma} = \{\{k_A\}_{k_S}, \{m\}_{\{k_B\}_{k_A}}, k_B, \mathrm{H}(m)\}$ and the added term is $\rho = k_S$. Note that in the second step, the premises of rule (33) are indeed satisfied, because $\mathtt{r}(\{k_B\}_{k_A}, \Sigma_1)$ is true, even if $\{k_B\}_{k_A} \notin \Sigma_1$.

4. Commutative and associative operators

The introduction of operators with special properties has several subtle implications in the intruder's knowledge representation, where the necessary and sufficient condition for the equivalence of two terms simply relies on

Table 5. A reduction involving a non-atomic key

i	Σ_i	$\mathcal{R}$	Σ_I	Σ_O
0	$\{\{k_A\}_{k_S}, \{m\}_{\{k_B\}_{k_A}}, k_B, \mathrm{H}(m), k_S\}$	(33)	$\{\{k_A\}_{k_S}\}$	$\{k_A\}$
1	$\{\{m\}_{\{k_B\}_{k_A}}, k_B, \mathrm{H}(m), k_S, k_A\}$	(33)	$\{\{m\}_{\{k_B\}_{k_A}}\}$	$\{m\}$
2	$\{k_B, \mathrm{H}(m), k_S, k_A, m\}$	(26)	$\{\mathrm{H}(m)\}$	$\emptyset$
3	$\{k_B, k_S, k_A, m\}$			

their syntactic identity. In fact, when a free term algebra is extended with commutative and/or associative operators, two or more syntactic representations of the same term are allowed. For example, if $\odot$ is a commutative, binary operator, $\overline{\Sigma}_1 = \{\sigma \odot \rho\}$ and $\overline{\Sigma}_2 = \{\rho \odot \sigma\}$ express exactly the same knowledge and are both minimum and canonical according to the original definition of $\overline{\Sigma}$ presented in Sect. 3, but clearly they are *not* equal in a syntactical sense.

In order to correctly handle such operators, we introduce the notion of *canonical term representation* and we assume that all algorithms presented in Sect. 3, such as Σ minimisation and $\mathtt{r}(\cdot, \cdot)$ always act on canonical terms. Moreover, the intruder's knowledge closure rules, the predicates defining $\overline{\Sigma}$, the $\mathtt{r}(\cdot, \cdot)$ algorithm, and the $\overline{\Sigma}$ minimisation rules have to be extended to take the operator's properties into account. Having done this, it can be proved that all the properties of $\overline{\Sigma}$ mentioned in Sect. 3, including *incremental computability*, still hold, as shown in Appendix D.

The canonical term representation leverages on the concept of *term equivalence class* induced by the operators' properties: two terms belong to the same equivalence class, induced by a given property, iff they can be made equal by applying that property. For example $\mathrm{H}(a \odot b)$ and $\mathrm{H}(b \odot a)$ are in the same equivalence class if $\odot$ is a commutative operator, because they can be made syntactically equal by applying the commutative property.

The canonical term representation has the important role of selecting, for each term equivalence class induced by the operators' properties, a unique element that will represent the class as a whole in all contexts.

From now on it is assumed that two terms that can be obtained from each other by a suitable application of one or more operators' properties are indistinguishable from the intruder's point of view. For this reason, any additional observation the intruder can make on term generation besides the final value, such as timing or accuracy, is neglected.

In addition to the main notational conventions of Table 1, note also that in this section:

- When multiple atoms, terms, and operators are needed in the same context, a unique, numeric subscript is used to distinguish them. For example, a_1 and a_2 are two distinct, and possibly different, atoms.
- When appropriate, the arity of the operator is explicitly denoted with a subscript, for example: $\odot_n$ is an n-ary operator. When more than one operator is needed in the same context, the *first* subscript singles out the operator, and the *second* one gives its arity. For example, $\odot_{1n}$ and $\odot_{2m}$ are two distinct, and possibly different operators; the first one has arity n, the second has arity m.
- The infix and prefix operator notation are used interchangeably, that is, $a \odot b = \odot(a, b)$.

4.1. Canonical term representation

Term canonicalisation has the purpose of determining a unique, canonical form for each $\sigma \in \mathcal{T}$; the canonical form is used when inserting a term into the intruder's knowledge, and when checking whether the intruder is able or not to synthesise a term from a given knowledge.

Term canonicalisation selects one representative element from each equivalence class induced on $\mathcal{T}$ by operator properties according to the following informal rules:

- The canonical form of an atom is the atom itself.
- The canonical form of the invocation of an operator without special properties is the invocation of the same operator on the same operands, put into canonical form.
- The canonical form of the invocation of a commutative operator $\odot$ on a list of operands $\sigma_1, \ldots, \sigma_n$ is the invocation of the same operator on the operands put into canonical form and then sorted according to $\leq_{\mathcal{T}}$, to ensure the uniqueness of representation (rule 37). The relation $\leq_{\mathcal{T}}$ is a total order relation on the *canonical subset* of $\mathcal{T}$ and will be defined in Sect. 4.2.

- The canonical form of the invocation of an associative operator $\odot$ on a list of operands $\sigma_1, \ldots, \sigma_n$ is the invocation of the same operator on the operands taken into canonical form. If, after canonicalisation, some operands have the same operator $\odot$ as their top-level operator, the hierarchy of invocations of $\odot$ is *flattened* (rule 38).

Formally, the term canonicalisation function $\mathcal{C}^* : \mathcal{T} \to \mathcal{T}$ is the result of the reduction performed by the term rewrite system defined according to the following set of rules, where $\mathcal{P}_C$ is the set of commutative operators, and $\mathcal{P}_A$ is the set of associative operators. The intersection of these sets may be non-empty because operators may have multiple properties.
In the following, a rewrite rule like $\frac{\pi}{\sigma \overset{\mathcal{C}}{\longmapsto} \sigma'}$ should be read as: when the prerequisite predicate π is true, then term σ can be rewritten as σ'.

$$\frac{\begin{array}{l}\odot \in \mathcal{P}_C \ \wedge \ (\exists\, i \mid \sigma_i \neq \mathcal{C}^*(\sigma_i) \ \vee \ \exists\, i, j, \ \ i < j, \ \ \mid \mathcal{C}^*(\sigma_i) \not\leq_{\mathcal{T}} \mathcal{C}^*(\sigma_j)) \ \wedge \\ \exists\, k_1, \ldots, k_n, k_i \neq k_l \ \ \forall\, i, l \mid \mathcal{C}^*(\sigma_{k_i}) \leq_{\mathcal{T}} \mathcal{C}^*(\sigma_{k_{i+1}}), \quad i = 1, \ldots, n-1\end{array}}{\odot(\sigma_1, \ldots, \sigma_n) \overset{\mathcal{C}}{\longmapsto} \odot(\mathcal{C}^*(\sigma_{k_1}), \ldots, \mathcal{C}^*(\sigma_{k_n}))} \tag{37}$$

$$\frac{\odot \in \mathcal{P}_A \ \wedge \ \exists\, i \mid \mathcal{C}^*(\sigma_i) = \odot(\sigma_{i1}, \ldots, \sigma_{im})}{\odot(\sigma_1, \ldots, \sigma_n) \overset{\mathcal{C}}{\longmapsto} \odot(\sigma_1, \ldots, \sigma_{i-1}\sigma_{i1}, \ldots, \sigma_{im}\sigma_{i+1}, \ldots, \sigma_n)} \tag{38}$$

Theorem 4.1 *The function $\mathcal{C}^* : \mathcal{T} \to \mathcal{T}$ is computable for any finite $\sigma \in \mathcal{T}$.*

The proof is based on the observation that the term rewrite system defined above is convergent.

4.2. Total order relation on terms

We assume that there is a total order relation $\leq_{\mathcal{P}}$ on the elements of $\mathcal{P}$, which is always possible when $\mathcal{P}$ is finite or is a countable infinity; similarly, we assume that there is a total order relation $\leq_{\mathcal{A}}$ on the elements of $\mathcal{A}$, as it is the case when the total number of atomic terms ever used in a session of the protocol is finite or is a countable infinity. Under these hypotheses, we define a relation $\leq_{\mathcal{T}}$ on the *canonical subset* of $\mathcal{T}$ as the smallest relation that satisfies the following implications:

$$\sigma_1 \in \mathcal{A} \ \wedge \ \sigma_2 \in \mathcal{A} \ \wedge \ \sigma_1 \leq_{\mathcal{A}} \sigma_2 \Rightarrow \sigma_1 \leq_{\mathcal{T}} \sigma_2 \tag{39}$$

$$\sigma_1 \in \mathcal{A} \ \wedge \ \sigma_2 \notin \mathcal{A} \Rightarrow \sigma_1 \leq_{\mathcal{T}} \sigma_2 \tag{40}$$

$$\sigma_1 = \odot_{1n}(\ldots) \ \wedge \ \sigma_2 = \odot_{2m}(\ldots) \ \wedge \ n < m \Rightarrow \sigma_1 \leq_{\mathcal{T}} \sigma_2 \tag{41}$$

$$\begin{array}{r}\sigma_1 = \odot_{1n}(\sigma_{11}, \ldots, \sigma_{1n}) \ \wedge \ \sigma_2 = \odot_{2n}(\sigma_{21}, \ldots, \sigma_{2n}) \\ \wedge \ \exists\, j \mid \sigma_{1j} \neq \sigma_{2j} \ \wedge \ \sigma_{1j} \leq_{\mathcal{T}} \sigma_{2j} \\ \wedge \ \sigma_{1i} = \sigma_{2i} \ \ \forall\, i < j \Rightarrow \sigma_1 \leq_{\mathcal{T}} \sigma_2\end{array} \tag{42}$$

$$\begin{array}{r}\sigma_1 = \odot_{1n}(\sigma_{11}, \ldots, \sigma_{1n}) \ \wedge \ \sigma_2 = \odot_{2n}(\sigma_{21}, \ldots, \sigma_{2n}) \\ \wedge \ \sigma_{1i} = \sigma_{2i} \ \ \forall\, i \ \wedge \ \ \odot_{1n} \leq_{\mathcal{P}} \odot_{2n} \Rightarrow \sigma_1 \leq_{\mathcal{T}} \sigma_2\end{array} \tag{43}$$

It can be shown that the $\leq_{\mathcal{T}}$ relation, as defined above, is computable.

Theorem 4.2 *The relation $\leq_{\mathcal{T}}$ is a total order relation on the canonical subset of $\mathcal{T}$.*

4.3. Commutative operators

Let us assume, without loss of generality, that $\odot$ is a binary, commutative operator. The ability to synthesise a compound term by means of operator $\odot$ can be captured by the following closure rule, to be added to rules (1–14):

$$\sigma_1 \in \widehat{\Sigma} \ \wedge \ \sigma_2 \in \widehat{\Sigma} \ \wedge \ \sigma_1 \leq_{\mathcal{T}} \sigma_2 \ \Rightarrow \ \sigma_1 \odot \sigma_2 \in \widehat{\Sigma}. \tag{44}$$

Accordingly, the following, additional predicate for the definition of $\overline{\Sigma}$ must be added to (17–25):

$$\sigma_1 \odot \sigma_2 \in \overline{\Sigma} \Leftrightarrow (\sigma_1 \notin \widehat{\Sigma} \ \vee \ \sigma_2 \notin \widehat{\Sigma}) \ \wedge \ \sigma_1 \leq_{\mathcal{T}} \sigma_2. \tag{45}$$

This predicate has been defined with reference to the closure rule (44) in the same way as predicates (17–25) were derived from (1–14), also taking canonicalisation into account. For example, predicate (20) states that $\{\sigma_1\}_{\sigma_2} \in \overline{\Sigma}$ iff $\sigma_2 \notin \widehat{\Sigma}$ because otherwise it would be possible to reduce $\{\sigma_1\}_{\sigma_2}$ into simpler terms by rule (11), and rebuild it from the reduction results by rule (4).
In this case, $\sigma_1 \odot \sigma_2$ can possibly belong to $\overline{\Sigma}$ iff it cannot be synthesised from its operands σ_1 and σ_2, that is, iff $\sigma_1 \notin \widehat{\Sigma} \ \vee \ \sigma_2 \notin \widehat{\Sigma}$. Moreover, $\sigma_1 \odot \sigma_2$ can possibly belong to $\overline{\Sigma}$ iff it is, or has been put, in canonical form, that is, from canonicalisation rule (37), iff $\sigma_1 \leq_{\mathcal{T}} \sigma_2$. In turn, this implies that only canonical terms belong to $\overline{\Sigma}$.
Similarly, closure rule (44) gives rise to the following additional $\overline{\Sigma}$ reduction rule to be added to (26–36):

$$\sigma_1 \odot \sigma_2 \in \Sigma_i \ \wedge \ \mathtt{r}(\sigma_1, \Sigma_i) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma_i) \ \rightarrow \ \langle \{\sigma_1 \odot \sigma_2\}, (44), \emptyset \rangle \tag{46}$$

When its premises are satisfied, rule (46) has a premise term $\sigma_1 \odot \sigma_2$ which can be generated using other elements of Σ_i, because the truth of $\mathtt{r}(\sigma_1, \Sigma_i) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma_i)$ makes closure rule (44) applicable, and simply removes it. Therefore, it guarantees that no loops are introduced in the reduction process, preserves the closure $\widehat{\Sigma}$ and ensures that Σ_{i+1} only contains canonical terms if the same is true for Σ_i. Closure rule (44) is also the starting point to extend function $\mathtt{r}(\cdot, \cdot)$ presented in Sect. 3:

```
boolean r(σ, Σ) {
  ...
  else if σ = ⊙(σ1, σ2) ∧ ⊙ ∈ P_C then
     return r(σ1, Σ) ∧  r(σ2, Σ) ;
  ...
}
```

4.4. Associative operators

Let us assume, without loss of generality, that $\odot \in \mathcal{P}_A$ is a n-ary, associative operator, with $n \geq 2$. In addition, let us assume that the (degenerate) invocation of operator $\odot$ with one operand is the operand itself: $\odot(\sigma) = \sigma$. The ability to synthesise a compound term by means of operator $\odot$, possibly leveraging the associative property, is captured by the following closure rule:

$$\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \ \wedge \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma} \Rightarrow \odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma} \tag{47}$$

by the additional predicate for the definition of $\overline{\Sigma}$:

$$\odot(\sigma_1, \ldots, \sigma_n) \in \overline{\Sigma} \Leftrightarrow \forall\, i \in [1, n-1] \ \ \odot(\sigma_1, \ldots, \sigma_i) \notin \widehat{\Sigma} \ \vee \ \odot(\sigma_{i+1}, \ldots, \sigma_n) \notin \widehat{\Sigma} \tag{48}$$

by the additional $\overline{\Sigma}$ reduction rule:

$$\left.\begin{array}{ll} & \odot(\sigma_1, \ldots, \sigma_n) \in \Sigma_i \\ \wedge & \exists\, i \in [1, n-1] \mid \mathtt{r}(\odot(\sigma_1, \ldots, \sigma_i), \Sigma_i) \\ \wedge & \mathtt{r}(\odot(\sigma_{i+1}, \ldots, \sigma_n), \Sigma_i) \end{array}\right\} \rightarrow \ \langle \{\odot(\sigma_1, \ldots, \sigma_n)\}, (47), \emptyset \rangle \tag{49}$$

and by the following extension to the $\mathtt{r}(\cdot, \cdot)$ function:

```
boolean r(σ, Σ) {
  ...
  else if σ = ⊙(σ1, ..., σn) ∧ ⊙ ∈ P_A
  then return =∃ i ∈ [1, n − 1] |
              r(⊙(σ1, ..., σi), Σ) ∧ r(⊙(σi+1, ..., σn), Σ);
  ...
}
```

If $\odot$ is *both* associative and commutative, the rules outlined above must be complemented because if the lists: $(\sigma_1, \ldots, \sigma_i)$ and $(\sigma_{i+1}, \ldots, \sigma_n)$ are both sorted according to $\leq_{\mathcal{T}}$, this does *not* imply that their concatenation $(\sigma_1, \ldots, \sigma_i, \sigma_{i+1}, \ldots, \sigma_n)$ is also sorted according to the same relation.

(1)	$A \to B$	:	$X = G^{n_1} \ mod \ N$
(2)	$B \to A$	:	$Y = G^{n_2} \ mod \ N$
(3)	$A \to B$	:	$\{M\}_{Y^{n_1} \ mod \ N}$

Fig. 2. The Diffie–Hellman key exchange protocol

So, for example, assuming that $\sigma_1 \leq_T \sigma_2 \leq_T \sigma_3$ it can be $\odot(\sigma_1, \sigma_2, \sigma_3) \in \widehat{\Sigma}$ even if $\sigma_1 \notin \widehat{\Sigma} \ \wedge \ \odot(\sigma_2, \sigma_3) \notin \widehat{\Sigma} \ \wedge \ \odot(\sigma_1, \sigma_2) \notin \widehat{\Sigma} \ \wedge \ \sigma_3 \notin \widehat{\Sigma}$ and closure rule (47) cannot be applied. In fact, this happens when $\sigma_2 \in \widehat{\Sigma} \ \wedge \ \odot(\sigma_1, \sigma_3) \in \widehat{\Sigma}$. On the other hand, the same closure rule can introduce non-canonical terms in $\widehat{\Sigma}$, for the same reason. The refined closure rule for a commutative and associative operator therefore is:

$$\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \ \wedge \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma} \Rightarrow \mathcal{C}^*(\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n})) \in \widehat{\Sigma}, \tag{50}$$

where the invocation of the canonicalisation operator $\mathcal{C}^*$ avoids non-canonical terms in $\widehat{\Sigma}$. On the other hand, when defining the predicate for the definition of $\overline{\Sigma}$ and the corresponding reduction rules on terms in the form $\odot(\sigma_1, \ldots, \sigma_n)$, all possible partitions of $S = \{\sigma_1, \ldots, \sigma_n\}$ into two subsets must be considered:

$$\begin{aligned} &\odot(\sigma_1, \ldots, \sigma_n) \in \overline{\Sigma} \Leftrightarrow \\ &\forall S_1, S_2 \mid S_1 \neq \emptyset \ \wedge \ S_2 \neq \emptyset \ \wedge \ S_1 \cap S_2 = \emptyset \ \wedge \ S_1 \cup S_2 = \{\sigma_1, \ldots, \sigma_n\} \ \ \odot(S_1) \notin \widehat{\Sigma} \ \vee \ \odot(S_2) \notin \widehat{\Sigma} \end{aligned} \tag{51}$$

$$\left.\begin{array}{ll} & \odot(\sigma_1, \ldots, \sigma_n) \in \Sigma_i \\ \wedge & \exists S_1, S_2 \mid S_1 \neq \emptyset \ \wedge \ S_2 \neq \emptyset \\ \wedge & S_1 \cap S_2 = \emptyset \\ \wedge & S_1 \cup S_1 = \{\sigma_1, \ldots, \sigma_n\} \\ \wedge & \mathtt{r}(\odot(S_1), \Sigma_i) \ \wedge \ \mathtt{r}(\odot(S_2), \Sigma_i) \end{array}\right\} \to \ \langle\{\odot(\sigma_1, \ldots, \sigma_n)\}, (50), \emptyset\rangle \tag{52}$$

In the above rules, with a little abuse of notation, we let $\odot(S) = \odot(\sigma_1, \ldots, \sigma_n)$, where $S = \{\sigma_1, \ldots, \sigma_n\}$ is a set of terms. In the same manner, the extension to the $\mathtt{r}(\cdot, \cdot)$ function must be:

```
boolean r(σ, Σ) {
  ...
  else if σ = ⊙(σ1, ..., σn) ∧ ⊙ ∈ P_A ∩ P_C
  then return  ∃ S1, S2 | S1 ≠ ∅ ∧ S2 ≠ ∅ ∧  S1 ∩ S2 = ∅ ∧
               S1 ∪ S2 = {σ1, ..., σn} ∧
               r(⊙(S1), Σ)  ∧  r(⊙(S2), Σ) ;
  ...
}
```

4.5. Modular exponentiation

In the Diffie–Hellman protocol [DH76], depicted in Fig. 2, two numbers G and N are publicly agreed on by the communicating principals A and B. A chooses $X = G^{n_1} \ mod \ N$ for some random n_1 and sends the result to B as message (1). B chooses $Y = G^{n_2} \ mod \ N$ for some random n_2 and sends the result to A as message (2). A computes $k = Y^{n_1} \ mod \ N$ and B computes $k = X^{n_2} \ mod \ N$. The result of these two calculations is the same and is equal to the new session key k. This provides a means for exchanging keys but gives no guarantees of authenticity.

The last step (3) does not belong to the key-exchange protocol itself, but shows how the new key shared among agents can be used: A sends a message M encrypted by means of $k = (G^{n_2} \ mod \ N)^{n_1} \ mod \ N$ to B which in turn can decrypt it by means of the same key $k = (G^{n_1} \ mod \ N)^{n_2} \ mod \ N$.

To model the computation carried out in this protocol, we define a new operator as:

$$\ll \sigma \gg_N^{\sigma_1, \ldots, \sigma_n} = \sigma^{\sigma_1, \ldots, \sigma_n} \ mod \ N. \tag{53}$$

The new operator has the following properties:

$$\ll \sigma \gg_N^{\sigma_1, \ldots, \sigma_n} = \ll \sigma \gg_N^{\Pi(\sigma_1, \ldots, \sigma_n)}, \tag{54}$$

where $\Pi(\sigma_1, \ldots, \sigma_n)$ is a permutation, which captures the commutative property of the exponent product, and:

$$\ll \ll \sigma \gg_N^{\sigma_{11},\ldots,\sigma_{1n}} \gg_N^{\sigma_{21},\ldots,\sigma_{2m}} = \ll \sigma \gg_N^{\sigma_{11},\ldots,\sigma_{1n}\sigma_{21},\ldots,\sigma_{2m}}, \tag{55}$$

which reflects the ability to associate nested invocations of the operator in base position.

The generic term canonicalisation rewrite rules outlined in Sect. 4.1 can be specialised for the Diffie–Hellman operator, taking into account its well-known operator's algebraic properties recalled in (53–55). Since these rules have been derived by specialisation, they preserve the properties of the generic rules they derive from, namely, all the properties of $\overline{\Sigma}$ mentioned in Sect. 3 still hold.

