Automated verification of equivalence properties of cryptographic protocols*

Rohit Chadha[1], Vincent Cheval[2], Ştefan Ciobâcă[3], and Steve Kremer[4]

[1] University of Missouri
[2] University of Kent
[3] University "Alexandru Ioan Cuza"
[4] INRIA Nancy - Grand-Est

Abstract. Indistinguishability properties are essential in formal verification of cryptographic protocols. They are needed to model anonymity properties, strong versions of confidentiality and resistance against offline guessing attacks, which can be conveniently modeled using process equivalences. We present a novel procedure to verify equivalence properties for a bounded number of sessions of cryptographic protocols. As in the applied pi-calculus, our protocol specification language is parametrized by a first-order sorted term signature and an equational theory which allows formalization of algebraic properties of cryptographic primitives. Our procedure is able to verify trace equivalence for determinate cryptographic protocols. On determinate protocols, trace equivalence coincides with observational equivalence which can therefore be automatically verified for such processes. When protocols are not determinate our procedure can be used for both under- and over-approximations of trace equivalence, which proved successful on examples. The procedure can handle a large set of cryptographic primitives, namely those that can be modeled by an optimally reducing convergent rewrite system. The procedure is based on a fully abstract modelling of the traces of a bounded number of sessions of the protocols into first-order Horn clauses on which a dedicated resolution procedure is used to decide equivalence properties. We have shown that our procedure terminates for the class of subterm convergent equational theories. Moreover, the procedure has been implemented in a prototype tool A-KiSs (Active Knowledge in Security Protocols) and has been effectively tested on examples. Some of the examples were outside the scope of existing tools, including checking anonymity of an electronic voting protocol.

1 Introduction

Cryptographic protocols are distributed programs that rely on the use of cryptography to secure electronic transactions such as those that arise in electronic commerce and wireless communication. They are also being applied in new domains such as in Internet voting—legally binding political elections in Estonia, Norway and Switzerland offered the possibility for Internet voting in 2011. This has led to increasing demands on the complexity of desired security properties, leading to more complex cryptographic protocols. Given the socio-economic-political consequences and the history of incorrect design of cryptographic protocols, the need for formal proofs of correctness of protocols is of great importance and has been widely recognized. Formal reasoning about cryptographic protocols is challenging as one has to reason against all potentially malicious behavior—all communication between protocol participants is assumed to be under the control of an adversary.

In order to make the task of formal analysis amenable to automation, usually the assumption of back-box cryptography and unbounded computational power on the part of the adversary is made. This adversarial model is often called the Dolev-Yao model as it is derived from the positions that Dolev and Yao took in their seminal paper [41]. It has proved extremely successful, and there are several automated tools [17,9,34,43] that can automatically check trace-properties such as (weak forms of) confidentiality and authentication. While these trace-based properties are certainly important, many crucial security properties can only be expressed in terms of *indistinguishability* (or equivalence). They include strong flavors of confidentiality

* Parts of this work has been done while the first, third and fourth author were affiliated to LSV, CNRS & Inria & ENS Cachan and the second author was affiliated to LORIA, CNRS & Inria & Université de Lorraine.

[18]; resistance to guessing attacks in password based protocols [14]; and anonymity properties in private authentication [3], electronic voting [39,13], vehicular networks [35,36] and RFID protocols [7,22]. More generally, indistinguishability allows to model security by the means of ideal systems, which are correct by construction [4,37]. Indistinguishability properties of cryptographic protocols are naturally modeled by the means of *observational* and *testing equivalences* in cryptographic extensions of process calculi, e.g., the spi [4] and the applied-pi calculus [2]. While we have good tools for automated verification of trace properties, the situation is different for indistinguishability properties. This paper is an attempt to address this concern.

State-of-the-art Many results have been obtained in the restricted case of a pure eavesdropper, i.e., a passive adversary: for *static equivalence* many decidability results have been shown [1,32,10] and exact [15,30] and approximate [19] tools exist for a variety of cryptographic primitives. In the case we consider indistinguishability in the presence of an active adversary, who can interact in an arbitrary way with honest participants less results are known. Hüttel [47] showed undecidability of observational equivalence in the spi calculus, even for the finite control fragment, as well as decidability for the finite, i.e., replication-free, fragment of the spi calculus. The decidability result however only holds for a fixed set of cryptographic primitives and does not yield a practical algorithm. Current results [19,53] allow to approximate observational equivalence for an unbounded number of sessions. However, this approximation does not suffice to conclude for many applications, *e.g.*, [39,7]. Our approach overcomes these limitations for some applications in [39]. We still cannot conclude for the e-passport example in [7], albeit for a different reason: our procedure does not currently handle else branches in protocols.

Symbolic bisimulations have also been devised for the spi [21,20,54] and applied pi calculus [38,49] to avoid unbounded branching due to adversary inputs. However, only [38,54] and [21] yield a decision procedure, but again only approximating observational equivalence. The results of [38] have been further refined to show a decision procedure on a restricted class of *simple* processes [33]. In particular they rely on a procedure deciding the equivalence of constraint systems, introduced by Baudet [14], for the special case of verifying the existence of guessing attacks. Baudet's procedure allows arbitrary cryptographic primitives that can be modeled as a subterm convergent rewrite systems [1]. An alternate procedure achieving the same goal was proposed by Chevalier and Rusinowitch [26]. However, both procedures are highly non-deterministic and do not yield a reasonable algorithm which could be implemented. Therefore, Cheval *et al.* [24] have designed a new procedure and a prototype tool to decide the equivalence of constraint systems, but only for a fixed set of primitives. Tools have also been implemented for checking testing equivalence [42], open bisimulation [54] and trace equivalence [25] for a bounded number of sessions but only a limited set of primitives. One may note that [25] is the only decision procedure to consider negative tests, i.e., else branches, which are crucial in several case studies [7,3].

Our contribution In this paper we introduce a new procedure for verifying equivalence properties for processes specified in a cryptographic process calculus (without replication). The messages in the calculus are modeled as terms equipped with an equational theory, similar to the applied pi calculus. Our main contributions are as follows.

- Our procedure checks for two equivalences which over- and under-approximate the standard notion of trace equivalence $\approx_t$ for cryptographic protocols: the under-approximation can be used to prove protocols correct while the over-approximation can be used to rule out incorrect protocols.
- Cortier and Delaune have shown in [33] that observational equivalence coincides with $\approx_t$ for the class of *determinate* processes. They also give a decision procedure for a strict sub-class of determinate processes, namely, *simple* processes. We show that the coarser relation coincides with $\approx_t$, and thus our procedure can be used to verify observational equivalence for the whole class of determinate processes.
- A novelty of our procedure is that it is based on a *fully abstract* modeling of symbolic traces for a *bounded* number of sessions in *first-order Horn clauses*. This is in contrast to the constraint-solving techniques employed by Tiu *et al.* [54], Cheval *et al.* [24,25], Baudet [14] and Chevalier et al. [26] for verifying under-approximations of observational equivalence. Techniques based on Horn clauses have been extensively used, e.g., by Blanchet [17], Weidenbach [55] and Goubault [46], in the case of an unbounded number

of sessions. Affeldt and Comon [5] faithfully encode a bounded protocol into Horn clauses with rigid variables. Of these tools, only Blanchet [17] can verify an equivalence property, which happens to be an under-approximation of observational equivalence. Horn clause modeling of an unbounded number of sessions of security protocols may allow false attacks. On the other hand, we have proven our modeling of a bounded number of sessions to be precise.

- Our modelling is fully abstract for arbitrary cryptographic primitives that can be modeled as a convergent rewrite system which has the *finite variant property* [31,44]. This allows us to handle a larger class of cryptographic primitives than [54,24,25,14,26,19]. Following our work, the recent work by Santiago et al. [53] also provides support for rewrite systems which have the *finite variant property*. They additionally cover associative-commutative theories, even though their experimental evaluation suggests that these theories yield frequent non termination problems for the associative-commutative theories. Moreover, they only provide support for a restricted class of processes. We were also able to show termination of our procedure for the sub-class of subterm convergent rewrite systems. Please note that static equivalence is undecidable even for the class of optimally reducing convergent rewrite systems [6]. Optimally reducing convergent rewrite theories generalize subterm convergent rewrite systems, while maintaining the finite variant property. Moreover, even though our termination proof does not apply, our tool terminated on specific protocols whose cryptographic primitives can be modeled as a convergent rewrite theories. These included the electronic voting protocols by Okamoto [52] and Fujioka et al. [45] which use trapdoor commitment and blind signature respectively.
- Our procedure is implemented in the AKiSs (Active Knowledge in Security protocols) prototype tool and we used it among others to successfully prove anonymity in an electronic voting protocol [45]. For this electronic voting protocol, this is the first automated proof.

An extended abstract of the paper [23] authored by R. Chadha, S. Kremer and Ş. Ciobâcă appeared in European Symposium of Programming. In addition to the proofs that were not present in the extended abstract, this paper also contains the proof of termination for subterm convergent rewrite theories. The proof of termination is due to V. Cheval.

2 Preliminaries

We recall some standard definitions and establish some notations that we shall be using throughout the paper.

2.1 Terms

Let $\mathcal{F}$ be a signature, i.e., a finite set of function symbols and let ar be a function which assigns to each function symbol a natural number. Given a function symbol $f \in \mathcal{F}$, we say $ar(f) \in \mathbb{N}$ is the arity of f. A function symbol of arity 0 is called a *constant*. Given a set of *atoms* $\mathcal{A}$ and a signature $\mathcal{F}$, we denote by $\mathcal{T}_{\mathcal{F},\mathcal{A}}$ the set of terms built inductively from $\mathcal{A}$ by applying functions symbols in $\mathcal{F}$. Given sets of atoms $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$, we denote the set $\mathcal{T}_{\mathcal{F},\cup_{1\leq i\leq n}\mathcal{A}_i}$ by $\mathcal{T}_{\mathcal{F},\mathcal{A}_1,\mathcal{A}_2,\ldots\mathcal{A}_n}$. We assume that we have the following countably infinite pairwise disjoint sets: a set $\mathcal{N}$ of *private names*, a set $\mathcal{M}$ of *public names*, a set $\mathcal{C}$ of public *channel names*, a set $\mathcal{W}$ of *parameters*, and a set $\mathcal{X}$ of message variables. Intuitively, elements of the set $\mathcal{N}$ represent nonces generated by honest principals of a protocol, elements of $\mathcal{M}$ represent nonces available both to the adversary and to the honest participants and elements of $\mathcal{C}$ represent names of public channel (e.g. the name of a public network). Elements of $\mathcal{W}$ are pointers used by the adversary to refer to messages output by the honest participants in a protocol. We fix an enumeration $w_1, w_2, \ldots$ of the elements of $\mathcal{W}$. We let x, y, z range over $\mathcal{X}$. We also define the following sets.

Definition 1. *The set $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M},\mathcal{W},\mathcal{X}}$, denoted* Terms*, is the set of all terms, the set $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M}}$, denoted* Messages*, is the set of* messages *and the set $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M},\mathcal{X}}$, denoted* SMessages*, is the set of* symbolic messages*.*

If t is a term, we denote by $vars(t)$ the set of variables appearing in t, by $names(t)$ the set of names (public or private) appearing in t and $\mathsf{st}(t)$ the set of all subterms of t. The functions $vars$, $names$ and st are extended as expected to sequences and sets of terms. A *position* is a string of positive natural numbers and ϵ denotes the empty string. The set $pos(t)$ of positions of a term t is defined as usual [11]. If $p \in pos(t)$ then $t|_p$ is the subterm of t at position p.

Example 1. Consider the signature $\mathcal{F} = \{\mathsf{enc}, \mathsf{dec}, \mathsf{pair}, \mathsf{fst}, \mathsf{snd}\}$ where $ar(\mathsf{enc}) = 3$, $ar(\mathsf{dec}) = ar(\mathsf{pair}) = 2$ and $ar(\mathsf{fst}) = ar(\mathsf{snd}) = 1$. The term $t = \mathsf{pair}(\mathsf{enc}(a, k_1, r_1), \mathsf{enc}(b, k_2, r_2))$ models the pair of the encryptions of public names a and b with keys k_1, resp. k_2 and randomness r_1, resp. r_2. The set of positions $pos(t) = \{\epsilon, 1, 11, 12, 13, 2, 21, 22, 23\}$ and $t_\epsilon = t$, $t_1 = \mathsf{enc}(a, k_1, r_1)$ and $t_{23} = r_2$.

Substitutions A substitution is a partial function $\sigma : \mathcal{W} \cup \mathcal{X} \to \mathsf{Terms}$. We only consider substitutions which map elements of $\mathcal{W}$ to elements in $\mathsf{Messages}$ and elements of $\mathcal{X}$ to elements of $\mathsf{SMessages}$. The domain of σ shall be denoted by $dom(\sigma)$. For our purposes, we only consider substitutions with finite domains. We let $range(\sigma) = \{\, \sigma(u) \in \mathcal{T} \mid u \in dom(\sigma) \,\}$. If $dom(\sigma) = \{\, u_1, u_2, \ldots, u_n \,\}$ and $t_i = \sigma(u_i)$ for each $1 \leq i \leq n$ then we shall write σ as $\{\, u_1 \mapsto t_1, \ldots, u_n \mapsto t_n \,\}$. σ is said to be *ground* if $range(\sigma) \subseteq \mathsf{Messages}$. The notation $names(\sigma)$ will denote the set $names(range(\sigma))$. As usual, a substitution extends homomorphically to a function $\mathsf{apply}_\sigma : \mathsf{Terms} \to \mathsf{Terms}$ obtained by "applying" σ. Given $t \in \mathsf{Terms}$, we denote $\mathsf{apply}_\sigma(t)$ by $t\sigma$. If σ is a substitution and $X \subseteq dom(\sigma)$, we denoted by $\sigma[X]$ the substitution whose domain is restricted to X. Given two substitutions σ and τ, the substitution obtained by *composing* σ and τ, denoted $\sigma\tau$, is the unique substitution such that $\sigma\tau(x) = \tau(\sigma(x))$.

2.2 Rewriting and unification

Two terms s and t are (syntactically) *unifiable* if there exists a substitution σ such that $s\sigma = t\sigma$. We denote by mgu a function which associates to any two unifiable terms s and t a most general unifier σ of s and t such that $\sigma = \sigma[vars(s,t)]$. It is well known [11,12] that for any two unifiable terms s and t, there is a most general unifier, unique up to variable renaming.

A rewrite system R is a set of rewrite rules of the form $\ell \to r$ where $\ell, r \in \mathsf{Terms}$, $names(l, r) = \emptyset$ and $vars(r) \subseteq (\ell)$. A term t can be rewritten in one step to u, denoted $t \to_\mathsf{R} u$, if there exist a position $p \in pos(t)$, a rule $\ell \to r$ in R and a substitution σ such that $t|_p = \ell\sigma$ and u is obtained from t by replacing the subterm $t|_p$ by $r\sigma$. $\to_\mathsf{R}^*$ denotes the transitive and reflexive closure of $\to_\mathsf{R}$. A rewrite system is said to be *confluent* if for any t, t_1, t_2 such that $t \to_\mathsf{R}^* t_1$ and $t \to_\mathsf{R}^* t_2$ there exists u such that $t_1 \to_\mathsf{R}^* u$ and $t_2 \to_\mathsf{R}^* u$. A rewrite system is said to be *terminating* if it does not admit any infinite sequence $t_0 \to_\mathsf{R} t_1 \to_\mathsf{R} t_2 \to_\mathsf{R} \ldots$. It is said to be *convergent* if it is both confluent and terminating. In a convergent rewrite system R, for every term t there is a unique term t' such that $t \to_R^* t'$ and there is no u such that $t' \to_\mathsf{R} u$. t' is said to be the *normal form* of t. We denote by $t{\downarrow_\mathsf{R}}$ the normal form of the term t. Two terms s and t are said to be equal modulo R, written $s =_\mathsf{R} t$, if $s{\downarrow_\mathsf{R}} = t{\downarrow_\mathsf{R}}$. Given a substitution σ we denote by $\sigma{\downarrow}$ a substitution such that $dom(\sigma{\downarrow}) = dom(\sigma)$ and for all $u \in dom(\sigma)$ $\sigma{\downarrow}(u) = \sigma(u){\downarrow}$.

Example 2. Continuing Example 1, consider the rewrite system $\mathsf{R} = \{\mathsf{dec}(\mathsf{enc}(x, y, z), y) \to x, \mathsf{fst}(\mathsf{pair}(x, y)) \to x, \mathsf{snd}(\mathsf{pair}(x, y)) \to y\}$. The first rewrite rule models that a message can be decrypted, provided decryption uses the same key (represented by variable y) as encryption. The two last rules model projection of the first and second component of a pair. Then we have that $t = \mathsf{fst}(\mathsf{pair}(\mathsf{dec}(\mathsf{enc}(a, k, r), k), b)) \to_\mathsf{R} \mathsf{fst}(\mathsf{pair}(a, b)) \to_\mathsf{R} a = t{\downarrow_\mathsf{R}}$.

We recall the notions of *optimally reducing* [51] and *subterm convergent* [1] rewrite systems.

Definition 2. *A rewrite system* R *is said to be* optimally reducing *if for any* $\ell \to r \in \mathsf{R}$ *and any substitution* θ *such that all proper subterms of* $\ell\theta$ *are in normal form, we have that* $r\theta$ *is in normal form.*

Definition 3. *A rewrite system* R *is said to be* subterm convergent *if* R *is convergent and for each rule* $\ell \to r \in \mathsf{R}$*, we have that either* $r \in \mathsf{st}(\ell)$ *or* r *is a constant.*

We immediately note that any subterm convergent rewrite system is also optimally reducing.

Example 3. The rewrite system $\mathsf{R} = \{\mathsf{dec}(\mathsf{enc}(x, y, z), y) \to x, \mathsf{fst}(\mathsf{pair}(x, y)) \to x, \mathsf{snd}(\mathsf{pair}(x, y)) \to y\}$ given in Example 2 is subterm convergent. We shall examples of convergent rewrite systems that are not subterm convergent when we discuss the case studies on electronic voting in Section 6.2.

Remark 1. When R is clear from the context or unimportant we will simply drop the subscript R in $\to_{\mathsf{R}}$ and $\downarrow_{\mathsf{R}}$.

2.3 The finite variant property

Given a convergent rewrite system, we now define the notion of complete set of variants, which was introduced by Common-Lundh and Delaune [31]. Our notion is slightly stronger than the notion defined in [31] and was first introduced in [44].

Definition 4. *A set of substitutions* $\mathrm{variants}(t_1, \ldots, t_k)$ *is called a* complete set of variants *of terms* $t_1, \ldots, t_k$ *if for any substitution* ω *there exist* $\sigma \in \mathrm{variants}(t_1, \ldots, t_k)$ *and a substitution* τ *such that for all* $1 \leq j \leq k$ *we have that* $\omega[vars(t_j)]\downarrow = (\sigma\downarrow\tau)[vars(t_j)]$ *and* $(t_j\omega)\downarrow = (t_j\sigma)\downarrow\tau$.

Intuitively, the set of variants of a term t represents a pre-computation such that any instance of t in normal form is *syntactically* equal to an instance of $t\sigma\downarrow$ for some variant σ of t, without the need to apply further rewrite steps. A rewrite system has the *finite variant property* if for any sequence of terms a finite, complete set of variants exists.

Example 4. Consider the rewrite system introduced in Example 2 and let $t = \mathsf{dec}(\mathsf{fst}(x), y)$. We have that $\mathrm{variants}\, t = \{\emptyset, \sigma_1, \sigma_2\}$ where $\emptyset$ denotes the identity substitution and

$$\sigma_1 = \{x \mapsto \mathsf{pair}(z_1, z_2)\}, \text{ and}$$
$$\sigma_2 = \{x \mapsto \mathsf{pair}(\mathsf{enc}(z_1, y, z_2), z_3)\}$$

Intuitively, the substitution $\emptyset$ covers the cases where both the decryption and projection may fail, σ_1 corresponds to the situation where the projection succeeds, but decryption may fail, and σ_2 accounts for the situations where both projection and encryption succeed. Note that the case where projection fails and encryption succeeds is not possible.

In [28], Ciobâcă presents an algorithm for computing such complete sets of variants which is correct whenever the rewrite system is *optimally reducing* [51]. Optimally reducing rewrite systems include subterm convergent systems [1] (and hence the classical Dolev Yao theories for encryption, signatures and hash functions), as well as a theory for modeling blind signatures [48]. Moreover, complete sets of variants can be used to perform unification modulo R [44,28].

Definition 5. *Given sets of terms* $\{s_i\}_{i\in I}$ *and* $\{t_i\}_{i\in I}$, *let* $X = vars(\{s_i, t_i\}_{i\in I})$. *A set of substitutions* $\mathrm{mgu}_{\mathsf{R}}(\{s_i \stackrel{?}{=} t_i\}_{i\in I})$ *is called a* complete set of unifiers modulo R *of the system of equations* $\{s_i \stackrel{?}{=} t_i\}_{i\in I}$ *if each of the following holds:*

1. $dom(\sigma) \subseteq vars(X)$ *for each* $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_i \stackrel{?}{=} t_i\}_{i\in I})$
2. $s_i\sigma =_{\mathsf{R}} t_i\sigma$ *for each* $i \in I$ *and for each* $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_i \stackrel{?}{=} t_i\}_{i\in I})$.
3. *For any substitution* θ *such that* $s_i\theta =_{\mathsf{R}} t_i\theta$ *for every* $i \in I$, *there exists a substitution* $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_i \stackrel{?}{=} t_i\}_{i\in I})$ *and a substitution* τ *with* $\theta[X] =_{\mathsf{R}} (\sigma\tau)[X]$.

For singleton systems, we also write $\mathrm{mgu}_{\mathsf{R}}(s, t)$ instead of $\mathrm{mgu}_{\mathsf{R}}(\{s, t\})$.

For the remaining of the paper, we assume that the rewrite system is convergent and has the finite variant property.

2.4 Frames, deducibility and static equivalence

Recall that we have fixed an enumeration $w_1, w_2, \ldots$ of the elements of the set $\mathcal{W}$. As in [2], we will use the notion of a *frame* to represent messages which have been recorded by the attacker.

Definition 6. *A* frame φ *is a substitution* $\{w_1 \mapsto t_1, \ldots, w_n \mapsto t_n\}$ *where* $t_i \in \mathsf{Messages}$ *(*$1 \leq i \leq n$*).*

Intuitively, w_i in a frame points to the the i message recorded by the attacker in a protocol run. Note that in our definition, every frame with $|dom(\varphi)| = n$ has $dom(\varphi) = \{w_1, \ldots, w_n\}$. We denote the set of all frames as Frames. The adversary can use the public names as well as recorded messages to construct new messages. This is modeled as the deducibility relation.

Definition 7. *Any term in* $\mathcal{T}_{\mathcal{F},\mathcal{M},\mathcal{W}}$ *is said to be a recipe. We say that* a message t is deducible from φ with a recipe r *(written as* $\varphi \vdash^r t$*) if* $t \in \mathsf{Messages}$ *and* $r\varphi =_{\mathsf{R}} t$*. We write* $\mathsf{Recipes}$ *for the set* $\mathcal{T}_{\mathcal{F},\mathcal{M},\mathcal{W}}$*.*

Intuitively, the recipe r tells how the attacker can construct the messages t from the recorded messages. Note that the same term t can be constructed using different recipes. A frame $\varphi' = \{w_1 \mapsto t'_1, \ldots, w_m \mapsto t'_m\}$ *extends* a frame $\varphi = \{w_1 \mapsto t_1, \ldots, w_n \mapsto t_n\}$ if $m \geq n$ and if $t'_i = t_i$ for all $1 \leq i \leq n$. It is easy to see that if φ' extends φ and if $\varphi \vdash^r t$ then $\varphi' \vdash^r t$.

Example 5. Consider the signature $\mathcal{F}$ and the rewrite system R in Example 2. Let $\varphi = \{w_1 \mapsto \mathsf{enc}(s, k, r), w_2 \mapsto k\}$ where $s, k, r \in \mathcal{N}$ are private names. Then we have that $\varphi \vdash^{\mathsf{dec}(w_1,w_2)} s$. Note that $\mathsf{dec}(w_1, k) \notin \mathsf{Recipes}$ as $k \in \mathcal{N}$. If we had that $s \in \mathcal{M}$ we would also have that $\varphi \vdash^s s$ reflecting that public names are always deducible.

We now recall *static equivalence* of frames [2] to capture indistinguishability of frames. Recall that two terms can be indistinguishable to an attacker even if the two terms are distinct. For example, 0 encrypted using a symmetric key unknown to the attacker and 1 encrypted using the same key are indistinguishable to the attacker. Thus, instead of checking of direct equality between messages, the attacker can perform a series of tests to distinguish between two frames. This is the intuition behind the following definition:

Definition 8. *Let* $r_1, r_2 \in \mathsf{Recipes}$*. A* test $r_1 \stackrel{?}{=} r_2$ holds *in a frame* φ *(written* $(r_1 = r_2)\varphi$*) if* $\varphi \vdash^{r_1} t$ *and* $\varphi \vdash^{r_2} t$ *for some* t*,* i.e.*,* r_1 *and* r_2 *are recipes for the same term in* φ*.*

Frames φ_1 *and* φ_2 *are* statically equivalent *(written* $\varphi_1 \approx_s \varphi_2$*) iff for all* $r_1, r_2 \in \mathsf{Recipes}$ *we have that* $(r_1 = r_2)\varphi_1$ *iff* $(r_1 = r_2)\varphi_2$*.*

Example 6. Let $a, b \in \mathcal{M}$ and $r, k, k' \in \mathcal{N}$. We have that $\{w_1 \mapsto \mathsf{enc}(a, k, r), w_2 \mapsto k\} \not\approx_s \{w_1 \mapsto \mathsf{enc}(b, k, r), w_2 \mapsto k\}$ because the test $(dec(w_1, w_2) \stackrel{?}{=} a)$ distinguishes the two frames. However, $\{w_1 \mapsto \mathsf{enc}(a, k, r), w_2 \mapsto k'\} \approx_s \{w_1 \mapsto \mathsf{enc}(b, k, r), w_2 \mapsto k'\}$.

3 A cryptographic process calculus

We shall assume that cryptographic protocols are modeled using a simple process calculus which has similarities with the applied pi-calculus [2]. The applied pi-calculus has shown to be useful for specifying and verifying cryptographic protocols; there are tools that automate verification of protocols in this model [17]. We shall further restrict our attention to the finite, *i.e.*, replication-free fragment of applied pi-calculus. This restriction is important because observational equivalence becomes undecidable with replication [47]. However, even with this restriction, one can model a bounded number of protocol instances.

In this section we define our process calculus. We begin by defining its syntax.

Syntax Recall that we have fixed a first-order signature $\mathcal{F}$, a set $\mathcal{N}$ of *private names*, $\mathcal{M}$ of *public names*, a set $\mathcal{C}$ of public channel names, a set $\mathcal{W}$ of *parameters*, and a set $\mathcal{X}$ of message variables (see Section 2). The terms of the set $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M},\mathcal{W},\mathcal{X}}$ are also identified modulo a fixed subterm convergent rewrite system R (see Section 2).

We model a bounded number of instances of a cryptographic protocol as a *finite* set of traces. Traces are defined using sequences of *actions* generated by the following grammar (note that here **in** and **out** are fresh symbols not occurring in $\mathcal{F}$):

$$\begin{array}{rll} a ::= & \mathbf{in}(c, x) & \text{receive action} \\ & \mathbf{out}(c, t) & \text{send action} \\ & [s \stackrel{?}{=} t] & \text{test action} \end{array}$$

where $x \in \mathcal{X}, s, t \in \mathsf{SMessages}, c \in \mathcal{C}$. A *trace* T is a sequence of actions $T = a_1.a_2.\ldots.a_n$. As usual, a receive action $\mathbf{in}(c, x)$ acts as a binding construct for the variable x. We assume the usual definitions of free and bound variables for traces. We also assume that each variable is bound at most once. A trace is *ground* if it does not contain any free variables. The set of ground traces shall be represented as $\mathsf{GndTraces}$. We also assume the usual definition of a name occurring in a trace.

A *process* P is defined to be a set of traces $P = \{T_1, \ldots, T_n\}$. We say that a process is ground if all of its traces are ground. We identify traces with singleton processes.

Remark 2. Contrary to the applied pi calculus [2] we do not have an ν operator for generating new names: as we only consider a finite number of sessions we can simply use private names in $\mathcal{N}$. We have also not explicitly included the parallel operator | and the choice operator +. One could include these and generate the corresponding set of traces. Thus, there is no loss in expressivity. However, we note that an explicit enumeration of the traces can result in an exponential number of traces.

Semantics The semantics of a process is defined using the semantics of its traces. The semantics of a trace is given in terms of a labeled transition system T. We assume that all interactions between protocol participants are mediated by the adversary. The labeled transition system records the interaction of the protocol participants with the adversary. The set of labels of T is defined using the set $\mathsf{Recipes}$. Recall that the set $\mathsf{Recipes}$ is the set $\mathcal{T}_{\mathcal{F},\mathcal{M},\mathcal{W}}$ (see Section 2). The set of labels, Labels, is

$$\mathsf{Labels} = \{\, \mathbf{in}(c, r), \mathbf{out}(c), \mathbf{test} \mid r \in \mathsf{Recipes}, c \in \mathcal{C} \,\}.$$

The labeled transition system T is a subset of $(\mathsf{GndTraces} \times \mathsf{Frames}) \times \mathsf{Labels} \times (\mathsf{GndTraces} \times \mathsf{Frames})$ and we shall write $(T, \varphi) \xrightarrow{\ell} (T', \varphi')$ whenever $((T, \varphi), \ell, (T', \varphi')) \in \mathsf{T}$. The frame in the transition system is used to record the messages that the protocol participants have sent in the past. The relation $\xrightarrow{\ell}$ is defined as follows:

$$\textsc{Receive}\ \frac{\varphi \vdash^r t}{(\mathbf{in}(c, x).T, \varphi) \xrightarrow{\mathbf{in}(c,r)} (T\{x \mapsto t\}, \varphi)}$$

$$\textsc{Send}\ \frac{}{(\mathbf{out}(c, t).T, \varphi) \xrightarrow{\mathbf{out}(c)} (T, \varphi \cup \{w_{|dom(\varphi)|+1} \mapsto t\})}$$

$$\textsc{Test}\ \frac{s =_{\mathsf{R}} t}{([s \stackrel{?}{=} t].T, \varphi) \xrightarrow{\mathbf{test}} (T, \varphi)}$$

The label $\mathbf{in}(c, r)$ indicates a message sent by the adversary over the channel c and r is the recipe that adversary uses to create this message. The label $\mathbf{out}(c)$ indicates a message sent over the public channel c and the transition rule Send records the message sent in the frame. Finally, the rule Test is an internal action.

As usual, we shall write $(T_0,\varphi_0) \xrightarrow{\ell_1,\ldots,\ell_n} (T_n,\varphi_n)$ when $(T_0,\varphi_0) \xrightarrow{\ell_1} (T_1,\varphi_1)\ldots \xrightarrow{\ell_n} (T_n,\varphi_n)$ and we say that $\ell_1\ldots\ell_n$ is a *run* of (T_0,φ_0). We shall write $(T,\varphi) \stackrel{\ell}{\Rightarrow} (T',\varphi')$ when either $(T,\varphi) \xrightarrow{\mathbf{test}^*,\ell,\mathbf{test}^*} (T',\varphi')$ and $\ell \neq \mathbf{test}$ or $(T,\varphi) \xrightarrow{\mathbf{test}^*} (T',\varphi')$ and $\ell = \mathbf{test}$, where $\mathbf{test}^*$ denotes an arbitrary number of **test** actions. We write $(T,\varphi) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T_n,\varphi_n)$ when $(T,\varphi) \stackrel{\ell_1}{\Rightarrow} (T_1,\varphi_1) \stackrel{\ell_2}{\Rightarrow} \ldots \stackrel{\ell_n}{\Rightarrow} (T_n,\varphi_n)$. If $P = \{T_1,\ldots,T_m\}$ is a process, we write $(P,\varphi) \xrightarrow{\ell_1,\ldots,\ell_n} (T',\varphi')$ (resp. $\stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T',\varphi')$) if there exists a trace $T \in P$ such that $(T,\varphi) \xrightarrow{\ell_1,\ldots,\ell_n} (T',\varphi')$ (resp. $(T,\varphi) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T',\varphi')$).

Process equivalences In this section we will define different flavors of trace equivalence which will be useful in this paper. We first recall the standard definition of trace equivalence in cryptographic process algebras.

Definition 9 (Trace equivalence). *A ground process P is said to be* trace-included *in a ground process Q (written $P \sqsubseteq_t Q$) if whenever $(P,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T,\varphi)$ then there exist T',φ' such that $(Q,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T',\varphi')$ and $\varphi \approx_s \varphi'$. Two processes P and Q are* trace-equivalent *(written $P \approx_t Q$) if $P \sqsubseteq_t Q$ and $Q \sqsubseteq_t P$.*

We will also define two other notions of trace equivalence, one coarser and one more fine-grained. The coarser trace equivalence, which we denote by $\approx_{ct}$ is the trace equivalence that can actually be verified by our procedure.

Definition 10 (Coarse trace equivalence). *Given ground processes P and Q, we say that $P \sqsubseteq_{ct} Q$ if whenever $(P,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T,\varphi)$ and $(r_1 = r_2)\varphi$ for some recipes r_1, r_2 then there exist T',φ' such that $(Q,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T',\varphi')$ and $(r_1 = r_2)\varphi'$. We say that $P \approx_{ct} Q$ if $P \sqsubseteq_{ct} Q$ and $Q \sqsubseteq_{ct} P$.*

The following example illustrates the difference between $\approx_t$ and $\approx_{ct}$.

Example 7. Let P and Q be the ground processes defined as follows:

$$P = \{\, \mathbf{out}(c,a).\mathbf{out}(c,a) \,\} \text{ and}$$
$$Q = \{\, \mathbf{out}(c,a).\mathbf{out}(c,a),\ \mathbf{out}(c,a).\mathbf{out}(c,b) \,\}$$

Clearly $P \sqsubseteq_{ct} Q$. Observe also that $Q \sqsubseteq_{ct} P$. Indeed, only trivial equalities hold on the frame $\{w_1 \mapsto a, w_2 \mapsto b\}$, and therefore these also hold on $\{w_1 \mapsto a, w_2 \mapsto a\}$. Thus, we have that $P \approx_{ct} Q$ while $P \not\approx_t Q$.

We will however show that these two notions coincide for a class of *determinate processes*. In the context of the applied pi calculus determinate processes were previously studied by Cortier and Delaune in [33].

Definition 11 (Determinate process). *We say that a ground process P is determinate if whenever $(P,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T,\varphi)$ and $(P,\emptyset) \stackrel{\ell_1,\ldots,\ell_n}{\Longrightarrow} (T',\varphi')$ then $\varphi \approx_s \varphi'$.*

Intuitively, determinate processes are processes in which the adversary's static knowledge at any instance is completely determined by its past interaction with the protocol participants. The following is immediate from the definition.

Proposition 1. *A ground trace, i.e, a ground process consisting of single trace, is determinate.*

As already mentioned above, it was demonstrated in [33] that trace equivalence coincides with observational equivalence for determinate processes. We show that $\approx_t$ and $\approx_{ct}$ also coincide for this class of processes.

Theorem 1. *If P and Q are ground processes then $P \approx_t Q$ implies $P \approx_{ct} Q$. Furthermore if P and Q are determinate, then $P \approx_{ct} Q$ implies $P \approx_t Q$.*

Proof. Let P and Q be determinate processes.

($\Rightarrow$) Follows immediately from definition of $\approx_t$ and $\approx_{ct}$.

($\Leftarrow$) We need to show that $P \approx_{ct} Q$ implies $P \approx_t Q$. We proceed by contradiction. Suppose that $P \approx_{ct} Q$ and $P \not\approx_t Q$. We suppose $P \not\sqsubseteq_t Q$ (the case of $Q \not\sqsubseteq_t P$ being symmetric). As $P \not\sqsubseteq_t Q$ we have that there exist $\ell_1, \ldots, \ell_n$, T, φ, such that $(P, \emptyset) \xRightarrow{\ell_1,\ldots,\ell_n} (T, \varphi)$ and

1. either there exist no φ', T' such that $(Q, \emptyset) \xRightarrow{\ell_1,\ldots,\ell_n} (T', \varphi')$,
2. or for all φ', T' such that $(Q, \emptyset) \xRightarrow{\ell_1,\ldots,\ell_n} (T', \varphi')$ we have that $\varphi \not\approx_s \varphi'$.

In the first case we have that $P \not\approx_{ct} Q$, contradicting our hypothesis. In the second case, as $\varphi \not\approx_s \varphi'$, there exist r, r' such that $(r = r')\varphi$ and $(r \neq r')\varphi'$ (or vice-versa, the other case is symmetric). As $P \sqsubseteq_{ct} Q$, we have that there exist T'', φ'' such that $(Q, \emptyset) \xRightarrow{\ell_1,\ldots,\ell_n} (T'', \varphi'')$ and $(r = r')\varphi''$. As Q is determinate, we have that $\varphi' \approx_s \varphi''$. This yields a contradiction, as $(r \neq r')\varphi'$ and $(r = r')\varphi''$ would imply $\varphi' \not\approx_s \varphi''$. □

Additionally, we introduce a more fine-grained notion of trace equivalence, denoted $\approx_{ft}$.

Definition 12 (fine-grained trace equivalence). *Given ground processes P and Q, we say that $P \sqsubseteq_{ft} Q$ whenever for all trace $T \in P$ there exists a trace $T' \in Q$ such that $T \approx_t T'$. We say that $P \approx_{ft} Q$ if $P \sqsubseteq_{ft} Q$ and $Q \sqsubseteq_{ft} P$.*

It follows directly from the definition that $\approx_{ft} \subset \approx_t$. The difference between these two relations is illustrated by the following example.

Example 8. Let P and Q be ground processes defined as follows:

$$P = \{\ \mathbf{out}(c, enc(a,k)).\mathbf{out}(c, enc(b,k)).\mathbf{in}(c,x).[x = enc(a,k)].\mathbf{out}(c,k),\\ \quad \mathbf{out}(c, enc(a,k)).\mathbf{out}(c, enc(b,k)).\mathbf{in}(c,x).[x = enc(b,k)].\mathbf{out}(c,k)\}$$

$$Q = \{\ \mathbf{out}(c, enc(a,k)).\mathbf{out}(c, enc(b,k)).\mathbf{in}(c,x).[x = enc(dec(x,k),k)].\mathbf{out}(c,k)\}$$

where $k \in \mathcal{N}$ is a private name and a, b are constants. The test $x = enc(dec(x,k),k)$ simply checks whether x is an encryption with key k. It is not difficult to see that $P \approx_t Q$ but $P \not\approx_{ft} Q$.

As already mentioned our procedure is able check $\approx_{ct}$ which coincides with $\approx_t$ when processes are determinate. In the case where processes are not determinate we can use our procedure to check $\approx_{ct}$ and $\approx_{ft}$ in order to over- and under-approximate $\approx_t$. Indeed, as traces are determinate processes a procedure for checking $\approx_{ct}$ can be used to verify $\approx_{ft}$.

4 Modeling traces as Horn clauses

Our decision procedure is based on a fully abstract modelling of a trace in first-order Horn clauses. We give the details of this modelling; we start by giving some definitions that we need for defining the predicates used in the logic.

Symbolic labels and symbolic runs We define the set of *symbolic labels* as

$$\mathsf{SLabels} = \{\mathbf{in}(c,t), \mathbf{out}(c), \mathbf{test} \mid t \in \mathsf{SMessages}, c \in \mathcal{C}\}$$

and the set of *symbolic runs* as the set of finite sequences of symbolic labels (see Figure 1). The empty sequence is denoted by ϵ. Sometimes we simply write (empty space) for ϵ. Intuitively, a symbolic label stands for a set of possible labels, and a symbolic run stands for a set of possible runs of the protocol.

Symbolic Recipes We assume a set $\mathcal{Y}$ of *recipe variables* disjoint from $\mathcal{X}$. The set of terms $\mathcal{T}_{\mathcal{F},\mathcal{M},\mathcal{W},\mathcal{Y}}$ shall be called *symbolic recipes* and denoted by $\mathsf{SRecipes}$. We use capital letters X, Y, Z to range over $\mathcal{Y}$. Intuitively, a symbolic recipe stands for a set of recipes.

We can extend the definition of substitutions to include variables from $\mathcal{Y}$ in its domain. However, we only consider substitutions that map variables in $\mathcal{Y}$ to $\mathsf{SRecipes}$. A ground substitution must map variables in $\mathcal{Y}$ to $\mathsf{Recipes}$. The notion of most general unifiers is extended to symbolic recipes as expected.

Predicates The predicates used in our modelling and the semantics of the predicates are given in Figure 1. The predicates are interpreted over a triple– a trace T, a frame φ and a substitution σ. We consider four kinds of predicates, all of which have a symbolic run as an argument. Intuitively, the *reachability predicate* r_w says that each run represented by w is possible, i.e., does not block due to a test that fails. The intruder knowledge predicate $\mathsf{k}_w(R,t)$ says that whenever a run represented by w happens, the (symbolic) message t can be constructed by the intruder using the (symbolic) recipe R. The identity predicate $\mathsf{i}_w(R,R')$ says that whenever the (symbolic) run w is executed, the (symbolic) recipes R and R' are recipes for the same (symbolic) term. The reachable identity predicate $\mathsf{ri}_w(R,R')$ is a short form for the conjunction of the predicates r_w and $\mathsf{i}_w(R,R')$.

Formulas and statements We consider first-order formulas built using the above predicates and the usual connectives (conjunction, disjunction, negation, implication, existential and universal quantification). As in the case of predicates, a formula is interpreted over a triple consisting of a trace T, a frame φ and a substitution σ; and the semantics is defined as expected. For ground formulas we do not need the substitution σ and when a formula f is ground we simply write $(T,\varphi) \models f$ to denote that this formula holds for (T,φ). If moreover, $dom(\varphi) = \emptyset$, we simply write $T \models f$ for $(T,\emptyset) \models f$.