The specialised rewrite rules are:

$$\frac{\begin{array}{l}(\exists\, i \mid \sigma_i \neq \mathcal{C}^*(\sigma_i) \ \vee \ \exists\, i, j, \ \ i < j, \ \ \mid \mathcal{C}^*(\sigma_i) \not\leq_{\mathcal{T}} \mathcal{C}^*(\sigma_j)) \ \wedge \\ \exists\, k_1, \ldots, k_n, \ \ k_i \neq k_l \ \ \forall\, i, l \mid \mathcal{C}^*(\sigma_{k_i}) \leq_{\mathcal{T}} \mathcal{C}^*(\sigma_{k_{i+1}}), \quad i = 1, \ldots, n-1\end{array}}{\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n} \overset{\mathcal{C}}{\longmapsto} \ll \sigma \gg_N^{\mathcal{C}^*(\sigma_{k_1}),\ldots,\mathcal{C}^*(\sigma_{k_n})}} \tag{56}$$

$$\frac{\sigma = \ll \sigma' \gg_N^{\sigma_{11},\ldots,\sigma_{1n}}}{\ll \sigma \gg_N^{\sigma_{21},\ldots,\sigma_{2m}} \overset{\mathcal{C}}{\longmapsto} \ll \sigma' \gg_N^{\sigma_{11},\ldots,\sigma_{1n},\sigma_{21},\ldots,\sigma_{2m}}} \tag{57}$$

The intruder's ability to compute the Diffie–Hellman operator can be expressed by the following closure rules which are the specialisation of (44) and (50):

$$\sigma' \in \widehat{\Sigma} \ \wedge \ \sigma_1, \ldots, \sigma_n \in \widehat{\Sigma} \ \wedge \ \sigma_i \leq_{\mathcal{T}} \sigma_{i+1} \ \ \forall\, i \in [1, n-1] \Rightarrow \ll \sigma' \gg_N^{\sigma_1,\ldots,\sigma_n} \in \widehat{\Sigma} \tag{58}$$

$$\ll \sigma' \gg_N^{\sigma_{11},\ldots,\sigma_{1n}} \in \widehat{\Sigma} \ \wedge \ \sigma_{21}, \ldots, \sigma_{2m} \in \widehat{\Sigma} \Rightarrow \mathcal{C}^*(\ll \sigma' \gg_N^{\sigma_{11},\ldots,\sigma_{1n},\sigma_{21},\ldots,\sigma_{2m}}) \in \widehat{\Sigma} \tag{59}$$

Rule (58) captures the commutative property of the Diffie–Hellman operator with respect to $\sigma_1, \ldots, \sigma_n$; similarly, rule (59) captures the ability to flatten nested invocations of the Diffie–Hellman operator in base position by grouping exponents $\sigma_{11}, \ldots, \sigma_{1n}$ and $\sigma_{21}, \ldots, \sigma_{2m}$ together.

Notice that, in rule (58), canonicalisation is unnecessary, because the rule's prerequisites implicitly ensure that the right-hand term is canonical.

According to the closure rules above, we can introduce the following additional predicates for the definition of $\overline{\Sigma}$; below, S_1 and S_2 represent any binary partition of $S = \{\sigma_1, \ldots, \sigma_n\}$ and, with a little abuse of notation, we let $\ll \sigma \gg_N^{S_1} = \ll \sigma \gg_N^{\sigma_{11},\ldots,\sigma_{1k}}$:

$$\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n} \in \overline{\Sigma} \Leftrightarrow (\sigma \notin \widehat{\Sigma} \ \vee \ \exists\, i \mid \sigma_i \notin \widehat{\Sigma}) \ \wedge \ \sigma_j \leq_{\mathcal{T}} \sigma_{j+1} \quad \forall\, j \in [1, n-1] \tag{60}$$

$$\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n} \in \overline{\Sigma} \Leftrightarrow \forall\, S_1, S_2 \mid \left\{ \begin{array}{l} S_1 = \{\sigma_{11}, \ldots, \sigma_{1k}\}, \\ S_2 = \{\sigma_{21}, \ldots, \sigma_{2l}\}, \\ S_1 \neq \emptyset \ \wedge \ S_2 \neq \emptyset \ \wedge \ S_1 \cap S_2 = \emptyset \ \wedge \\ S_1 \cup S_2 = \{\sigma_1, \ldots, \sigma_n\} \ \wedge \\ (\ll \sigma \gg_N^{S_1} \notin \widehat{\Sigma} \ \vee \ \exists\, i \mid \sigma_{2i} \notin \widehat{\Sigma}) \end{array} \right. \tag{61}$$

Last, closure rules (58) and (59) induce the following $\overline{\Sigma}$ reduction rules, which are the specialisation of (46) and (52):

$$\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n} \in \Sigma_i \ \wedge \ \mathtt{r}(\sigma, \Sigma_i) \ \wedge \ \mathtt{r}(\sigma_j, \Sigma_i) \quad \forall\, j \in [1, n] \rightarrow \langle \{\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n}\}, (58), \emptyset \rangle \tag{62}$$

Table 6. An example of canonicalisation

$\overline{\Sigma}$	ρ	$\mathcal{C}$
$\{c, G, N\}$	$\ll G \gg_N^{n_1}$	
$\{c, G, N, \ll G \gg_N^{n_1}\}$	$\ll G \gg_N^{n_2}$	
$\{c, G, N, \ll G \gg_N^{n_1}, \ll G \gg_N^{n_2}\}$	$\{M\}_{\ll\ll G \gg_N^{n_2} \gg_N^{n_1}}$	(57)
	$\{M\}_{\ll G \gg_N^{n_2 n_1}}$	(56)
	$\{M\}_{\ll G \gg_N^{n_1 n_2}}$	
$\{c, G, N, \ll G \gg_N^{n_1}, \ll G \gg_N^{n_2}, \{M\}_{\ll G \gg_N^{n_1 n_2}}\}$		

$$\left.\begin{array}{l} \ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n} \in \Sigma_i \ \wedge\ \exists\, S_1, S_2 \mid S_1 = \{\sigma_{11}, \ldots, \sigma_{1k}\}, \\ S_2 = \{\sigma_{21}, \ldots, \sigma_{2l}\}, S_1 \neq \emptyset \ \wedge\ S_2 \neq \emptyset \ \wedge\ S_1 \cap S_2 = \emptyset \\ \wedge\ S_1 \cup S_2 = \{\sigma_1, \ldots, \sigma_n\} \\ \wedge\ \mathtt{r}(\ll \sigma \gg_N^{S_1}, \Sigma_i) \ \wedge\ \mathtt{r}(\sigma_{2j}, \Sigma_i) \quad \forall\, j \in [1, l] \end{array}\right\} \rightarrow \langle \{\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n}\}, (59), \emptyset \rangle \tag{63}$$

and the following extension to the $\mathtt{r}(\cdot, \cdot)$ function:

```
boolean r(σ, Σ) {
  ...
  else if σ = ≪ σ′≫_N^{σ1,...,σn}
  then return  ( r(σ′, Σ)  ∧  r(σj, Σ)  ∀j )  ∨
               (∃S1 = {σ11, ..., σ1n}, S2 = {σ21, ..., σ2m} |
               S1 ≠ ∅ ∧ S2 ≠ ∅ ∧
               S1 ∩ S2 = ∅ ∧ S1 ∪ S2 = {σ1, ..., σn} ∧
               r(≪ σ′≫_N^{S1}, Σ)  ∧  r(σ2j, Σ)  ∀j);
  ...
}
```

It should be noted that if we allow $S_1 = \emptyset$ and assume that the degenerate application of modular exponentiation to any σ with an empty set of exponents yields σ itself (that is, $\ll \sigma \gg_N^{\emptyset} \equiv \sigma \quad \forall \sigma$), then (60) becomes a special case of (61). However, it may still be convenient to keep them separate, both to adhere to the general theory developed in Sects. 4.3 and 4.4 more closely, and to have distinct reduction rules, (62) and (63), for the two cases. In this way, by evaluating these rules in the given sequence, it becomes possible to avoid the expensive computation of all the partitions of $\{\sigma_1, \ldots, \sigma_2\}$ into two subsets S_1 and S_2, entailed by the evaluation of (63), when the simpler rule (62) applies and renders that computation useless. In fact, the conclusions of (62) remove $\ll \sigma \gg_N^{\sigma_1,\ldots,\sigma_n}$ from σ and thus unconditionally invalidate the premises of (63). From the implementation point of view, (62) can also be seen as an efficient shortcut to (63) that covers the simplest situations.

It can also be noted that the ability of handling any generic commutative operator allows this method to handle, at least in principle, other forms of encryption commutativity besides Diffie–Hellman. However, this possibility has not been further investigated in the present work.

4.6. Example of intruder's knowledge management

Table 6 shows how the intruder's knowledge grows up when the intruder intercepts the messages exchanged between A and B of Fig. 2: $\overline{\Sigma}$ is the minimised intruder's knowledge, ρ is the eavesdropped message, and column $\mathcal{C}$ contains the number of the canonicalisation rules needed to put ρ in canonical form.

The first and second message of Fig. 2 are already in canonical form, thus no canonicalisation is needed. Moreover, they cannot be split into simpler sub-messages, thus the new minimal intruder's knowledge is obtained simply by adding these messages to the initial one.

The last message of Fig. 2 has been encrypted by a nested invocation of the new operator thus, by rule (57), the associative property is exploited and $\{M\}_{\ll G \gg_N^{n_2 n_1}}$ is obtained. Last, rule (56) gives the canonical form $\{M\}_{\ll G \gg_N^{n_1 n_2}}$ where each exponent precedes the next one with respect to the total order relation among terms (we assume $n_1 \leq_{\mathcal{T}} n_2$).

Table 7. An example of reductions

	$\overline{\Sigma} = \{c, G, N, n_1\}$		$\mathcal{C}^*(\rho) = \ll G \gg_N^{n_1}$	
i	Σ_i	$\mathcal{R}$	Σ_I	Σ_O
0	$\{c, G, N, n_1, \ll G \gg_N^{n_1}\}$	(62)	$\{\ll G \gg_N^{n_1}\}$	$\emptyset$
1	$\{c, G, N, n_1\}$			
	$\overline{\Sigma} = \{c, G, N, n_1\}$		$\mathcal{C}^*(\rho) = \ll G \gg_N^{n_2}$	
i	Σ_i	$\mathcal{R}$	Σ_I	Σ_O
0	$\{c, G, N, n_1, \ll G \gg_N^{n_2}\}$			
	$\overline{\Sigma} = \{c, G, N, n_1, \ll G \gg_N^{n_2}\}$		$\mathcal{C}^*(\rho) = \{M\}_{\ll G \gg_N^{n_1 n_2}}$	
i	Σ_i	$\mathcal{R}$	Σ_I	Σ_O
0	$\{c, G, N, n_1, \ll G \gg_N^{n_2}, \{M\}_{\ll G \gg_N^{n_1 n_2}}\}$	(33)	$\{\{M\}_{\ll G \gg_N^{n_1 n_2}}\}$	$\{M\}$
1	$\{c, G, N, n_1, \ll G \gg_N^{n_2}, M\}$			
	$\overline{\Sigma} = \{c, G, N, n_1, \ll G \gg_N^{n_2}, M\}$			

Table 7 shows what happens to the intruder's knowledge when n_1 belongs to the initial $\overline{\Sigma}$, too. Since the canonicalisation details have already been analysed in the previous example, here we assume that each message the intruder intercepts is already in canonical form ($\mathcal{C}^*(\rho)$), and we focus on the intruder's knowledge minimisation steps, induced by the presence of n_1 in the initial $\overline{\Sigma}$.

At each protocol step $\overline{\Sigma}$ (obtained from the previous step) and the intercepted message $\mathcal{C}^*(\rho)$ are listed in a row. Below, the sequence of reduction steps follows, starting from $\Sigma_0 = \overline{\Sigma} \cup \{\mathcal{C}^*(\rho)\}$ and ending when a new minimised Σ_i has been obtained. For each step, the reduction rule used ($\mathcal{R}$) and the corresponding Σ_I and Σ_O sets are shown.

Starting from $\overline{\Sigma} = \{c, G, N, n_1\}$, the intruder captures the first message $\ll G \gg_N^{n_1}$ sent from A to B, thus $\Sigma_0 = \{c, G, N, n_1, \ll G \gg_N^{n_1}\}$ is obtained. $\ll G \gg_N^{n_1}$ can be synthesised starting from G and n_1 (N is embedded into the operator itself, but this is correct since N and G are *publicly agreed* between A and B), in fact rule (62) allows it to be removed from Σ_0.

The next intercepted message is $\ll G \gg_N^{n_2}$; it is added to $\overline{\Sigma}$ without any reduction, since it can neither be synthesised from simpler messages, nor it allows to decode some message already in $\overline{\Sigma}$. In fact, premises of both rules (62) and (63) fail.

The last intercepted message is $\{M\}_{\ll G \gg_N^{n_1 n_2}}$; since $\mathtt{r}(\Sigma_0, \ll G \gg_N^{n_1 n_2})$ is true, then rule (33) allows to remove $\{M\}_{\ll G \gg_N^{n_1 n_2}}$ from, and add M to the intruder's knowledge. In fact, $\mathtt{r}(\ll G \gg_N^{n_1 n_2}, \Sigma_0)$ corresponds to the last extension to $\mathtt{r}(\cdot, \cdot)$ made before the examples. By defining $S_1 = \{n_2\}$ and $S_2 = \{n_1\}$, we have that both $\mathtt{r}(\ll G \gg_N^{S_1}, \Sigma_0)$ and $\mathtt{r}(n_1, \Sigma_0)$ are true.

5. Complexity analysis

The complexity analysis is carried out in three steps: first, we describe a basic complexity model for operators with no special properties. Then, we extend the model to encompass operator's properties and take a sequence of reductions into account. A comprehensive set of experimental results concludes the section.

5.1. Basic computational complexity

5.1.1. On the computation of the question $\sigma \in \Sigma$

In this and in the following sections, let $\mathrm{op}(\sigma)$ be the number of operators in term σ and $\mathrm{n}(\Sigma)$ the number of elements in set Σ. Moreover, we extend the domain of the operator $\mathrm{op}(\cdot)$ to sets of terms, by defining it as:

$$\mathrm{op}(\Sigma) = \max_{\sigma \in \Sigma}(\mathrm{op}(\sigma))$$

in that case. Assuming that the comparison between atomic terms can be carried out in constant time, and the lookup of a term in Σ is sequential, then the computational complexity to check whether $\sigma \in \Sigma$ is $O(nm)$, where $n = \mathrm{n}(\Sigma)$ and $m = \mathrm{op}(\sigma)$.

5.1.2. *On the computation of* $\mathtt{r}(\sigma, \Sigma)$

The worst case happens when the operator's tree in σ is fully unbalanced, that is, each invocation of $\mathtt{r}(\sigma, \Sigma)$ on a compound term σ with m operators entails the recursive computation of $\mathtt{r}(\sigma_1, \Sigma)$ and $\mathtt{r}(\sigma_2, \Sigma)$, where σ_1 is atomic, and σ_2 has $m - 1$ operators.
In this case, each recursion step executes in $O(nm)$, where $n = \mathrm{n}(\Sigma)$ and $m = \mathrm{op}(\sigma)$ as shown above, and the recursion depth is m. So, the computational complexity of $\mathtt{r}(\sigma, \Sigma)$ is $O(nm^2)$, as it has also been proved in [McA93] in the more general framework of local inference rule sets, of which the definition of $\mathtt{r}(\cdot, \cdot)$ is a special case.

5.1.3. *On the incremental computation of* $\overline{\Sigma}$

For the sake of this discussion, and without loss of generality, let us define a *reduction step* as the simultaneous application of all independent reduction rules and let us denote it with $\rightarrow$. The incremental reduction of $\overline{\Sigma}$ after the addition of the new term ρ can be seen as a finite sequence of reduction steps starting from $\overline{\Sigma} \cup \{\rho\} = \Sigma_0$; reduction step i acts on set Σ_i and produces the (partially) reduced set Σ_{i+1}:

$$\overline{\Sigma} \cup \{\rho\} = \Sigma_0 \rightarrow \cdots \rightarrow \Sigma_i \rightarrow \Sigma_{i+1} \rightarrow \cdots$$

Let $n_i = \mathrm{n}(\Sigma_i)$ be the number of terms in Σ_i, and $m_i = \mathrm{op}(\Sigma_i)$ the maximum number of operators of terms in Σ_i. Then, we have the initial condition:

$$\begin{cases} n_0 = \mathrm{n}(\overline{\Sigma} \cup \{\rho\}) \\ m_0 = \mathrm{op}(\overline{\Sigma} \cup \{\rho\}) \end{cases}$$

In the worst case, up to n_i reduction rules can be applied at step i, one for each term in Σ_i; assuming that we can determine which reduction rule must be applied in constant time, each application of such rule entails one invocation of the $\in$ operator and up to two invocations of $\mathtt{r}(\cdot, \cdot)$; therefore, each application of a reduction rule at reduction step i has a computational complexity of $O(n_i m_i) + O(n_i m_i^2) = O(n_i m_i^2)$, and the computational complexity of reduction step i is $O(n_i^2 m_i^2)$.
At each reduction step i, whenever we remove a term σ from Σ_i, and add some other terms $\sigma_1, \ldots, \sigma_n$ derived from it, the added terms will always have one operator less than the term they originated from, that is, $\mathrm{op}(\{\sigma_1, \ldots, \sigma_n\}) = \mathrm{op}(\sigma) - 1$.
Therefore, as a result of reduction step i we remove n_i terms with m_i operators and add up to $2n_i$ terms with up to $m_i - 1$ operators; rules (34) and (35) are the only exceptions in this respect, because they do not remove any term. However, their application does not lead to the worst-case complexity because they leave in Σ_{i+1} the compound term $\{[\sigma_1]\}_{\sigma_2^+}$ or $\{[\sigma_1]\}_{\sigma_2^-}$ that cannot be further reduced, because $\sigma_1 \in \Sigma_{i+1}$ as a consequence of the application of the rule itself.
So, we can write:

$$\begin{cases} n_{i+1} = 2n_i \\ m_{i+1} = m_i - 1 \end{cases}$$

After a maximum of m_0 reduction steps, Σ_{m_0} is reduced to contain only atoms and no further reductions are possible. So the computational complexity of the reduction as a whole is:

$$\sum_{i=0}^{m_0} O(n_i^2 m_i^2) = \sum_{i=0}^{m_0} O((2^i n_0)^2 (m_0 - i)^2) = O(n_0^2 2^{2m_0}) = O\left((n_0 2^{m_0})^2\right) \tag{64}$$

5.1.4. *Comparison with normal, natural deductions*

When we assume that encryption keys are atomic, neglect public/private cryptosystems, and restrict our scope to normal, natural deductions only, as in [CJM98], we can replace all invocations of $\mathtt{r}(\sigma, \Sigma)$ in pre-conditions (26)–(36) with the simpler check $\sigma \in \Sigma$, and we can drop out reduction rules (27)–(30) and (34)–(36).

Accordingly, the complexity of a reduction step as defined in the previous section is reduced to $O(n_i m_i)$, because function $\mathtt{r}(\cdot, \cdot)$ is never invoked in this case. So, the complexity of the reduction process as a whole reduces to:

$$\sum_{i=0}^{m_0} O(n_i m_i^2) = \sum_{i=0}^{m_0} O(2^i n_0 (m_0 - i)^2) = O(n_0 2^{m_0}) \tag{65}$$

5.2. Commutative and associative operators

5.2.1. *Computation of* $\sigma_1 \leq_{\mathcal{T}} \sigma_2$

The total order relation on terms is defined on canonical terms only; therefore, we can assume that both σ_1 and σ_2 are canonical.
The computation of the relation when at least one operand is an atom is carried out in constant time, because only rule (39) or (40) may possibly apply.
When neither operand is an atom, rules (41–43) may apply, and the complexity is then linear with respect to the number of operators. In this respect, we consider a k-ary operator equivalent to the nesting of $k - 1$ applications of a binary operator. In summary, the complexity of $\sigma_1 \leq_{\mathcal{T}} \sigma_2$ is:

$$O(\mathrm{op}(\{\sigma_1, \ \sigma_2\}))$$

5.2.2. *Computation of* $\mathcal{C}^*(\sigma)$

The complexity of the canonicalisation depends on the actual term structure, because the properties of each operator determine which of rules (37–38) must be applied at the corresponding step of the recursive descent through the structure of the term.
The worst case is rule (37), which entails sorting the top-level operands, taken into canonical form, according to relation $\leq_{\mathcal{T}}$. When using a naive sort algorithm, the complexity of such a sort is quadratic in the complexity of $\leq_{\mathcal{T}}$ applied on the top-level operands. The worst case for a term with $m = \mathrm{op}(\sigma)$ commutative and associative operators happens when the operators can be flattened into a single application of the operator to $m + 1$ atomic operands, and the corresponding complexity is:

$$O((m + 1)^2) = O(m^2) \tag{66}$$

barring lower-order addenda.

5.2.3. *Computation of the question* $\sigma \in \Sigma$

This computation is not affected by the extension to the intruder's knowledge representation for associative and commutative operators, so its complexity is $O(mn)$ as before.

5.2.4. *Computation of* $\mathtt{r}(\sigma, \Sigma)$

The computation of $\mathtt{r}(\sigma, \Sigma)$ does not involve, per se, any canonicalisation, though the canonicalisation of σ must be performed before the invocation of $\mathtt{r}(\cdot, \cdot)$, so the complexity of the latter can be computed independently.
The worst case occurs when the operators in σ are both associative and commutative, because in this case the computation of $\mathtt{r}(\cdot, \cdot)$ involves the recursive invocation of the same function on all possible partitions of the operands in σ into two subsets. If a given σ_1 has k operands, the number P of such partitions is:

$$P = \frac{1}{2} \sum_{i=1}^{k-1} \left(\begin{array}{c} k \\ i \end{array} \right) = 2^{k-1} - 1, \quad k \geq 2 \tag{67}$$

Taking into account that the worst case for the complexity of the recursion still happens when the expression tree is fully unbalanced, and equating the complexity of all cases to the worst one, the complexity of $\mathtt{r}(\sigma_1, \Sigma)$, where $\mathrm{op}(\sigma_1) = k$ and we let $\sigma_1 = a \odot \sigma_2$, is:

$$[\mathbf{cmplx}(\mathtt{r}(a, \Sigma)) + \mathbf{cmplx}(\mathtt{r}(\sigma_2, \Sigma))]\,(2^{(k-1)} - 1),$$

where $\mathrm{op}(\sigma_2) = k - 1$ and $\mathbf{cmplx}(\cdot)$ denotes the complexity of computation $\cdot$. Barring terms of lower order and bringing the summation to closed form we obtain for the overall complexity of $\mathtt{r}(\sigma, \Sigma)$:

$$O(nm^2 2^{\frac{m^2}{2}}) \tag{68}$$

With respect to this complexity, the additional complexity introduced by the canonicalisation step to be performed beforehand (66) can be neglected.