Symbolic Runs ($\ell \in \mathsf{SLabels}$):

$u, v, w := \epsilon \mid \ell, w$

Predicates ($w \in \mathsf{SRuns}, R \in \mathsf{SRecipes}, t \in \mathsf{SMessages}$):

r_w	(Reachability predicate)
$\mathsf{k}_w(R,t)$	(Intruder knowledge predicate)
$\mathsf{i}_w(R,R')$	(Identity predicate)
$\mathsf{ri}_w(R,R')$	(Reachable identity predicate)

Semantics ($\ell_i \in \mathsf{SLabels}, R \in \mathsf{SRecipes}, t \in \mathsf{SMessages}, T \in \mathsf{GndTraces}$, $\varphi \in \mathsf{Frames}, \sigma$ a ground substitution):

$(T,\varphi_0,\sigma) \models \mathsf{r}_{\ell_1,\ldots,\ell_n}$	if $(T,\varphi_0) \xrightarrow{L_1} (T_1,\varphi_1) \xrightarrow{L_2} \ldots \xrightarrow{L_n} (T_n,\varphi_n)$ such that $\ell_i\sigma =_{\mathsf{R}} L_i\varphi_{i-1}$ for all $1 \le i \le n$
$(T,\varphi_0,\sigma) \models \mathsf{k}_{\ell_1,\ldots,\ell_n}(R,t)$	if when $(T,\varphi_0) \xrightarrow{L_1} (T_1,\varphi_1) \xrightarrow{L_2} \ldots \xrightarrow{L_n} (T_n,\varphi_n)$ such that $\ell_i\sigma =_{\mathsf{R}} L_i\varphi_{i-1}$ for all $1 \le i \le n$ then $\varphi_n \vdash^{R\sigma} t\sigma$
$(T,\varphi_0,\sigma) \models \mathsf{i}_{\ell_1,\ldots,\ell_n}(R,R')$	if there exists t such that $(T,\varphi_0,\sigma) \models \mathsf{k}_{\ell_1,\ldots,\ell_n}(R,t)$ and $(T,\varphi_0,\sigma) \models \mathsf{k}_{\ell_1,\ldots,\ell_n}(R',t)$
$(T,\varphi_0,\sigma) \models \mathsf{ri}_{\ell_1,\ldots,\ell_i}(R,R')$	if $(T,\varphi_0,\sigma) \models \mathsf{r}_{\ell_1,\ldots,\ell_n}$ and $(T,\varphi_0,\sigma) \models \mathsf{i}_{\ell_1,\ldots,\ell_n}(R,R')$

Fig. 1: Predicates

We now identify a subset of the formulas, which we shall call *statements*. Statements will take the form of Horn clauses, and we shall be mainly concerned with them.

Definition 13. *A* statement *is a Horn clause of the form* $H \Leftarrow B_1,\ldots,B_n$ *where:*

1. $H \in \{\mathsf{r}_{l_1,\ldots,l_k}, \mathsf{k}_{l_1,\ldots,l_k}(R,t), \mathsf{i}_{l_1,\ldots,l_k}(R,R'), \mathsf{ri}_{l_1,\ldots,l_k}(R,R')\}$
2. *For each* $1 \le i \le n, B_i = \mathsf{k}_{l_1,\ldots,l_{j_i}}(X_i,t_i)$

for some $l_1,\ldots,l_k \in \mathsf{SLabels}$, $t \in \mathsf{SMessages}$, $R,R' \in \mathsf{SRecipes}$, $j_i \le k$, $t_1,\ldots,t_n \in \mathsf{SMessages}$ *and* $X_1,\ldots,X_n \in \mathcal{Y}$. *Furthermore* $X_1,\ldots,X_n$ *are distinct variables and if* $H = \mathsf{k}_{\ell_1,\ldots,\ell_k}(R,t)$ *then* $vars(t) \subseteq vars(t_1,\ldots,t_n)$.

We implicitly assume that in a Horn clause all variables are universally quantified. Hence, all statements are closed formulas.

4.1 The set of seed statements

As mentioned above, our decision procedure is based on a fully abstract modelling of a trace in first-order Horn clauses. In this section, given a trace T we will give a set of statements $\mathsf{seed}(T)$ which will serve as a starting point for the modelling. We shall also establish that the set of statements $\mathsf{seed}(T)$ is a sound and (partially) complete abstraction of the trace T. In order to formally define $\mathsf{seed}(T)$, we start by fixing some notational conventions.

Let $T = a_1.a_2.\ldots.a_n$ be a ground trace. We assume w.l.o.g. the following naming conventions:

1. if a_i is a receive action then $a_i = \mathbf{in}(c_i, x_i)$.
2. $x_i \neq x_j$ for any $i \neq j$.
3. if a_i is a send action then $a_i = \mathbf{out}(c_i, t_i)$.
4. if a_i is a test actions then $a_i = [s_i \stackrel{?}{=} t_i]$.

Moreover, for each $1 \leq i \leq n$ let $\ell_i \in \mathsf{SLabels}$ be as follows:

$$\ell_i = \begin{cases} \mathbf{in}(c_i, x_i) & \text{if } a_i = \mathbf{in}(c_i, x_i) \\ \mathbf{out}(c_i) & \text{if } a_i = \mathbf{out}(c_i, t_i) \\ \mathbf{test} & \text{if } a_i = [s_i \stackrel{?}{=} t_i] \end{cases}.$$

For each $0 \leq m \leq n$, let the sets $R(m)$, $S(m)$ and $T(m)$ respectively denote the indices of the receive actions, send actions and test actions amongst $a_1, \ldots, a_m$. Formally,

$$\begin{cases} R(m) = \{i \mid 1 \leq i \leq m, a_i = \mathbf{in}(c_i, x_i)\} \\ S(m) = \{i \mid 1 \leq i \leq m, a_i = \mathbf{out}(c_i, t_i)\} \\ T(m) = \{i \mid 1 \leq i \leq m, a_i = [s_i \stackrel{?}{=} t_i]\} \end{cases}.$$

Given a set of public names $\mathcal{M}_0 \subseteq \mathcal{M}$, the *set of seed statements* associated to T and $\mathcal{M}_0$, denoted $\mathsf{seed}(T, \mathcal{M}_0)$, is defined to be the set of statements given in Figure 2. If $\mathcal{M}_0 = \mathcal{M}$, then $\mathsf{seed}(T, \mathcal{M})$ is said to be the set of seed statements associated to T and in this case we write $\mathsf{seed}(T)$ as a shortcut for $\mathsf{seed}(T, \mathcal{M})$.

Remark 3. Please note that while constructing the set of seed statements, we apply the most general unifier modulo R to all tests. In addition, we also apply finite variants. This allows us to *get rid* of rewriting in our procedure.

We shortly show that the set of seed statements is a sound and (partially) complete modelling of a trace. However, we need one more definition to state this fact.

Definition 14. *Let K be a set of statements. We define $\mathcal{H}(K)$ to be the smallest set of ground terms such that:*

$$\textsc{Simple Consequence}\ \frac{f = \Big(H \Leftarrow B_1, \ldots, B_n\Big) \in K \qquad \sigma \text{ grounding for } f \qquad B_1\sigma \in \mathcal{H}(K) \qquad \ldots \qquad B_n\sigma \in \mathcal{H}(K)}{H\sigma \in \mathcal{H}(K)}$$

$$\textsc{ExtendK}\ \frac{\mathsf{k}_u(R, t) \in \mathcal{H}(K)}{\mathsf{k}_{uv}(R, t) \in \mathcal{H}(K)}$$

(Equivalently, $\mathcal{H}(K)$ is the least Herbrand model of $K \cup \{\mathsf{k}_{\ell_1,\ldots,\ell_{n+1}}(X, x) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_n}(X, x)\}_{n\in\mathbb{N}}$.)

$$\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow} \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)\}_{j\in R(m)}$$
for all $0 \leq m \leq n$
for all $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k\in T(m)})$
for all $\tau \in \mathrm{variants}(\ell_1\sigma, \ldots, \ell_m\sigma)$

$$\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_{|S(m)|}, t_m\sigma\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)\}_{j\in R(m)}$$
for all $m \in S(n)$
for all $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k\in T(m)})$
for all $\tau \in \mathrm{variants}(\ell_1\sigma, \ldots, \ell_m\sigma, t_m\sigma)$

$$\mathsf{k}(c, c) \Leftarrow$$
for all public names $c \in \mathcal{M}_0$

$$\mathsf{k}_{\ell_1,\ldots,\ell_m}(f(Y_1, \ldots, Y_k), f(y_1, \ldots, y_k)\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1,\ldots,\ell_m}(Y_j, y_j\tau\downarrow)\}_{j\in\{1,\ldots,k\}}$$
for all $0 \leq m \leq n$
for all function symbols f of arity k
for all $\tau \in \mathrm{variants}(f(y_1, \ldots, y_k))$.

Fig. 2: Seed statements

We show that as far as reachability predicates and intruder knowledge predicates are concerned, the set $\mathsf{seed}(T)$ is a complete abstraction of a trace.

Theorem 2. *Let T be a ground trace.*

- *(Soundness.) For any statement $f \in \mathsf{seed}(T) \cup \mathcal{H}(\mathsf{seed}(T))$, $T \models f$.*
- *(Completeness.) If $(T, \emptyset) \xrightarrow{L_1,\ldots,L_m} (S, \varphi)$ then:*
  1. *$\mathsf{r}_{L_1\varphi\downarrow,\ldots,L_m\varphi\downarrow} \in \mathcal{H}(\mathsf{seed}(T))$.*
  2. *if $\varphi \vdash^R t$ then $\mathsf{k}_{L_1\varphi\downarrow,\ldots,L_m\varphi\downarrow}(R, t\downarrow) \in \mathcal{H}(\mathsf{seed}(T))$.*

Remark 4. Please note that the set $\mathsf{seed}(T)$ is only partially complete in that we have not shown in Theorem 2 that if $\varphi \vdash^R t$ and $\varphi \vdash^{R'} t$ then $\mathsf{i}_{L_1\varphi\downarrow,\ldots,L_m\varphi\downarrow}(R, R') \in \mathcal{H}(\mathsf{seed}(T))$.

We will shortly show how the completeness of $\mathsf{seed}(T)$ can be built upon to achieve a) full abstraction of the trace T and b) a procedure for checking equivalences $\approx_{ct}$ and $\approx_{ft}$.

5 Procedure for deciding trace equivalence

We shall now describe a procedure for deciding trace equivalence. At a high level, this consists of two steps.

1. A saturation procedure which constructs a set of *simple* statements from the set $\mathsf{seed}(T)$ which we will call *solved* statements. The saturation procedure ensures that the set of solved statements is a complete abstraction of T.
2. Given two processes P and Q, we saturate the set of seed statements for traces of P and Q and then use the solved statements to decide whether P and Q are trace equivalent.

We shall now give the details of the procedure. We start by the saturation procedure.

5.1 Knowledge bases and saturation

The saturation procedure manipulates a set of statements called a knowledge base:

Definition 15. *Given a statement $f = H \Leftarrow B_1, \ldots, B_n$,*

$$\textsc{Resolution}\ \frac{\begin{array}{c} f \in K, g \in K_{\mathsf{solved}}, \qquad f = \Big(H \Leftarrow \mathsf{k}_{uv}(X,t), B_1, \ldots, B_n\Big) \\ g = \Big(\mathsf{k}_w(R,t') \Leftarrow B_{n+1}, \ldots, B_m\Big) \qquad \sigma = \mathrm{mgu}(\mathsf{k}_u(X,t), \mathsf{k}_w(R,t')) \qquad t \notin \mathcal{X} \end{array}}{K = K \oplus h \text{ where } h = \Big((H \Leftarrow B_1, \ldots, B_m)\sigma\Big)}$$

$$\textsc{Equation}\ \frac{\begin{array}{c} f, g \in K_{\mathsf{solved}}, \\ f = \Big(\mathsf{k}_u(R,t) \Leftarrow B_1, \ldots, B_n\Big) \qquad g = \Big(\mathsf{k}_{u'v'}(R',t') \Leftarrow B_{n+1}, \ldots, B_m\Big) \qquad \sigma = \mathrm{mgu}(\mathsf{k}_u(\_,t), \mathsf{k}_{u'}(\_,t')) \end{array}}{K = K \oplus h \text{ where } h = \Big((\mathsf{i}_{u'v'}(R,R') \Leftarrow B_1, \ldots, B_m)\sigma\Big)}$$

$$\textsc{Test}\ \frac{f, g \in K_{\mathsf{solved}}, \qquad f = \Big(\mathsf{i}_u(R,R') \Leftarrow B_1, \ldots, B_n\Big) \qquad g = \Big(\mathsf{r}_{u'v'} \Leftarrow B_{n+1}, \ldots, B_m\Big) \qquad \sigma = \mathrm{mgu}(u,u')}{K = K \oplus h \text{ where } h = \Big((\mathsf{ri}_{u'v'}(R,R') \Leftarrow B_1, \ldots, B_m)\sigma\Big)}$$

Fig. 3: Saturation rules

- *f is said to be* solved *if for all $1 \leq i \leq n$, $B_i = \mathsf{k}_{\ell_1,\ldots,\ell_{j_i}}(X_i, x_i)$ for some variables $x_i \in \mathcal{X}, X_i \in \mathcal{Y}$.*
- *f is said to be* well-formed *if whenever it is solved and $H = \mathsf{k}_{\ell_1,\ldots,\ell_k}(R,t)$, we have that $t \notin \mathcal{X}$.*

A set of well-formed *statements is called a* knowledge base. *If K is a knowledge base, we define $K_{\mathsf{solved}} = \{f \in K \mid f \text{ is solved}\}$ to be the knowledge base restricted to the solved statements.*

Given an initial knowledge base K, the saturation procedure produces another knowledge base $\mathsf{sat}(K)$. The saturation procedure proceeds as follows. First new statements are *generated* and then the knowledge base is *updated* with the new statements. This two-step process continues until a fixed-point is achieved. We describe the two steps in the procedure.

Generating new statements Given a knowledge base K, new statements f are generated by applying the rules in Figure 3.

Update The first step while updating the knowledge base by f is to convert f into a canonical form.

Definition 16. *Given a solved deduction statement f, we define the* canonical form *of f to be the statement $f\Downarrow$ obtained by first applying Rule* Rename *below as many times as possible and then applying Rule* Remove *below as many times as possible:*

$$\textsc{Rename}\ \frac{H \Leftarrow \mathsf{k}_u(X,x), \mathsf{k}_{uv}(Y,x), B_1, \ldots, B_n}{(H \Leftarrow \mathsf{k}_u(X,x), B_1, \ldots, B_n)\{Y \mapsto X\}}$$

$$\textsc{Remove}\ \frac{H \Leftarrow \mathsf{k}_u(X,x), B_1, \ldots, B_n \qquad x \notin \mathit{vars}(H)}{H \Leftarrow B_1, \ldots, B_n}$$

For any other type of statement f, the canonical form $f\Downarrow$ is defined to be equal to f.

It is easy to see that any fact f can be converted into a canonical form. After a canonical form has been obtained, we perform another check before $f\Downarrow$ can be added to the knowledge base. Intuitively, this check ensures that we add enough identity predicates in the knowledge base.

Definition 17. *The set of* consequences *of a knowledge base K, denoted* $\mathbf{conseq}(K)$*, is the smallest set such that*

$$\textsc{Axiom}\ \frac{}{\mathsf{k}_{uv}(R,t) \Leftarrow \mathsf{k}_u(R,t), B_1, \ldots, B_m \in \mathbf{conseq}(K)}$$

$$\textsc{Res}\ \frac{\mathsf{k}_u(R,t) \Leftarrow B_1, \ldots, B_n \in K \qquad \sigma \text{ a substitution} \quad B_1\sigma \Leftarrow C_1, \ldots, C_m \in \mathbf{conseq}(K) \ \ldots \ B_n\sigma \Leftarrow C_1, \ldots, C_m \in \mathbf{conseq}(K)}{\mathsf{k}_{uv}(R,t)\sigma \Leftarrow C_1, \ldots, C_m \in \mathbf{conseq}(K)}$$

Given a knowledge base K and a statement f, the *update of K by f*, denoted $K \oplus f$, is defined to be $K \cup \{f\Downarrow\}$ if the head of f is not of the form $\mathsf{k}_{\ell_1,\ldots,\ell_k}(R,t)$. Otherwise, let

$$f\Downarrow = \mathsf{k}_{\ell_1,\ldots,\ell_k}(R,t) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1,t_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n,t_n)$$

and

$$K \oplus f = \begin{cases} K \cup \{f\Downarrow\} & \text{if } f \text{ is solved and for any } R' \text{ we have that} \\ & \mathsf{k}_{\ell_1,\ldots,\ell_k}(R',t) \Leftarrow \{\mathsf{k}_{\ell_1,\ldots,\ell_{i_j}}(X_j,t_j)\}_{j\in\{1,\ldots,n\}}\} \notin K' \\ K \cup \{\mathsf{i}_{\ell_1,\ldots,\ell_k}(R,R') & \text{if } f \text{ is solved and } R' \text{ is such that} \\ \quad \Leftarrow \{\mathsf{k}_{\ell_1,\ldots,\ell_{i_j}}(X_j,t_j)\}_{j\in\{1,\ldots,n\}}\} & \mathsf{k}_{\ell_1,\ldots,\ell_k}(R',t) \Leftarrow \{\mathsf{k}_{\ell_1,\ldots,\ell_{i_j}}(X_j,t_j)\}_{j\in\{1,\ldots,n\}}\} \in K' \\ K \cup \{f\Downarrow\} & \text{if } f \text{ is not solved} \end{cases}$$

where $K' = \mathbf{conseq}(K_{\mathsf{solved}})$.

Please note that update is not a function, namely that there may be several $R', i_1, \ldots, i_n$ such that $\mathsf{k}_{\ell_1,\ldots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1,t_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n,t_n) \in \mathbf{conseq}(K_{\mathsf{solved}})$. However, we need to compute only one such $R', i_1, \ldots, i_n$.

Initial knowledge base One question that naturally arises is what is the initial knowledge base for the saturation procedure. Given a trace T, the initial knowledge base for the saturation procedure is defined as follows.

Definition 18. *Given a set of statements S, the* initial knowledge base *associated to S, denoted $K_i(S)$, is defined to be the empty knowledge base updated by the set S, i.e., $K_i(S) = \emptyset \oplus_{f\in S} f$. If T is a ground trace, we write $K_i(T)$ for $K_i(\mathsf{seed}(T))$.*

Please observe that $K_i(T)$ depends on the order in which statements in $\mathsf{seed}(T)$ are updated. The exact order, however, is not important and our results shall hold regardless of the order chosen. The saturation procedure takes $K_i(T)$ as an input and produces a knowledge base $\mathsf{sat}(K_i(T))$. The reason for choosing $K_i(T)$ instead of $\mathsf{seed}(T)$ as the starting point of the saturation procedure is that $\mathsf{seed}(T)$ may not be a knowledge base (recall that a knowledge base is a set of well-formed statements). The set $K_i(T)$ is, however, a knowledge base.

Proposition 2. *Given a trace T, the set $K_i(T)$ is a knowledge base.*

Soundness and completeness of the saturation procedure We shall now show that the set of solved statements in $\mathsf{sat}(K_i(T))$ is a sound and complete abstraction of a trace T. We need one more definition which extends $\mathcal{H}(K)$ and allows us to establish that $\mathsf{sat}(K_i(T))$ is a complete abstraction of T.

Definition 19. *Let K be a set of statements. We define $\mathcal{H}_{\mathsf{e}}(K)$ to be the smallest set of ground terms such that $\mathcal{H}(K) \subseteq \mathcal{H}_{\mathsf{e}}(K)$ and closed under the following rules.*

$$\textsc{Refl}\ \frac{}{\mathsf{i}_w(R,R) \in \mathcal{H}_\mathsf{e}(K)} \qquad \textsc{Sym}\ \frac{\mathsf{i}_w(R_1,R_2) \in \mathcal{H}_\mathsf{e}(K)}{\mathsf{i}_w(R_2,R_1) \in \mathcal{H}_\mathsf{e}(K)}$$

$$\textsc{Tran}\ \frac{\mathsf{i}_w(R_1,R_2) \in \mathcal{H}_\mathsf{e}(K) \qquad \mathsf{i}_w(R_1,R_3) \in \mathcal{H}_\mathsf{e}(K)}{\mathsf{i}_w(R_1,R_3) \in \mathcal{H}_\mathsf{e}(K)}$$

$$\textsc{Cong}\ \frac{\mathsf{i}_w(R_1,R_1') \in \mathcal{H}_\mathsf{e}(K), \cdots, \mathsf{i}_w(R_n,R_n') \in \mathcal{H}_\mathsf{e}(K) \qquad f \in \mathcal{F}, ar(f) = n}{\mathsf{i}_w(f(R_1,\dots R_n), f(R_1',\dots R_n')) \in \mathcal{H}_\mathsf{e}(K)}$$

$$\textsc{Extend}\ \frac{\mathsf{i}_u(R,R') \in \mathcal{H}_\mathsf{e}(K)}{\mathsf{i}_{uv}(R,R') \in \mathcal{H}_\mathsf{e}(K)}$$

$$\textsc{Equational Consequence}\ \frac{\mathsf{k}_w(R,t) \in \mathcal{H}(K) \qquad \mathsf{i}_w(R,R') \in \mathcal{H}_\mathsf{e}(K)}{\mathsf{k}_w(R',t) \in \mathcal{H}_\mathsf{e}(K)}$$

We have that the set of solved statements produced by the saturation procedure is a sound and complete abstraction of the trace T.

Theorem 3. *Let T be a ground trace and let $K = \mathsf{sat}(K_i(T))$.*

- *(Soundness.) For any $f \in K \cup \mathcal{H}_\mathsf{e}(K)$, $T \models f$.*
- *(Completeness.) If $(T,\emptyset) \xrightarrow{L_1,\dots,L_n} (S,\varphi)$ then*
  1. $\mathsf{r}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow} \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved})$.
  2. *if $\varphi \vdash^R t$ then $\mathsf{k}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow}(R, t\downarrow) \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved})$.*
  3. *if $\varphi \vdash^R t$ and $\varphi \vdash^{R'} t$, then $\mathsf{i}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow}(R,R') \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved})$.*

Effectiveness of the saturation procedure We have shown that the set of solved statements in $\mathsf{sat}(K_i(T))$ form a sound and complete abstraction for the trace T. However, the set $\mathsf{sat}(K_i(T))$ may, a priori, not be computable for several reasons.

- As the set of public names $\mathcal{M}$ is infinite, the set $\mathsf{seed}(T)$ for a ground trace T is infinite as well.
- For the update rule, we have to check that given a knowledge base K, a term t, labels $\ell_1,\dots,\ell_k$, indices $1 \le i_1,\dots i_n, \le k$, variables $x_1,\dots,x_n \in \mathcal{X}$ and recipe variables $X_1,\dots,X_n \in \mathcal{Y}$, whether

$$\exists R.\ \mathsf{k}_{\ell_1,\dots,\ell_k}(R,t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \in \mathbf{conseq}(K_\mathsf{solved}).$$

  Furthermore, if the check succeeds then we have to compute one such R.
- The saturation procedure may itself not terminate even if the initial knowledge base is finite.

We will now address each of these three reasons.

First, we show that we only need to consider the saturation of the set $K_i(\mathsf{seed}(T,\mathcal{M}_T))$ where $\mathcal{M}_T$ is the (finite) set of public names occurring in T. The set $\mathsf{sat}(K_i(T))$ can then be computed from the set $\mathsf{sat}(K_i(seed(M_T,T)))$ by adding the set of clauses $K_{\mathcal{M},R}$ which is not required for the saturation and is defined as follows.

Definition 20. *Given a set of public names $M \subseteq \mathcal{M}$ and a set of solved reach statements R we define*

$$\begin{aligned} K_{M,R} = {} & \{\mathsf{k}(m,m) \Leftarrow\}_{m\in M} \cup \{\mathsf{i}(m,m) \Leftarrow\}_{m\in M} \cup \\ & \{\mathsf{ri}_u(m,m) \Leftarrow B_1,\dots B_n \mid m \in M, \mathsf{r}_u \Leftarrow B_1,\dots B_n \in R\} \end{aligned}$$

Lemma 1. *Let T be a trace and $M_T \subseteq \mathcal{M}$ be the public names occurring in T. Then*

$$\mathsf{sat}(K_i(T)) = \mathsf{sat}(K_i(seed(M_T, T))) \cup K_{\mathcal{M},R}$$

where R is the set of solved reach statements in $\mathsf{sat}(K_i(seed(M_T, T)))$.

Since the set $K_i(\mathsf{seed}(T, \mathcal{M}_T))$ is finite, this means that all intermediate knowledge bases in the saturation procedure are finite.

Second, we show that the update step can be decided if we only have a finite number of statements in the knowledge base.

Lemma 2. *Given a finite set of solved statements K, term t, labels $\ell_1, \ldots, \ell_k$, indices $1 \leq i_1, \ldots i_n \leq k$, variables $x_1, \ldots, x_n \in \mathcal{X}$ and recipe variables $X_1, \ldots, X_n \in \mathcal{Y}$, it is decidable if there is an R such that $\mathsf{k}_{\ell_1,\ldots,\ell_k}(R, t) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}(K_{\mathsf{solved}})$. If the answer to the decision procedure is "Yes", then we can compute one such R.*

It has been shown in [6] that deducibility is undecidable for convergent optimally reducing rewrite systems. As observed in [1] static equivalence is even harder to decide, as soon as the signature may contain a free symbol (which does not change the fact that the rewrite system is convergent and optimally reducing). As our algorithm would allow to decide static equivalence as a particular case we cannot expect a general termination result. However, we prove that the saturation procedure does terminate for the class of subterm convergent rewrite systems.

Theorem 4. *Let T be a ground trace and $S = \mathsf{seed}(T)$. For a subterm convergent rewrite system the computation of $\mathsf{sat}(K_i(S))$ terminates in a finite number of steps.*

We remark that the saturation is nevertheless sound and complete for the more general class of convergent rewrite systems for which the finite variant property holds. Indeed, the procedure may also terminate on protocols that rely on rewrite systems that are not subterm convergent. This is demonstrated in our case studies when analysing protocols using blind signatures and trapdoor commitment schemes.

5.2 Algorithm

In this section we describe an algorithm to decide trace inclusion for determinate processes. In Figure 4, we describe the checks Reachability and Identity which allow us to test whether a trace–represented by the set K of solved facts in the saturated knowledge base associated to this trace–is included in a determinate process P.

Theorem 5. *Let T be a ground trace, P a ground process and $K = (\mathsf{sat}(K_i(T)))_{\mathsf{solved}}$. We have that*

- *if $T \sqsubseteq_{ct} P$ then* Reachability(K, P) *and* Identity(K, P) *hold.*
- *if P is determinate and* Reachability(K, P) *and* Identity(K, P) *hold then $T \sqsubseteq_{ct} P$.*

Note that performing the tests requires deciding if, given t, and w, $\mathsf{k}_w(R, t) \in \mathcal{H}(K)$ for some recipe R for a knowledge base K containing only solved statements. It is easy to see that this is equivalent to checking if $\left(\mathsf{k}_w(R, t) \Leftarrow\right) \in \mathbf{conseq}(K)$ and we have already shown that there is an effective procedure for this (which finds an R if such an R exists).

REACHABILITY(K, P)
For all $\mathsf{r}_{l_1,\ldots,l_n} \Leftarrow \{\mathsf{k}_{w_i}(X_i, x_i)\}_{i\in\{1,\ldots,m\}} \in K$ do
  let $c_1, \ldots, c_k$ be fresh public names
    such that $\sigma : vars(l_1, \ldots, l_n) \to \{c_1, \ldots, c_k\}$ is a bijection
  if for all i s.t. $l_i = \mathtt{in}(t_i)$ we have that $\mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R_i, t_i\sigma) \in \mathcal{H}(K)$ then
    let $M_i = \begin{cases} l_i & \text{if } l_i \in \{\mathtt{test}, \mathtt{out}(c) \mid c \in \mathcal{C}\} \\ \mathtt{in}(d_i, R_i) & \text{if } l_i = \mathtt{in}(d_i, t_i) \end{cases}$
    check that $(P, \emptyset) \xrightarrow{M_1,\ldots,M_n} (T', \varphi)$
  endif
enddo

IDENTITY(K, P)
For all $\mathsf{ri}_{l_1,\ldots,l_n}(R, R') \Leftarrow \{\mathsf{k}_{w_i}(X_i, x_i)\}_{i\in\{1,\ldots,m\}} \in K$ do
  let $c_1, \ldots, c_k$ be fresh public names
    such that $\sigma : vars(l_1, \ldots, l_n) \to \{c_1, \ldots, c_k\}$ is a bijection
  if for all i s.t. $l_i = \mathtt{in}(t_i)$ we have that $\mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R_i, t_i\sigma) \in \mathcal{H}(K)$ then
    let $M_i = \begin{cases} l_i & \text{if } l_i \in \{\mathtt{test}, \mathtt{out}(c) \mid c \in \mathcal{C}\} \\ \mathtt{in}(d_i, R_i) & \text{if } l_i = \mathtt{in}(d_i, t_i) \end{cases}$
    check that $(P, \emptyset) \xrightarrow{M_1,\ldots,M_n} (T', \varphi)$ and $(R\omega = R'\omega)\varphi$
      where $\omega = \{X_i \mapsto x_i\sigma\}$
  endif
enddo

Fig. 4: Tests for checking $T \sqsubseteq_{ct} P$

6 Prototype and case studies

6.1 The AKISS prototype

We implemented the procedure for checking equivalence in a prototype, AKISS (Active Knowledge IN Security protocolS). AKISS is written in OCaml and has about 2000 lines of source code, including code for computing complete sets of finite variants and complete sets of equational unifiers. We used AKISS to verify the equivalences in Examples 7 and 8. Using AKISS we were able to verify strong secrecy for Denning-Sacco-Blanchet [18] and Needham-Schroeder-Lowe (NSL) [50], resistance to guessing attacks in the EKE protocol [16], and, more interestingly, anonymity of the FOO [45] and Okamoto [52] electronic voting protocols.[5] To our knowledge, AKISS is the only tool that can verify FOO and Okamoto completely automatically. We discuss each of these examples in more details below. In [8] the tool has also been extended to verify a property called *everlasting privacy* that appears in electronic voting. Several other protocols were analysed in this context. AKISS along with all the discussed examples is available on:

`http://akiss.gforge.inria.fr`

To ease protocol specification, the process calculus syntax used for specifying protocol we allow for an operator *interleave*, denoted $\|$, which models parallel composition of processes and an operator *sequence*, denoted ;, for modeling protocols structured in phases. These constructs are merely syntactic sugar and are defined as follows. Given processes P and Q we define $P; Q$ as the sequential composition of each trace in P with each trace in Q, i.e.,

$$P; Q = \{T_1.T_2 \mid T_1 \in P, T_2 \in Q\}$$

[5] Please note that as defined in [52], modeling of Okamoto's protocol requires private channels. As we do not have private channels in our calculus, we transform the protocol so that every message sent by honest participants on a private channel is sent encrypted under a key not known to the adversary

Let ϵ denote the empty sequence, a_1, a_2 be actions and T, T_1, T_2 traces. The parallel composition of two traces is the process defined inductively as

$$T \parallel \epsilon = \epsilon \parallel T = T$$
$$a_1.T_1 \parallel a_2.T_2 = \{a_1; (T_1 \parallel a_2.T_2), a_2; (a_1.T_1 \parallel T_2)\}$$

The parallel composition is then naturally lifted to process, i.e. $P \parallel Q = \{T_1 \parallel T_2 \mid T_1 \in P, T_2 \in Q\}$.

The $\parallel$ operator reflects the usual notion of parallel composition in process calculi. One may note that the number of possible interleavings (and hence generated traces) is exponential. We can however slightly lower this number due to the fact that test actions are silent, i.e unobservable. We therefore define an optimised interleaving operator $\parallel_o$ which generates less interleavings. In practice this gain is substantial on several examples. In the following we let τ (and decorations of τ) range over test actions, i.e. actions of the form $[s \stackrel{?}{=} t]$ for some terms s, t. α (and decorations of α) range over input and output actions. The optimized parallel composition of two traces is the process defined inductively as

$$\tau_1 \dots \tau_n \parallel_o T = T.\tau_1 \dots \tau_n$$
$$\tau_1 \dots \tau_n.\alpha.T \parallel_o \tau'_1 \dots \tau'_m.\alpha'.T' = \left\{ \begin{array}{l} \tau_1 \dots \tau_n.\alpha; (T \parallel_o \tau'_1 \dots \tau'_m.\alpha'.T'), \\ \tau'_1 \dots \tau'_m.\alpha'; (\tau_1 \dots \tau_n.\alpha.T \parallel_o T') \end{array} \right\}$$

Intuitively we consider sequences of silent actions together with the following visible action as atomic. We will now show that this is indeed a sound optimization when checking trace equivalence by showing that $P_1 \parallel P_2 \approx_t P_1 \parallel_o P_2$.

Proposition 3. *Let T_1, T_2 be two ground traces.*

$$(T_1 \parallel T_2) \stackrel{l_1,\dots l_k}{\Longrightarrow} (T, \varphi) \text{ iff } (T_1 \parallel_o T_2) \stackrel{l_1,\dots l_k}{\Longrightarrow} (T_o, \varphi)$$

From this proposition it is easy to conclude that $(P \parallel Q) \approx_t (P \parallel_o Q)$.

6.2 Security properties and case studies

We now give more details about our case studies.

Strong flavors of confidentiality The *strong secrecy* property was introduced by Blanchet in [18] and we rephrase it here in our setting. Let P be a protocol with x as the only free variable of P. Then x is said to be *strongly secret* if

$$\mathbf{in}(c, x_1).\mathbf{in}(c, x_2).(P\{x \mapsto x_1\}) \approx_t \mathbf{in}(c, x_1).\mathbf{in}(c, x_2).(P\{x \mapsto x_2\}).$$

Intuitively, the attacker cannot distinguish the processes using variables x_1 and x_2 even though it can choose arbitrary (public) values for these variables. The definition generalizes to multiple variables in the expected way. We illustrate this property on a Denning-Sacco-Blanchet protocol. Informally, the protocol can be described as follows.

$$\mathsf{A} \to \mathsf{B} : \mathsf{aenc}(\mathsf{sign}(\mathsf{pair}(\mathsf{pk}(ska), \mathsf{pair}(\mathsf{pk}(skb, k))), ska), \mathsf{pk}(skb))$$
$$\mathsf{B} \to \mathsf{A} : \mathsf{enc}(x, k)$$

A sends to B a fresh symmetric session key k together with A's and B's public keys. This is signed with A's secret key and (asymmetrically) encrypted with B's public key. Upon receiving this message, B decrypts it, checks the signature and uses the fresh session key to symmetrically encrypt a secret x. The detailed protocol model is given in Figure 5. We note that the rewrite system is subterm convergent. We used AKISs to verify this protocol for strong secrecy of x (with one session of A and B). This protocol is determinate, and hence we used $\approx_{ct}$ to verify that $P_1 \approx_{ct} P_2$. The verification succeeds as expected.

Rewrite System:

$$\begin{array}{lll} \mathsf{fst}(\mathsf{pair}(x,y)) \to x & \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) \to x & \mathsf{check}(\mathsf{sign}(x,y),pk(y)) \to ok \\ \mathsf{snd}(\mathsf{pair}(x,y)) \to y & \mathsf{dec}(\mathsf{enc}(x,y),y) \to x & \mathsf{msg}(\mathsf{sign}(x,y)) \to x \end{array}$$

Processes:

$$\begin{aligned} P_i &= \mathit{Setup}; (A \parallel B_i) \qquad (i \in \{1,2\}) \\ \mathit{Setup} &= \mathbf{out}(c,\mathsf{pk}(skA)).\mathbf{out}(c,\mathsf{pk}(ekB)).\mathbf{in}(c,x_1).\mathbf{in}(c,x_2) \\ A &= \mathbf{out}(c,\mathsf{aenc}(\mathsf{sign}(\mathsf{pair}(\mathsf{pk}(skA),\mathsf{pair}(\mathsf{pk}(ekB),k)),skA),\mathsf{pk}(ekB))) \\ B_i &= \mathbf{in}(c,z). \\ &\quad [\mathsf{check}(\mathsf{adec}(z,ekB),\mathsf{pk}(skA)) \stackrel{?}{=} ok]. \\ &\quad [\mathsf{fst}(\mathsf{msg}(\mathsf{adec}(z,ekB))) \stackrel{?}{=} \mathsf{pk}(skA)]. \\ &\quad [\mathsf{fst}(\mathsf{snd}(\mathsf{msg}(\mathsf{adec}(z,ekB)))) \stackrel{?}{=} \mathsf{pk}(ekB)]. \\ &\quad \mathbf{out}(c,\mathsf{enc}(x_i,\mathsf{snd}(\mathsf{snd}(\mathsf{msg}(\mathsf{adec}(z,ekB)))))) \end{aligned}$$

Fig. 5: Formal description of the protocol by Blanchet

A variant of the protocol [18] consists in letting A also send out a secret y encrypted with k changing the first message to

$$\mathsf{A} \to \mathsf{B} : \mathsf{pair}(\mathsf{aenc}(\mathsf{sign}(\mathsf{pair}(\mathsf{pk}(ska),\mathsf{pair}(\mathsf{pk}(skb,k))),ska),\mathsf{pk}(skb)),\mathsf{enc}(y,k))$$

In this case the protocol does not respect strong secrecy of x, y as, by choosing $x_1 = y_1$ and $x_2 \neq y_2$, the attacker can distinguish the two situations by testing the equality of the encryptions of x and y. The detailed model is given in Figure 6. This attack is again found by AKISS.

Rewrite System:

$$\begin{array}{lll} \mathsf{fst}(\mathsf{pair}(x,y)) \to x & \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) \to x & \mathsf{check}(\mathsf{sign}(x,y),pk(y)) \to ok \\ \mathsf{snd}(\mathsf{pair}(x,y)) \to y & \mathsf{dec}(\mathsf{enc}(x,y),y) \to x & \mathsf{msg}(\mathsf{sign}(x,y)) \to x \end{array}$$

Processes:

$$\begin{aligned} P_i &= \mathit{Setup}; (A_i \parallel B_i) \qquad (i \in \{1,2\}) \\ \mathit{Setup} &= \mathbf{out}(c,\mathsf{pk}(skA)).\mathbf{out}(c,\mathsf{pk}(ekB)).\mathbf{in}(c,x_1).\mathbf{in}(c,x_2).\mathbf{in}(c,y_1).\mathbf{in}(c,y_2) \\ A_i &= \mathbf{out}(c,\mathsf{pair}(\mathsf{aenc}(\mathsf{sign}(\mathsf{pair}(\mathsf{pk}(skA),\mathsf{pair}(\mathsf{pk}(ekB),k)),skA),\mathsf{pk}(ekB))), \\ &\qquad\qquad \mathsf{enc}(y_1,k)) \\ B_i &= \mathbf{in}(c,z). \\ &\quad [\mathsf{check}(\mathsf{adec}(z,ekB),\mathsf{pk}(skA)) \stackrel{?}{=} ok]. \\ &\quad [\mathsf{fst}(\mathsf{msg}(\mathsf{adec}(z,ekB))) \stackrel{?}{=} \mathsf{pk}(skA)]. \\ &\quad [\mathsf{fst}(\mathsf{snd}(\mathsf{msg}(\mathsf{adec}(z,ekB)))) \stackrel{?}{=} \mathsf{pk}(ekB)]. \\ &\quad \mathbf{out}(c,\mathsf{enc}(x_i,\mathsf{snd}(\mathsf{snd}(\mathsf{msg}(\mathsf{adec}(z,ekB))))))) \end{aligned}$$

Fig. 6: Formal description of the variant protocol by Blanchet

AKISS also verifies strong secrecy of the nonce generated by the responder in the Needham-Schroeder-Lowe (NSL) [50] protocol. The NSL protocol is a two-way handshake protocol relying only on public encryp-

tion of fresh nonces and can be informally described as follows.

$$\begin{aligned}
&\mathsf{A} \to \mathsf{B} : \mathsf{aenc}(\mathsf{pair}(n_a, A), \mathsf{pk}(skb)) \\
&\mathsf{B} \to \mathsf{A} : \mathsf{aenc}(\mathsf{pair}(n_b, \mathsf{pair}(n_a, B)), \mathsf{pk}(ska)) \\
&\mathsf{A} \to \mathsf{B} : \mathsf{aenc}(\mathsf{pair}(n_b), \mathsf{pk}(skb))
\end{aligned}$$

Once again, the modelling of NSL leads to a subterm convergent rewrite system and determinate processes. We therefore used $\approx_{ct}$ for our verification. The detailed model is given in Figure 7.

Rewrite System:

$$\mathsf{fst}(\mathsf{pair}(x, y)) \to x \qquad \mathsf{snd}(\mathsf{pair}(x, y)) \to y \qquad \mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y), z), y) \to x$$

Processes:

$$\begin{aligned}
P_i &= \mathit{Setup}; (A \parallel B_i) \qquad (i \in \{1, 2\}) \\
\mathit{Setup} &= \mathbf{out}(c, \mathsf{pk}(skA)).\mathbf{out}(c, \mathsf{pk}(ekB)).\mathbf{out}(c, skD)\mathbf{in}(c, x_1).\mathbf{in}(c, x_2) \\
A &= \mathbf{out}(c, \mathsf{aenc}(\mathsf{pair}(n_a, a), \mathsf{pk}(skD), r_1)). \\
&\quad\ \mathbf{in}(c, y).[\mathsf{snd}(\mathsf{snd}(\mathsf{adec}(y, skA))) = d]. \\
&\quad\ \mathbf{out}(c, \mathsf{aenc}(\mathsf{fst}(\mathsf{snd}(\mathsf{adec}(y, skA))), \mathsf{pk}(skD), r_2)) \\
B_i &= \mathbf{in}(c, z).[\mathsf{snd}(\mathsf{adec}(z, skB)) = a]. \\
&\quad\ \mathbf{out}(c, \mathsf{aenc}(\mathsf{pair}(\mathsf{fst}(\mathsf{adec}(z, skB)), \mathsf{pair}(x_1, b)), pk(skA), r_3))
\end{aligned}$$

Fig. 7: Formal description of the NSL protocol

This model includes a session of the initiator who is willing to engage with any participant (including the attacker to allow man-in-the-middle attacks) and a session of B who is willing to engage a session with A. Note that if B was willing to start a session with an arbitrary initiator the secrecy of n_b would be trivially broken in a session with the attacker. (In a more complex model one could of course add additional sessions for B with an arbitrary initiator.) We note that for the verification of NSL, one needs to explicitly model randomness for asymmetric encryption since the protocol is insecure if deterministic asymmetric encryption is used. Indeed, as the attacker may choose the value of n_b he could simply recompute the last message and compare it with the message sent by the initiator.