5.2.5. *Incremental computation of* $\overline{\Sigma}$

The framework is the same as in the complexity analysis of the basic knowledge representation; the difference is that, from (67) and reduction rule (52), the application of each reduction rule may now involve up to $2^{(m_i-1)} - 1$ invocations of $\mathtt{r}(\cdot, \cdot)$ instead of two, so the overall complexity of reduction step i is:

$$n_i[2^{(m_i-1)} - 1]O\left(n_i {m_i}^2 2^{\frac{m_i^2}{2}}\right) = O\left(n_i^2 m_i^2 2^{\frac{m_i^2}{2}}\right)$$

provided we neglect the lower-order term in the exponent of 2. The overall complexity is therefore:

$$\sum_{i=0}^{m_0} O\left(n_i^2 m_i^2 2^{\frac{m_i^2}{2}}\right) = \sum_{i=0}^{m_0} O\left((2^i n_0)^2 (m_0 - i)^2 2^{\frac{(m_0-i)^2}{2}}\right)$$

that is, barring lower order terms in the summation:

$$O(n_0^2 2^{2m_0^2}) \tag{69}$$

With respect to the original complexity for a free-term algebra, we have now $2m_0^2$ instead of $2m_0$ at the exponent.

5.3. Further refinements

The most important shortcoming of the basic computational complexity analysis carried out in Sects. 5.1 and 5.2 is that it takes only a single reduction into account, that is, the incremental computation of the updated $\overline{\Sigma}$ after the addition of a single term ρ.

This approach neglects the effects that each reduction can possibly have on the complexity of subsequent ones. For example, if a given reduction splits some compound terms into atoms and thereby removes the formers from the new $\overline{\Sigma}$, these very same terms will no longer have to be considered in any reduction that follows, thus intuitively lowering its complexity.

The model of a sequence of reductions presented in this section tries to solve these issues to some extent, and provide a more realistic asymptotic complexity figure. The computation will be carried out with respect to the basic knowledge representation strategy of Sect. 3; however, the same model is still true under the extension presented in Sect. 4.

5.3.1. *Multiple reduction model*

Let us make, for the sake of the following discussion, some further definitions and simplifying assumptions:

m is the maximum term length ever occurring in the intruder's knowledge. As a consequence, $m - 1$ is the maximum number of operators in non-atomic terms.

$\overline{\Sigma}_i$ is the minimised intruder's knowledge at the beginning of the ith reduction. In order to simplify the analysis, the number of operators in each non-atomic term $\sigma \in \overline{\Sigma}_i$ is approximated with the worst-case value $m - 1$, that is, $\mathrm{op}(\sigma) = m - 1$.

x_i is the number of non-atomic terms in $\overline{\Sigma}_i$.

y_i is the number of atoms in $\overline{\Sigma}_i$.

α_i represents the fraction of non-atomic terms in $\overline{\Sigma}_i$ that is split into atoms by the ith reduction. By definition, $0 \leq \alpha_i \leq 1, \quad \forall\, i$.

β_i represents the fraction of the atoms produced by the ith reduction that are "fresh", that is, were not already known to the intruder; these atoms will become new elements of $\overline{\Sigma}_{i+1}$. The fraction of fresh atoms added in the ith reduction is then $\alpha_i\beta_i$. By definition, $0 \leq \beta_i \leq 1, \forall\, i$.

ρ_i represents the term added to $\overline{\Sigma}_i$ to trigger the ith reduction. It is assumed, as a worst-case hypothesis, that all ρ_i are non-atomic, that $\mathrm{op}(\rho_i) = m - 1$, and that they are always "fresh", that is, they were not already known to the intruder, even if some of their constituents possibly were.

It should be noted that none of the simplifications above affects the behavior of the algorithm that computes $\overline{\Sigma}$; they are merely worst-case assumptions introduced to simplify its complexity analysis.
Under the set of definitions and assumptions discussed above, we can write the following recurrence relations:

$$\begin{cases} x_{i+1} = (x_i + 1)(1 - \alpha_i) \\ y_{i+1} = y_i + m(x_i + 1)\alpha_i\beta_i \end{cases} \qquad i \geq 1 \tag{70}$$

with initial values x_1 and y_1. The relations above hold because the ith reduction starts, after the addition of ρ_i, with $x_i + 1$ non-atomic terms in $\overline{\Sigma}_i$. By definition of α_i, the reduction process leaves $(x_i + 1)(1 - \alpha_i)$ of them untouched, whilst splitting $(x_i + 1)\alpha_i$ of them into atoms, thus producing m atoms for each non-atomic term. Among the atoms produced, by definition of β_i, only $m(x_i + 1)\alpha_i\beta_i$ are actually fresh, and increase the number of atoms in $\overline{\Sigma}_{i+1}$, denoted by y_{i+1}.

5.3.2. Asymptotic convergence

If $\alpha_\mathrm{m} = \min_i(\alpha_i)$, from the first recurrence in (70) we can write:

$$x_{i+1} = (x_i + 1)(1 - \alpha_i) \leq (x_i + 1)(1 - \alpha_\mathrm{m}) \tag{71}$$

and then, solving the recurrence relations under the additional restriction that $0 < \alpha_\mathrm{m} \leq 1$, as shown in the appendix, it results that:

$$\lim_{i \to +\infty} x_i \leq \frac{1 - \alpha_\mathrm{m}}{\alpha_\mathrm{m}} \tag{72}$$

iff the limit on the left-hand side of the equation exists, thus ensuring that both the elements of the succession and its limit have a finite upper bound; it can also be shown that the same result holds when $\alpha_i = 0$ for a finite number of choices of i.
In addition, it can easily be shown that when the values of α_i are chosen according to a uniform or normal probability distribution, a reasonable approximation of the asymptotic limit of the succession x_i, when it exists, is:

$$\frac{1 - \overline{\alpha}}{\overline{\alpha}}, \tag{73}$$

where $\overline{\alpha}$ is the distribution's mean.
Let us now consider the succession y_i and introduce the additional assumption:

$$\beta_i = 1 - \frac{y_i}{N}, \tag{74}$$

where N is finite and represents the total number of atoms ever known in the system. Basically, this corresponds to assume that the frequency of collision is proportional to the fraction y_i/N of atoms already known to the intruder, and that when all existing atoms are known, no further atoms can add up into the intruder's knowledge. Assuming N to be a finite quantity is reasonable, because the non-decidability of the problem with an infinite N has already been proved otherwise.
Then, we can solve the recurrence as described in the appendix and obtain:

$$\lim_{i \to +\infty} y_i \leq N, \tag{75}$$

Under the initial, obviously true, condition $y_1 \leq N$, and assuming $N \geq 1$, it can easily be proved that the succession y_i is monotonic, non-decreasing and its upper limit is N.
From Eqs. (72) and (75), and assuming that both successions have a finite limit, denoted by x_{lim} and y_{lim}, respectively, we can write the following asymptotic relations:

$$\begin{cases} x_{\text{lim}} \leq \frac{1-\alpha_{\text{m}}}{\alpha_{\text{m}}} \\ y_{\text{lim}} \leq N \end{cases} \tag{76}$$

At equilibrium, $\overline{\Sigma}_i$ has up to $(1-\alpha_{\text{m}})/\alpha_{\text{m}}$ compound terms, and up to N atoms. Then, on each subsequent reduction, substituting (76) into (70) and keeping in mind the definition of α_{m}, up to

$$\left(\frac{1-\alpha_{\text{m}}}{\alpha_{\text{m}}} + 1\right)\alpha_{\text{m}} = 1 \tag{77}$$

terms are split into atoms, but none of these is fresh.

5.3.3. Refined complexity analysis

From the point of view of the computational complexity, starting from the asymptotic behaviour analysis carried out in Sect. 5.3.2, combined with the discussion of Sect. 5.1 about the complexity of the basic reduction algorithms, we have:

- The worst-case complexity of the lookup of a term σ in Σ (Sect. 5.1.1) is: $O(m(x_{\text{lim}}+1))+O(N) = O(mx_{\text{lim}}+N)$.
- The worst-case complexity of $\mathtt{r}(\sigma, \Sigma)$ (Sect. 5.1.2) is: $mO(mx_{\text{lim}} + N) = O(m^2 x_{\text{lim}} + mN)$.
- The splitting into atoms of a single compound term with $m-1$ operators entails up to m applications of a reduction rule. In turn, neglecting the complexity of the lookup in Σ with respect to the $\mathtt{r}(\cdot,\cdot)$ operator, this implies up to $2m$ applications of the $\mathtt{r}(\cdot,\cdot)$ operator. Thus, the overall complexity is:

$$2mO\left(m^2 x_{\text{lim}} + mN\right) = O\left(m^3 x_{\text{lim}} + m^2 N\right) \tag{78}$$

  that is polynomial in m.

It can easily be shown that, in the same framework, limiting the reduction process to natural deductions only, the complexity is:

$$2mO\left(m^2 x_{\text{lim}} + mN\right) = O\left(m^2 x_{\text{lim}} + mN\right) \tag{79}$$

that, again, is polynomial in m, albeit with a lesser exponent.
Comparing (78) and (79) it can be seen that, asymptotically, the complexity of our reduction method is still greater than methods based on natural deductions, but the disadvantage is considerably lesser than it would seem to be when considering a single reduction alone, as was done in Sect. 5.1. Similar results can be obtained for commutative and associative operators.

5.4. Experimental results

5.4.1. Comparison against a reduction simulator

In order to compare the results about asymptotic behaviour derived in Sect. 5.3.2 with the actual evolution of the reduction model presented in Sect. 5.3.1, a simulator of the reduction model was used to produce two sets of simulation results. In both sets, the value assigned to parameters α_i, N, and m is derived from, and representative of, real-world protocols, as will be discussed in detail in the next section.
In the first set, all α_i are assumed to be constant. Figure 3 plots the values of x_i and y_i for 30 reductions obtained from the simulator, and compares them against their asymptotic upper bounds.
In the second set, the values α_i are chosen at random according to two different probability distribution functions. Figure 4 compares the simulator's values for x_i and y_i against both the asymptotic limit and the upper bound of x_i, and the upper bound of y_i, when the α_i are uniformly distributed in a given range. The asymptotic limit of x_i has been computed as discussed in Sect. 5.3.2.

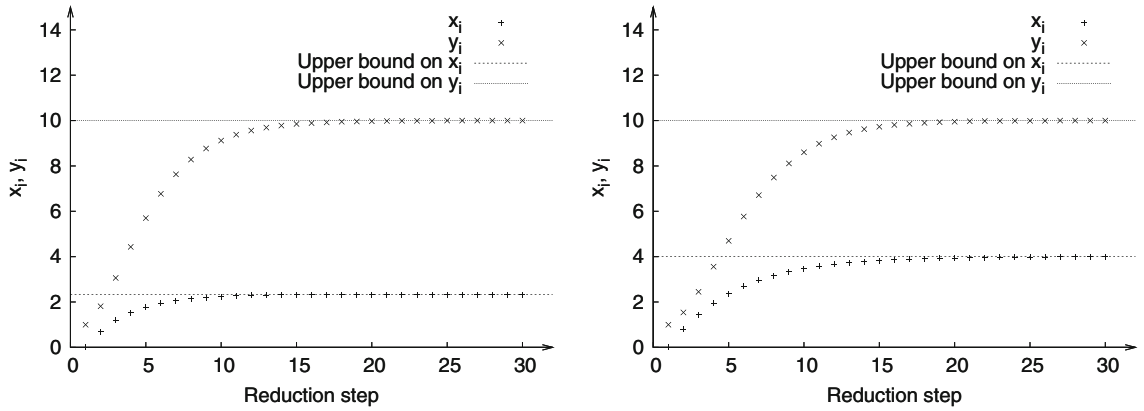

Fig. 3. Simulation results with constant $\alpha_i = 0.3$ to the *left*, and $\alpha_i = 0.2$ to the *right*; in both cases, $N = 10$ and $m = 3$

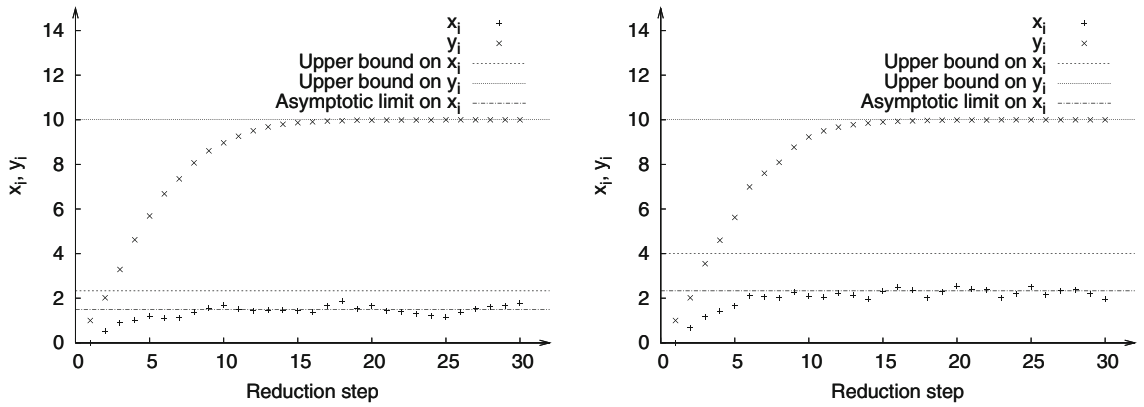

Fig. 4. Simulation results with uniformly distributed $\alpha_i \in [0.3, 0.5]$ (i.e. $\overline{\alpha} = 0.4$) to the *left*, and $\alpha_i \in [0.2, 0.4]$ (i.e. $\overline{\alpha} = 0.3$) to the *right*; in both cases, $N = 10$ and $m = 3$

Figure 5 makes the same comparison, but with normally distributed α_i. In this case, the upper bound on the x_i has been computed assuming $\alpha_{\mathrm{m}} = \overline{\alpha} - \sigma_\alpha$.

All plots show a good correspondence between the simulated and the asymptotic behaviours. All simulations start from the initial conditions $x_1 = 0$ and $y_1 = 1$, thus assuming that, at the very beginning, the intruder's knowledge includes a single atom, e.g. a public communications channel, and no compound terms.

5.4.2. Comparison with the actual minimisation process

To better assess the conformance of the simulator to the behaviour of the intruder's knowledge minimisation process in real protocol analysis, simulator's results were compared with some test runs of a prototype, automatic model-checking tool [DSV03], with source code instrumentation added to dump the contents of the intruder's knowledge after each reduction, thus enabling us to gather statistics about its contents. Since the tool only works on the basic knowledge representation described in Sect. 3 and does not support the extensions presented in Sect. 4 we did not use commutative and associative operators in the test runs.

Figure 6 compares the solution for x_i, i.e. the number of non-atomic terms in the intruder's knowledge, computed by the simulator and measured with the tool's instrumentation for two parallel sessions of the well-known WMF protocol that appears in [AG99] and is recalled in Fig. 7. The simulator's parameters have been determined by

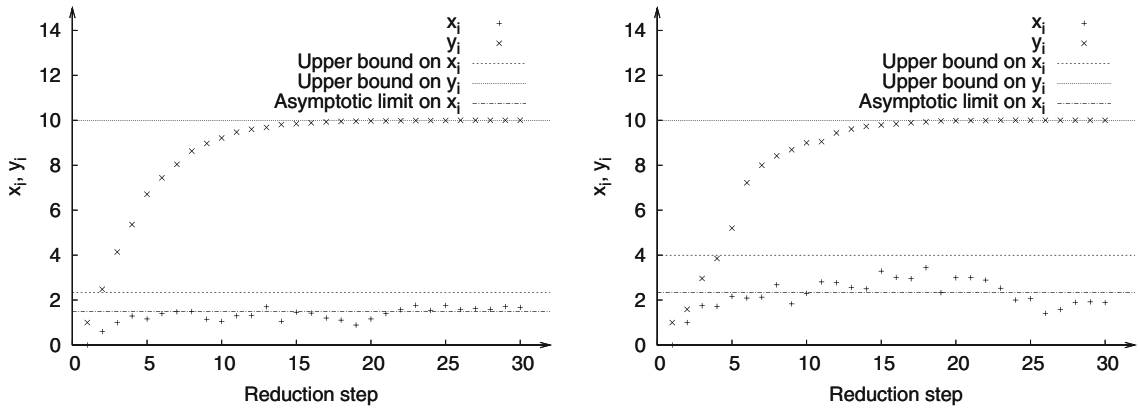

Fig. 5. Simulation results with normally distributed α_i, with $\overline{\alpha} = 0.4$ and $\sigma_\alpha = 0.1$ to the *left*, and $\overline{\alpha} = 0.3$ and $\sigma_\alpha = 0.1$ to the *right*; in both cases, $N = 10$ and $m = 3$

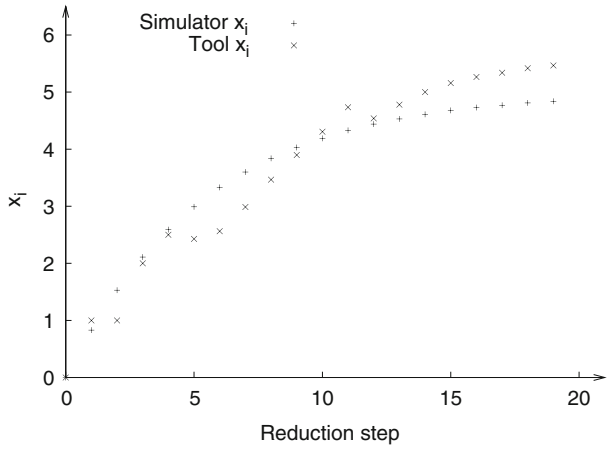

Fig. 6. Comparison between simulation and tool results for the WMF protocol. Simulator's parameters are $N = 12$, $m = 2$, and a constant $\alpha_i = 1/6$

$A(M, K_{as}, C_{as}, C_{ab})$	$\triangleq$	$(\nu K_{ab})(\overline{C_{as}}\langle\{K_{ab}\}_{K_{as}}\rangle.\ \overline{C_{ab}}\langle\{M\}_{K_{ab}}\rangle.\ 0)$
$S(K_{as}, K_{sb}, C_{as}, C_{sb})$	$\triangleq$	$C_{as}(x_1).\ case\ x_1\ of\ \{x_2\}_{K_{as}}\ in$ $\overline{C_{sb}}\langle\{x_2\}_{K_{sb}}\rangle.\ 0$
$B(K_{sb}, C_{sb}, C_{ab}, C_F)$	$\triangleq$	$C_{sb}(y_1).\ case\ y_1\ of\ \{y_2\}_{K_{sb}}\ in$ $C_{ab}(y_3).\ case\ y_3\ of\ \{y_4\}_{y_2}\ in$ $\overline{C_F}\langle y_4\rangle.\ 0$
$\mathrm{Sys}(M, K_{as}, K_{sb},$ $C_{as}, C_{sb}, C_{ab}, C_F)$	$\triangleq$	$A(M, K_{as}, C_{as}, C_{ab})\ \mid$ $S(K_{as}, K_{sb}, C_{as}, C_{sb})\ \mid$ $B(K_{sb}, C_{sb}, C_{ab}, C_F)$
$\mathrm{Inst}()$	$\triangleq$	$(\nu K_{as})(\nu K_{sb})($ $\mathrm{Sys}(M_1, K_{as}, K_{sb}, C_{as1}, C_{sb1}, C_{ab1}, C_{F1})\mid$ $\mathrm{Sys}(M_2, K_{as}, K_{sb}, C_{as2}, C_{sb2}, C_{ab2}, C_{F2}))$

Fig. 7. Spi specification of the WMF protocol, two sessions

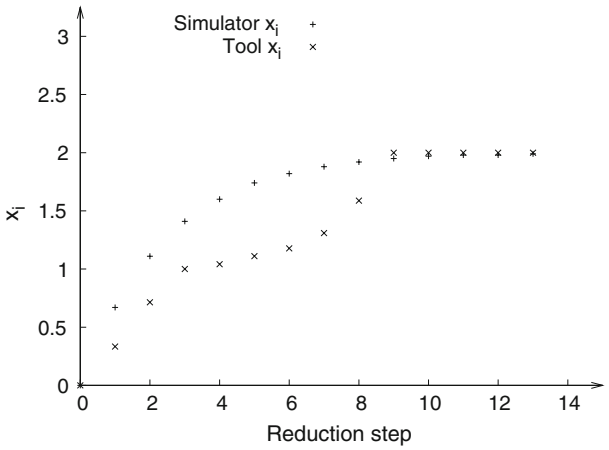

Fig. 8. Comparison between simulation and tool results for the protocol of Fig. 9. Simulator's parameters are $N = 3$, $m = 2$, and a constant $\alpha_i = 1/3$

$P_A(M)$	$\triangleq$	$\overline{c}\langle\{M\}_k\rangle.\ c(x).\ [x\ is\ H(M)]F(M)$
P_B	$\triangleq$	$c(y_1).\ case\ y_1\ of\ \{y_2\}_k\ in\ \overline{c}\langle H(y_2)\rangle.\ 0$
$\mathrm{Inst}(M)$	$\triangleq$	$(\nu k)(P_A(M) \mid P_B \mid P_A(M) \mid P_B)$

Fig. 9. Spi specification of two parallel sessions of the protocol of Fig. 1

fitting the model to the spi specification of the protocol itself, for parameters N and m, and estimating α_i from one of the protocol traces with maximum length. In particular:

- Parameter N has been determined by counting the total number of atoms in the protocol specification ($N = 12$).
- Parameter m has been determined by inspecting the structure of all messages exchanged in the protocol, and counting the maximum number of operators they contain ($m = 2$).
- Parameter α_i has been estimated from one of the traces of the protocol with maximum length, by computing the fraction of output terms that went into the intruder's knowledge without being split by the reduction process.

Figure 8 repeats the same comparison, but for two parallel sessions of the protocol already presented in Fig. 1 and whose spi specification can be found in Fig. 9. It can be noticed that, even when the number of reductions in the actual protocol is very small, like in this case, thus placing the simulator in its worst-case, the simulator's result are still near the measured ones, and the error is below unity.

6. Concluding remarks and related work

Most finite [CJM98, CJM00, Low96] and infinite-state [AL00, BDNP02, Bor01] protocol analysis methods restrict encryption operators to atomic keys only. For example, in [CJM98] and [Pau98], this restriction comes from the adoption of the closure of Σ under $\mathcal{E}$ rules as a representation of the intruder's knowledge.
Other approaches based on theorem proving do not pose this restriction but the trade-off in the general case is between incompleteness and possible non-termination of the analysis, as pointed out by [MS01]. For example, the approach taken by Proverif [Bla01] admits non-atomic keys, but may give false positives.
It is worth noting that support for constructed, non-atomic keys is becoming increasingly important to be able to analyse real-world protocols, since it is common for such protocols to build a symmetric key from shared secrets and other data exchanged between parties during a run of the protocol itself.