We also used AKiSs to verify the above protocols for *real-or-random* secrecy. Let P be a protocol and $n \in \mathit{names}(P)$. Then n is said to be *real-or-random secret* if

$$P; \mathbf{out}(c, n) \approx_t P; \mathbf{out}(c, n')$$

where n' is a fresh name, i.e. a name that does not appear in P. Real-or-random secrecy is particularly useful to model resistance to offline guessing attacks in password protocols [14]. Intuitively, an offline guessing attacks works in two phases. In the first, online phase the attacker interacts with the protocol P in an arbitrary way. In a second, offline phase the attacker tries all possible passwords against the data recorded in the first phase. Our property states that the attacker cannot distinguish the case where he tests the real password (n) from the case where he tests a wrong password (n'). We show that the EKE protocol [16] is resistant to offline guessing attacks. The protocol can be described informally as follows:

$$\begin{aligned}
&\mathsf{A} \to \mathsf{B} : \mathsf{enc}(\mathsf{pk}(k)), w) \\
&\mathsf{B} \to \mathsf{A} : \mathsf{enc}(\mathsf{aenc}(r, \mathsf{pk}(k)), w) \\
&\mathsf{A} \to \mathsf{B} : \mathsf{enc}(na, r) \\
&\mathsf{B} \to \mathsf{A} : \mathsf{enc}(\langle na, nb \rangle, r) \\
&\mathsf{A} \to \mathsf{B} : \mathsf{enc}(nb, r)
\end{aligned}$$

In the first step A generates a new private session key k and sends the corresponding public key $\mathsf{pk}(k)$ to B, encrypted (using symmetric encryption) with the shared password w. Then, B generates a fresh symmetric

session key r, which he encrypts (using asymmetric encryption) with the previously received public key $\mathsf{pk}(k)$. Finally, he encrypts the resulting ciphertext with the password w and sends the result to A. The last three steps perform a handshake to avoid replay attacks. Using AKISs we have shown that the protocol resists to offline guessing attacks on the password w. As EKE is modelled by a subterm convergent rewrite system and determinate processes, we used the $\approx_{ct}$ relation. The detailed description of our model is given in Figure 8.

Rewrite System:

$$\mathsf{fst}(\mathsf{pair}(x,y)) \to x \qquad \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y),z),y) \to x$$
$$\mathsf{snd}(\mathsf{pair}(x,y)) \to y \qquad \mathsf{dec}(\mathsf{enc}(x,y),y) \to x$$

Processes:

$$\begin{aligned}
P_1 &= (A \parallel B); \mathbf{out}(c,w)\\
P_2 &= (A \parallel B); \mathbf{out}(c,w')\\
A &= \mathbf{out}(c,\mathsf{enc}(\mathsf{pk}(k),w)).\\
&\quad \mathbf{in}(c,x_1).\\
&\quad \mathbf{out}(c,\mathsf{enc}(n_a,\mathsf{adec}(dec(x_1,w),k))).\\
&\quad \mathbf{in}(c,x_2).[\mathsf{fst}(\mathsf{dec}(x_2,\mathsf{adec}(\mathsf{dec}(x_1,w),k))) \stackrel{?}{=} n_a].\\
&\quad \mathbf{out}(c,\mathsf{snd}(\mathsf{dec}(x_2,\mathsf{adec}(\mathsf{dec}(x1,w),k)))).\\
B &= \mathbf{in}(c,y_1).\\
&\quad \mathbf{out}(c,\mathsf{enc}(\mathsf{aenc}(r,\mathsf{dec}(y_1,w),r_b),w)).\\
&\quad \mathbf{in}(c,y_2).\\
&\quad \mathbf{out}(c,\mathsf{enc}(\mathsf{pair}(\mathsf{dec}(y_2,r),n_b),r)).
\end{aligned}$$

Fig. 8: Formal description of the EKE protocol

Anonymity for electronic voting protocol A voting protocol must respect voter privacy: the adversary should not be able to learn how each voter voted. AKISs can automatically verify voter privacy in the FOO electronic voting protocol [45] and the Okamoto protocol [52]. Voter privacy is naturally modeled as an equivalence property [39,13]: it is not possible to distinguish the situation where honest voter A votes 'yes' and honest B votes 'no' from the situation that A votes 'no' and B votes 'yes'. Note that our modeling of the protocols, that we precise below, is exactly the same as in [39]. We assume that *only* voters A and B are honest while all other entities are dishonest. An arbitrary number of dishonest voters are however subsumed by the attacker and need not be modeled directly.

We now briefly describe the two protocols. The FOO protocol relies on blind signatures and a commitment function. The rewrite system is specified in Figure 9. We note that the rewrite system is not subterm convergent, but it is optimally reducing. The protocol consists in 3 phases informally described as follows.

$$\begin{aligned}
&\textit{Phase } 1:\\
&\mathsf{V} \to \mathsf{A} : \mathsf{sign}(\mathsf{blind}(\mathsf{commit}(v,r),b),skV))\\
&\mathsf{A} \to \mathsf{V} : \mathsf{sign}(\mathsf{blind}(\mathsf{commit}(v,r),b),skA))\\
&\textit{Phase } 2:\\
&\mathsf{V} \to \mathsf{C} : \mathsf{sign}(\mathsf{commit}(v,r),skA)\\
&\textit{Phase } 3:\\
&\mathsf{V} \to \mathsf{C} : r
\end{aligned}$$

In the first phase the voter V commits to his vote v which he blindly signs and sends to the election administrator A. A checks eligibility of V and then signs the blinded commitment. Blinding the commitment ensures that A cannot trace the ballot. V unblinds the signature and obtains a ballot which is signed by A. In the second phase A submits the signed ballot to a collector C who publishes all the submitted ballots on a public bulletin board. Finally, in the 3rd phase, V submits the random r which allows to open the

commitment to C who again publishes this value on the bulletin board. The election can now be tallied by any observer. The detailed model is given in Figure 9. Note that only two honest voters need to be modelled for showing anonymity. All remaining voters and election authorities are subsumed by the adversary. The processes $A_{yes}B_{no}$ and $A_{no}B_{yes}$ model the situation where these two honest voters have swapped their vote. The protocols do not lead to determinate processes. Therefore, we proved the relation $A_{yes}B_{no} \approx_{ft} A_{no}B_{yes}$.

As the processes are not

Rewrite System:

$$\mathsf{open}(\mathsf{commit}(x,y),y) \to x \qquad \mathsf{check}(\mathsf{sign}(x,y),pk(y)) \to x$$
$$\mathsf{unblind}(\mathsf{sign}(\mathsf{blind}(x,y),z),y) \to \mathsf{sign}(x,z)$$

Processes:

$$\begin{aligned}
P_1 &= \mathit{Setup}; A_{yes}B_{no}\\
P_2 &= \mathit{Setup}; A_{no}B_{yes}\\
\mathit{Setup} &= \mathbf{out}(c,\mathsf{pk}(sk_A)).\mathbf{out}(c,\mathsf{pk}(sk_B))\\
\mathit{V\text{-}Phase1} &= \mathbf{out}(V,\mathsf{sign}(\mathsf{blind}(\mathsf{commit}(v,r),b),sk)).\\
&\quad \mathbf{in}(A,x).[\mathsf{check}(x,\mathsf{pk}(skA)) \stackrel{?}{=} \mathsf{blind}(\mathsf{commit}(v,r),b)]\\
\mathit{V\text{-}Phase2} &= \mathbf{out}(C,\mathsf{unblind}(x,b))\\
\mathit{V\text{-}Phase3} &= \mathbf{out}(C,r)\\
AB &= (\mathit{V\text{-}Phase1}\{^{v_a}/_{v},^{sk_A}/_{sk},^{r_a}/_{r},^{b_a}/_{b},^{x_a}/_{x},\} \parallel\\
&\quad \mathit{V\text{-}Phase1}\{^{v_b}/_{v},^{sk_B}/_{sk},^{r_b}/_{r},^{b_b}/_{b},^{x_b}/_{x},\});\\
&\quad (\mathit{V\text{-}Phase2}\{^{b_a}/_{b},^{x_a}/_{x}\} \parallel \mathit{V\text{-}Phase2}\{^{b_b}/_{b},^{x_b}/_{x}\});\\
&\quad (\mathit{V\text{-}Phase3}\{^{r_a}/_{r}\} \parallel \mathit{V\text{-}Phase3}\{^{r_b}/_{r}\})\\
A_{yes}B_{no} &= AB\{^{yes}/_{v_a},^{no}/_{v_b}\}\\
A_{no}B_{yes} &= AB\{^{no}/_{v_a},^{yes}/_{v_b}\}
\end{aligned}$$

Fig. 9: Formal description of the FOO protocol

We will not give a detailed description of the Okamoto protocol and refer the reader to [39]. The protocol is a variant of the FOO protocol which aims at achieving receipt-freeness. To avoid vote selling a voter should not be able to provide a receipt of how he voted to a potential coercer. In the FOO protocol this is possible by sending all private names to a coercer. The main tool to avoid this problem in the Okamoto protocol is the use of trapdoor commitment functions. These functions allow to change the value of committed vote using a secret value called the trapdoor. Following [29] we model trapdoor commitment by the following rewrite system:

$$\mathsf{open}(\mathsf{tdcommit}(x,y,z),y) \to x \qquad \mathsf{tdcommit}(x,\mathsf{f}(x_1,y,z,x),z) \to \mathsf{tdcommit}(x_1,y,z)$$
$$\mathsf{open}(\mathsf{tdcommit}(x,y,z),\mathsf{f}(x,y,z,x_1)) \to x_1 \qquad \mathsf{f}(x_1,\mathsf{f}(x,y,z,x_1),z,x_2) \to \mathsf{f}(x,y,z,x_2)$$

Intuitively, a trapdoor commitment $\mathsf{tdcommit}(x,y,z)$ commits to x using the key y and trapdoor z. The commitment can be opened using key y to x. However, knowing the trapdoor z one may compute an altermate key $f(x_1,y,z,x)$ which opens the commitment $\mathsf{tdcommit}(x,y,z)$ to x_1 rather than x. This rewrite system is again optimally reducing but not subterm convergent and out of the scope of most tools, even in the simpler case of a passive adversary. The only result we are aware of that can verify this rewrite system is [29]. As for the FOO protocol we used the relation $\approx_{ft}$ to prove anonymity.

To our knowledge, no other tool can handle this automatically. We are aware of two other attempts for verifying the FOO protocol. Using ProVerif [18], Delaune *et al.* [40], verify a transformation of the protocol. However, the soundness of this transformation has never been proven. Chothia *et al.* [27] verify a different notion of anonymity (also based on process equivalence) using the μCRL tool. However, the attacker they consider is only an observer that cannot interact with the protocol participants, yielding a finite state system.

Efficiency On a standard modern laptop, AKiSs takes a few seconds to carry out the above verification, except for the verification of the Okamoto protocol which takes about 30 seconds. Most of the computational effort goes into the saturation of the traces. Interleaving individual roles of a protocol introduces an exponential blowup on the number of traces and saturations to perform. However, we believe that we can scale to larger protocols and more sessions by parallelizing the saturation of these traces (e.g. on clusters of machines). An implementation performing saturations in parallel is currently in progress.

7 Conclusion

In this paper we present a novel procedure for verifying equivalence properties for a bounded number of sessions of cryptographic protocols. The procedure has been implemented in a tool which is able to handle examples which are out of the scope of existing tools.

There are several directions for future work. The implementation of the tool should be optimized and we plan to analyze more examples coming from electronic voting, RFID protocols and auction protocols which all have requirements stated in terms of equivalences.

We would also like to extend the procedure to be able to take disequalities into account. On the one hand, disequalities will allow to verify processes with else branches which are important in a number of practical examples. On the other hand, characterizing disequalities in our decision procedure would allow to directly decide trace equivalence based on static equivalence (rather than static inclusion). Another direction would be to extend the procedure to allow AC operators in order to treat protocols based on exclusive or Diffie-Hellman exponentiations.

References

1. Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 387(1-2):2–32, 2006.
2. Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *Proc. 28th ACM Symposium on Principles of Programming Languages (POPL'01)*. ACM, 2001.
3. Martín Abadi and Cédric Fournet. Private authentication. *Theoretical Computer Science*, 322(3):427–476, 2004.
4. Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Inf. Comput.*, 148(1):1–70, 1999.
5. Reynald Affeldt and Hubert Comon-Lundh. Verification of security protocols with a bounded number of sessions based on resolution for rigid variables. In *Formal to Practical Security*, volume 5458 of *LNCS, State-of-the-Art Survey*, pages 1–20. Springer, 2009.
6. Siva Anantharaman, Paliath Narendran, and Michaël Rusinowitch. Intruders with caps. In *Proc. 18th International Conference on Term Rewriting and Applications (RTA'07)*, volume 4533 of *LNCS*. Springer, 2007.
7. Myrto Arapinis, Tom Chothia, Eike Ritter, and Mark D. Ryan. Analysing unlinkability and anonymity using the applied pi calculus. In *Proc. 23rd Computer Security Foundations Symposium (CSF'10)*, pages 107–121. IEEE Comp. Soc. Press, 2010.
8. Myrto Arapinis, Véronique Cortier, Steve Kremer, and Mark D. Ryan. Practical Everlasting Privacy. In *Proc. 2nd Conference on Principles of Security and Trust (POST'13)*, volume 7796 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2013.
9. Alessandro Armando, David A. Basin, Yohan Boichut, Yannick Chevalier, Luca Compagna, Jorge Cuéllar, Paul Hankes Drielsma, Pierre-Cyrille Héam, Olga Kouchnarenko, Jacopo Mantovani, Sebastian Mödersheim, David von Oheimb, Michaël Rusinowitch, Judson Santiago, Mathieu Turuani, Luca Viganò, and Laurent Vigneron. The AVISPA tool for the automated validation of internet security protocols and applications. In *Proc. 17th International Conference on Computer Aided Verification (CAV'05)*, Lecture Notes in Computer Science, pages 281–285. Springer, 2005.
10. Mathilde Arnaud, Véronique Cortier, and Stéphanie Delaune. Combining algorithms for deciding knowledge in security protocols. In *Proc. 6th International Symposium on Frontiers of Combining Systems (FroCoS'07)*, volume 4720 of *Lecture Notes in Artificial Intelligence*, pages 103–117. Springer, 2007.
11. Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.

12. Franz Baader and Wayne Snyder. Unification theory. In *Handbook of Automated Reasoning, volume I, chapter 8*, pages 445–532. Elsevier Science, 2001.
13. Michael Backes, Catalin Hritcu, and Matteo Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *Proc. 21st IEEE Computer Security Foundations Symposium (CSF'08)*, 2008.
14. Mathieu Baudet. Deciding security of protocols against off-line guessing attacks. In *12th ACM Conference on Computer and Communications Security (CCS'05)*, 2005.
15. Mathieu Baudet, Véronique Cortier, and Stéphanie Delaune. YAPA: A generic tool for computing intruder knowledge. In *Proc. 20th International Conference on Rewriting Techniques and Applications (RTA'09)*, volume 5595 of *Lecture Notes in Computer Science*, pages 148–163. Springer, 2009.
16. Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Symposium on Security and Privacy (S&P'92)*, pages 72–84. IEEE Comp. Soc., 1992.
17. Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *14th Computer Security Foundations Workshop (CSFW'01)*, pages 82–96. IEEE Comp. Soc. Press, 2001.
18. Bruno Blanchet. Automatic proof of strong secrecy for security protocols. In *Symposium on Security and Privacy (S&P'04)*, pages 86–100, 2004.
19. Bruno Blanchet, Martín Abadi, and Cédric Fournet. Automated Verification of Selected Equivalences for Security Protocols. In *Symposium on Logic in Computer Science*, pages 331–340. IEEE Comp. Soc. Press, 2005.
20. Johannes Borgström. *Equivalences and Calculi for Formal Verifiation of Cryptographic Protocols*. Phd thesis, EPFL, Switzerland, 2008.
21. Johannes Borgström, Sébastien Briais, and Uwe Nestmann. Symbolic bisimulation in the spi calculus. In *Proc. 15th Int. Conference on Concurrency Theory*, volume 3170 of *LNCS*, pages 161–176. Springer, 2004.
22. Mayla Bruso, Konstantinos Chatzikokolakis, and Jerry den Hartog. Analysing unlinkability and anonymity using the applied pi calculus. In *Proc. 23rd Computer Security Foundations Symposium (CSF'10)*, pages 107–121. IEEE Comp. Soc. Press, 2010.
23. Rohit Chadha, Ştefan Ciobâcă, and Steve Kremer. Automated verification of equivalence properties of cryptographic protocols. In Helmut Seidl, editor, *21st European Symposium on Programming, ESOP 2012*, volume 7211 of *Lecture Notes in Computer Science*, pages 108–127. Springer, 2012.
24. Vincent Cheval, Hubert Comon-Lundh, and Stephanie Delaune. Automating security analysis: symbolic equivalence of constraint systems. In *Proc. International Joint Conference on Automated Reasoning (IJCAR'10)*, Lecture Notes in Artificial Intelligence. Springer, 2010. To appear.
25. Vincent Cheval, Hubert Comon-Lundh, and Stéphanie Delaune. Trace equivalence decision: Negative tests and non-determinism. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*, Chicago, Illinois, USA, October 2011. ACM Press. To appear.
26. Yannick Chevalier and Michaël Rusinowitch. Decidability of equivalence of symbolic derivations. *Journal of Automated Reasoning*, 2010. To appear.
27. Tom Chothia, Simona Orzan, Jun Pang, and Muhammad Torabi Dashti. A framework for automatically checking anonymity with *mu* crl. In *2nd Symposium on Trustworthy Global Computing (TGC'06)*, volume 4661 of *Lecture Notes in Computer Science*, pages 301–318. Springer, 2007.
28. Ştefan Ciobâcă. Computing finite variants for subterm convergent rewrite systems. Research Report LSV-11-06, Laboratoire Spécification et Vérification, ENS Cachan, France, April 2011. 16 pages.
29. Ştefan Ciobâcă, Stéphanie Delaune, and Steve Kremer. Computing knowledge in security protocols under convergent equational theories. In Renate Schmidt, editor, *Proc. 22nd International Conference on Automated Deduction (CADE'09)*, Lecture Notes in Artificial Intelligence, pages 355–370, Montreal, Canada, August 2009. Springer.
30. Ştefan Ciobâcă, Stéphanie Delaune, and Steve Kremer. Computing knowledge in security protocols under convergent equational theories. *Journal of Automated Reasoning*, 2011. To appear.
31. Hubert Comon-Lundh and Stéphanie Delaune. The finite variant property: How to get rid of some algebraic properties. In *Proceedings of the 16th International Conference on Rewriting Techniques and Applications (RTA'05)*, volume 3467 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2005.
32. Véronique Cortier and Stéphanie Delaune. Deciding knowledge in security protocols for monoidal equational theories. In *Proc. 14th Int. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'07)*, LNAI. Springer, 2007.
33. Véronique Cortier and Stéphanie Delaune. A method for proving observational equivalence. In *Proc. 22nd IEEE Computer Security Foundations Symposium (CSF'09)*, pages 266–276, Port Jefferson, NY, USA, July 2009. IEEE Computer Society Press.
34. Cas J.F. Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Proc. 20th International Conference on Computer Aided Verification (CAV'08)*, volume 5123 of *Lecture Notes in Computer Science*, pages 414–418. Springer, 2008.

35. Morten Dahl, Stéphanie Delaune, and Graham Steel. Formal analysis of privacy for vehicular mix-zones. In *Proc. 15th European Symposium on Research in Computer Security (ESORICS'10)*, volume 6345 of *Lecture Notes in Computer Science*, pages 55–70. Springer, 2010.
36. Morten Dahl, Stéphanie Delaune, and Graham Steel. Formal analysis of privacy for anonymous location based services. In *Proc. Workshop on Theory of Security and Applications (TOSCA'11)*, 2011. To appear.
37. Stéphanie Delaune, Steve Kremer, and Olivier Pereira. Simulation based security in the applied pi calculus. In Ravi Kannan and K. Narayan Kumar, editors, *Proceedings of the 29th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'09)*, volume 4 of *Leibniz International Proceedings in Informatics*, pages 169–180, Kanpur, India, December 2009. Leibniz-Zentrum für Informatik.
38. Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Symbolic bisimulation for the applied pi calculus. *Journal of Computer Security*, 2009. To appear.
39. Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, July 2009.
40. Stéphanie Delaune, Mark D. Ryan, and Ben Smyth. Automatic verification of privacy properties in the applied pi-calculus. In Yuecel Karabulut, John Mitchell, Peter Herrmann, and Christian Damsgaard Jensen, editors, *Proceedings of the 2nd Joint iTrust and PST Conferences on Privacy, Trust Management and Security (IFIPTM'08)*, volume 263 of *IFIP Conference Proceedings*, pages 263–278, Trondheim, Norway, June 2008. Springer.
41. Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols. In *Proc. of the 22nd Symp. on Foundations of Computer Science*, pages 350–357. IEEE Comp. Soc. Press, 1981.
42. Luca Durante, Riccardo Sisto, and Adriano Valenzano. Automatic testing equivalence verification of spi calculus specifications. *ACM Transactions on Software Engineering and Methodology*, 12(2):222–284, 2003.
43. Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-npa: Cryptographic protocol analysis modulo equational properties. In *Foundations of Security Analysis and Design V*, volume 5705 of *Lecture Notes in Computer Science*, pages 1–50. Springer, 2009.
44. Santiago Escobar, Ralf Sasse, and José Meseguer. Folding variant narrowing and optimal variant termination. *Journal of Logic and Algebraic Programming*, 81(7-8):898–928, 2012.
45. Atsushi Fujioka, Tatsuaki Okamoto, and Kazui Ohta. A practical secret voting scheme for large scale elections. In *Advances in Cryptology — AUSCRYPT '92*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992.
46. Jean Goubault-Larrecq. Deciding $\mathcal{H}_1$ by resolution. *Information Processing Letters*, 95(3):401–408, August 2005.
47. Hans Hüttel. Deciding framed bisimilarity. In *Proc. 4th International Workshop on Verification of Infinite-State Systems (INFINITY'02)*, pages 1–20, 2002.
48. Steve Kremer and Mark D. Ryan. Analysis of an electronic voting protocol in the applied pi-calculus. In *14th European Symposium on Programming (ESOP'05)*, volume 3444 of *LNCS*, pages 186–200. Springer, 2005.
49. Jia Liu and Huimin Lin. A complete symbolic bisimulation for full applied pi calculus. In *Proc. 36th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'10)*, volume 5901 of *Lecture Notes in Computer Science*, pages 552–563. Springer, 2010.
50. Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, volume 1055 of *LNCS*, pages 147–166. Springer-Verlag, 1996.
51. Paliath Narendran, Frank Pfenning, and Richard Statman. On the unification problem for cartesian closed categories. *J. Symb. Log.*, 62(2):636–647, 1997.
52. Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Proc. 5th Int. Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*. Springer, 1997.
53. Sonia Santiago, Santiago Escobar, Catherine Meadows, and José Meseguer. A formal definition of protocol indistinguishability and its verification using maude-npa. In *Proc. 10th International Workshop on Security and Trust Management (STM'14)*, volume 8743 of *Lecture Notes in Computer Science*, pages 162–177. Springer, 2014.
54. Alwen Tiu and Jeremy Dawson. Automating open bisimulation checking for the spi-calculus. In *Proc. 23rd Computer Security Foundations Symposium (CSF'10)*, pages 307–321. IEEE Comp. Soc. Press, 2010.
55. Christoph Weidenbach. Towards an automatic analysis of security protocols in first-order logic. In *Proc. 16th International Conference on Automated Deduction (CADE'99)*, volume 1632 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 1999.