Other papers, such as [FA01, Hui99, BMV05], relax this restriction but neither explicitly introduce the notion of $\overline{\Sigma}$, that is, the minimised, canonical representation of the intruder's knowledge, nor analyse and exploit its properties. As a consequence their intruder's knowledge representation is less than optimal.
The free term algebra of [MS01], too, allows any term to be used as an encryption key for both public-key and symmetric key encryption, but the attacker's knowledge representation is not minimised and some other slight restrictions are in effect, such as for example the assumption that private keys are never leaked. This assumption seems quite reasonable, but cannot easily be guaranteed by hand when dealing with complex protocols; so, we believe that such property is best checked with the aid of a formal, automated method.
On the other hand, [RT01] proves that the maximum length of a derivation of a term σ from an arbitrary set of terms Σ is bounded by the size of the Directed Acyclic Graph (DAG) representation of σ and Σ, thus implying the existence of a polynomial algorithm to carry out a single reduction step. However, the proof of this result heavily relies on the assumption that asymmetric encryption keys are restricted to be atomic, i.e. the validity of this result has not been proved in the general case of non-atomic keys. The same restriction affects [CKRT03] too, which extends [RT01] to also cope with the exclusive or operator.
By contrast, our approach does not pose any restriction on the internal structure and construction operators of symmetric and asymmetric encryption keys, and supports the full term language of the spi calculus itself. This result comes at the expense of a greater computational complexity for a single reduction step; however, the overall complexity of our method for a whole reduction has been shown to be asymptotically polynomial anyway.
The additional expressive power and flexibility of our method more than outweighs this disadvantage in the communications protocols typically found in practice, as it has been shown by a refined model for complexity analysis. This result was relieved by the comparison of the refined model against the actual behaviour of a model-checking tool, showing good results.
Several recent papers, e.g. [CKRT03, CLS03, MN02, MS03], present methods to overcome the limitations of a free term algebra framework and support commutative, associative and self-cancelling operators; however, they have not been applied to model checking and do not keep the intruder's knowledge in a minimised form.
Last, it should be noted that, even if we adopted the term syntax of spi calculus in this paper, our method is easily amenable to work with other term representation languages with similar sets of term composition operators.
As a future work, it would be interesting and useful to further extend this method to handle self-cancelling operators and, more in general, cancellation. However, at present, this possibility has been little investigated.

Appendix A: Refined computation analysis details

This section works out the mathematical details of the refined computational analysis described in Sect. 5.3, under the same assumptions already outlined in Sect. 5.3.1. If $\alpha_{\mathrm{m}} = \min_i(\alpha_i)$, from the first recurrence in (70) we can write:

$$x_{i+1} = (x_i + 1)(1 - \alpha_i) \leq (x_i + 1)(1 - \alpha_{\mathrm{m}}) \tag{80}$$

and then, solving the recurrence relation:

$$x_{i+1} \leq (1 - \alpha_{\mathrm{m}})^i x_1 + \sum_{j=1}^{i} (1 - \alpha_{\mathrm{m}})^j \tag{81}$$

Under the additional restriction that $0 < \alpha_{\mathrm{m}} \leq 1$:

$$\lim_{i \to +\infty} (1 - \alpha_{\mathrm{m}})^i x_1 = 0, \quad \forall\, x_1 \tag{82}$$

and

$$\sum_{j=1}^{+\infty} (1 - \alpha_{\mathrm{m}})^j = \frac{1 - \alpha_{\mathrm{m}}}{\alpha_{\mathrm{m}}} \tag{83}$$

So, substituting (82) and (83) into (81), we obtain (72), iff the limit on the left-hand side of the equation exists. We have shown that when the succession x_i has a limit for $i \to +\infty$, then both the elements of the succession and its limit have a finite upper bound, provided that $0 < \alpha_{\mathrm{m}} \leq 1$; it can also be shown that the same result holds when $\alpha_i = 0$ for a finite number of choices of i.

Let us now consider the succession y_i, with the additional assumption (74), where N is finite and represents the total number of atoms ever known in the system.
If $\alpha_{\mathbf{M}} = \max_i(\alpha_i)$, substituting (74) into the second recurrence in (70) we can write:

$$y_{i+1} = y_i + m(x_i + 1)\alpha_i\beta_i \leq y_i + m(x_i + 1)\alpha_{\mathbf{M}}\left(1 - \frac{y_i}{N}\right) \tag{84}$$

Since the succession x_i has a finite upper bound (72), then we can choose a suitable M to write:

$$m(x_i + 1)\alpha_{\mathbf{M}} \leq M, \quad \forall\, i \tag{85}$$

Substituting (85) into (84) and solving the recurrence, we obtain:

$$y_{i+1} \leq y_i + M\left(1 - \frac{y_i}{N}\right) = (y_1 - N)\left(1 - \frac{M}{N}\right)^i + N \tag{86}$$

and, therefore, when $|1 - M/N| < 1$, that is, being $M > 0$ by definition, when $M < 2N$, (75) holds since, in this case:

$$\lim_{i\to+\infty}\left(1 - \frac{M}{N}\right)^i = 0. \tag{87}$$

Under the initial, obviously true condition $y_1 \leq N$, and assuming $N \geq 1$, it can easily be proved that the succession y_i is monotonic, non-decreasing and its upper limit is N.

Appendix B: Summary of Theorems

Theorem C.1 *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, with closure $\widehat{\Sigma}$, and $\overline{\Sigma}$ defined by (17)–(25), given $\overline{\sigma} \in \widehat{\Sigma}$, then*

$\overline{\sigma}$ satisfies a predicate among (17)–(25) $\Leftrightarrow$ $\overline{\sigma}$ cannot be built by a rule among (2)–(8) from $\widehat{\Sigma}$

i.e. each $\overline{\sigma} \in \overline{\Sigma}$ is computed from $\widehat{\Sigma}$ by means of closure rules (1), and (9)–(14) only.

Corollary C.0.1 *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, with closure $\widehat{\Sigma}$, $\overline{\Sigma}$ defined by (17)–(25), and given $\overline{\sigma} \in \overline{\Sigma}$ then, $\overline{\sigma} \in \Sigma \vee \overline{\sigma}$ is a sub-term of some $\sigma \in \Sigma$.*

Uses Theorem C.1.

Theorem 3.1 (Finiteness) *For each finite set of terms $\Sigma \subseteq \mathcal{T}$, $\overline{\Sigma}$ is finite and unique.*

Uses Corollary C.0.1.

Theorem 3.2 (Minimality) *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, and $\sigma \in \overline{\Sigma}$. Then $(\widehat{\overline{\Sigma}\setminus\{\sigma\}})^I \subset \widehat{\overline{\Sigma}}$.*

Uses Theorem C.1.

Theorem C.2 *Given a finite set of terms $\Sigma \subseteq \mathcal{T}$, with closure $\widehat{\Sigma}$, a finite message $\sigma \in \mathcal{T}$, and a finite subset of $\widehat{\Sigma}$ S, then*

$$\mathtt{r}(\sigma, S) \;\Rightarrow\; \sigma \in \widehat{\Sigma}$$

and $\mathtt{r}(\sigma, S)$ can be computed in a finite number of steps.

Theorem C.3 *Given a finite set of finite messages $\Sigma \subseteq \mathcal{T}$ with closure $\widehat{\Sigma}$,*

$$\sigma \in \widehat{\Sigma\setminus\{\sigma\}} \;\Rightarrow\; \widehat{\Sigma\setminus\{\sigma\}} \equiv \widehat{\Sigma}$$

Proposition 3.1 *Given a finite set of finite messages $\Sigma \subseteq \mathcal{T}$ with closure $\widehat{\Sigma}$, and a reduction rule $R = \langle \Sigma_I,\ C,\ \Sigma_O\rangle$, then*

$$\Sigma \xrightarrow{R} \Sigma' \;\Rightarrow\; \widehat{\Sigma} \equiv \widehat{\Sigma'}$$

Uses Theorems C.2 and C.3.

Theorem C.4 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*, and a finite message* $\sigma \in \mathcal{T}$*, then*

$$\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I$$

and $\mathtt{r}(\sigma, \Sigma)$ *is computed in a finite number of steps.*

Theorem C.5 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*,* $\overline{\Sigma}$ *defined by* (17)–(25)*, and a finite message* $\sigma \in \mathcal{T}$*, then*

$$\mathtt{r}(\sigma\,,\ \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}$$

and the computation of $\mathtt{r}(\sigma\,,\ \overline{\Sigma})$ *takes a finite number of steps.*

Theorem C.6 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*, and* $\overline{\Sigma}$ *defined by* (17)–(25) $\Rightarrow$ *rules* (26)–(36) *do not hold in* $\overline{\Sigma}$*.*

Uses Theorems C.3, C.4 and C.5.

Theorem C.7 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*,*

$$\textit{if rules (26)–(36) do not hold in } \Sigma \Rightarrow \widehat{\Sigma} \setminus \widehat{\Sigma}^I = \emptyset$$

i.e. from such a Σ *we can build* $\widehat{\Sigma}$ *by using* $\mathcal{I}$ *rules only* ($\widehat{\Sigma} \equiv \widehat{\Sigma}^I$)*.*

Uses Theorem C.4.

Theorem C.8 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*, and* $\overline{\Sigma}$ *defined by* (17)–(25)*,*

$$\textit{if rules (26)–(36) do not hold in } \Sigma \Rightarrow \Sigma \equiv \overline{\Sigma}$$

Uses Theorems C.1, C.3 and C.7.

Appendix C: Proofs of Theorems

Theorem C.1 *Let* $\Sigma \subseteq \mathcal{T}$ *be a finite set of terms, with closure* $\widehat{\Sigma}$*, and* $\overline{\Sigma}$ *defined by* (17)–(25)*, given* $\overline{\sigma} \in \widehat{\Sigma}$*, then*

$\overline{\sigma}$ *satisfies a predicate among* (17)–(25) $\Leftrightarrow$ $\overline{\sigma}$ *cannot be built by a rule among* (2)–(8) *from* $\widehat{\Sigma}$*.*

Proof. We need to prove that each $\overline{\sigma}$ cannot be built by combining simpler terms, i.e. when rules (17)–(25) hold, the preconditions of closure rules (2)–(8) are always invalid (since $\overline{\Sigma} \subseteq \widehat{\Sigma}$, then $\overline{\sigma} \in \widehat{\Sigma}$ and closure rules apply) and vice-versa. We address each syntactic form of $\overline{\sigma}$.

$\Rightarrow$

- if $\overline{\sigma} = a$, rule (17) holds and no rule among (2)–(8) can build a name belonging to $\widehat{\Sigma}$;
- rule (18) states that $\overline{\sigma}$ cannot be a *successor*, then it cannot have been built by means of closure rule (2);
- rule (19) states that $\overline{\sigma}$ cannot be a *pair*, then it cannot have been built by means of closure rule (3);
- if $\overline{\sigma} = \{\sigma_1\}_{\sigma_2}$, rule (20) states that $\sigma_2 \notin \widehat{\Sigma}$, then the preconditions of closure rule (4) are always invalid;
- if $\overline{\sigma} = \mathrm{H}(\sigma)$, rule (21) states that $\sigma \notin \widehat{\Sigma}$, then the preconditions of closure rule (5) are always invalid;
- if $\overline{\sigma} = \{[\sigma_1]\}_{\sigma_2^+}$, rule (22) states that $\sigma_2^+ \notin \widehat{\Sigma} \ \vee\ \sigma_1 \notin \widehat{\Sigma}$, then the preconditions of closure rule (6) are always invalid;
- if $\overline{\sigma} = [\{\sigma_1\}]_{\sigma_2^-}$, rule (23) states that $\sigma_2^- \notin \widehat{\Sigma} \ \vee\ \sigma_1 \notin \widehat{\Sigma}$, then the preconditions of closure rule (7) are always invalid;
- if $\overline{\sigma} = \sigma^+$, rule (24) states that $\sigma \notin \widehat{\Sigma}$, then the preconditions of closure rule (8) are always invalid;
- if $\overline{\sigma} = \sigma^-$, rule (25) states that $\sigma \notin \widehat{\Sigma}$, then the preconditions of closure rule (8) are always invalid.

$\Leftarrow$

- if $\overline{\sigma} = a$, by (17) ($a \in \overline{\Sigma} \Leftrightarrow a \in \widehat{\Sigma}$) we have the thesis, without the need of assuming that rules (2)–(8) do not hold;

- if $\overline{\sigma} = \mathrm{suc}(\sigma)$, only closure rules (2) and (9) deal with $\mathrm{suc}(\cdot)$, moreover closure rule (2) does not hold, i.e. its premises are false by hypothesis. We claim that this leads to $\mathrm{suc}(\sigma) \notin \widehat{\Sigma}$, and, since $\overline{\Sigma} \subseteq \widehat{\Sigma}$, then (18) holds too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma \in \widehat{\Sigma} \;\Rightarrow\; \mathrm{suc}(\sigma) \in \widehat{\Sigma} \\ \wedge \\ \mathrm{suc}(\sigma) \in \widehat{\Sigma} \;\Rightarrow\; \sigma \in \widehat{\Sigma} \\ \wedge \\ \sigma \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \mathrm{suc}(\sigma) \notin \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = (\sigma_1, \sigma_2)$, only closure rules (3) and (10) deal with $(\cdot, \cdot)$, moreover closure rule (3) does not hold, i.e. its premises are false by hypothesis. We claim that this implies $(\sigma_1, \sigma_2) \notin \widehat{\Sigma}$, and, since $\overline{\Sigma} \subseteq \widehat{\Sigma}$, also (19) holds too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; (\sigma_1, \sigma_2) \in \widehat{\Sigma} \\ \wedge \\ (\sigma_1, \sigma_2) \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \notin \widehat{\Sigma} \vee \sigma_2 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow (\sigma_1, \sigma_2) \notin \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = \{\sigma_1\}_{\sigma_2}$, only closure rules (4) and (11) deal with $\{\cdot\}_\cdot$, moreover closure rule (4) does not hold, i.e. its premises are false, and $\{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$. We claim that this implies $\sigma_2 \notin \widehat{\Sigma} \wedge \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$, allowing (20) to hold too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \;\Rightarrow\; \sigma_1 \in \widehat{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \notin \widehat{\Sigma} \vee \sigma_2 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \sigma_2 \notin \widehat{\Sigma} \wedge \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = \mathrm{H}(\sigma)$, we have that the premises of closure rule (5) are invalid by hypothesis, then rule (21) holds, and the thesis comes from the truth of:

$$\left.\begin{array}{c} \sigma \in \widehat{\Sigma} \;\Rightarrow\; \mathrm{H}(\sigma) \in \widehat{\Sigma} \\ \wedge \\ \sigma \notin \widehat{\Sigma} \\ \wedge \\ \mathrm{H}(\sigma) \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \sigma \notin \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = \{[\sigma_1]\}_{\sigma_2^+}$, we have closure rule (6) whose premises are invalid by hypothesis, and $\{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$. We claim that this implies $\left(\sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^+ \notin \widehat{\Sigma}\right) \wedge \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$, allowing (22) to hold too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \;\Rightarrow\; \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \\ \wedge \\ \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^+ \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^+ \notin \widehat{\Sigma}\right) \wedge \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.

- if $\overline{\sigma} = [\{\sigma_1\}]_{\sigma_2^-}$, we have closure rule (7) whose premises are invalid by hypothesis, and $[\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma}$. We claim that this implies $\left(\sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^- \notin \widehat{\Sigma}\right) \wedge [\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma}$, allowing (23) to hold too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^- \in \widehat{\Sigma} \;\Rightarrow\; [\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma} \\ \wedge \\ [\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^- \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \notin \widehat{\Sigma} \vee \sigma_2^- \notin \widehat{\Sigma}\right) \wedge [\{\sigma_1\}]_{\sigma_2^-} \in \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = \sigma^+$, we have closure rule (8) whose premises are invalid by hypothesis, and $\sigma^+ \in \widehat{\Sigma}$. We claim that this implies $\sigma \notin \widehat{\Sigma} \wedge \sigma^+ \in \widehat{\Sigma}$, allowing (24) to hold too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma \in \widehat{\Sigma} \;\Rightarrow\; \sigma^+ \in \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma} \\ \wedge \\ \sigma^+ \in \widehat{\Sigma} \\ \wedge \\ \sigma \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \sigma \notin \widehat{\Sigma} \wedge \sigma^+ \in \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions.
- if $\overline{\sigma} = \sigma^-$, we have closure rule (8) whose premises are invalid by hypothesis, and $\sigma^- \in \widehat{\Sigma}$. We claim that this implies $\sigma \notin \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma}$, allowing (25) to hold too, leading to the thesis. The truth of the claim comes from the truth of:

$$\left.\begin{array}{c} \sigma \in \widehat{\Sigma} \;\Rightarrow\; \sigma^+ \in \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma} \\ \wedge \\ \sigma^- \in \widehat{\Sigma} \\ \wedge \\ \sigma \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \sigma \notin \widehat{\Sigma} \wedge \sigma^- \in \widehat{\Sigma}$$

  which can be easily proved by using boolean expressions. □

Corollary C.0.1 *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, with closure $\widehat{\Sigma}$, $\overline{\Sigma}$ defined by (17)–(25), and $\overline{\sigma} \in \overline{\Sigma}$ then, $\overline{\sigma} \in \Sigma \vee \overline{\sigma}$ is a sub-term of some $\sigma \in \Sigma$.*

Proof. From Theorem C.1, it comes that $\overline{\sigma}$ can be built from $\widehat{\Sigma}$ by means of closure rules (1), and (9)–(14) only. Since $\Sigma \subseteq \widehat{\Sigma}$, the thesis can be deduced by inspection of closure rules (1) and (9)–(14). □

Theorem 3.1 (Finiteness) *For each finite set of terms $\Sigma \subseteq \mathcal{T}$, $\overline{\Sigma}$ is finite and unique.*

Proof. Here a set of terms is finite if it contains a finite number of finite terms. So let us divide the proof: first of all we prove that each term in $\overline{\Sigma}$ is finite, then that $\overline{\Sigma}$ contains a finite number of terms.

Each term in $\overline{\Sigma}$ is finite: Absurdly, let us assume that $\exists\ \overline{\sigma}_\infty \in \overline{\Sigma} \mid \overline{\sigma}_\infty$ is made by an infinite number of sub-terms. Corollary C.0.1 states that such $\overline{\sigma}_\infty \in \Sigma \vee \overline{\sigma}$ is a sub-term of some $\sigma \in \Sigma$, but this means that Σ contains at least an infinite term, and this violates the hypothesis of finiteness of Σ. □

The cardinality of $\overline{\Sigma}$ is finite: Absurdly, let us assume that the cardinality of $\overline{\Sigma}$ is infinite: then, by Corollary C.0.1, we have that Σ contains infinite terms, or some term with infinite sub-terms, but this violates the hypothesis of finiteness of Σ.

$\overline{\Sigma}$ is unique: Let us absurdly assume that $\exists$ two distinct $\overline{\Sigma}_1$ and $\overline{\Sigma}_2$ both satisfying rules (17)–(25): let $\overline{\sigma} \in \overline{\Sigma}_1$ and $\overline{\sigma} \notin \overline{\Sigma}_2$ (or vice-versa): $\overline{\sigma}$ cannot be a *pair* or a *successor* by rules (18) and (19), on the other hand, the remaining rules among (17) and (25) are necessary and sufficient conditions on $\widehat{\Sigma}$, then, given a $\widehat{\Sigma}$, just one $\overline{\Sigma}$ exists. □

Definition C.1 Let $\Sigma \subseteq \mathcal{T}$ be a set of terms, we define $\widehat{\Sigma}^I$ as the *closure* of Σ under rules (1) and (2)–(8) only.

Theorem 3.2 (Minimality) *Let $\Sigma \subseteq \mathcal{T}$ be a finite set of terms, and $\sigma \in \overline{\Sigma}$. Then $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I \subset \widehat{\overline{\Sigma}}$.*

Proof. Absurdly, let us assume that $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I \not\subset \widehat{\overline{\Sigma}}$.
In general, given two sets H, $K|H \not\subset K$, it means that $H = K \vee \exists\ h \in H|h \notin K$. In this case we know how that $\overline{\Sigma}\backslash\{\sigma\} \subset \overline{\Sigma}$, moreover the set of rules under which we close $\overline{\Sigma}\backslash\{\sigma\}$ is a subset of the set of rules under which we close $\overline{\Sigma}$. Then $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I$ cannot be larger than $\widehat{\overline{\Sigma}}$, and, in particular, $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I$ cannot include terms not included in $\widehat{\overline{\Sigma}}$ also, then we can reduce our absurd hypothesis $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I \not\subset \widehat{\overline{\Sigma}}$ to $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I = \widehat{\overline{\Sigma}}$.
Since $\sigma \in \overline{\Sigma}$, we have that $\sigma \in \widehat{\overline{\Sigma}}$ and, absurdly being $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I = \widehat{\overline{\Sigma}}$, it also holds that $\sigma \in (\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I$. Then we should be able to build σ from $\overline{\Sigma}\backslash\{\sigma\}$ by using rules (1) and (2)–(8) only.
We have that $\sigma \notin \overline{\Sigma}\backslash\{\sigma\}$ by definition, then we cannot use rule (1). On the other hand, being both $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I$ and $\widehat{\overline{\Sigma}}$ subsets of $\widehat{\Sigma}$, Theorem C.1 holds and states that premises of (2)–(8) are not satisfied by elements of $\overline{\Sigma}$, and, even more so, this holds for the subset $\overline{\Sigma}\backslash\{\sigma\}$, i.e. we are not able to compute σ from $\overline{\Sigma}\backslash\{\sigma\}$ by means of rules (1) and (2)–(8) only, and this comes from our initial, absurd hypothesis. □

The absurd hypothesis $(\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I = \widehat{\overline{\Sigma}} \Rightarrow \sigma \in (\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I$, but Theorem C.1 proves that this last statement is *false*, and this implies that our absurd hypothesis is *false* $(\sigma \in (\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I \Rightarrow (\widehat{\overline{\Sigma}\backslash\{\sigma\}})^I = \widehat{\overline{\Sigma}})$.

Theorem C.2 *Given a finite set of terms $\Sigma \subseteq \mathcal{T}$, with closure $\widehat{\Sigma}$, a finite message $\sigma \in \mathcal{T}$, and a finite subset of $\widehat{\Sigma}$ S, then*

$$\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{\Sigma}$$

and $\mathtt{r}(\sigma, S)$ can be computed in a finite number of steps.

Proof. The basic idea is to carry out the proof inductively: the theorem will be proved directly for the two disjoint[2] cases $\sigma \in S$, and $\sigma \in \mathcal{A}\backslash S$ (i.e. the base of the induction) and inductively the remaining cases: $\sigma \notin \mathcal{A} \wedge \sigma \notin S$ (i.e. $\sigma \notin \mathcal{A} \cup S$).
Base $(\sigma \in S \ \vee\ \sigma \in \mathcal{A}\backslash S)$

- $\sigma \in S$: we have that
  - $\sigma \in S$ as hypothesis;
  - $\sigma \in S \Rightarrow \mathtt{r}(\sigma, S)$ from the definition of `r` (`if` $\sigma \in$ `S then return TRUE`);
  - $\sigma \in S \Rightarrow \sigma \in \widehat{\Sigma}$ since $S \subseteq \widehat{\Sigma}$ as hypothesis.

  Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{\Sigma}$ holds, i.e.:

  $$\left.\begin{array}{l} \sigma \in S \\ \wedge \\ \sigma \in S \Rightarrow \mathtt{r}(\sigma, S) \\ \wedge \\ \sigma \in S \Rightarrow \sigma \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{\Sigma}\right)$$

  which can be easily proved by using boolean expressions.
- $\sigma \in \mathcal{A}\backslash S$: we have that
  - $\sigma \in \mathcal{A} \wedge \sigma \notin S$ as hypothesis;
  - $\sigma \in \mathcal{A} \wedge \sigma \notin S \Rightarrow \overline{\mathtt{r}(\sigma, S)}$ from the definition of `r` (`else` $(\sigma \in \mathcal{A}\backslash$`S)` `return FALSE`);
  - $\sigma \in S \Rightarrow \sigma \in \widehat{\Sigma}$ since $S \subseteq \widehat{\Sigma}$ as hypothesis.

[2] $S \cap (\mathcal{A}\backslash S) = \emptyset$.

Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{\Sigma}$ holds, i.e.:

$$\left.\begin{array}{l} \sigma \in \mathcal{A} \wedge \sigma \notin S \\ \wedge \\ \sigma \in \mathcal{A} \wedge \sigma \notin S \Rightarrow \overline{\mathtt{r}(\sigma, S)} \\ \wedge \\ \sigma \in S \Rightarrow \sigma \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions.

Induction ($\sigma \notin \mathcal{A} \cup S$)

- Let $\sigma = \mathrm{suc}(\sigma_1) \wedge \mathrm{suc}(\sigma_1) \notin S$: trivially $\mathtt{r}(\sigma_1, S) \Rightarrow \mathtt{r}(\mathrm{suc}(\sigma_1), S)$, moreover $\mathrm{suc}(\sigma_1) \notin S$, and this makes true also the backward implication[3]

$$\mathtt{r}(\sigma_1, S) \Leftarrow \mathtt{r}(suc(\sigma_1), S)$$

  Closure rule (2) states that $\sigma_1 \in \widehat{\Sigma} \Rightarrow suc(\sigma_1) \in \widehat{\Sigma}$.
  By inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$, we have the thesis:

$$\mathtt{r}(suc(\sigma_1), S) \Rightarrow \mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma} \Rightarrow suc(\sigma_1) \in \widehat{\Sigma}$$

- Let $\sigma = (\sigma_1, \sigma_2) \wedge (\sigma_1, \sigma_2) \notin S$: trivially $\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2, S) \Rightarrow \mathtt{r}((\sigma_1, \sigma_2), S)$, moreover $(\sigma_1, \sigma_2) \notin S$, and this makes true also the backward implication

$$\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2, S) \Leftarrow \mathtt{r}((\sigma_1, \sigma_2), S)$$

  Closure rule (3) states that $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}$ and, by inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2, S) \Rightarrow \sigma_2 \in \widehat{\Sigma}$, it holds that

$$\begin{array}{lcl} \mathtt{r}((\sigma_1, \sigma_2), S) & \Rightarrow & \mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2, S) \\ & \wedge & \\ \mathtt{r}(\sigma_1, S) & \Rightarrow & \sigma_1 \in \widehat{\Sigma} \\ & \wedge & \\ \mathtt{r}(\sigma_2, S) & \Rightarrow & \sigma_2 \in \widehat{\Sigma} \\ & \wedge & \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} & \Rightarrow & (\sigma_1, \sigma_2) \in \widehat{\Sigma} \end{array}$$

  which easily implies $\mathtt{r}((\sigma_1, \sigma_2), S) \Rightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}$.
- Let $\sigma = \{\sigma_1\}_{\sigma_2} \wedge \{\sigma_1\}_{\sigma_2} \notin S$: trivially $\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2, S) \Rightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, S)$, moreover $\{\sigma_1\}_{\sigma_2} \notin S$, and this makes true also the backward implication

$$\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2, S) \Leftarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, S)$$

  Closure rule (4) states that $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$ and, by inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2, S) \Rightarrow \sigma_2 \in \widehat{\Sigma}$, we have the same case of $\sigma = (\sigma_1, \sigma_2)$ (where $\{\sigma_1\}_{\sigma_2}$ replaces (σ_1, σ_2)), which easily implies $\mathtt{r}(\{\sigma_1\}_{\sigma_2}, S) \Rightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$.
- Let $\sigma = \mathrm{H}(\sigma_1) \wedge \mathrm{H}(\sigma_1) \notin S$: trivially $\mathtt{r}(\sigma_1, S) \Rightarrow \mathtt{r}(\mathrm{H}(\sigma_1), S)$, moreover $\mathrm{H}(\sigma_1) \notin S$, and this makes true also the backward implication

$$\mathtt{r}(\sigma_1, S) \Leftarrow \mathtt{r}(\mathrm{H}(\sigma_1), S)$$

  Closure rule (5) states that $\sigma_1 \in \widehat{\Sigma} \Rightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}$.
  By inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$, we have the thesis:

$$\mathtt{r}(\mathrm{H}(\sigma_1), S) \Rightarrow \mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma} \Rightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}$$

[3] we know that $\mathtt{r}(suc(\sigma_1), S)$ has given `TRUE`, and that the result has been computed by means of $\mathtt{r}(\sigma_1, S)$.

- Let $\sigma = \{[\sigma_1]\}_{\sigma_2^+} \wedge \{[\sigma_1]\}_{\sigma_2^+} \notin S$: trivially $\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2^+, S) \Rightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, S)$, moreover $\{[\sigma_1]\}_{\sigma_2^+} \notin S$, and this makes true also the backward implication

$$\mathtt{r}(\sigma_1, S) \wedge \mathtt{r}(\sigma_2^+, S) \Leftarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, S)$$

  Closure rule (6) states that $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Rightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$ and, by inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2^+, S) \Rightarrow \sigma_2^+ \in \widehat{\Sigma}$, we have the same case of $\sigma = (\sigma_1, \sigma_2)$ (where $\{[\sigma_1]\}_{\sigma_2^+}$ replaces (σ_1, σ_2)), which easily implies $\mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, S) \Rightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$.
- Let $\sigma = [\{\sigma_1\}]_{\sigma_2^-} \wedge [\{\sigma_1\}]_{\sigma_2^-} \notin S$ is handled exactly as the previous one [where closure rule (7) replaces (6)].
- Let $\sigma = \sigma_1^+ \wedge \sigma_1^+ \notin S$: trivially $\mathtt{r}(\sigma_1, S) \Rightarrow \mathtt{r}(\sigma_1^+, S)$, moreover $\sigma_1^+ \notin S$, and this makes true also the backward implication

$$\mathtt{r}(\sigma_1, S) \Leftarrow \mathtt{r}(\sigma_1^+, S)$$

  Closure rule (8) states that $\sigma_1 \in \widehat{\Sigma} \Rightarrow \sigma_1^+ \in \widehat{\Sigma}$.
  By inductively assuming $\mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma}$, we have the thesis:

$$\mathtt{r}(\sigma_1^+, S) \Rightarrow \mathtt{r}(\sigma_1, S) \Rightarrow \sigma_1 \in \widehat{\Sigma} \Rightarrow \sigma_1^+ \in \widehat{\Sigma}$$

- $\sigma = \sigma_1^- \wedge \sigma_1^- \notin S$ is handled exactly as the previous one.

Since, by hypothesis, σ is a finite message, and S is a finite set containing finite messages only, the check $\mathtt{r}(\sigma, S)$ can be carried out in a finite number of steps. □

Theorem C.3 *Given a finite set of finite messages $\Sigma \subseteq \mathcal{T}$ with closure $\widehat{\Sigma}$,*

$$\sigma \in \widehat{\Sigma \backslash \{\sigma\}} \Rightarrow \widehat{\Sigma \backslash \{\sigma\}} \equiv \widehat{\Sigma}$$

Proof. Since $\sigma \in \widehat{\Sigma \backslash \{\sigma\}}$ by hypothesis, closure rules (2)–(14) that hold for a term in $\widehat{\Sigma \backslash \{\sigma\}}$, also hold for the same term in $\widehat{\Sigma}$. Closure rule (1) does not hold in $\widehat{\Sigma \backslash \{\sigma\}}$, but it has been replaced by the hypothesis itself. □

Proposition 3.1 *Given a finite set of finite messages $\Sigma \subseteq \mathcal{T}$ with closure $\widehat{\Sigma}$, and a reduction rule $R = \langle \Sigma_I, C, \Sigma_O \rangle$, then*

$$\Sigma \xrightarrow{R} \Sigma' \Rightarrow \widehat{\Sigma} \equiv \widehat{\Sigma'}$$

Proof. The proof can be carried out by taking into account each rule at a time, and by reasoning on the related Σ and Σ'.

(26) $\Sigma_I = \{\mathrm{H}(\sigma)\}$, $\Sigma_O = \emptyset$, then $\Sigma' = \Sigma \backslash \{\mathrm{H}(\sigma)\}$.
$\mathtt{r}(\sigma, \Sigma) = \mathtt{TRUE}$ by rule (26): while doing such a computation, $\mathtt{r}(\cdot)$ starts from σ, and recursively computes on σ's subterms, then it will never be called on $\mathrm{H}(\sigma)$. Then it also holds that $\mathtt{r}(\sigma, \Sigma \backslash \{\mathrm{H}(\sigma)\}) = \mathtt{TRUE}$ and, by Theorem C.2, we have that $\sigma \in \widehat{\Sigma \backslash \{\mathrm{H}(\sigma)\}}$, and this implies that closure rule (5) holds, i.e. $\mathrm{H}(\sigma) \in \widehat{\Sigma \backslash \{\mathrm{H}(\sigma)\}}$ too, and by Theorem C.3 we have that $\widehat{\Sigma \backslash \{\mathrm{H}(\sigma)\}} \equiv \widehat{\Sigma}$.

(27) $\Sigma_I = \{\{[\sigma_1]\}_{\sigma_2^+}\}$, $\Sigma_O = \emptyset$, then $\Sigma' = \Sigma \backslash \{\{[\sigma_1]\}_{\sigma_2^+}\}$.
$\mathtt{r}(\sigma_1, \Sigma) = \mathtt{TRUE}$ and $\mathtt{r}(\sigma_2^+, \Sigma) = \mathtt{TRUE}$ by rule (27): while doing such a computation, $\mathtt{r}(\cdot)$, respectively starts from σ_1, and σ_2^+, and recursively computes on σ_1's and σ_2^+'s subterms, then it will never be called on $\{[\sigma_1]\}_{\sigma_2^+}$. Then it also holds that $\mathtt{r}(\sigma_1, \Sigma \backslash \{\{[\sigma_1]\}_{\sigma_2^+}\}) = \mathtt{TRUE}$ and $\mathtt{r}(\sigma_2^+, \Sigma \backslash \{\{[\sigma_1]\}_{\sigma_2^+}\}) = \mathtt{TRUE}$ and, by Theorem C.2, we have that $\sigma_1 \in \Sigma \backslash \widehat{\{\{[\sigma_1]\}_{\sigma_2^+}\}}$ and $\sigma_2^+ \in \Sigma \backslash \widehat{\{\{[\sigma_1]\}_{\sigma_2^+}\}}$, and this implies that closure rule (6) holds, i.e. $\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \backslash \widehat{\{\{[\sigma_1]\}_{\sigma_2^+}\}}$ too, and by Theorem C.3 we have that $\Sigma \backslash \widehat{\{\{[\sigma_1]\}_{\sigma_2^+}\}} \equiv \widehat{\Sigma}$.

(28) see (27).

(29) $\Sigma_I = \{\sigma^+\}$, $\Sigma_O = \emptyset$, then $\Sigma' = \Sigma \backslash \{\sigma^+\}$.
$\mathtt{r}(\sigma, \Sigma) = \mathtt{TRUE}$ by rule (29): while doing such a computation, $\mathtt{r}(\cdot)$ starts from σ, and recursively computes on σ's subterms, then it will never be called on σ^+. Then it also holds that $\mathtt{r}(\sigma, \Sigma \backslash \{\sigma^+\}) = \mathtt{TRUE}$ and, by Theorem C.2, we have that $\sigma \in \widehat{\Sigma \backslash \{\sigma^+\}}$, and this implies that closure rule (8) holds, i.e. $\sigma^+ \in \widehat{\Sigma \backslash \{\sigma^+\}}$ too, and by Theorem C.3 we have that $\widehat{\Sigma \backslash \{\sigma^+\}} \equiv \widehat{\Sigma}$.

(30) see (29).

(31) $\Sigma_I = \{suc(\sigma)\}$, $\Sigma_O = \{\sigma\}$, then $\Sigma' = \Sigma \setminus \{suc(\sigma)\} \cup \{\sigma\}$.
Σ' has been obtained from Σ by replacing $suc(\sigma)$ with σ, on the other hand, closure rules (2) and (9) state that $\text{suc}(\sigma) \in \widehat{\Sigma} \Leftrightarrow \sigma \in \widehat{\Sigma}$, i.e. starting from Σ, by means of closure rule (9), we obtain $\Sigma \cup \{\sigma\}$, on the other hand, starting from Σ', by means of closure rule (2), we obtain $\Sigma' \cup \{suc(\sigma)\}$, and $\Sigma \cup \{\sigma\} \equiv \Sigma' \cup \{suc(\sigma)\}$.

(32) $\Sigma_I = \{(\sigma_1, \sigma_2)\}$, $\Sigma_O = \{\sigma_1, \sigma_2\}$, then $\Sigma' = \Sigma \setminus \{(\sigma_1, \sigma_2)\} \cup \{\sigma_1, \sigma_2\}$.
Σ' has been obtained from Σ by replacing (σ_1, σ_2) with σ_1 and σ_2, on the other hand, closure rules (3) and (10) state that $(\sigma_1, \sigma_2) \in \widehat{\Sigma} \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \ \wedge \ \sigma_2 \in \widehat{\Sigma}$, i.e. starting from Σ, by means of closure rule (10), we obtain $\Sigma \cup \{\sigma_1, \sigma_2\}$, on the other hand, starting from Σ', by means of closure rule (3), we obtain $\Sigma' \cup \{(\sigma_1, \sigma_2)\}$, and $\Sigma \cup \{\sigma_1, \sigma_2\} \equiv \Sigma' \cup \{(\sigma_1, \sigma_2)\}$.

(33) $\Sigma_I = \{\{\sigma_1\}_{\sigma_2}\}$, $\Sigma_O = \{\sigma_1\}$, then $\Sigma' = \Sigma \setminus \{\{\sigma_1\}_{\sigma_2}\} \cup \{\sigma_1\}$.
Σ' has been obtained from Σ by replacing $\{\sigma_1\}_{\sigma_2}$ with σ_1, provided that $\mathtt{r}(\sigma_2, \Sigma)$ is TRUE. $\mathtt{r}(\sigma_2, \Sigma) = \mathtt{TRUE}$ by rule (33): while doing such a computation, $\mathtt{r}(\cdot)$ starts from σ_2, and recursively computes on σ_2's subterms, then it will never be called on $\{\sigma_1\}_{\sigma_2}$. Then it also holds that $\mathtt{r}(\sigma_2, \Sigma') = \mathtt{TRUE}$ and, by Theorem C.2, we have that both $\sigma_2 \in \widehat{\Sigma}$ and $\sigma_2 \in \widehat{\Sigma'}$. Then closure rules (11) and (4), respectively hold, enabling us to respectively, build $\Sigma \cup \{\sigma_1\}$ and $\Sigma' \cup \{\{\sigma_1\}_{\sigma_2}\}$, and the last two sets are equal.

(34) $\Sigma_I = \{\{[\sigma_1]\}_{\sigma_2^+}\}$, $\Sigma_O = \{\sigma_1, \{[\sigma_1]\}_{\sigma_2^+}\}$, then $\Sigma' = \Sigma \cup \{\sigma_1\}$.
Σ' has been obtained from Σ by adding σ_1, provided that $\mathtt{r}(\sigma_2^-, \Sigma)$ is TRUE. $\mathtt{r}(\sigma_2^-, \Sigma) = \mathtt{TRUE}$ by rule (34) (and trivially it holds that $\mathtt{r}(\sigma_2^-, \Sigma') = \mathtt{TRUE}$, being Σ' a superset of Σ), and, by Theorem C.2, we have that $\sigma 2^- \in \widehat{\Sigma}$. Then closure rule (12) holds, enabling us to add σ_1 to Σ, thus obtaining Σ'.

(35) see (34).

(36) $\Sigma_I = \{\sigma^+, \sigma^-\}$, $\Sigma_O = \{\sigma\}$, then $\Sigma' = \Sigma \setminus \{\sigma^+, \sigma^-\} \cup \{\sigma\}$.
Σ' has been obtained from Σ by replacing σ^+, σ^- with σ, on the other hand, closure rules (14) and (8) state that $\sigma \in \widehat{\Sigma} \Leftrightarrow \sigma^+ \in \widehat{\Sigma} \ \wedge \ \sigma^- \in \widehat{\Sigma}$, i.e. starting from Σ, by means of closure rule (14), we obtain $\Sigma \cup \{\sigma\}$, on the other hand, starting from Σ', by means of closure rule (8), we obtain $\Sigma' \cup \{\sigma^+, \sigma^-\}$, and $\Sigma \cup \{\sigma\} \equiv \Sigma' \cup \{\sigma^+, \sigma^-)\}$.

In all cases we are able to obtain the same set as an intermediate result during the computation of the closure of Σ and Σ', thus they have the same closure. □

Theorem C.4 *Given a finite set of terms $\Sigma \subseteq \mathcal{T}$, and a finite message $\sigma \in \mathcal{T}$, then*

$$\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I$$

and $\mathtt{r}(\sigma, \Sigma)$ is computed in a finite number of steps.

Proof. The basic idea is to carry out the proof inductively: the theorem will be proved directly for the two disjoint[4] cases $\sigma \in \Sigma$, and $\sigma \in \mathcal{A} \setminus \Sigma$ (i.e. the base of the induction), and inductively the remaining cases: $\sigma \notin \mathcal{A} \wedge \sigma \notin \Sigma$ (i.e. $\sigma \notin \mathcal{A} \cup \Sigma$).
Base ($\sigma \in \Sigma \ \vee \ \sigma \in \mathcal{A} \setminus \Sigma$)

- $\sigma \in \Sigma$: we have that
  - $\sigma \in \Sigma$ as hypothesis;
  - $\sigma \in \Sigma \ \Rightarrow \ \mathtt{r}(\sigma, \Sigma)$ from the definition of $\mathtt{r}$ (`if` $\sigma \in \Sigma$ `then return TRUE`);
  - $\sigma \in \Sigma \ \Rightarrow \ \sigma \in \widehat{\Sigma}^I$ by rule (1).

  Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I$ holds, i.e.:

$$\left.\begin{array}{l} \sigma \in \Sigma \\ \wedge \\ \sigma \in \Sigma \ \Rightarrow \ \mathtt{r}(\sigma, \Sigma) \\ \wedge \\ \sigma \in \Sigma \ \Rightarrow \ \sigma \in \widehat{\Sigma}^I \end{array}\right\} \ \Rightarrow \ \left(\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I\right)$$

  which can be easily proved by using boolean expressions.

[4] $\Sigma \cap (\mathcal{A} \setminus \Sigma) = \emptyset$.

- $\sigma \in \mathcal{A} \setminus \Sigma$: we have that
  - $\sigma \in \mathcal{A} \wedge \sigma \notin \Sigma$ as hypothesis;
  - $\sigma \in \mathcal{A} \wedge \sigma \notin \Sigma \Rightarrow \overline{\mathtt{r}(\sigma, \Sigma)}$ from the definition of `r` (`else` $(\sigma \in \mathcal{A} \setminus \Sigma)$ `return FALSE`);
  - $\sigma \in \mathcal{A} \wedge \sigma \notin \Sigma \Rightarrow \sigma \notin \widehat{\Sigma}^I$ since rule (1) does not hold by hypothesis, and rules (2)–(8) cannot be used to build a name.

  Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I$ holds, i.e.:

$$\left.\begin{array}{l} \sigma \in \mathcal{A} \wedge \sigma \notin \Sigma \\ \wedge \\ \sigma \in \mathcal{A} \wedge \sigma \notin \Sigma \Rightarrow \overline{\mathtt{r}(\sigma, \Sigma)} \\ \wedge \\ \sigma \in \mathcal{A} \wedge \sigma \notin \Sigma \Rightarrow \sigma \notin \widehat{\Sigma}^I \end{array}\right\} \Rightarrow \left(\mathtt{r}(\sigma, \Sigma) \Leftrightarrow \sigma \in \widehat{\Sigma}^I\right)$$

  which can be easily proved by using boolean expressions.

Induction ($\sigma \notin \mathcal{A} \cup \Sigma$)

- $\sigma = \mathrm{suc}(\sigma_1) \wedge \mathrm{suc}(\sigma_1) \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \Rightarrow \mathtt{r}(suc(\sigma_1), \Sigma)$, moreover $\mathrm{suc}(\sigma_1) \notin \Sigma$, and this makes true also the backward implication,[5] thus leading to

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \mathtt{r}(suc(\sigma_1), \Sigma)$$

  Moreover, by induction, we are allowed to assume

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I$$

  Then, we still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \Leftrightarrow \mathrm{suc}(\sigma_1) \in \widehat{\Sigma}^I$$

  $\Rightarrow$ Trivial, since rule (2) allows us to build $\mathrm{suc}(\sigma_1)$ from σ_1.
  $\Leftarrow$ $\mathrm{suc}(\sigma_1) \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $\mathrm{suc}(\sigma_1) \in \widehat{\Sigma}^I$, then it holds that $\mathrm{suc}(\sigma_1)$ has been built by means of a rule among (2)–(8), i.e. rule (2), and this leads to $\sigma_1 \in \widehat{\Sigma}^I$.
  We can now state that

$$\mathtt{r}(suc(\sigma_1), \Sigma) \Leftrightarrow \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \Leftrightarrow \mathrm{suc}(\sigma_1) \in \widehat{\Sigma}^I$$

- $\sigma = (\sigma_1, \sigma_2) \wedge (\sigma_1, \sigma_2) \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \wedge \mathtt{r}(\sigma_2, \Sigma) \Rightarrow \mathtt{r}((\sigma_1, \sigma_2), \Sigma)$, moreover $(\sigma_1, \sigma_2) \notin \Sigma$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \Sigma) \wedge \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \mathtt{r}((\sigma_1, \sigma_2), \Sigma)$$

  Moreover, by induction, we are allowed to assume

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I$$
$$\mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}^I$$

  We still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}^I$$

  $\Rightarrow$ Trivial, since rule (3) allows us to build (σ_1, σ_2) from σ_1 and σ_2 separately.
  $\Leftarrow$ $(\sigma_1, \sigma_2) \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $(\sigma_1, \sigma_2) \in \widehat{\Sigma}^I$, then it holds that (σ_1, σ_2) has been built by means of a rule among (2)–(8), i.e. rule (3), and this leads to $\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I$.