A Optimally reducing convergent rewrite theories and static equivalence

In this section, we shall sketch the proof of undecidability of static equivalence for equational theories when the equational theory form an optimally reducing convergent rewrite system. We shall first show that the deducibility relation is undecidable. Then we can use the results of [1] to conclude that checking static equivalence is also undecidable.

Lemma 3. *Given a signature $\mathcal{F}$, an optimally reducing convergent rewrite system R over $\mathcal{F}$, a frame φ that maps elements of $\mathcal{W}$ to elements of $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M}}$ and a term $t \in T_{\mathcal{F},\mathcal{N},\mathcal{M}}$, the problem of checking whether there exists a recipe $r \in \mathcal{T}_{\mathcal{F},\mathcal{M},\mathcal{W}}$ such that $\varphi \vdash^r t$ is undecidable.*

Proof. (Sketch.) We shall reduce the problem of checking whether a 2-counter deterministic Minsky machine halts on empty inputs to our problem. Recall that a 2-counter Minsky Machine is tuple $\mathcal{CM} = (Q, q_s, q_f, \delta^1_{inc}, \delta^2_{inc}, \delta^1_{jzdec}, \delta^2_{jzdec})$ where

- Q is a finite set of control states.
- $q_s \in Q$ is the initial state.
- $q_f \in Q$ is the final state.
- $\delta^i_{inc} \subseteq Q \times Q$ is the increment of counter i for $i = 1, 2$.
- $\delta^i_{jzdec} \subseteq Q \times Q \times Q$ is the conditional jump of counter i for $i = 1, 2$.

$\mathcal{CM}$ is said to be deterministic if from each state q, there is almost transition out of q. The semantics of $\mathcal{CM}$ is defined in terms of a transition system $(Conf, (q_s, 0, 0), \rightarrow)$ where $Conf = Q \times \mathbb{N} \times \mathbb{N}$ is the set of configurations, $(q_s, 0, 0)$ is the initial configuration and $\rightarrow$ is defined as follows:

$(q, i, j) \rightarrow (q', i+1, j)$ if $(q, q') \in \delta^1_{inc}$,
$(q, i, j) \rightarrow (q', i, j+1)$ if $(q, q') \in \delta^2_{inc}$,
$(q, i, j) \rightarrow (q', i, j)$ if $i = 0$ and $(q, q', q'') \in \delta^1_{jzdec}$,
$(q, i, j) \rightarrow (q'', i-1, j)$ if $i \neq 0$ and $(q, q', q'') \in \delta^1_{jzdec}$,
$(q, i, j) \rightarrow (q', i, j)$ if $j = 0$ and $(q, q', q'') \in \delta^2_{jzdec}$, and
$(q, i, j) \rightarrow (q'', i, j-1)$ if $j \neq 0$ and $(q, q', q'') \in \delta^2_{jzdec}$.

A sequence of configurations $s_0, s_1, \ldots s_k$ is said to be a computation of $\mathcal{CM}$ is $s_0 = (q_s, 0, 0)$ and $s_i \rightarrow s_{i+1}$ for $i = 0, 1, \ldots k-1$. A computation $s_0, s_1, \ldots s_k$ is sad to be a halting computation of $\mathcal{CM}$ if $s_k = (q_f, 0, 0)$.

The halting problem asks given a 2-counter machine $\mathcal{CM}$ if there is a halting computation of $\mathcal{CM}$.

Now, given a deterministic 2-counter $\mathcal{CM} = (Q, q_s, q_f, \delta^1_{inc}, \delta^2_{inc}, \delta^1_{jzdec}, \delta^2_{jzdec})$, we construct $\mathcal{F}, \mathsf{R}, \varphi, t$ as follows. For each $q \in Q$, $\mathcal{F}$ will consist of a unary symbol q. In addition, $\mathcal{F}$ will consist of a 4-ary symbol h, a 0-ary symbol Z, a unary symbol s and a unary symbol Run. Intuitively a term $h(q, x, y, w)$ will stand for a configuration of the counter machine $\mathcal{CM}$ with q as the control state, x and y as the contents of the two counters and w some auxiliary information. The term Z will stand for the natural number 0. The term $s(x)$ will stand for increment of counter x. Finally $\mathsf{Run}(\cdot)$ will encode a command to execute one step of the counter machine.

The rewrite system R consists of the following rewrite rules:

$\mathsf{Run}(h(q, x, y, w)) \rightarrow h(q', s(x), y, w)$ if $(q, q') \in \delta^1_{inc}$.
$\mathsf{Run}(h(q, x, y, w)) \rightarrow h(q', x, s(y), w)$ if $(q, q') \in \delta^2_{inc}$.
$\mathsf{Run}(h(q, Z, y, w)) \rightarrow h(q', Z, y, w)$ if $(q, q', q'') \in \delta^1_{jzdec}$.
$\mathsf{Run}(h(q, s(x), y, w)) \rightarrow h(q'', x, y, w)$ if $(q, q', q'') \in \delta^1_{jzdec}$.
$\mathsf{Run}(h(q, x, Z, w)) \rightarrow h(q', x, Z, w)$ if $(q, q', q'') \in \delta^2_{jzdec}$.
$\mathsf{Run}(h(q, x, s(y), w)) \rightarrow h(q'', x, y, w)$ if $(q, q', q'') \in \delta^2_{jzdec}$.

R can be shown to be convergent and optimally reducing. Fix a secret name $n \in \mathcal{N}$ and let $\varphi = \{ w_1 \mapsto h(q_s, Z, Z, n) \}$ and let t be $h(q_f, Z, Z, n)$. Now, it can be shown that for each configuration (q, i, j) of $\mathcal{CM}$ there is a computation $s_0, s_1, \ldots s_k$ such that $s_k = (q, i, j)$ iff there is a recipe r_q such that $\varphi \vdash^{r_q} h(q, s^i(Z), s^j(Z), n)$. It follows that there is is an recipe r such that $\varphi \vdash^r t$ if and only if $\mathcal{CM}$ halts.

It is shown in [1] that given a signature $\mathcal{F}$ and a rewrite system R, the problem of checking whether a term is deducible from a frame can be reduced to the problem of checking of static equivalence of two frames by just adding one unary function symbol to the signature, while keeping the same equational rewrite system R over the extended signature. This yields:

Corollary 1. *Given a signature $\mathcal{F}$, an optimally reducing convergent rewrite system R over $\mathcal{F}$, frame φ_1 and $varphi_2$ that map elements of $\mathcal{W}$ to elements of $\mathcal{T}_{\mathcal{F},\mathcal{N},\mathcal{M}}$, the problem of checking whether $\varphi_1 \approx_s \varphi_2$ is undecidable.*

B Proof of Theorem 2: Soundness and completeness of the set of seed Statements

We prove soundness (see Lemma 4 and Proposition 4) and completeness (see Lemma 5) for the set of seed statements.

Lemma 4 (Soundness of the set of seed statements). *Let T be a ground trace. For any statement f in the set of seed statements* $\mathsf{seed}(T)$ *we have that $T \models f$.*

Proof. We suppose the same naming conventions for T as in the definition of the set of seed statements (see Section 4.1). We prove that for each statement $f \in \mathsf{seed}(T)$ we have that $T \models f$. There are four kinds of seed statements (see Figure 2) which we consider one-by-one.

1. Let m be such that $0 \leq m \leq n$, let σ and τ be substitutions such that $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k \in T(m)})$ and $\tau \in \mathrm{variants}(l_1\sigma, \ldots, l_m\sigma)$. We show that

$$f = \Big((\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow} \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)\}_{j \in R(m)}) \Big)$$

   is a statement that is true in T.
   Let ω be an arbitrary substitution grounding for f. Assume furthermore that $T \models (\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow))\omega$ for all $j \in R(m)$. We show that $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow})\omega$. In fact we will show a stronger statement. In particular, we to show that

$$T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_p\sigma\tau\downarrow})\omega$$

   for all $0 \leq p \leq m$. We proceed by induction on p.
   Base case: $p = 0$. We have $(\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_p\sigma\tau\downarrow})\omega = \mathsf{r}.$ and $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_p\sigma\tau\downarrow})\omega$ trivially.
   Inductive case: $p > 0$. We assume that $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{p-1}\sigma\tau\downarrow})\omega$ and we show that $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_p\sigma\tau\downarrow})\omega$ by case analysis on a_p. Before, we do the case analysis, let us first fix some notations.
   Let $T_1 = T$ and $\varphi_1 = \varphi$. As $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{p-1}\sigma\tau\downarrow})\omega$, we have that there exist $L_1, \ldots, L_{p-1}$ such that

$$(T_i, \varphi_i) \xrightarrow{L_i} (T_{i+1}, \varphi_{i+1})$$

   and $L_i\varphi_i =_{\mathsf{R}} \ell_i\sigma\tau\downarrow\omega$ for all $1 \leq i < p$, where $T_i = (a_i. \ldots . a_n)\{x_j \mapsto x_j\sigma\tau\downarrow\omega\}_{j \in R(i-1)}$ and where φ_i extends φ_{i-1} (for all $1 < i \leq p$). We can now do the case analysis.
   (a) if $a_p = \mathbf{out}(c_p, t_p)$, then $\ell_p = \mathbf{out}(c_p)$ by definition. Let $T_{p+1} = (a_{p+1}. \ldots . a_n)\{x_j \mapsto x_j\sigma\tau\downarrow\omega\}_{j \in R(p)}$ and let $\varphi_{p+1} = \varphi_p \cup \{w_{dom(\varphi_p)+1} \mapsto t_p\sigma\tau\downarrow\omega\}$. Let $L_p = \mathbf{out}(c_p)$. By the definition, we have that

$$(T_p, \varphi_p) \xrightarrow{L_p} (T_{p+1}, \varphi_{p+1}),$$

   which is was we wanted to prove.
   (b) if $a_p = [s_p \stackrel{?}{=} t_p]$, then $\ell_p = \mathbf{test}$. Let $T_{p+1} = (a_{p+1}. \ldots . a_n)\{x_j \mapsto x_j\sigma\tau\downarrow\omega\}_{j \in R(p)}$ and let $\varphi_{p+1} = \varphi_p$. As $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k \in T(m)})$, we have that $s_p\sigma =_{\mathsf{R}} t_p\sigma$ and therefore $s_p\sigma\tau\downarrow\omega =_{\mathsf{R}} t_p\sigma\tau\downarrow\omega$. Hence,

$$(T_p, \varphi_p) \xrightarrow{\mathbf{test}} (T_{p+1}, \varphi_{p+1}),$$

   as we wanted to prove.
   (c) If $a_p = \mathbf{in}(c_p, x_p)$, we know that $p \in R(p)$. Let $T_{p+1} = (a_{p+1}. \ldots . a_n)\{x_j \mapsto x_j\sigma\tau\downarrow\omega\}_{j \in R(p)}$ and let $\varphi_{p+1} = \varphi_p$. As $p \in R(p)$, we have that $T \models (\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{p-1}\sigma\tau\downarrow}(X_p, x_p\sigma\tau\downarrow))\omega$ (this is an antecendent of f). Therefore $\varphi_p \vdash^{X_p\omega} x_p\sigma\tau\downarrow\omega$ and, by letting $L_p = \mathbf{in}(c_p, x_p\sigma\tau\downarrow\omega)$, we obtain by the definition of $\rightarrow$ that

$$(T_p, \varphi_p) \xrightarrow{L_p} (T_{p+1}, \varphi_{p+1}),$$

   which is what we wanted to prove.

We have shown that $T \models (\mathsf{r}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_p\sigma\tau\downarrow})\omega$.

2. Let $m \in S(n)$, $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k\in T(m)})$ and $\tau \in \mathrm{variants}(t_m)$. We show that the statement

$$f = \Big((\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_{|S(m)|}, (t_m\sigma\tau)\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)\}_{j\in R(m)})\Big)$$

holds in T.
Let ω be a substitution grounding for f. We assume that

$$T \models (\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow))\omega$$

for all $j \in R(m)$ and we show that $T \models (\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_{|S(m)|}, (t_m\sigma\tau)\downarrow))\omega$.
Let $T_i = (a_i.\ldots.a_n)\{x_j \mapsto x_j\sigma\tau\omega\}_{j\in R(i-1)}$ and $\varphi_i = \cup_{1\le j\le|S(i-1)|}\{w_j \mapsto t_{o(j)}\sigma\tau\omega\}$, where $o(j) = \min\{x \mid |S(x)| = j\}$, *i.e.* $o(j)$ denotes the index of the jth send action.
We distinguish two cases:

(a) if there exist $L_1, \ldots, L_m$ such that $(T_1, \varphi_1) \xrightarrow{L_1} (T_2, \varphi_2) \xrightarrow{L_2} \ldots \xrightarrow{L_m} (T_{m+1}, \varphi_{m+1})$ and $L_i\varphi_i =_{\mathsf{R}} l_i\sigma\tau\downarrow\omega$ for all $1 \le i \le m$, we have that

$$\varphi_m(w_{|S(m)|}) = t_{o(S(m))}\sigma\tau\omega = t_m\sigma\tau\omega$$

and we have that $\varphi \vdash^{w_{|S(m)|}} t_m\sigma\tau\omega$ and therefore $\varphi \vdash^{w_{|S(m)|}} (t_m\sigma\tau)\downarrow\omega$ which implies that $T \models (\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_{|S(m)|}, t_m\sigma\tau\downarrow))\omega$.

(b) otherwise, we trivially have that $T \models \mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_{|S(m)|}, (t_m\sigma\tau)\downarrow)\omega$.

We have shown that $T \models f$.

3. Let c be a public name. $T \models \Big(\mathsf{k}(c, c) \Leftarrow\Big)$ trivially holds because $\emptyset \vdash^c c$.

4. Let g be a function symbol of arity k and let $\sigma \in \mathrm{variants}(g(x_1, \ldots, x_k))$. We show that the statement

$$f = \Big(\mathsf{k}(g(X_1, \ldots, X_k), g(x_1, \ldots, x_k)\sigma\downarrow) \Leftarrow \{\mathsf{k}(X_j, x_j\sigma\downarrow)\}_{j\in\{1,\ldots,k\}}\Big)$$

is true in T.
Let ω be an arbitrary substitution grounding for f. We assume that $T \models \mathsf{k}(X_j, x_j\sigma\downarrow)\omega$ for all $1 \le j \le k$ and we show that

$$T \models (\mathsf{k}(g(X_1, \ldots, X_k), g(x_1, \ldots, x_k)\sigma\downarrow))\omega.$$

We have that

$$\emptyset \vdash^{X_j\omega} x_j\sigma\downarrow\omega$$

for all $1 \le j \le k$ by our hypothesis. But this implies

$$\emptyset \vdash^{g(X_1\omega,\ldots,X_k\omega)} g(x_1\sigma\downarrow\omega, \ldots, x_k\sigma\downarrow\omega) =_{\mathsf{R}} g(x_1, \ldots, x_k)\sigma\downarrow\omega$$

which immediately implies that $T \models (\mathsf{k}(g(X_1, \ldots, X_k), g(x_1, \ldots, x_k)\sigma\downarrow))\omega$.
We have shown that $T \models f$.

We have shown for every statement $f \in \mathsf{seed}(T)$ that $T \models f$. □

Proposition 4 (Soundness of $\mathcal{H}(\,)$). *Let T be a ground trace and K be a set of statements such that for all $f \in K$ we have that $T \models f$. Then for all $f \in \mathcal{H}(K)$ we also have that $T \models f$.*

Proof. The proof of this proposition is a straightforward induction on the size of the smallest proof of $f \in \mathcal{H}(K)$.

Base case. The proof of $f \in \mathcal{H}(K)$ is obtained by applying the rule Simple Consequence when $n = 0$. We have that $f' = (H \Leftarrow) \in K$ and $f = f'\sigma$ where σ is a substitution grounding for f'. As $f' \in K$, by hypothesis, $T \models f'$. Hence, as all variables in f' are universally quantified, $T \models f'\sigma$.

Inductive case. We proceed by case distinction on the last rule which has been applied.

- Simple Consequence: We have that $f' = (H \Leftarrow B_1 \dots B_n) \in K$, σ is a substitution grounding for f' such that $f = H\sigma$ and $B_i\sigma \in \mathcal{H}(K)$ for $1 \leq i \leq n$. As $H \Leftarrow B_1 \dots B_n \in K$ we have by hypothesis that $T \models H \Leftarrow B_1 \dots B_n$ and hence $T \models (H \Leftarrow B_1 \dots B_n)\sigma$. By induction hypothesis we also have that $T \models B_i\sigma$. Hence, we conclude that $T \models H\sigma$.
- ExtendK: We have that $\mathsf{k}_u(R,t) \in \mathcal{H}(K)$. By induction hypothesis $T \models \mathsf{k}_u(R,t)$. It follows from the semantics of k that $T \models \mathsf{k}_{uv}(R,t)$.

Lemma 5. *Let T and S be traces and let φ be a frame. If $(T,\emptyset) \xrightarrow{L_1,\dots,L_n} (S,\varphi)$ then*

(A) $\mathsf{r}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow} \in \mathcal{H}(\mathsf{seed}(T))$*;*
(B) if $\varphi \vdash^R t$ *then* $\mathsf{k}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow}(R, t\downarrow) \in \mathcal{H}(\mathsf{seed}(T))$.

Proof. We prove the two statements by induction on n. We assume that the two statements hold for any index less than n and we prove them for n. As $(T,\emptyset) \xrightarrow{L_1,\dots,L_n} (S,\varphi)$, we have that

- there exists ω such that $(L_1\varphi\downarrow, \dots, L_n\varphi\downarrow) = (\ell_1, \dots, \ell_n)\omega$,
- $s_k\omega =_{\mathsf{R}} t_k\omega$ for all $k \in T(n)$.

We prove each of statements in turn:

(A) As $s_k\omega =_{\mathsf{R}} t_k\omega$ for all $k \in T(n)$, by the definition of $\mathrm{mgu}_{\mathsf{R}}$ there exists $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k \stackrel{?}{=} t_k\}_{k\in T(n)})$ such that:
  (a) $dom(\sigma) \subseteq X$,
  (b) $s_k\sigma =_{\mathsf{R}} t_k\sigma$ for all $k \in T(n)$ and
  (c) $\omega[X] =_{\mathsf{R}} (\sigma\pi)[X]$ for some substitution π
  where $X = vars(\{s_k, t_k\}_{k\in T(n)})$.
  It follows that $(\ell_1, \dots, \ell_n)\omega\downarrow = (\ell_1, \dots, \ell_n)\sigma\pi\downarrow$ for some substitution π. By the definition of variants$((\ell_1, \dots, \ell_n)\sigma)$, there exists $\tau \in$ variants$((\ell_1, \dots, \ell_n)\sigma)$ such that $(\ell_1, \dots, \ell_n)\sigma\pi\downarrow = (\ell_1, \dots, \ell_n)\sigma\tau\downarrow\tau'$ for some substitution τ'. By the definition of $\mathsf{seed}(T)$, we have that

$$f = \left(\mathsf{r}_{\ell_1\sigma\tau\downarrow,\dots,\ell_n\sigma\tau\downarrow} \Leftarrow \mathsf{k}_{\ell_1\sigma\tau\downarrow,\dots,\ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)_{j\in R(n)}\right) \in \mathsf{seed}(T).$$

  Let τ'' be the substitution that extends τ' by $\{X_j \mapsto R_j\}_{j\in R(n)}$ where R_j are recipes for $x_j\omega$.
  We have by the induction hypothesis that each antecedent of $f\tau''$ is in $\mathcal{H}(\mathsf{seed}(T))$. Therefore

$$\mathsf{r}_{\ell_1\sigma\tau\downarrow\tau'',\dots,\ell_n\sigma\tau\downarrow\tau''} = \mathsf{r}_{\ell_1\sigma\tau\downarrow\tau',\dots,\ell_n\sigma\tau\downarrow\tau'} \in \mathcal{H}(\mathsf{seed}(T)).$$

(B) By induction on R, we show that:

$$\mathsf{k}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow}(R, R\varphi\downarrow) \in \mathcal{H}(\mathsf{seed}(T))$$

  (a) If $R = c$ is a public name, and as the statement $f = \left(\mathsf{k}(c,c) \Leftarrow \right)$ is in the set of seed statements by definition, we have that $\mathsf{k}(R, R\varphi\downarrow) = \mathsf{k}(c,c) \in \mathcal{H}(\mathsf{seed}(T))$ by definition and therefore $\mathsf{k}_{L_1\varphi\downarrow,\dots,L_n\varphi\downarrow}(R, R\varphi\downarrow) \in \mathcal{H}(\mathsf{seed}(T))$ by the ExtendK rule.
  (b) If $R = w_j$, let m be the smallest index such that $|S(m)| = j$ (i.e. m is the index of the action a_m that output the content of w_j) and let t_m be the term such that $a_m = \mathbf{out}(c, t_m)$ for some channel c.
  As for item A, we choose $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k \stackrel{?}{=} t_k\}_{k\in T(m)})$ such that $(\ell_1, \dots, \ell_n)\omega\downarrow = (\ell_1, \dots, \ell_n)\sigma\pi\downarrow$ for some substitution π. Let $\tau \in$ variants$((\ell_1, \dots, \ell_m, t_m)\sigma)$ and τ' be substitutions such that $(\ell_1, \dots, \ell_m, t_m)\omega = (\ell_1, \dots, \ell_m, t_m)\sigma\tau\downarrow\tau'$.