[5] we know that $\mathtt{r}(suc(\sigma_1), \Sigma)$ has given `TRUE`, and that the result has been computed by means of $\mathtt{r}(\sigma_1, \Sigma)$.

Thus we shall prove that under these hypotheses $\mathtt{r}((\sigma_1, \sigma_2), \Sigma) \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}^I$ holds, i.e.:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \mathtt{r}((\sigma_1, \sigma_2), \Sigma) \\ \wedge \\ \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \\ \wedge \\ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}^I \\ \wedge \\ \sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I \ \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}^I \end{array}\right\} \Rightarrow \ \left(\mathtt{r}((\sigma_1, \sigma_2), \Sigma) \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}^I\right)$$

which can be easily proved by using boolean expressions.

- $\sigma = \{\sigma_1\}_{\sigma_2} \wedge \{\sigma_1\}_{\sigma_2} \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \ \Rightarrow \ \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \Sigma)$, moreover $\{\sigma_1\}_{\sigma_2} \notin \Sigma$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \Sigma)$$

Moreover, by induction, we are allowed to assume

$$\begin{array}{l} \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \\ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}^I \end{array}$$

We still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I \ \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}^I$$

$\Rightarrow$ Trivial, since rule (4) allows us to build $\{\sigma_1\}_{\sigma_2}$ from σ_1 and σ_2 separately.
$\Leftarrow$ $\{\sigma_1\}_{\sigma_2} \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $\{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}^I$, then it holds that $\{\sigma_1\}_{\sigma_2}$ has been built by means of a rule among (2)–(8), i.e. rule (4), and this leads to $\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I$.
Thus we shall prove that under these hypotheses $\mathtt{r}(\{\sigma_1\}_{\sigma_2}, \Sigma) \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}^I$ holds, i.e.:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \Sigma) \\ \wedge \\ \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \\ \wedge \\ \mathtt{r}(\sigma_2, \Sigma) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}^I \\ \wedge \\ \sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2 \in \widehat{\Sigma}^I \ \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}^I \end{array}\right\} \Rightarrow \ \left(\mathtt{r}(\{\sigma_1\}_{\sigma_2}, \Sigma) \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}^I\right)$$

which can be easily proved by using boolean expressions.

- $\sigma = \mathrm{H}(\sigma_1) \wedge \mathrm{H}(\sigma_1) \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \ \Rightarrow \ \mathtt{r}(\mathrm{H}(\sigma_1), \Sigma)$, moreover $\mathrm{H}(\sigma_1) \notin \Sigma$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \mathtt{r}(\mathrm{H}(\sigma_1), \Sigma)$$

Moreover, by induction, we are allowed to assume

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I$$

We still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \ \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}^I$$

$\Rightarrow$ Trivial, since rule (5) allows us to build $\mathrm{H}(\sigma_1)$ from σ_1.
$\Leftarrow$ $\mathrm{H}(\sigma_1) \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $\mathrm{H}(\sigma_1) \in \widehat{\Sigma}^I$, then it holds that $\mathrm{H}(\sigma_1)$ has been built by means of a rule among (2)–(8), i.e. rule (5), and this leads to $\sigma_1 \in \widehat{\Sigma}^I$.
We can now state that

$$\mathtt{r}(\mathrm{H}(\sigma_1), \Sigma) \Leftrightarrow \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}^I$$

- $\sigma = \{[\sigma_1]\}_{\sigma_2^+} \wedge \{[\sigma_1]\}_{\sigma_2^+} \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \wedge \mathtt{r}(\sigma_2^+, \Sigma) \Rightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \Sigma)$, moreover $\{[\sigma_1]\}_{\sigma_2^+} \notin \Sigma$, thus leading us to have

$$\mathtt{r}(\sigma_1, \Sigma) \wedge \mathtt{r}(\sigma_2^+, \Sigma) \Leftrightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \Sigma)$$

  Moreover, by induction, we are allowed to assume

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I$$
$$\mathtt{r}(\sigma_2^+, \Sigma) \Leftrightarrow \sigma_2^+ \in \widehat{\Sigma}^I$$

  We still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2^+ \in \widehat{\Sigma}^I \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}^I$$

  $\Rightarrow$ Trivial, since rule (6) allows us to build $\{[\sigma_1]\}_{\sigma_2^+}$ from σ_1 and σ_2^+ separately.
  $\Leftarrow$ $\{[\sigma_1]\}_{\sigma_2^+}) \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $\{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}^I$, then it holds that $\{[\sigma_1]\}_{\sigma_2^+}$ has been built by means of a rule among (2)–(8), i.e. rule (6), and this leads to $\sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2^+ \in \widehat{\Sigma}^I$.
  Thus we shall prove that under these hypotheses $\mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \Sigma) \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}^I$ holds, i.e.:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \Sigma) \wedge \mathtt{r}(\sigma_2^+, \Sigma) \Leftrightarrow \\ \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \Sigma) \\ \wedge \\ \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \\ \wedge \\ \mathtt{r}(\sigma_2^+, \Sigma) \Leftrightarrow \sigma_2^+ \in \widehat{\Sigma}^I \\ \wedge \\ \sigma_1 \in \widehat{\Sigma}^I \wedge \sigma_2^+ \in \widehat{\Sigma}^I \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}^I \end{array}\right\} \Rightarrow \left(\mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \Sigma) \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}^I\right)$$

  which can be easily proved by using boolean expressions.
- $\sigma = [\{\sigma_1\}]_{\sigma_2^-} \wedge [\{\sigma_1\}]_{\sigma_2^-} \notin \Sigma$ is handled exactly as the previous one (where closure rule (7) replaces (6)).
- $\sigma = \sigma_1^+ \wedge \sigma_1^+ \notin \Sigma$: trivially $\mathtt{r}(\sigma_1, \Sigma) \Rightarrow \mathtt{r}(\sigma_1^+, \Sigma)$, moreover $\sigma_1^+ \notin \Sigma$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \mathtt{r}(\sigma_1^+, \Sigma)$$

  Moreover, by induction, we are allowed to assume

$$\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I$$

  We still need to prove that

$$\sigma_1 \in \widehat{\Sigma}^I \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}^I$$

  $\Rightarrow$ Trivial, since rule (8) allows us to build σ_1^+ from σ_1.
  $\Leftarrow$ $\sigma_1^+ \notin \Sigma$ by hypothesis, then rule (1) is not responsible for $\sigma_1^+ \in \widehat{\Sigma}^I$, then it holds that σ_1^+ has been built by means of a rule among (2)–(8), i.e. rule (8), and this leads to $\sigma_1 \in \widehat{\Sigma}^I$.
  We can now state that

$$\mathtt{r}(\sigma_1^+, \Sigma) \Leftrightarrow \mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}^I$$

- $\sigma = \sigma_1^- \wedge \sigma_1^- \notin \Sigma$ then return $\mathtt{r}(\sigma_1, \Sigma)$ is handled exactly as the previous one.

Since, by hypothesis, σ is a finite message, and Σ is a finite set containing finite messages only, the check $\mathtt{r}(\sigma, \Sigma)$ can be carried out in a finite number of steps. □

Theorem C.5 *Given a finite set of terms $\Sigma \subseteq \mathcal{T}$, $\overline{\Sigma}$ defined by (17)–(25), and a finite message $\sigma \in \mathcal{T}$, then*

$$\mathtt{r}(\sigma, \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}$$

and the computation of $\mathtt{r}(\sigma, \overline{\Sigma})$ takes a finite number of steps.

Proof. The basic idea is to carry out the proof inductively: the theorem will be proved directly for the two disjoint[6] cases $\sigma \in \overline{\Sigma}$, and $\sigma \in \mathcal{A}\backslash\overline{\Sigma}$ (i.e. the base of the induction), and inductively for the remaining cases: $\sigma \notin \mathcal{A} \wedge \sigma \notin \overline{\Sigma}$ (i.e. $\sigma \notin \mathcal{A} \cup \overline{\Sigma}$).

Base ($\sigma \in \overline{\Sigma} \ \vee \ \sigma \in \mathcal{A}\backslash\Sigma$)

- $\sigma \in \overline{\Sigma}$: we have that
  - $\sigma \in \overline{\Sigma}$ as hypothesis;
  - $\sigma \in \overline{\Sigma} \ \Rightarrow \ \mathtt{r}(\sigma, \overline{\Sigma})$ from the function (`if` $\sigma \in \overline{\Sigma}$ `then return TRUE`);
  - $\sigma \in \overline{\Sigma} \ \Rightarrow \ \sigma \in \widehat{\Sigma}$, since $\overline{\Sigma} \subseteq \widehat{\Sigma}$.

  Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}$ holds, i.e.:

  $$\left.\begin{array}{l} \sigma \in \overline{\Sigma} \\ \wedge \\ \sigma \in \overline{\Sigma} \ \Rightarrow \ \mathtt{r}(\sigma, \overline{\Sigma}) \\ \wedge \\ \sigma \in \overline{\Sigma} \ \Rightarrow \ \sigma \in \widehat{\Sigma} \end{array}\right\} \ \Rightarrow \ \left(\mathtt{r}(\sigma, \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}\right)$$

  which can be easily proved by using boolean expressions.
- $\sigma \in \mathcal{A}\backslash\overline{\Sigma}$: we have that
  - $\sigma \in \mathcal{A} \ \wedge \ \sigma \notin \overline{\Sigma}$ as hypothesis;
  - $\sigma \in \mathcal{A} \ \wedge \ \sigma \notin \overline{\Sigma} \ \Rightarrow \ \overline{\mathtt{r}(\sigma, \overline{\Sigma})}$ from the function (`else` $(\sigma \in \mathcal{A}\backslash\overline{\Sigma})$ `return FALSE`);
  - $\sigma \in \mathcal{A} \ \Rightarrow \ (\sigma \in \overline{\Sigma} \Leftrightarrow \sigma \in \widehat{\Sigma})$ from rule (17).

  Thus we shall prove that under these hypotheses $\mathtt{r}(\sigma, \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}$ holds, i.e.:

  $$\left.\begin{array}{l} \sigma \in \mathcal{A} \ \wedge \ \sigma \notin \overline{\Sigma} \\ \wedge \\ \sigma \in \mathcal{A} \ \wedge \ \sigma \notin \overline{\Sigma} \ \Rightarrow \ \overline{\mathtt{r}(\sigma, \overline{\Sigma})} \\ \wedge \\ \sigma \in \mathcal{A} \ \Rightarrow \ (\sigma \in \overline{\Sigma} \Leftrightarrow \sigma \in \widehat{\Sigma}) \end{array}\right\} \ \Rightarrow \ \left(\mathtt{r}(\sigma, \overline{\Sigma}) \Leftrightarrow \sigma \in \widehat{\Sigma}\right)$$

  which can be easily proved by using boolean expressions. □

Induction ($\sigma \notin \mathcal{A} \cup \overline{\Sigma}$)

- $\sigma = \mathrm{suc}(\sigma_1) \wedge \mathrm{suc}(\sigma_1) \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \ \Rightarrow \ \mathtt{r}(suc(\sigma_1), \overline{\Sigma})$, moreover $\mathrm{suc}(\sigma_1) \notin \overline{\Sigma}$, and this makes true also the backward implication,[7] leading us to have:

  $$\mathtt{r}(suc(\sigma_1), \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\sigma_1, \overline{\Sigma})$$

  Closure rules (2) and (9) state that $\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{suc}(\sigma_1) \in \widehat{\Sigma}$.
  By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$, we have the thesis:

  $$\mathtt{r}(suc(\sigma_1), \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\sigma_1, \overline{\Sigma}) \ \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{suc}(\sigma_1) \in \widehat{\Sigma}$$

- $\sigma = (\sigma_1, \sigma_2) \wedge (\sigma_1, \sigma_2) \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \ \wedge \ \mathtt{r}(\sigma_2, \overline{\Sigma}) \ \Rightarrow \ \mathtt{r}((\sigma_1, \sigma_2), \overline{\Sigma})$, moreover $(\sigma_1, \sigma_2) \notin \overline{\Sigma}$, and this makes true also the backward implication, leading us to have:

  $$\mathtt{r}((\sigma_1, \sigma_2), \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\sigma_1, \overline{\Sigma}) \ \wedge \ \mathtt{r}(\sigma_2, \overline{\Sigma})$$

  Closure rules (3) and (10) state that $\sigma_1 \in \widehat{\Sigma} \ \wedge \ \sigma_2 \in \widehat{\Sigma} \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}$.

---

[6] $\overline{\Sigma} \cap (\mathcal{A}\backslash\overline{\Sigma}) = \emptyset$.

[7] we know that $\mathtt{r}(suc(\sigma_1), \Sigma)$ has given `TRUE`, and that the result has been computed by means of $\mathtt{r}(\sigma_1, \Sigma)$.

By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}$, we have:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \mathtt{r}((\sigma_1, \sigma_2), \overline{\Sigma}) \\ \wedge \\ \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \sigma_2 \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}((\sigma_1, \sigma_2), \overline{\Sigma}) \Leftrightarrow (\sigma_1, \sigma_2) \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

- $\sigma = \{\sigma_1\}_{\sigma_2} \wedge \{\sigma_1\}_{\sigma_2} \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma}) \Rightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \overline{\Sigma})$, moreover $\{\sigma_1\}_{\sigma_2} \notin \overline{\Sigma}$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \overline{\Sigma})$$

Now we have to deal with the two following cases:

  - $\{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$: being $\overline{\Sigma}$ a subset of $\widehat{\Sigma}$, rules relating $\overline{\Sigma}$ with $\widehat{\Sigma}$ hold, i.e. (20) in particular; closure rules (4) and (11) always hold, and we have $\{\sigma_1\}_{\sigma_2} \notin \overline{\Sigma}$ by hypothesis, leading us to

$$\left.\begin{array}{l} \{\sigma_1\}_{\sigma_2} \notin \overline{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow \sigma_1 \in \widehat{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \overline{\Sigma} \Leftrightarrow \sigma_2 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions. □

  - $\{\sigma_1\}_{\sigma_2} \notin \widehat{\Sigma}$: here we have no constraints coming from $\overline{\Sigma}$, and just closure rules (4) and (11) hold, leading us to

$$\left.\begin{array}{l} \{\sigma_1\}_{\sigma_2} \notin \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Rightarrow \sigma_1 \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions. □

and both of them lead to $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}$. By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}$, we have:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\{\sigma_1\}_{\sigma_2}, \overline{\Sigma}) \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2 \in \widehat{\Sigma} \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_2, \overline{\Sigma}) \Leftrightarrow \sigma_2 \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\{\sigma_1\}_{\sigma_2}, \overline{\Sigma}) \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

- $\sigma = \mathrm{H}(\sigma_1) \wedge \mathrm{H}(\sigma_1) \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Rightarrow \mathtt{r}(\mathrm{H}(\sigma_1), \overline{\Sigma})$, moreover $\mathrm{H}(\sigma_1) \notin \overline{\Sigma}$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\mathrm{H}(\sigma_1), \overline{\Sigma}))$$

Now we have to deal with the two following cases:

  - $\mathrm{H}(\sigma_1) \in \widehat{\Sigma}$: being $\overline{\Sigma}$ a subset of $\widehat{\Sigma}$, rules relating $\overline{\Sigma}$ with $\widehat{\Sigma}$ hold, i.e. (21) in particular; closure rule (5) always holds, and we have $\mathrm{H}(\sigma_1) \notin \overline{\Sigma}$ by hypothesis, leading us to

$$\left.\begin{array}{l} \mathrm{H}(\sigma_1) \notin \overline{\Sigma} \\ \wedge \\ \mathrm{H}(\sigma_1) \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Rightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma} \\ \wedge \\ \mathrm{H}(\sigma_1) \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions. □

  - $\mathrm{H}(\sigma_1) \notin \widehat{\Sigma}$: here we have no constraints coming from $\overline{\Sigma}$, and just closure rule (5) holds, leading us to

$$\left.\begin{array}{l} \mathrm{H}(\sigma_1) \notin \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Rightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions, □

and both of them lead to $\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}$. By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$, we have:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\mathrm{H}(\sigma_1), \overline{\Sigma}) \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\mathrm{H}(\sigma_1), \overline{\Sigma}) \Leftrightarrow \mathrm{H}(\sigma_1) \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

- $\sigma = \{[\sigma_1]\}_{\sigma_2^+} \wedge \{[\sigma_1]\}_{\sigma_2^+} \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2^+, \overline{\Sigma}) \Rightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \overline{\Sigma})$, moreover $\{[\sigma_1]\}_{\sigma_2^+} \notin \overline{\Sigma}$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2^+, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \overline{\Sigma})$$

Now we have to deal with the two following cases:

  - $\{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$: being $\overline{\Sigma}$ a subset of $\widehat{\Sigma}$, rules relating $\overline{\Sigma}$ with $\widehat{\Sigma}$ hold, i.e. (22) in particular; closure rule (6) always holds, and we have $\{[\sigma_1]\}_{\sigma_2} \notin \overline{\Sigma}$ by hypothesis, leading us to

$$\left.\begin{array}{l} \{[\sigma_1]\}_{\sigma_2^+} \notin \overline{\Sigma} \\ \wedge \\ \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Rightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \\ \wedge \\ \{[\sigma_1]\}_{\sigma_2^+} \in \overline{\Sigma} \Leftrightarrow \sigma_2^+ \notin \widehat{\Sigma} \vee \sigma_1 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(.\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions. □

  - $\{[\sigma_1]\}_{\sigma_2^+} \notin \widehat{\Sigma}$: here we have no constraints coming from $\overline{\Sigma}$, and just closure rule (6) holds, leading us to

$$\left.\begin{array}{l} \{[\sigma_1]\}_{\sigma_2^+} \notin \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Rightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}\right)$$

    which can be easily proved by using boolean expressions. □

and both of them lead to $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}$. By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$ and $\mathtt{r}(\sigma_2^+, \overline{\Sigma}) \Leftrightarrow \sigma_2^+ \in \widehat{\Sigma}$, we have:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2^+, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \overline{\Sigma}) \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \wedge \sigma_2^+ \in \widehat{\Sigma} \Leftrightarrow \{\sigma_1\}_{\sigma_2^+} \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_2^+, \overline{\Sigma}) \Leftrightarrow \sigma_2^+ \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\{[\sigma_1]\}_{\sigma_2^+}, \overline{\Sigma}) \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

- $\sigma = [\{\sigma_1\}]_{\sigma_2^-} \wedge [\{\sigma_1\}]_{\sigma_2^-} \notin \Sigma$ is handled exactly as the previous one [where closure rule (7) replaces (6), and (23) replaces (22)].
- $\sigma = \sigma_1^+ \wedge \sigma_1^+ \notin \overline{\Sigma}$: trivially $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Rightarrow \mathtt{r}(\sigma_1^+, \overline{\Sigma})$, moreover $\sigma_1^+ \notin \overline{\Sigma}$, and this makes true also the backward implication, thus leading us to have

$$\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\sigma_1^+, \overline{\Sigma}))$$

Now we have to deal with the two following cases:

– $\sigma_1^+ \in \widehat{\Sigma}$: being $\overline{\Sigma}$ a subset of $\widehat{\Sigma}$, rules relating $\overline{\Sigma}$ with $\widehat{\Sigma}$ hold, i.e. (24) in particular; closure rule (8) always holds, and we have $\sigma_1^+ \notin \overline{\Sigma}$ by hypothesis, leading us to

$$\left.\begin{array}{l} \sigma_1^+ \notin \overline{\Sigma} \\ \wedge \\ \sigma_1^+ \in \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Rightarrow \sigma_1^+ \in \widehat{\Sigma} \\ \wedge \\ \sigma_1^+ \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

– $\sigma_1^+ \notin \widehat{\Sigma}$: here we have no constraints coming from $\overline{\Sigma}$, and just closure rule (8) holds, leading us to

$$\left.\begin{array}{l} \sigma_1^+ \notin \widehat{\Sigma} \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Rightarrow \sigma_1^+ \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

and both of them lead to $\sigma_1 \in \widehat{\Sigma} \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}$. By inductively assuming $\mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}$, we have:

$$\left.\begin{array}{l} \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \mathtt{r}(\sigma_1^+, \overline{\Sigma}) \\ \wedge \\ \sigma_1 \in \widehat{\Sigma} \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma} \\ \wedge \\ \mathtt{r}(\sigma_1, \overline{\Sigma}) \Leftrightarrow \sigma_1 \in \widehat{\Sigma} \end{array}\right\} \Rightarrow \left(\mathtt{r}(\sigma_1^+, \overline{\Sigma}) \Leftrightarrow \sigma_1^+ \in \widehat{\Sigma}\right)$$

which can be easily proved by using boolean expressions. □

- $\sigma = \sigma_1^- \wedge \sigma_1^- \notin \Sigma$ is handled exactly as the previous one, where (25) replaces (24).

Since, by hypothesis, σ is a finite message, and $\overline{\Sigma}$ is a finite set containing finite messages only (Theorem 3.1), the check $\mathtt{r}(\sigma, \overline{\Sigma})$ can be carried out in a finite number of steps. □

Theorem C.6 *Given a finite set of terms $\Sigma \subseteq \mathcal{T}$, and $\overline{\Sigma}$ defined by (17)–(25) ⇒ rules (26)–(36) do not hold in $\overline{\Sigma}$.*

Proof. The basic idea is to carry out the proof for each syntactic form of $\sigma \in \overline{\Sigma}$

- $\sigma \in \mathcal{A}$: no rule among (26)–(36) deals with names, then we have the thesis.
- $\sigma = \mathrm{suc}(\sigma_1)$: rule (18) forbids $\mathrm{suc}(\cdot)$ to belong to $\overline{\Sigma}$, and makes rule (31) inapplicable, the only one dealing with $\mathrm{suc}(\cdot)$ among (26)–(36).

- $\sigma = (\sigma_1, \sigma_2)$: rule (19) forbids $(\cdot, \cdot)$ to belong to $\overline{\Sigma}$, and makes rule (32) inapplicable, the only one dealing with $(\cdot, \cdot)$ among (26)–(36).
- $\sigma = \{\sigma_1\}_{\sigma_2}$: from (20) we have that $\{\sigma_1\}_{\sigma_2} \in \overline{\Sigma} \Leftrightarrow \sigma_2 \notin \widehat{\Sigma}$, and, by Theorem C.2, we have $\sigma_2 \notin \widehat{\Sigma} \Rightarrow \overline{\mathtt{r}(\sigma_2, \overline{\Sigma})}$. Among rules (26)–(36), just (33) deals with $\{\cdot\}.$, but it needs $\mathtt{r}(\sigma_2, \overline{\Sigma})$, and we have $\overline{\mathtt{r}(\sigma_2, \overline{\Sigma})}$.
- $\sigma = \mathrm{H}(\sigma_1)$: from (21) we have that $\mathrm{H}(\sigma_1) \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}$, and, by Theorem C.2, we have $\sigma_1 \notin \widehat{\Sigma} \Rightarrow \overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$. Among rules (26)–(36), just (26) deals with $\mathrm{H}(\cdot)$, but it needs $\mathtt{r}(\sigma_1, \overline{\Sigma})$, and we have $\overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$.
- $\sigma = \sigma_1^+$: from (24) we have that $\sigma_1^+ \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}$, and, by Theorem C.2, we have $\sigma_1 \notin \widehat{\Sigma} \Rightarrow \overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$. Among rules (26)–(36), just (29) deals with $\cdot^+$, but it needs $\mathtt{r}(\sigma_1, \overline{\Sigma})$, and we have $\overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$.
- $\sigma = \sigma_1^-$: as the above case, where (25) and (30) replace (24) and (29), respectively.
- $\sigma = \{[\sigma_1]\}_{\sigma_2^+}$: from (22) we have that $\{[\sigma_1]\}_{\sigma_2} \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma} \vee \sigma_2 \notin \widehat{\Sigma}$, and, by Theorem C.2, we have $\sigma_1 \notin \widehat{\Sigma} \vee \sigma_2 \notin \widehat{\Sigma} \Rightarrow \overline{\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma})}$. Among rules (26)–(36), (27) deals with $\{[\cdot]\}_{\cdot^+}$, but it needs $\mathtt{r}(\sigma_2, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_1, \overline{\Sigma})$, and we have $\overline{\mathtt{r}(\sigma_1, \overline{\Sigma}) \wedge \mathtt{r}(\sigma_2, \overline{\Sigma})}$.
- $\sigma = [\{\sigma_1\}]_{\sigma_2^-}$: as the above case, where (23) and (28) replace (22) and (27), respectively.

Now we have proved that rules (26)–(32) and (33) are inapplicable, and we still need to prove that also (34), (35), and (36) are inapplicable:

- (34): if we absurdly assume that (34) holds, given $\{[\sigma_1]\}_{\sigma_2^+} \in \overline{\Sigma}$, we would have $\mathtt{r}(\sigma_2^-, \overline{\Sigma}) \wedge \overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$. In particular $\mathtt{r}(\sigma_2^-, \overline{\Sigma})$ would enable Theorem C.4, then $\mathtt{r}(\sigma_2^-, \overline{\Sigma}) \Leftrightarrow \sigma_2^- \in \widehat{\overline{\Sigma}}^I$ and, being $\widehat{\overline{\Sigma}}^I \subseteq \widehat{\overline{\Sigma}}$, $\sigma_2^- \in \widehat{\overline{\Sigma}}$. Since $\overline{\Sigma} \subseteq \widehat{\overline{\Sigma}}$, we would have $\{[\sigma_1]\}_{\sigma_2^+} \in \widehat{\overline{\Sigma}}$ and $\sigma_2^- \in \widehat{\overline{\Sigma}}$, i.e. rule (12) would hold, giving us $\sigma_1 \in \widehat{\overline{\Sigma}}$ and, by Theorem C.5 when $\overline{\Sigma}$ replaces a generic Σ, we would have $\mathtt{r}(\sigma_1, \overline{\Sigma})$, but, by assuming (34) holding, we also have $\overline{\mathtt{r}(\sigma_1, \overline{\Sigma})}$.
- (35): as the above case, where (13) and (35) replace (12) and (34), respectively.
- (36): if we absurdly assume that (36) holds, we would have $\sigma^+ \in \overline{\Sigma} \wedge \sigma^- \in \overline{\Sigma}$, and this would enable (14), leading us to have $\sigma \in \widehat{\Sigma}$, but (24) and (25) hold and prevent $\sigma \in \widehat{\Sigma}$.

Since the theorem holds for each syntactic form of $\sigma \in \overline{\Sigma}$, we have the proof. □

Theorem C.7 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$,

$$\textit{if rules (26)–(36) do not hold in } \Sigma \Rightarrow \widehat{\Sigma} \setminus \widehat{\Sigma}^I = \emptyset$$

i.e. from such a Σ *we can build* $\widehat{\Sigma}$ *by using* $\mathcal{I}$ *rules only* ($\widehat{\Sigma} \equiv \widehat{\Sigma}^I$).

Proof. Let us absurdly assume that exists $\sigma \in \widehat{\Sigma} \setminus \widehat{\Sigma}^I$. Then, in the sequence of closure rules leading to σ from Σ, at least a rule among (9)–(14) shall exist.
The basic idea is to carry out the proof inductively:

Base (we assume that a rule among (9)–(14) is the first one in the sequence of rules leading to σ from Σ)

- (9): if this rule holds in Σ, then also (31) holds, but this is forbidden by hypothesis.
- (10): if this rule holds in Σ, then also (32) holds, but this is forbidden by hypothesis.
- (11): if this rule holds in Σ, then we have $\{\sigma_1\}_{\sigma_2} \in \Sigma \wedge \sigma_2 \in \Sigma$, then also (33) holds, but this is forbidden by hypothesis.
- (12): if this rule holds in Σ, then we have $\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \wedge \sigma_2^- \in \Sigma$, so the first two hypotheses of rule (34) hold. Let us now investigate what happens of the third hypothesis of rule (34):
  - if $\overline{\mathtt{r}(\sigma_1, \Sigma)}$ then all the hypotheses of rule (34) hold, absurdly enabling it
  - if $\mathtt{r}(\sigma_1, \Sigma)$ then (34) does not hold, and, by Theorem C.4 it follows that $\sigma_1 \in \widehat{\Sigma}^I$, and it means that σ_1 can be built from Σ by means of rules (1) and (2)–(8), i.e. rule (12) is not needed to compute σ_1.

  In practice, by assuming (12) enabled in Σ, we have two scenarios: an absurd situation, or rule (12) produces a term which can be also computed by means of $\mathcal{I}$ rules only, i.e. (12) is not needed.

- (13): the same as above, where we replace (12), (34), $\{[\sigma_1]\}_{\sigma_2^+}$ and σ_2^- with (13), (35), $[\{\sigma_1\}]_{\sigma_2^-}$ and σ_2^+, respectively.
- (14): if this rule holds in Σ, then (36) is absurdly enabled.

Induction (we assume that, starting from Σ, rules (1) and (2)–(8) have been applied a number of times, leading to a new set $\Sigma \cup \delta_\Sigma$, where it holds that $\Sigma \cup \delta_\Sigma \subseteq \widehat{\Sigma}^I$, and $\delta_\Sigma \subseteq \widehat{\Sigma}^I$, and where a rule among (9)–(14) can be applied).

- (9): $\mathrm{suc}(\sigma) \in \Sigma \cup \delta_\Sigma$ leads to two disjoint sub-cases
  - $\mathrm{suc}(\sigma) \in \Sigma$ already managed in *Base*;
  - $\mathrm{suc}(\sigma) \in \delta_\Sigma$, then it has been computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute $\mathrm{suc}(\sigma)$ is (2) and this means that also σ shall belong to $\Sigma \cup \delta_\Sigma$, i.e. rule (9) is not needed to compute σ.
- (10): $(\sigma_1, \sigma_2) \in \Sigma \cup \delta_\Sigma$ leads to two disjoint sub-cases
  - $(\sigma_1, \sigma_2) \in \Sigma$ already managed in *Base*;
  - $(\sigma_1, \sigma_2) \in \delta_\Sigma$, then it has been computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute (σ_1, σ_2) is (3) and this means that both σ_1 and σ_2 shall belong to $\Sigma \cup \delta_\Sigma$, i.e. rule (10) is not needed to compute σ_1 and σ_2.
- (11): we have that

$$\begin{array}{rcl} \{\sigma_1\}_{\sigma_2} & \in & \Sigma \cup \delta_\Sigma \\ & \wedge & \\ \sigma_2 & \in & \Sigma \cup \delta_\Sigma \end{array}$$

  and, being $\Sigma \cup \delta_\Sigma$ built by means of $\mathcal{I}$ rules only, also $\sigma_2 \in \Sigma \cup \delta_\Sigma \Rightarrow \sigma_2 \in \widehat{\Sigma}^I$ holds, leading to two sub-cases:
  - $\{\sigma_1\}_{\sigma_2} \in \delta_\Sigma \wedge \sigma_2 \in \widehat{\Sigma}^I$, then $\{\sigma_1\}_{\sigma_2}$ has been computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute $\{\sigma_1\}_{\sigma_2}$ is (4) and this means that both σ_1 and σ_2 shall belong to $\Sigma \cup \delta_\Sigma$, i.e. rule (11) is not needed to compute σ_1;
  - $\{\sigma_1\}_{\sigma_2} \in \Sigma \wedge \sigma_2 \in \widehat{\Sigma}^I$: by Theorem C.4 we have that $\{\sigma_1\}_{\sigma_2} \in \Sigma \wedge \sigma_2 \in \widehat{\Sigma}^I \Leftrightarrow \{\sigma_1\}_{\sigma_2} \in \Sigma \wedge \mathtt{r}(\sigma_2, \Sigma)$, and this absurdly enables rule (33).
- (12): we have that

$$\begin{array}{rcl} \{[\sigma_1]\}_{\sigma_2^+} & \in & \Sigma \cup \delta_\Sigma \\ & \wedge & \\ \sigma_2^- & \in & \Sigma \cup \delta_\Sigma \end{array}$$

  and, being $\Sigma \cup \delta_\Sigma$ built by means of $\mathcal{I}$ rules only, also $\sigma_2^- \in \Sigma \cup \delta_\Sigma \Rightarrow \sigma_2^- \in \widehat{\Sigma}^I$ holds, leading to two sub-cases:
  - $\{[\sigma_1]\}_{\sigma_2^+} \in \delta_\Sigma \wedge \sigma_2^- \in \widehat{\Sigma}^I$, then $\{[\sigma_1]\}_{\sigma_2^+}$ has been computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute $\{[\sigma_1]\}_{\sigma_2^+}$ is (6) and this means that both σ_1 and σ_2^+ shall belong to $\Sigma \cup \delta_\Sigma$, i.e. rule (12) is not needed to compute σ_1;
  - $\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \wedge \sigma_2^- \in \widehat{\Sigma}^I$: by Theorem C.4 we have that $\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \wedge \sigma_2^- \in \widehat{\Sigma}^I \Leftrightarrow \{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \wedge \mathtt{r}(\sigma_2^-, \Sigma)$, so the first two hypotheses of rule (34) hold in Σ, and, even more so, they hold in $\Sigma \cup \delta_\Sigma$ ($\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \cup \delta_\Sigma \wedge \mathtt{r}(\sigma_2^-, \Sigma \cup \delta_\Sigma)$). Let us now investigate what happens of the third hypothesis of rule (34):
    - if $\overline{\mathtt{r}(\sigma_1, \Sigma \cup \delta_\Sigma)}$ then by Theorem C.4 it holds that $\sigma_1 \notin \widehat{\Sigma \cup \delta_\Sigma}^I$ and, being $\delta_\Sigma \subseteq \widehat{\Sigma}^I$, it also holds that $\sigma_1 \notin \widehat{\Sigma}^I$, leading to $\mathtt{r}(\sigma_1, \Sigma)$ (by Theorem C.4). So now we have that $\{[\sigma_1]\}_{\sigma_2^+} \in \Sigma \wedge \mathtt{r}(\sigma_2^-, \Sigma) \wedge \overline{\mathtt{r}(\sigma_1, \Sigma)}$, absurdly enabling rule (34) in Σ;
    - if $\mathtt{r}(\sigma_1, \Sigma \cup \delta_\Sigma)$ then by Theorem C.4 it holds that $\sigma_1 \in \widehat{\Sigma \cup \delta_\Sigma}^I$ and, being $\delta_\Sigma \subseteq \widehat{\Sigma}^I$, it also holds that $\sigma_1 \in \widehat{\Sigma}^I$, and it means that σ_1 can be built from Σ by means of rules (1) and (2)–(8), i.e. rule (12) is not needed to compute σ_1.

- (13): the same as above, where we replace (12), (34), $\{[\sigma_1]\}_{\sigma_2^+}$ and σ_2^- with (13), (35), $[\{\sigma_1\}]_{\sigma_2^-}$ and σ_2^+, respectively.
- (14): $\sigma^+ \in \Sigma \cup \delta_\Sigma \wedge \sigma^- \in \Sigma \cup \delta_\Sigma$. Let us consider the following four cases separately:
  - $\sigma^+ \in \Sigma \wedge \sigma^- \in \Sigma$: this absurdly enables (36) in Σ (see *Base*);
  - $\sigma^+ \in \delta_\Sigma \wedge \sigma^- \in \delta_\Sigma$: σ^+ and σ^- computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute both σ^+ and σ^- is (8) and this means that σ belongs to $\Sigma \cup \delta_\Sigma$, i.e. rule (14) is not needed to compute σ;
  - $\sigma^+ \in \Sigma \wedge \sigma^- \in \delta_\Sigma$: σ^- computed by a rule among (2)–(8), because of the hypotheses about the construction of δ_Σ. The only rule, among (2)–(8), able to compute σ^- is (8) and this means that σ belong to $\Sigma \cup \delta_\Sigma$, i.e. rule (14) is not needed to compute σ;
  - $\sigma^+ \in \delta_\Sigma \wedge \sigma^- \in \Sigma$: the same as above, where we exchange σ^+ with σ^-.

Theorem C.8 *Given a finite set of terms* $\Sigma \subseteq \mathcal{T}$*, and* $\overline{\Sigma}$ *defined by* (17)–(25),

$$\textit{if rules } (26)\text{–}(36) \textit{ do not hold in } \Sigma \Rightarrow \Sigma \equiv \overline{\Sigma}$$

Proof. By absurdly assuming that $\Sigma \not\equiv \overline{\Sigma}$, we have

$$\Sigma \not\equiv \overline{\Sigma} \Rightarrow (\exists\, \sigma \in \Sigma \mid \sigma \notin \overline{\Sigma}) \vee (\exists\, \overline{\sigma} \in \overline{\Sigma} \mid \overline{\sigma} \notin \Sigma)$$

Let us begin to prove that $\exists\, \sigma \in \Sigma \mid \sigma \notin \overline{\Sigma}$ leads to absurd. The proof is carried out for each syntactic form of $\sigma \in \Sigma$:

- $\sigma \in \mathcal{A}$ ($\sigma = a$): our absurd hypothesis gives $a \notin \overline{\Sigma}$ and, by rule (17) ($a \in \overline{\Sigma} \Leftrightarrow a \in \widehat{\Sigma}$), we would have $a \notin \widehat{\Sigma}$, leading us to violate closure rule (1) ($a \in \Sigma \;\Rightarrow\; a \in \widehat{\Sigma}$), so the σ we are looking for cannot be an atom.
- $\sigma = \mathrm{suc}(\sigma_1)$: if such a term would belong to Σ, it would enable rule (31), violating the hypothesis.
- $\sigma = (\sigma_1, \sigma_2)$: if such a term would belong to Σ, it would enable rule (32), violating the hypothesis.
- $\sigma = \{\sigma_1\}_{\sigma_2}$: our absurd hypothesis gives $\sigma \notin \overline{\Sigma}$ and, by rule (20) ($\{\sigma_1\}_{\sigma_2} \in \overline{\Sigma} \Leftrightarrow \sigma_2 \notin \widehat{\Sigma}$), we would have $\sigma_2 \in \widehat{\Sigma}$. On the other hand, since no reduction rules are enabled, in particular do not hold the preconditions of rule (33), i.e. $\overline{\mathtt{r}(\sigma_2, \Sigma)}$. Then, by Theorem C.5 ($\overline{\mathtt{r}(\sigma_2, \Sigma)} \Leftrightarrow \sigma_2 \notin \widehat{\Sigma}^I$), we would have $\sigma_2 \in \widehat{\Sigma} \setminus \widehat{\Sigma}^I$, i.e. a violation of the thesis of Theorem C.7.
- $\sigma = \mathrm{H}(\sigma_1)$: our absurd hypothesis gives $\sigma \notin \overline{\Sigma}$ and, by rule (21) ($\mathrm{H}(\sigma_1) \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}$), we would have $\sigma_1 \in \widehat{\Sigma}$. On the other hand, since no reduction rules are enabled, in particular do not hold the preconditions of rule (26), i.e. $\overline{\mathtt{r}(\sigma_1, \Sigma)}$. Then, by Theorem C.5 ($\overline{\mathtt{r}(\sigma_1, \Sigma)} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}^I$), we would have $\sigma_1 \in \widehat{\Sigma} \setminus \widehat{\Sigma}^I$, i.e. a violation of Theorem C.7.
- $\sigma = \{[\sigma_1]\}_{\sigma_2^+}$: our absurd hypothesis gives $\sigma \notin \overline{\Sigma}$ and, by rule (22) ($\{[\sigma_1]\}_{\sigma_2^+} \in \overline{\Sigma} \Leftrightarrow \sigma_2^+ \notin \widehat{\Sigma} \vee \sigma_1 \notin \widehat{\Sigma}$), we would have $\sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_1 \in \widehat{\Sigma}$. On the other hand, since no reduction rules are enabled, in particular do not hold preconditions of rules (27) and (34), i.e. $\overline{\mathtt{r}(\sigma_1, \Sigma_i)} \wedge \overline{\mathtt{r}(\sigma_2^+, \Sigma_i)} \wedge \overline{\mathtt{r}(\sigma_2^-, \Sigma_i)} \wedge \overline{\neg \mathtt{r}(\sigma_1, \Sigma_i)}$. The resulting boolean expression is

  $$\sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_1 \in \widehat{\Sigma} \wedge \left(\overline{\mathtt{r}(\sigma_1, \Sigma)} \vee \overline{\mathtt{r}(\sigma_2^+, \Sigma)}\right) \wedge \left(\overline{\mathtt{r}(\sigma_2^-, \Sigma)} \vee \mathtt{r}(\sigma_1, \Sigma)\right)$$

  By means of Theorem C.5, we replace $\overline{\mathtt{r}(\sigma_1, \Sigma)}$, $\overline{\mathtt{r}(\sigma_2^+, \Sigma)}$, $\overline{\mathtt{r}(\sigma_2^-, \Sigma)}$, and $\mathtt{r}(\sigma_1, \Sigma)$, respectively with $\sigma_1 \notin \widehat{\Sigma}^I$, $\sigma_2^+ \notin \widehat{\Sigma}^I$, $\sigma_2^- \notin \widehat{\Sigma}^I$, and $\sigma_1 \in \widehat{\Sigma}^I$, and obtain:

  $$\sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_1 \in \widehat{\Sigma} \wedge \left(\sigma_1 \notin \widehat{\Sigma}^I \vee \sigma_2^+ \notin \widehat{\Sigma}^I\right) \wedge \left(\sigma_2^- \notin \widehat{\Sigma}^I \vee \sigma_1 \in \widehat{\Sigma}^I\right)$$

  i.e.

  $$\left(\sigma_1 \in \widehat{\Sigma} \wedge \sigma_1 \notin \widehat{\Sigma}^I \wedge \sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_2^- \notin \widehat{\Sigma}^I\right) \vee \left(\sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_2^+ \notin \widehat{\Sigma}^I \wedge \sigma_1 \in \widehat{\Sigma} \wedge \left(\sigma_2^- \notin \widehat{\Sigma}^I \vee \sigma_1 \in \widehat{\Sigma}^I\right)\right),$$

  where in the two or-ed sub-expressions, we respectively, find $\sigma_1 \in \widehat{\Sigma} \wedge \sigma_1 \notin \widehat{\Sigma}^I$ and $\sigma_2^+ \in \widehat{\Sigma} \wedge \sigma_2^+ \notin \widehat{\Sigma}^I$, which violate Theorem C.7, and make *false* the whole expression.
- $\sigma = [\{\sigma_1\}]_{\sigma_2^-}$: the same as above, where we replace (22), (27), (34), $\{[\sigma_1]\}_{\sigma_2^+}$, σ_2^+ and σ_2^- with (23), (28), (35), $[\{\sigma_1\}]_{\sigma_2^-}$, σ_2^- and σ_2^+, respectively.

- $\sigma = \sigma_1^+$: our absurd hypothesis gives $\sigma \notin \overline{\Sigma}$ and, by rule (24) ($\sigma_1^+ \in \overline{\Sigma} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}$), we would have $\sigma_1 \in \widehat{\Sigma}$. On the other hand, since no reduction rules are enabled, in particular do not hold the preconditions of rule (29) i.e. $\overline{\mathtt{r}(\sigma_1, \Sigma)}$. Then, by Theorem C.5 ($\overline{\mathtt{r}(\sigma_1, \Sigma)} \Leftrightarrow \sigma_1 \notin \widehat{\Sigma}^I$), we would have $\sigma_1 \in \widehat{\Sigma} \setminus \widehat{\Sigma}^I$, i.e. a violation of Theorem C.7.
- $\sigma = \sigma_1^-$: the same as above, where we replace (24), (29), and σ_1^+ with (25), (30), and σ_1^-, respectively.