We have by the definition of the seed knowledge base that

$$h = \left(\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_m\sigma\tau\downarrow}(w_j, t_m\sigma\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow,\ldots,\ell_{k-1}\sigma\tau\downarrow}(X_k, x_k\sigma\tau\downarrow)\}_{k\in R(m)}\right) \in \mathsf{seed}(T).$$

For $k \in R(m)$ we let R_k be recipes of $x_k\sigma\tau\downarrow\tau' =_{\mathsf{R}} x_k\omega$ in the smallest possible prefix of φ. Let $\tau'' = \tau' \cup \{X_k \mapsto R_k\}_{k\in R(m)}$. We have that the antecedents of $h\tau''$ are in $\mathcal{H}(\mathsf{seed}(T))$ by the induction hypothesis. Therefore

$$\begin{aligned}&\mathsf{k}_{\ell_1\sigma\tau\downarrow\tau'',\ldots,\ell_m\sigma\tau\downarrow\tau''}(w_j, t_m\sigma\tau\downarrow\tau'')\\ =\ &\mathsf{k}_{\ell_1\sigma\tau\downarrow\tau',\ldots,\ell_m\sigma\tau\downarrow\tau'}(w_j, t_m\sigma\tau\downarrow\tau')\\ =\ &\mathsf{k}_{\ell_1\omega\downarrow,\ldots,\ell_m\omega\downarrow}(w_j, t_m\omega\downarrow) \in \mathcal{H}(\mathsf{seed}(T)).\end{aligned}$$

But $(\ell_1, \ldots, \ell_m)\omega\downarrow$ is a prefix of $w = (\ell_1, \ldots, \ell_n)\omega\downarrow$ and therefore by the ExtendK rule $\mathsf{k}_w(w_j, t_m\omega\downarrow) = \mathsf{k}_w(R_j, R_j\varphi\downarrow) \in \mathcal{H}(\mathsf{seed}(T))$, which is what we had to prove.

(c) If $R = f(R_1, \ldots, R_k)$, let $\tau \in \text{variants}(f(y_1, \ldots, y_k))$ and τ' be such that $R\varphi\downarrow = (f(y_1, \ldots, y_k)\tau)\downarrow\tau'$. By the definition of the seed knowledge base, we have that the statement

$$g = \left(\mathsf{k}_{\ell_1,\ldots,\ell_n}(f(Y_1, \ldots, Y_k), f(y_1, \ldots, y_k)\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1,\ldots,\ell_n}(Y_j, y_j\tau\downarrow)\}_{j\in\{1,\ldots,k\}}\right) \in \mathsf{seed}(T).$$

Let $\tau'' = \omega \cup \tau' \cup \{Y_j \mapsto R_j\}_{j\in\{1,\ldots,k\}}$. We have that all antecedents of $g\tau''$ are in $\mathcal{H}(\mathsf{seed}(T))$ by the induction hypothesis. Therefore, the head of $g\tau''$ is also in $\mathcal{H}(\mathsf{seed}(T))$.

C Soundness and completeness of saturation: Proof of Theorem 3

C.1 Soundness of saturation

In this section is we prove the soundness part of Theorem 3. Soundness is an immediate consequence of Lemma 6, Lemma 8, Lemma 12, Lemma 9, Lemma 10, Lemma 11 and Lemma 13 proved below.

Lemma 6 (Soundness of canonicalization). *Let T be a ground trace. If $T \models f$ then $T \models f{\Downarrow}$.*

Proof. We will show that each canonicalization rule is sound:

1. For the RENAME rule, consider a statement
$$f = \Big(H \Leftarrow \mathsf{k}_{t_1,\ldots,t_k}(X,x), \mathsf{k}_{t_1,\ldots,t_l}(Y,x), B_1, \ldots, B_n\Big)$$
where $k \leq l$ and we show that if $T \models f$ then $T \models g$ where
$$g = \Big((H \Leftarrow \mathsf{k}_{t_1,\ldots,t_k}(X,x), B_1, \ldots, B_n)\{Y \mapsto X\}\Big)$$
Let τ be a grounding substitution for g such that $T \models \mathsf{k}_{t_1,\ldots,t_k}(X,x)\{Y \mapsto X\}\tau, B_1\{Y \mapsto X\}\tau, \ldots, B_n\{Y \mapsto X\}\tau$. We show that if $T \models f$ then $T \models H\{Y \mapsto X\}\tau$.
Let τ' be a substitution identical to τ, except for $\tau'(Y) = \tau(X)$. We will show that all the antecedents in $f\tau'$ are true in T.
Indeed, $\mathsf{k}_{t_1,\ldots,t_k}(X,x)\tau' = \mathsf{k}_{t_1,\ldots,t_k}(X,x)\{Y \mapsto X\}\tau$ holds by hypothesis. As $k \leq l$ and $T \models \mathsf{k}_{t_1,\ldots,t_k}(X,x)\tau'$, we also have that $T \models \mathsf{k}_{t_1,\ldots,t_k}(X,x)\tau' = \mathsf{k}_{t_1,\ldots,t_l}(Y,x)\tau'$. Furthermore $T \models B_1\tau' = B_1\{Y \mapsto X\}\tau, \ldots, B_n\tau' = B_n\{Y \mapsto X\}\tau$ by hypothesis. As $T \models f$, and all antecedents of $f\tau'$ are true in T, we obtain that $T \models H\tau'$. But $H\tau' = H\{Y \mapsto X\}\tau$ and therefore we have that $T \models H\{Y \mapsto X\}\tau$. As we have chosen τ arbitrarily, it follows that $T \models g$.
2. For the REMOVE rule, consider a solved statement
$$f = \Big(H \Leftarrow \mathsf{k}_{t_1,\ldots,t_k}(X,x), B_1, \ldots, B_n\Big)$$
such that the rule RENAME does not apply to f and such that $x \notin \mathit{vars}(H)$. We show that if $T \models f$ then $T \models g$ where
$$g = \Big(H \Leftarrow B_1, \ldots, B_n\Big)$$
Let τ be an arbitrary substitution such that $T \models B_1\tau, \ldots, B_n\tau$. We will show that $T \models H\tau$ and hence $T \models g$.
Let $(T_1, \varphi_1) = (T, \emptyset)$. We distinguish between two cases:
   (a) If $(T_1, \varphi_1) \xrightarrow{L_1} (T_2, \varphi_2) \xrightarrow{L_2} \ldots \xrightarrow{L_k} (T_{k+1}, \varphi_{k+1})$ such that $L_i\varphi_i = t_i\tau$ for all $1 \leq i \leq k$, we consider the substitution τ' to be identical to τ except for $\tau'(x) = (X\tau)\varphi_{k+1}$.
   As $x \notin \mathit{vars}(H)$ and because f is solved and the rule RENAME does not apply, we have that $x \notin \mathit{vars}(B_1, \ldots, B_n)$ and therefore $T \models B_1\tau' = B_1\tau, \ldots, B_n\tau' = B_n\tau$.
   Furthermore, we have that $T \models \mathsf{k}_{t_1,\ldots,t_k}(X,x)\tau'$ by the definition of k.
   As all antecedents of $f\tau'$ are true in T and $T \models f$, it follows that $T \models H\tau'$. But $H\tau = H\tau'$ since $x \notin \mathit{vars}(H)$ and therefore $T \models H\tau$.
   (b) Otherwise, we trivially have that $T \models \mathsf{k}_{t_1,\ldots,t_k}(X,x)\tau$. We have that all antecedents of $f\tau$ are true in T and therefore, as $T \models f$, it follows that $T \models H\tau$.

   We have shown that $T \models g$, therefore the rule REMOVE is sound.

We have shown that both rules for computing the canonical form are sound and therefore $T \models f{\Downarrow}$ whenever $T \models f$. □

Lemma 7 (Monotonicity of k). *Let T be a ground trace. If $T \models \mathsf{k}_u(R,t)$ then $T \models \mathsf{k}_{uv}(R,t)$.*

Proof. Immediate by the semantics of k.

Lemma 8 (Soundness of the consequence). *Let T be a ground trace and K a knowledge base. If for all $f \in K$ we have that $T \models f$, then for all $f \in \mathbf{conseq}(K)$ we have that $T \models f$.*

Proof. We show that both inference rules are sound.

For the AXIOM rule, soundness follows immediately from the semantics of k.

For the RES rule, let $f = \Big(\mathsf{k}_u(R,t) \Leftarrow B_1, \ldots, B_n\Big)$ and $g_i = \Big(B_i\sigma \Leftarrow C_1, \ldots, C_m\Big)$ for $1 \leq i \leq n$ be statements such that $T \models f$ and $T \models g_i$ $(1 \leq i \leq n)$. We will show that $T \models \Big(\mathsf{k}_u(R,t)\sigma \Leftarrow C_1, \ldots, C_m\Big)$ by letting τ be a substitution such that $T \models C_1\tau, \ldots, C_m\tau$ and proving that $T \models \mathsf{k}_u(R,t)\sigma\tau$. Indeed, as $T \models C_1\tau, \ldots, C_m\tau$ and as $T \models g_i$ $(1 \leq i \leq n)$, we have that $T \models B_i\sigma\tau$ $(1 \leq i \leq n)$. But $T \models f$ and therefore $T \models \mathsf{k}_u(R,t)\sigma\tau$ as well. By monotonicity of k (Lemma 7) we conclude that $T \models \mathsf{k}_{uv}(R,t)\sigma\tau$.

Lemma 9 (Soundness of the Resolutionsaturation rule). *Let T be a ground trace and f, g and h be defined as in the* RESOLUTION*rule. If $T \models f$ and $T \models g$ then $T \models h$.*

Proof. We consider the following statements:

$$
\begin{aligned}
f &= \Big(H \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_i}(X,t), B_1, \ldots, B_n\Big)\\
g &= \Big(\mathsf{k}_{\ell'_1,\ldots,\ell'_j}(R,t') \Leftarrow B_{n+1}, \ldots, B_m\Big)\\
h &= \Big((H \Leftarrow B_1, \ldots, B_m)\sigma\Big)
\end{aligned}
$$

with $j \leq i$ and where $\sigma = \mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_j}(R,t'), \mathsf{k}_{\ell_1,\ldots,\ell_j}(X,t))$. We will show that if $T \models f$ and $T \models g$ then $T \models h$.

Indeed, let τ be an arbitrary substitution grounding for h and assume that $T \models B_1\sigma\tau, \ldots, B_m\sigma\tau$. We will show that $T \models H\sigma\tau$. As $T \models B_{n+1}\sigma\tau, \ldots, B_m\sigma\tau$ and because $T \models g$, we have that $T \models \mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_j}(R,t')\sigma\tau$. But $\mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_j}(R,t')\sigma\tau = \mathsf{k}_{\ell_1,\ldots,\ell_j}(X,t)\sigma\tau$ by choice of $\sigma = \mathrm{mgu}(\mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_j}(R,t'), \mathsf{k}_{\ell_1,\ldots,\ell_j}(X,t)$. As $j \leq i$, it follows by Lemma 7 that $T \models \mathsf{k}_{\ell_1,\ldots,\ell_i}(X,t)\sigma\tau$ as well. As all antecedents of $f\sigma\tau$ are true in T and because $T \models f$, we have that $T \models H\sigma\tau$. As τ was chosen arbitrarily, it follows that $T \models h$. □

Lemma 10 (Soundness of the Equation saturation rule). *Let T be a ground trace and f, g and h be defined as in the* EQUATION*rule. If $T \models f$ and $T \models g$ then $T \models h$.*

Proof. We consider the following statements:

$$
\begin{aligned}
f &= \Big(\mathsf{k}_u(R,t) \Leftarrow B_1, \ldots, B_n\Big)\\
g &= \Big(\mathsf{k}_{u'v'}(R',t') \Leftarrow B_{n+1}, \ldots, B_m\Big)\\
h &= \Big((\mathsf{i}_{u'v'}(R,R') \Leftarrow B_1, \ldots, B_m)\sigma\Big)
\end{aligned}
$$

where $\sigma = \mathrm{mgu}(\mathsf{k}_u(R,t), \mathsf{k}_{u'}(R,t'))$.

We will show that if $T \models f$ and $T \models g$ then $T \models h$. Let τ be an arbitrary substitution grounding for h. We assume that $T \models B_1\sigma\tau, \ldots, B_m\sigma\tau$ and we show that $T \models \mathsf{i}_{u'v'}(R,R')\sigma\tau$. As $T \models B_1\sigma\tau, \ldots, B_n\sigma\tau$ and because $T \models f$ we have that $T \models \mathsf{k}_u(R,t)\sigma\tau$. But $\mathsf{k}_u(R,t)\sigma\tau = \mathsf{k}_{u'}(R,t)\sigma\tau$ by choice of $\sigma = \mathrm{mgu}(\mathsf{k}_u(R,t), \mathsf{k}_{u'}(R,t))$ and therefore $T \models \mathsf{k}_{u'}(R,t)\sigma\tau$. By monotonicity of k (Lemma 7) we also have that $T \models \mathsf{k}_{u'v'}(R,t)\sigma\tau$. As $T \models B_{n+1}\sigma\tau \ldots, B_m\sigma\tau$ and because $T \models g$ we also obtain that $T \models \mathsf{k}_{u'v'}(R',t')\sigma\tau$. As $T \models \mathsf{k}_{u'v'}(R,t)\sigma\tau$ and $T \models \mathsf{k}_{u'v'}(R',t')\sigma\tau$, we have by definition that $T \models \mathsf{i}_{u'v'}(R,R')\sigma\tau$.

We have shown that the head of $h\tau$ is true in T. As τ was chosen arbitrarily, it follows that h holds in T. □

Lemma 11 (Soundness of the Test saturation rule). *Let T be a ground trace and f, g, h be statements as in the* TEST*saturation rule. If $T \models f$ and $T \models g$ then $T \models h$.*

Proof. We consider the following statements:

$$f = \Big(\mathsf{i}_u(R, R') \Leftarrow B_1, \ldots, B_n\Big)$$
$$g = \Big(\mathsf{r}_{u'v'} \Leftarrow B_{n+1}, \ldots, B_m\Big)$$
$$h = \Big((\mathsf{ri}_{u'v'}(R, R') \Leftarrow B_1, \ldots, B_m)\sigma\Big)$$

where $\sigma = \mathrm{mgu}(u, u')$).

Let τ be an arbitrary substitution grounding for h. We assume that $T \models B_1\sigma\tau, \ldots, B_m\sigma\tau$ and we show that $T \models \mathsf{ri}_u(R, R')\tau$. Indeed, as $T \models B_1\sigma\tau, \ldots, B_n\sigma\tau$ and as $T \models f$, we have that $T \models \mathsf{i}_u(R, R')\sigma\tau$. As $T \models B_{n+1}\sigma\tau, \ldots, B_m\sigma\tau$ and as $T \models g$, we have that $T \models \mathsf{r}_{u'v'}\sigma\tau$.

But $\sigma = \mathrm{mgu}(u, u')$ and therefore $u\sigma\tau = u'\sigma\tau$. Hence, we immediately obtain $T \models \mathsf{ri}_{u'v'}(R, R')\sigma\tau$, which is what we wanted. As τ was chosen arbitrarily, it follows that $T \models h$.

Lemma 12 (Soundness of the update). *Let T be a ground trace and K a knowledge base. If for all $f \in K$ we have that $T \models f$ and if $T \models g$, then for any $f \in (K \oplus g)$ we have that $T \models f$.*

Proof. If $K \oplus g = K \cup \{g\Downarrow\}$, we immediately conclude by Lemma 8. Otherwise, it must be that

$$g\Downarrow = \Big(\mathsf{k}_{\ell_1,\ldots,\ell_k}(R, t) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n)\Big)$$

for some $R, t, \ell_1, \ldots, \ell_k, i_1, \ldots, i_n, X_1, \ldots, X_n, x_1, \ldots, x_n$ and $K \oplus g = K \cup \{h\}$, where

$$h = \Big(\mathsf{i}_{\ell_1,\ldots,\ell_k}(R, R') \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n)\Big)$$

and where

$$g' = \Big(\mathsf{k}_{\ell_1,\ldots,\ell_k}(R', t) \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n)\Big) \in \mathbf{conseq}(K_{\mathsf{solved}}).$$

It is sufficient to show that $T \models h$. As $K_{\mathsf{solved}} \subseteq K$, it immediately follows that $g' \in \mathbf{conseq}(K)$ and, by Lemma 8, $T \models g'$. We now show that $T \models h$. Let τ be an arbitrary substitution grounding for h such that the antecedents of $h\tau$ are true in T. As the antecedents of $h\tau$ are the same as the antecedents of $g\Downarrow\tau$ and those of $g'\tau$, and as $T \models g$ and $T \models g'$ we have that $T \models \mathsf{k}_{\ell_1,\ldots,\ell_k}(R, t)\tau$ and $T \models \mathsf{k}_{\ell_1,\ldots,\ell_k}(R', t)\tau$. But this immediately implies that $T \models \mathsf{i}_{\ell_1,\ldots,\ell_k}(R, R')\tau$ (the head of $h\tau$). As τ was chosen arbitrarily, it follows that $T \models h$. □

Lemma 13. *Let T be a ground trace and K a knowledge base such that for all $f \in K$ we have that $T \models f$. Then for all $H \in \mathcal{H}_{\mathsf{e}}(K)$ we also have that $T \models H$.*

Proof. This result is proved by structural induction on the prooftree witnessing the fact that $H \in \mathcal{H}_{\mathsf{e}}(K)$. □

C.2 Completeness of saturation

In this section we prove the completeness part of Theorem 3. The first two items of the completeness are immediate consequences of Theorem 2 and Lemma 20 proved below. The third item follows from the second item, direct applications of the definition of $\mathcal{H}_{\mathsf{e}}$, Corollary 2 (proved below) and applications of the SYM and TRAN rules.

Proposition 5. *Let K be a knowledge base, $f = \left(\mathsf{k}_{uv}(R,t) \Leftarrow C_1,\ldots,C_m\right)$ a statement such that $f \in \mathbf{conseq}(K)$ and τ a substitution that is grounding for f such that $C_i\tau \in \mathcal{H}(K)$ for all $1 \leq i \leq n$. Then $\mathsf{k}_{uv}(R,t)\tau \in \mathcal{H}(K)$.*

Proof. By induction on the proof tree of $f \in \mathbf{conseq}(K)$.

- If the AXIOM rule was used, we have that $C_i = \mathsf{k}_u(R,t)$ for some i and, by hypothesis, $C_i\tau \in \mathcal{H}(K)$. Using the EXTENDK rule we conclude.
- If the RES rule was used, we have that there exists $\left(\mathsf{k}_{u'}(R',t') \Leftarrow B_1,\ldots,B_n\right) \in K$ and a substitution σ such that $\mathsf{k}_u(R,t) = \mathsf{k}_{u'}(R',t')\sigma$ and $B_i\sigma \Leftarrow C_1,\ldots,C_m \in \mathbf{conseq}(K)$ $(1 \leq i \leq n)$. By the induction hypothesis, we have that $B_i\sigma\tau \in \mathcal{H}(K)$. As $\left(\mathsf{k}_{u'}(R',t') \Leftarrow B_1,\ldots,B_n\right) \in K$, it follows that $\mathsf{k}_{u'}(R',t')\sigma\tau = \mathsf{k}_u(R,t)\tau \in \mathcal{H}(K)$. Using the EXTENDK rule we conclude.