Let us now prove that $\exists\ \overline{\sigma} \in \overline{\Sigma} \mid \overline{\sigma} \notin \Sigma$ leads to absurd.
Because of $\overline{\sigma} \in \overline{\Sigma}$, Theorem C.1 ($\overline{\sigma} \in \overline{\Sigma} \Leftrightarrow \overline{\sigma}$ cannot have been built by rules (2)–(8) from $\widehat{\Sigma}$) holds, and, being $\Sigma \subseteq \widehat{\Sigma}$, even more so, it holds in Σ, preventing us from building $\overline{\sigma}$ from Σ by means of rules (2)–(8).
By definition of $\overline{\Sigma}$, we have $\overline{\Sigma} \subseteq \widehat{\Sigma}$, and, since $\overline{\sigma} \in \overline{\Sigma}$, we have $\overline{\sigma} \in \widehat{\Sigma}$. Moreover, hypotheses of Theorem C.7 hold here, so we have that $\widehat{\Sigma} \equiv \widehat{\Sigma}^I$, leading to $\overline{\sigma} \in \widehat{\Sigma}^I$. This means that $\overline{\sigma}$ can be computed from Σ by means of closure rules (1) and (2)–(8) only (definition C.1). On the other hand, our absurd hypothesis $\overline{\sigma} \notin \Sigma$ prevents the use of closure rule (1) during the computation of $\overline{\sigma}$ from Σ, leaving us closure rules (2)–(8) only for computing $\overline{\sigma}$ from Σ: this makes the thesis of Theorem C.1 *false*, leading to make its hypotheses false. Since they are a subset of our hypotheses, also the latter ones become false. □

Proposition 3.2 *Given a finite set of terms* Σ*, there exists a finite reduction of* Σ *such that:*

$$\Sigma = \Sigma_0 \xrightarrow{R_0} \Sigma_1 \cdots \Sigma_{k-1} \xrightarrow{R_{k-1}} \overline{\Sigma}.$$

Proof. First of all we prove that each sequence of one-step reductions is finite, then we prove that the last computed set is $\overline{\Sigma}$.
Each rule in (26)–(36) in particular needs that a certain term σ belong to Σ_i . Anyway, when all the premises are satisfied, each one add to Σ_{i+1} a sub-term of σ, i.e. a simpler term.
Moreover, rules (26)–(33) and (36) always remove σ from Σ_{i+1}, while (34) and (35) do not do it, but the presence of the subterm of σ (σ_1 in this case) in Σ_{i+1} prevents the rule from being enabled again on the same term.
In practice almost all rules replace a term σ with a simpler one, while a couple of them does not remove σ, but prevent the rule itself from entering an infinite loop on the same term.
Then, being Σ finite, just a finite number of reduction steps can be done.
Above we have proved that always we reach a Σ_k where no reduction rules are enabled, then Theorem C.8 holds, leading us to state that $\Sigma_k \equiv \overline{\Sigma}$. Theorem C.6 states that in $\overline{\Sigma}$ no reduction rules are enabled. □

Proof of Theorem 3.3 Since Theorem 3.1 guarantees the closure preservation after each reduction step, by means of Proposition 3.2 we have that $\widehat{\overline{\Sigma}} \equiv \widehat{\Sigma}$. □
Proof of Theorem 3.4 Trivially from Proposition 3.2. □
Proof of Theorem 3.5 Trivially from Theorem 3.4, Theorem 3.3 and Theorem C.5. □

Appendix D: Extension to associative and commutative operators

In order to be sound, the extensions to the knowledge representation method described in Sect. 4 must not invalidate any of the theorems and propositions introduced in Sect. 3.
Figure 10 depicts the relationships among the theorems, corollaries and propositions proved in Appendix C. There, the main theorems and propositions, discussed in the main text, are highlighted by means of a grey box, and double-stroked boxes mark the theorems whose proof also depends on one or more properties of rules (1)–(14), (17)–(25), and (26)–(36). Finally, dependencies of a theorem on another are denoted by an arrow going from the dependent to the dependee.
By means of Fig. 10, it is possible to locate the theorems whose proof is influenced by an extension of the above-mentioned rules, identify the requirements they pose on the extension itself, and confirm that they are still satisfied after the extension. In particular:

Theorem C.1 requires that the predicate of each rule within (17)–(25) and their extensions must invalidate the corresponding $\mathcal{I}$ rules (2)–(8) and their extensions. Moreover, from the negation of the premises of rules (2)–(8), it must be possible to derive the truth of the right-hand side of the corresponding rules in (17)–(25). For a commutative operator, (45) implies $\sigma_1 \notin \widehat{\Sigma} \ \vee\ \sigma_2 \notin \widehat{\Sigma}$, which invalidates the premises of (44). On the other hand, the negation of the premises of (44) implies the truth of the right-hand side of (45).

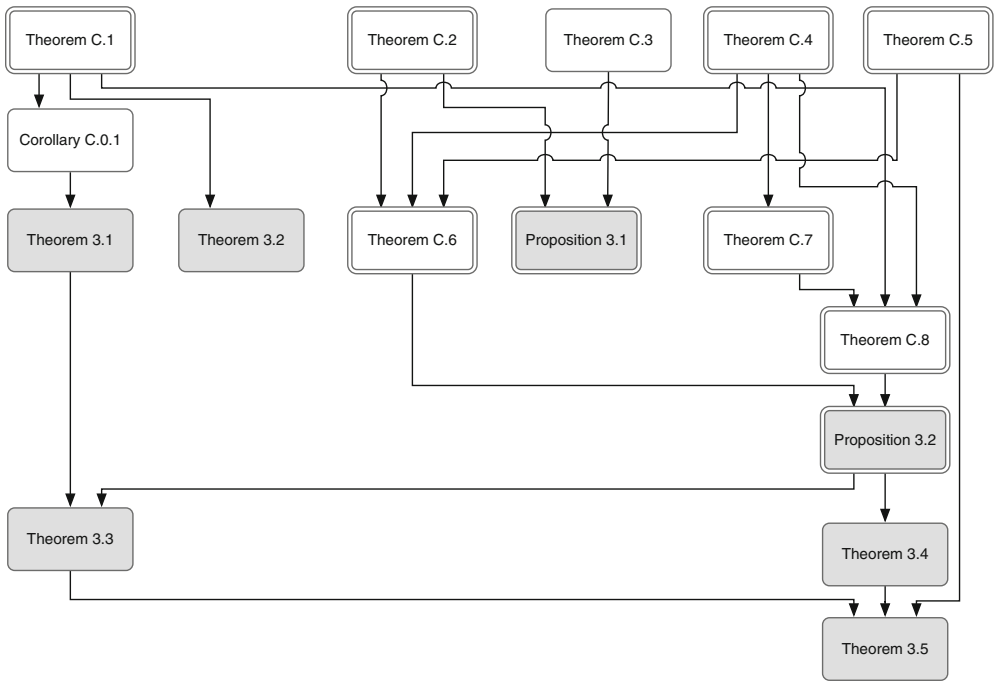

Fig. 10. Dependencies of the theorems, corollaries and propositions proved in Appendix C, among themselves and with respect to rules (1)–(14), (17)–(25), (26)–(36), and their extensions

For an associative operator, (48) implies $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \notin \widehat{\Sigma} \vee \odot(\sigma_{21}, \ldots, \sigma_{2n}) \notin \widehat{\Sigma}$, hence it cannot be $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \wedge \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}$ as required by (47). On the other hand, the negation of the premises of (47) yields $\odot(\sigma_1, \ldots, \sigma_i) \notin \widehat{\Sigma} \vee \odot(\sigma_{i+1}, \ldots, \sigma_n) \notin \widehat{\Sigma}$ for any i, which coincides with the right-hand side of (48).
When considering an operator which is both commutative and associative, (51) implies $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \notin \widehat{\Sigma} \vee \odot(\sigma_{21}, \ldots, \sigma_{2n}) \notin \widehat{\Sigma}$, hence it cannot be $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \wedge \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}$ as required by (50). On the other hand, the negation of the premises of (50) yields $\odot(S_1) \notin \widehat{\Sigma} \vee \odot(S_2) \notin \widehat{\Sigma}$ for any choice of $S_1, S_2 \mid S_1 \neq \emptyset \wedge S_2 \neq \emptyset \wedge S_1 \cap S_2 = \emptyset \wedge S_1 \cup S_2 = \{\sigma_1, \ldots, \sigma_n\}$, which makes the right-hand side of (51) true.

Theorem C.2 proceeds by induction, with a separate case for each possible syntactic form of the term being considered. From the similarities between the proofs of Theorems C.2 and C.4, it is possible to conclude that Theorem C.4 is a stronger form of Theorem C.2:

$$\mathtt{r}(\sigma, S) \Leftrightarrow \sigma \in \widehat{S}^I \quad \text{by Theorem C.4} \tag{88}$$

$$S \subseteq \widehat{\Sigma} \Rightarrow \widehat{S} \subseteq \widehat{\Sigma} \quad \text{hypothesis of C.2, property of closures} \tag{89}$$

$$\widehat{S}^I \subseteq \widehat{S} \quad \text{restriction of a closure} \tag{90}$$

$$\mathtt{r}(\sigma, S) \Rightarrow \sigma \in \widehat{S}^I \Rightarrow \sigma \in \widehat{S} \Rightarrow \sigma \in \widehat{\Sigma} \quad \text{thesis of C.2} \tag{91}$$

Therefore, the same proof technique that will be described for Theorem C.4 applies to Theorem C.2 as well.

Proposition 3.1 considers the reduction rules (26)–(36) one at a time. The rules added by the extension are similar, for example, to (26) because they all have $\Sigma_O = \emptyset$. The proof requires that, for each added rule:

- In the preconditions of the rule, $\mathtt{r}(\cdot, \cdot)$ is always invoked on a proper subterm of the term in Σ_i being considered.
- By using an $\mathcal{I}$ rule, it must be possible to rebuild the terms which were removed from Σ_i by the application of the rule.

By inspection, it can be seen that it is possible to rebuild the term $\sigma_1 \odot \sigma_2$ removed by (46) through rule (44). In the same way, it is possible to rebuild $\odot(\sigma_1, \ldots, \sigma_n)$ removed by (49) through rule (47) by letting $\sigma_{11} = \sigma_1, \ldots, \sigma_{1m} = \sigma_i, \sigma_{21} = \sigma_{i+1}, \ldots$ With respect to rule (52), the removed term is in canonical form, because it belonged to Σ_i, and will be reintroduced in canonical form by (50) due to the forced canonicalisation on its right-hand side, with an appropriate assignment of $\sigma_1, \ldots, \sigma_n$ to $\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots$ This consideration also justifies the addition of the canonicalisation to rule (50) with respect to (47).

Theorem C.4 proceeds by induction, with a separate case for each possible syntactic form of the term being considered. For a commutative operator $\odot$, the following must be proved:

- $\mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \Rightarrow \mathtt{r}(\sigma_1 \odot \sigma_2, \Sigma)$
- Assuming that $\sigma_1 \odot \sigma_2 \notin \Sigma$, $\mathtt{r}(\sigma_1 \odot \sigma_2, \Sigma) \Rightarrow \mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma)$
- From the inductive hypothesis $\mathtt{r}(\sigma_1, \Sigma) \Leftrightarrow \sigma_1 \in \widehat{\Sigma}^I \ \wedge \ \mathtt{r}(\sigma_2, S) \Leftrightarrow \sigma_2 \in \widehat{\Sigma}^I$, *it must be* $\mathtt{r}(\sigma_1 \odot \sigma_2, \Sigma) \Rightarrow \mathtt{r}(\sigma_1, \Sigma) \ \wedge \ \mathtt{r}(\sigma_2, \Sigma) \Rightarrow \sigma_1 \in \widehat{\Sigma}^I \ \wedge \ \sigma_2 \in \widehat{\Sigma}^I \Rightarrow \sigma_1 \odot \sigma_2 \in \widehat{\Sigma}^I$

All these claims can easily be proved by examining the extension to the $\mathtt{r}(\cdot, \cdot)$ function presented in Sect. 4.3 and the additional closure rule (44), which concurs to the construction of $\widehat{\Sigma}^I$ being an $\mathcal{I}$ rule.
When associative operators are considered, the requirements become:

- $\mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}), \Sigma) \ \wedge \ \mathtt{r}(\odot(\sigma_{21}, \ldots, \sigma_{2n}), \Sigma) \Rightarrow \mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}), \Sigma)$
- Assuming that $\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}) \notin \Sigma$,

  $$\mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}), \Sigma) \Rightarrow \mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}), \Sigma) \ \wedge \ \mathtt{r}(\odot(\sigma_{21}, \ldots, \sigma_{2n}), \Sigma)$$

- From the inductive hypothesis $\mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}), \Sigma) \Leftrightarrow \odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma}^I \ \wedge \ \mathtt{r}(\odot(\sigma_{21}, \ldots, \sigma_{2n}), \Sigma) \Leftrightarrow \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}^I$, it must be

  $$\mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}), \Sigma) \Leftrightarrow \mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}), \Sigma) \ \wedge \ \mathtt{r}(\odot(\sigma_{21}, \ldots, \sigma_{2n}), \Sigma) \Leftrightarrow \odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma}^I \ \wedge \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}^I \Leftrightarrow \odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}^I$$

Again, these claims can be proved by examining the extension to the $\mathtt{r}(\cdot, \cdot)$ function presented in Sect. 4.4 and the additional closure rule (47).
If the operator is both associative and commutative, then the above requirements must be satisfied *regardless of the order* of the operands $\sigma_{11}, \ldots, \sigma_{1m}$ and $\sigma_{21}, \ldots, \sigma_{2n}$. This justifies the introduction of subsets S_1 and S_2 in the extension to the $\mathtt{r}(\cdot, \cdot)$ function made in this case.

Theorem C.5 requires the same properties as Theorem C.4.

Theorem C.6 uses Theorem C.2 to state that $\sigma_1 \notin \widehat{\Sigma} \ \vee \ \sigma_2 \notin \widehat{\Sigma} \Rightarrow \overline{\mathtt{r}(\sigma_1, \overline{\Sigma})} \ \vee \ \overline{\mathtt{r}(\sigma_2, \overline{\Sigma})}$.
Then, all the reduction rules in (26)–(36) and their extension which can possibly apply to the commutative operator $\sigma_1 \odot \sigma_2$ must require $\mathtt{r}(\sigma_1, \overline{\Sigma}) \ \wedge \ \mathtt{r}(\sigma_2, \overline{\Sigma})$, and therefore be inapplicable. This constraint is trivially satisfied by rule (46).
For an associative operator, rule (49) must be inapplicable if

$$\forall\, i \in [1, n-1] \mid \overline{\mathtt{r}(\odot(\sigma_1, \ldots, \sigma_i), \overline{\Sigma})} \ \vee \ \overline{\mathtt{r}(\odot(\sigma_{i+1}, \ldots, \sigma_n), \overline{\Sigma})},$$

but the preconditions of the rule satisfy this constraint. If the operator is also commutative, the same statement must be true *regardless of* the order in which the operands $\sigma_1, \ldots, \sigma_n$ are considered: this justifies the introduction of subsets S_1 and S_2 in rule (52).

TheoremC.7 uses several properties of $\mathcal{E}$ rules, but no additional rules of this kind are added by the extension.

TheoremC.8 proceeds by absurd with a separate path for each syntactic form of the term σ begin considered.
For a commutative operator $\odot$ the proof requires the following steps:

- If $\sigma_1 \odot \sigma_2 \in \Sigma$ *but* $\sigma_1 \odot \sigma_2 \notin \overline{\Sigma}$, then it must be $\sigma_1 \in \widehat{\Sigma} \ \wedge \ \sigma_2 \in \widehat{\Sigma}$. This is guaranteed by rule (45).
- The inapplicability of the corresponding reduction rule (46) implies that $\overline{\mathtt{r}(\sigma_1, \Sigma)} \ \vee \ \overline{\mathtt{r}(\sigma_2, \Sigma)}$.
- By Theorem C.4, this implies $\sigma_1 \notin \widehat{\Sigma}^I \ \vee \ \sigma_2 \notin \widehat{\Sigma}^I$. Then, it would be $\sigma_1 \in \widehat{\Sigma} \backslash \widehat{\Sigma}^I \ \vee \ \sigma_1 \in \widehat{\Sigma} \backslash \widehat{\Sigma}^I$, but this is absurd because it violates Theorem C.7.

For an associative operator, the steps become:

- If $\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}) \in \Sigma$ but $\odot(\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}) \notin \overline{\Sigma}$, then it must be $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \ \wedge \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma}$. This is guaranteed by rule (48).
- The inapplicability of the corresponding reduction rule (49) implies that

  $$\overline{\mathtt{r}(\odot(\sigma_{11}, \ldots, \sigma_{1m}), \Sigma)} \ \vee \ \overline{\mathtt{r}(\odot(\sigma_{21}, \ldots, \sigma_{2n}), \Sigma)}.$$

- By Theorem C.4, this implies $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \notin \widehat{\Sigma}^I \ \vee \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \notin \widehat{\Sigma}^I$. Then, it would be $\odot(\sigma_{11}, \ldots, \sigma_{1m}) \in \widehat{\Sigma} \backslash \widehat{\Sigma}^I \ \vee \ \odot(\sigma_{21}, \ldots, \sigma_{2n}) \in \widehat{\Sigma} \backslash \widehat{\Sigma}^I$,but this is absurd because it violates Theorem C.7.

If the associative operator is also commutative, the steps must be valid regardless of the order in which $\sigma_{11}, \ldots, \sigma_{1m}, \sigma_{21}, \ldots, \sigma_{2n}$ are considered. This is guaranteed by the introduction of subsets S_1 *and* S_2 in (51) and (52).

Proposition 3.2 could be invalidated only by additional reduction rules with a non-empty Σ_O, but this is not the case for any of the extended rules.

References

[AG99] Abadi M, Gordon AD (1999) A calculus for cryptographic protocols: The spi calculus. Inf Comput 148(1):1–70. doi:10.1006/inco.1998.2740

[AL00] Amadio RM, Lugiez D (2000) On the reachability problem in cryptographic protocols. In: Proceedings of the 11th international conference on concurrency theory (CONCUR 2000), vol 1877 of Lecture Notes in Computer Science, pp 380–394, Springer, Berlin

[BB02] Boreale M, Buscemi MG (2002) A framework for the analysis of security protocols. In: Proceedings of the 13th International Conference on Concurrency Theory (CONCUR 2002). Lecture Notes in Computer Science, vol 2421. Springer, Berlin, pp 483–498

[BDNP02] Boreale M, De Nicola R, Pugliese R (2002) Proof techniques for cryptographic processes. SIAM J Comput 31(3):947–986. doi:10.1137/S0097539700377864

[Bla01] Blanchet B (2001) An efficient cryptographic protocol verifier based on prolog rules. In: Proceedings of the 14th IEEE computer security foundations workshop (CSFW-14), Cape Breton. IEEE Computer Society, Washington, pp 82–96

[BMV05] Basin D, Mödersheim S, Viganò L (2005) OFMC: a symbolic model checker for security protocols. Int J Inf Secur 4(3):181–208, Special issue on ESORICS 2003

[Bor01] Boreale M (2001) Symbolic trace analysis of cryptographic protocols. In: Proceedings of the 28th international colloquium on automata, languages, and programming (ICALP 2001). Lecture Notes in Computer Science, vol 2076. Springer, Berlin, pp 667–681

[CDSV03a] Cibrario Bertolotti I, Durante L, Sisto R, Valenzano A (2003) Introducing commutative and associative operators in cryptographic protocol analysis. In: Proceedings of the 23rd IFIP international conference on formal techniques for networked and distributed systems (FORTE 2003). Lecture Notes in Computer Science, vol 2767. Springer, Berlin, pp 224–239

[CDSV03b] Cibrario Bertolotti I, Durante L, Sisto R, Valenzano A (2003) A new knowledge representation strategy for cryptographic protocol analysis. In: Proceedings of tools and algoritms for the construction and analysis of systems (TACAS 2003). Lecture Notes in Computer Science, vol 2619. Springer, Berlin, pp 284–298

[CJM98] Clarke EM, Jha S, Marrero W (1998) Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In: Proceedings of the IFIP working conference on programming concepts and methods (PROCOMET 1998). Chapman & Hall, London, pp 87–106

[CJM00] Clarke EM, Jha S, Marrero W (2000) Verifying security protocols with Brutus. ACM Trans Softw Eng Methods 9(4):443–487. doi:10.1145/363516.363528

[CKRT03] Chevalier Y, Küsters R, Rusinowitch M, Turuani M (2003) An NP decision procedure for protocol insecurity with XOR. In: Proceedings of the 18th IEEE symposium on logic in computer science (LICS 2003). IEEE Computer Society Press, Washington, pp 261–170. doi:10.1109/LICS.2003.1210066

[CLS03] Comon-Lundh H, Shmatikov V (2003) Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In: Proceedings of the 18th IEEE symposium on logic in computer science (LICS 2003). IEEE Computer Society Press, Washington, pp 271–280. doi:10.1109/LICS.2003.1210067

[DH76] Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654

[DSV03] Durante L, Sisto R, Valenzano A (2003) Automatic testing equivalence verification of spi calculus specifications. ACM Trans Softw Eng Methodology 12(2):222–284. doi:10.1145/941566.941570

[DY83] Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208

[FA01] Fiore M, Abadi M (2001) Computing symbolic models for verifying cryptographic protocols. In: Proceedings of the 14th IEEE computer security foundations workshop (CSFW 2001). IEEE Computer Society Press, Washington, pp 160–173. doi:10.1109/CSFW.2001.930144

[Hui99] Huima A (1999) Efficient infinite-state analysis of security protocols. In: Proceedings of the FLOC workshop on formal methods and security protocols

[Low96] Lowe G (1996) Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In: Proceedings of tools and algoritms for the construction and analysis of systems (TACAS 1996). Lecture Notes in Computer Science, vol 1055. Springer, Berlin, pp 147–166

[Low97] Lowe G (1997) Casper: a compiler for the analysis of security protocols. In: Proceedings of the 10th IEEE computer security foundations workshop (CSFW 1997). IEEE Computer Society Press, Washington, pp 18–30. doi:10.1109/CSFW.1997.596779

[Low99] Lowe G (1999) Towards a completeness result for model checking security protocols. J Comput Sec 7(2–3):89–146

[McA93] McAllester D (1993) Automatic recognition of tractability in inference relations. J ACM 40(2):284–303. doi:10.1145/151261.151265

[MCJ97] Marrero W, Clarke EM, Jha S (1997) A model checker for authentication protocols. In: Proceedings of the DIMACS workshop on design and formal verification of security protocols

[MN02] Meadows C, Narendran P (2002) A unification algorithm for the group Diffie–Hellman protocol. In: Proceedings of WITS'02

[Mon99] Monniaux D (1999) Abstracting cryptographic protocols with tree automata. In: Proceedings of the 6th international static analysis symposium (SAS 1999). Lecture Notes in Computer Science, vol 1694. Springer, Berlin, pp 149–163

[MPW92] Milner R, Parrow J, Walker D (1992) A calculus of mobile processes, parts I and II. Inf Comput 100(1):1–77. doi:10.1016/0890-5401(92)90008-4

[MS01] Millen JK, Shmatikov V (2001) Constraint solving for bounded-process cryptographic protocol analysis. In: Proceedings of the 8th ACM conference on computer and communications security (CCS 2001). ACM Press, New York, pp 166–175. doi:10.1145/501983.502007

[MS03] Millen JK, Shmatikov V (2003) Symbolic protocol analysis with products and Diffie–Hellman exponentiation. In: Proceedings of the 16th IEEE computer security foundations workshop (CSFW 2003). IEEE Computer Society Press, Washington, pp 47–61. doi:10.1109/CSFW.2003.1212704

[Pau98] Paulson LC (1998) The inductive approach to verifying cryptographic protocols. J Comput Sec 6:85–128

[Pra65] Prawitz D (1965) Natural deduction: a proof-theoretical study. Almqvist & Wiksell, Stockholm

[RT01] Rusinowitch M, Turuani M (2001) Protocol insecurity with finite number of sessions is NP-complete. In: Proceedings of the 14th IEEE computer security foundations workshop (CSFW 2001). IEEE Computer Society Press, Washington, pp 174–187. doi:10.1109/CSFW.2001.930145

[Sch98] Schneider S (1998) Verifying authentication protocols in CSP. IEEE Trans Softw Eng 24(9):741–758. doi:10.1109/32.713329

Received 2 November 2005
Accepted in revised form 22 March 2008 by C. B. Jones
Published online 29 April 2008