Definition 21. *When $H \in \mathcal{H}(K)$ we define $\mathcal{S}(H,K)$ to be the* size *of the smallest proof tree of $H \in \mathcal{H}(K)$.*

Proposition 6. *Let K be a knowledge base. If $\mathsf{k}_w(R,t) \in \mathcal{H}_{\mathsf{e}}(K)$ and $\mathsf{i}_w(R,R') \in \mathcal{H}_{\mathsf{e}}(K)$, then $\mathsf{k}_w(R',t) \in \mathcal{H}_{\mathsf{e}}(K)$.*

Proof. As $\mathsf{k}_w(R,t) \in \mathcal{H}_{\mathsf{e}}(K)$, it follows that there exist R'' such that

$$\mathsf{k}_w(R'',t) \in \mathcal{H}(K) \tag{1}$$

and such that $\mathsf{i}_w(R,R'') \in \mathcal{H}_{\mathsf{e}}(K)$. But $\mathsf{i}_w(R,R') \in \mathcal{H}_{\mathsf{e}}(K)$ and therefore, by the symmetry and transitivity of $\mathsf{i}_w(\_,\_)$, we have that

$$\mathsf{i}_w(R'',R') \in \mathcal{H}_{\mathsf{e}}(K). \tag{2}$$

Using Equations 1 and 2 we immediately obtain by the definition of $\mathcal{H}_{\mathsf{e}}$ that $\mathsf{k}_w(R',T) \in \mathcal{H}_{\mathsf{e}}(K)$.

Definition 22. *We write $w \sqsubseteq w'$ whenever w is a prefix of w': i.e. there exists $\ell_1,\ldots,\ell_n$ such that $w' = \ell_1,\ldots,\ell_n$ and $w = \ell_1,\ldots,\ell_m$ for some $0 \leq m \leq n$.*

Proposition 7. *Let K be a knowledge base. If $\mathsf{k}_w(R,t) \in \mathcal{H}(K)$ (resp. $\mathsf{i}_w(R,S) \in \mathcal{H}(K)$) then there exist a statement $f = \left(\mathsf{k}_{w'}(R',t') \Leftarrow B_1,\ldots,B_m\right) \in K$ (resp. $f = \left(\mathsf{i}_{w'}(R',S') \Leftarrow B_1,\ldots,B_m\right) \in K$) and a substitution σ such that $R'\sigma = R$, $t'\sigma = t$ (resp. $S'\sigma = S$), $w'\sigma \sqsubseteq w$, $B_i\sigma \in \mathcal{H}(K)$ for all $1 \leq i \leq m$ and $\sum_{1\leq i\leq m} \mathcal{S}(B_i\sigma,K) < \mathcal{S}(\mathsf{k}_w(R,t),K)$.*

Proof. We prove the proposition by induction on the smallest proof tree of $H = \mathsf{k}_w(R,t) \in \mathcal{H}(K)$ (resp. $H = \mathsf{i}_w(R,S) \in \mathcal{H}(K)$). We proceed by case distinction on the last proof rule that has been applied.

- SIMPLE CONSEQUENCE: In this case we have that there exist a statement $f = \left(H' \Leftarrow B_1,\ldots,B_m\right) \in K$ and a substitution σ such that $H'\sigma = H$, $B_i\sigma \in \mathcal{H}(K)$ for all $1 \leq i \leq m$ and $\sum_{1\leq i\leq m} \mathcal{S}(B_i\sigma,K) + 1 = \mathcal{S}(\mathsf{k}_{w'}(R',t')\sigma,K)$. Hence we directly conclude.
- EXTENDK: In this case $H = \mathsf{k}_w(R,t)$ and we have that $w = uv$ for some u,v and $\mathsf{k}_u(R,t) \in \mathcal{H}(K)$. By induction hypothesis, we have that there exists $f = \mathsf{k}_{u'}(R',t') \Leftarrow B_1,\ldots,B_m \in K$ and σ such that $R'\sigma = R$, $t'\sigma = t$, $u'\sigma \sqsubseteq u$, $B_i\sigma \in \mathcal{H}(K)$ for all $1 \leq i \leq m$ and $\sum_{1\leq i\leq m} \mathcal{S}(B_i\sigma,K) < \mathcal{S}(\mathsf{k}_w(R,t),K)$. As $u \sqsubseteq w$, we also have that $u'\sigma \sqsubseteq w$. Moreover, $\mathcal{S}(\mathsf{k}_w(R,t) \in \mathcal{H}(K)) = \mathcal{S}(\mathsf{k}_u(R,t) \in \mathcal{H}(K)) + 1 > \sum_{1\leq i\leq m} \mathcal{S}(B_i\sigma,K)$ which allows us to conclude.

Lemma 14. *Let K be a saturated knowledge base and $f \in K$ be a statement*

$$f = \Big(H \Leftarrow B_1, \ldots, B_n\Big)$$

where H is either $\mathsf{i}_w(R, R')$, $\mathsf{ri}_w(R, R')$ or r_w. If σ is a substitution grounding for f such that $B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ for all $1 \leq i \leq n$ then we have that

$$H\sigma \in \mathcal{H}(K_{\mathsf{solved}}).$$

Proof. We prove the lemma for the case where $H = \mathsf{i}_w(R, R')$. The proof for the two other cases is similar. Let $\mathcal{G} = \sum_{i \in \{1,\ldots,n\}} \mathcal{S}(B_i\sigma, K_{\mathsf{solved}})$. We prove the lemma by induction on $\mathcal{G}$. If f is a solved statement, the conclusion is immediate by the definition of $\mathcal{H}$.

Otherwise, if f is not a solved statement, there exists some B_j $(1 \leq j \leq n)$ such that $B_j = \mathsf{k}_{w_j}(X_j, t_j)$ and $t_j \notin \mathcal{X}$.

As $B_j\sigma \in \mathcal{H}(K_{\mathsf{solved}})$, it follows by Proposition 7 that $w_j = u_j v_j$ for some u_j, v_j and that there exists

$$g = \Big(\mathsf{k}_{u'_j}(R'_j, t'_j) \Leftarrow B_{n+1}, \ldots, B_m\Big) \in K_{\mathsf{solved}}$$

and a substitution σ' grounding for g such that $B_{n+1}\sigma', \ldots, B_m\sigma' \in \mathcal{H}(K_{\mathsf{solved}})$, $R'_j\sigma' = X_j\sigma$, $t'_j\sigma' = t_j\sigma$, $u'_j\sigma' = u_j\sigma$ and $\mathcal{S}(B_j\sigma) = 1 + \sum_{i \in \{n+1,\ldots,m\}} \mathcal{S}(B_i\sigma')$.

As $\omega = \sigma \cup \sigma'$ is a unifier of $H = \mathsf{k}_{u'_j}(R'_j, t'_j)$ and $\mathsf{k}_{u_j}(X_j, t_j)$, it follows that the two terms are unifiable. Let $\tau = \mathrm{mgu}(H, \mathsf{k}_{u_j}(X_j, t_j))$ denote their most general unifier. As K is saturated, it follows that the RESOLUTIONsaturation rule was applied to f and g and therefore the resulting equational statement

$$h = \Big(\mathsf{i}_w(R, R') \Leftarrow B_1, \ldots, B_{j-1}, B_{j+1}, \ldots, B_m\Big)\tau \in K$$

must be in K (by the update function, equational statements are added to the knowledge base).

As ω is a unifier of H and B_j and as $\tau = \mathrm{mgu}(H, \mathsf{k}_{u_j}(X_j, t_j))$, it follows that there exists ω' such that $\omega = \tau\omega'$. We have that ω' is a substitution grounding for h, that

$$B_i\tau\omega' \in \mathcal{H}(K_{\mathsf{solved}})$$

for $i \in \{1, \ldots, j-1, j+1, \ldots, m\}$ and that $\sum_{i \in \{1,\ldots,j-1,j+1,\ldots,m\}} \mathcal{S}(B_i\tau\omega') = \mathcal{G} - 1$.

Therefore we can apply the induction hypothesis to h and ω' and conclude.

Lemma 15. *Let K be a saturated knowledge base. If $\mathsf{r}_u \in \mathcal{H}(K_{\mathsf{solved}})$, $\mathsf{i}_{u'}(R, R') \in \mathcal{H}(K_{\mathsf{solved}})$ and $u' \sqsubseteq u$, then $\mathsf{ri}_u(R, R') \in \mathcal{H}(K_{\mathsf{solved}})$.*

Proof. As $\mathsf{r}_u \in \mathcal{H}(K_{\mathsf{solved}})$, there exists a solved statement $f = \Big(\mathsf{r}_v \Leftarrow B_1, \ldots, B_n\Big) \in K_{\mathsf{solved}}$ and a substitution σ grounding for f such that $B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ for all $1 \leq i \leq n$ and such that $u = v\sigma$.

As $\mathsf{i}_{u'}(R, R') \in \mathcal{H}(K_{\mathsf{solved}})$, there exists by Proposition 7 a solved statement $g = \Big(\mathsf{i}_w(T, T') \Leftarrow B_{n+1}, \ldots, B_m\Big)$ and a substitution τ grounding for g such that $B_i\tau \in \mathcal{H}(K_{\mathsf{solved}})$ for all $n+1 \leq i \leq m$ and such that $u \sqsupseteq u' \sqsupseteq w\tau$, $R = T\tau$ and $R' = T'\tau$.

As $v\sigma = u \sqsupseteq w\tau$, it follows that $v = v_0 v_1$ such that v_0 and w are unifiable ($\sigma \cup \tau$ is such a unifier). Let $\omega = \mathrm{mgu}(v_0, w)$ and let π be such that $\sigma \cup \tau = \omega \circ \pi$.

As the knowledge base is saturated, the TESTsaturation rule must have fired for f and g and therefore K must have been updated by h where

$$h = \Big((\mathsf{ri}_v(T, T') \Leftarrow B_1, \ldots, B_m)\omega\Big).$$

But as h is not a deduction fact, the update must have simply added h to K and therefore $h \in K$.

We have that $B_i\omega\pi = B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ for all $1 \leq i \leq n$ and that $B_i\omega\pi = B_i\tau \in \mathcal{H}(K_{\mathsf{solved}})$ for all $n+1 \leq i \leq m$. By applying Lemma 14 to the statement h and the substitution π, we obtain that $\mathsf{ri}_v(T, T')\omega\pi = \mathsf{ri}_u(R, R') \in \mathcal{H}(K_{\mathsf{solved}})$. □

Lemma 16. *Let K be a saturated knowledge base. If $\mathsf{k}_u(R,t) \in \mathcal{H}(K_{\mathsf{solved}})$ and $\mathsf{k}_{uv}(R',t) \in \mathcal{H}(K_{\mathsf{solved}})$ then $\mathsf{i}_w(R,R') \in \mathcal{H}(K_{\mathsf{solved}})$ for some $w \sqsubseteq uv$.*

Proof. Let $u = \ell_1, \ldots, \ell_k$ and $v = \ell_{k+1}, \ldots, \ell_l$. As $\mathsf{k}_u(R,t) \in \mathcal{H}(K_{\mathsf{solved}})$, it follows by Proposition 7 that there exist

$$f = \Big(\mathsf{k}_w(S,s) \Leftarrow B_1, \ldots, B_n\Big) \in K_{\mathsf{solved}}$$

and a substitution σ grounding for f such that $B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ $(1 \leq i \leq n)$ and $\mathsf{k}_w(S,s)\sigma = \mathsf{k}_{u'}(R,t)$ for some $u' \sqsubseteq u$ a prefix of u.

Similarly, as $\mathsf{k}_{uv}(R',t) \in \mathcal{H}(K_{\mathsf{solved}})$, it follows that there exist

$$f' = \Big(\mathsf{k}_{w'}(S',s') \Leftarrow B'_1, \ldots, B'_m\Big) \in K_{\mathsf{solved}}$$

and a substitution σ' grounding for f' such that $B'_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ $(1 \leq i \leq m)$ and $\mathsf{k}_{w'}(S',s')\sigma' = \mathsf{k}_{u''}(R',t)$ for $u'' \sqsubseteq uv$ a prefix of uv.

We have that $w\sigma \sqsubseteq u$, which trivially implies $w\sigma \sqsubseteq uv$. We also have $w'\sigma' \sqsubseteq uv$. Let $w = \ell'_1, \ldots, \ell'_p$ and $w' = \ell''_1, \ldots, \ell''_q$ and let $r = \max\{p,q\}$. We have that $(\ell'_1, \ldots, \ell'_r)\sigma = (\ell''_1, \ldots, \ell''_r)\sigma'$.

We have that $\sigma \cup \sigma'$ is a unifier of $\mathsf{k}_{\ell'_1,\ldots,\ell'_r}(\_,s)$ and $\mathsf{k}_{\ell''_1,\ldots,\ell''_r}(\_,s')$, it follows that $\tau = \mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_r}(\_,s), \mathsf{k}_{\ell''_1,\ldots,\ell''_r}(\_,s'))$ exists. As K is saturated, it follows that the equational fact

$$h = \Big(\mathsf{i}_{\ell'_1,\ldots,\ell'_r}(S,S') \Leftarrow B_1, \ldots, B_n, B'_1, \ldots, B'_m\Big)\tau \in K$$

resulting from applying the EQUATIONsaturation rule to f and f' is in K.

As $\sigma \cup \sigma'$ is a unifier of $\mathsf{k}_{\ell'_1,\ldots,\ell'_r}(\_,s)$ and $\mathsf{k}_{\ell''_1,\ldots,\ell''_r}(\_,s')$ and as $\tau = \mathrm{mgu}(\mathsf{k}_{\ell'_1,\ldots,\ell'_r}(\_,s), \mathsf{k}_{\ell''_1,\ldots,\ell''_r}(\_,s'))$, it follows that there exists ω such that $\sigma \cup \sigma' = \tau\omega$.

We have that ω is grounding for h and that $B_1\tau\omega, \ldots, B_n\tau\omega, B'_1\tau\omega, \ldots, B'_m\tau\omega \in \mathcal{H}(K_{\mathsf{solved}})$. Therefore, we have by Lemma 14 that

$$\mathsf{i}_{\ell'_1,\ldots,\ell'_r}(S,S')\tau\omega = \mathsf{i}_{\ell'_1\sigma,\ldots,\ell'_r\sigma}(R,R') \in \mathcal{H}(K_{\mathsf{solved}}).$$

As $(\ell'_1, \ldots, \ell'_r)\sigma$ is a prefix of uv we conclude.

Corollary 2. *Let K be a saturated knowledge base. If $\mathsf{k}_u(R,t) \in \mathcal{H}(K_{\mathsf{solved}})$ and $\mathsf{k}_{uv}(R',t) \in \mathcal{H}(K_{\mathsf{solved}})$ then $\mathsf{i}_{uv}(R,R') \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$.*

Proof. The corollary follows from Lemma 16 by the EXTEND rule of the definition of $\mathcal{H}_{\mathsf{e}}$.

Lemma 17. *Let K be a saturated knowledge base, let*

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow B_1, \ldots, B_n\Big)$$

be a statement such that $f{\Downarrow} \in K_{\mathsf{solved}}$ and let σ be a substitution grounding for f such that $B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ for all $1 \leq i \leq n$. Then we have that

$$(\mathsf{k}_w(R,t))\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}).$$

Proof. We prove this by induction on the number of canonicalization steps.

If f is already in canonical form, then the conclusion is immediately true by definition of $\mathcal{H}$. Otherwise, there must be a canonicalization rule which can be applied to f. We distinguish between two cases:

1. If the RENAME canonicalization rule can be applied, then f must be of the form:

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), \mathsf{k}_{uv}(Y,x), B_3, \ldots, B_n\Big).$$

Let us consider the statement f' obtained by applying RENAME to f:

$$f' = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), B_3, \dots B_n\Big)\{Y \mapsto X\}.$$

By the definition of a statement, Y has at most one occurence in $B_1, \dots, B_n$ and therefore we have that $(B_1, B_3, \dots B_n)\{Y \mapsto X\} = (B_1, B_3, \dots, B_n)$. Therefore $(B_1, B_3, \dots B_n)\{Y \mapsto X\}\sigma = (B_1, B_3, \dots, B_n)\sigma$. We can therefore apply the induction hypothesis on f' and σ to obtain that

$$\mathsf{k}_w(R,t)\{Y \mapsto X\}\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}). \tag{3}$$

But $\mathsf{k}_u(X,x)\sigma \in \mathcal{H}(K_\mathsf{solved})$ and $\mathsf{k}_{uv}(Y,x)\sigma \in \mathcal{H}(K_\mathsf{solved})$. By Lemma 2, we have that

$$\mathsf{i}_{uv}(X,Y)\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}). \tag{4}$$

From Equation 3 and Equation 4 and as uv is a prefix of w by the definition of a statement, we conclude by Proposition 6 that

$$\mathsf{k}_w(R,t)\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}).$$

2. If the REMOVE canonicalization rule can be applied, then f must be of the form:

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), B_2, \dots, B_n\Big).$$

Let f' be the statement obtained from f by applying REMOVE . We have that

$$f' = \Big(\mathsf{k}_w(R,t) \Leftarrow B_2, \dots, B_n\Big).$$

By applying the induction hypothesis on f' and σ, we immediately obtain our conclusion:

$$\mathsf{k}_w(R,t)\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}).$$

Lemma 18. *Let K be a saturated knowledge base, let*

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow B_1, \dots, B_n\Big)$$

be a statement such that $f\Downarrow = \Big(\mathsf{k}_w(R',t) \Leftarrow C_1, \dots, C_m\Big)$ for some $R', C_1, \dots, C_m$ and let R'' be a recipe such that

$$g = \Big(\mathsf{k}_w(R'',t) \Leftarrow C_1, \dots, C_m\Big) \in \mathbf{conseq}(K_\mathsf{solved})$$

and such that

$$h = \Big(\mathsf{i}_w(R'',R') \Leftarrow C_1, \dots, C_m\Big) \in K_\mathsf{solved}.$$

Let σ be a substitution grounding for f such that $B_i\sigma \in \mathcal{H}(K_\mathsf{solved})$ for all $1 \leq i \leq n$. Then we have that

$$(\mathsf{k}_w(R,t))\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}).$$

Proof. We prove the lemma by induction on the number of steps to reach the canonical form.

If f is already in canonical form we have that $B_1, \dots, B_n = C_1, \dots, C_m$ and, by applying Proposition 5 to g and σ, we have that

$$\mathsf{k}_w(R',t)\sigma \in \mathcal{H}(K_\mathsf{solved}).$$

Furthermore, as $h \in K_\mathsf{solved}$ and as all antecedents $B_1\sigma, \dots, B_n\sigma = C_1\sigma, \dots, C_m\sigma$ of $h\sigma$ are in $\mathcal{H}(K_\mathsf{solved})$, we have that

$$\mathsf{i}_w(R'',R')\sigma \in \mathcal{H}(K_\mathsf{solved}).$$

It immediately follows that

$$\mathsf{k}_w(R'',t)\sigma \in \mathcal{H}_\mathsf{e}(K_\mathsf{solved}),$$

which is what we had the prove.

Otherwise, there must be a canonicalization rule which can be applied to f. We distinguish between two cases:

1. If the RENAME canonicalization rule can be applied, then f must be of the form:

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), \mathsf{k}_{uv}(Y,x), B_3, \ldots, B_n\Big).$$

Let us consider the statement f' obtained by applying RENAME to f:

$$f' = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), B_3, \ldots B_n\Big)\{Y \mapsto X\}.$$

By the definition of a statement, Y has at most one occurence in $B_1, \ldots, B_n$ and therefore we have that $(B_1, B_3, \ldots B_n)\{Y \mapsto X\} = (B_1, B_3, \ldots, B_n)$. Therefore $(B_1, B_3, \ldots B_n)\{Y \mapsto X\}\sigma = (B_1, B_3, \ldots, B_n)\sigma$. We can therefore apply the induction hypothesis on f' and σ to obtain that

$$\mathsf{k}_w(R,t)\{Y \mapsto X\}\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}). \tag{5}$$

But $\mathsf{k}_u(X,x)\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ and $\mathsf{k}_{uv}(Y,x)\sigma \in \mathcal{H}(K_{\mathsf{solved}})$. By Lemma 2, we have that

$$\mathsf{i}_{uv}(X,Y)\sigma \in \mathcal{H}(K_{\mathsf{solved}}). \tag{6}$$

From Equation 5 and Equation 6 and as uv is a prefix of w by the definition of a statement, we conclude by Proposition 6 that

$$\mathsf{k}_w(R,t)\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}).$$

2. If the REMOVE canonicalization rule can be applied, then f must be of the form:

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_u(X,x), B_2, \ldots, B_n\Big).$$

Let f' be the statement obtained from f by applying REMOVE . We have that

$$f' = \Big(\mathsf{k}_w(R,t) \Leftarrow B_2, \ldots, B_n\Big).$$

By applying the induction hypothesis on f' and σ, we immediately obtain our conclusion:

$$\mathsf{k}_w(R,t)\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}).$$

Lemma 19. *Let K be a saturated knowledge base, let $f \in K$ be a statement*

$$f = \Big(\mathsf{k}_w(R,t) \Leftarrow B_1, \ldots, B_n\Big)$$

and let σ be a substitution grounding for f such that $B_i\sigma \in \mathcal{H}(K_{\mathsf{solved}})$ for all $1 \leq i \leq n$. Then we have that

$$(\mathsf{k}_w(R,t))\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}).$$

Proof. Let $\mathcal{G} = \sum_{i\in\{1,\ldots,n\}} \mathcal{S}(B_i\sigma, K_{\mathsf{solved}})$. We prove the lemma by induction on $\mathcal{G}$.

If f is a solved statement, the conclusion is trivial by the definitions of $\mathcal{H}$, $\mathcal{H}_{\mathsf{e}}$.

Otherwise, there exists some $B_j = \mathsf{k}_{w_j}(X_j, t_j)$ (with $1 \leq j \leq n$) such that $t_j \notin \mathcal{X}$.

As $B_j\sigma \in \mathcal{H}(K_{\mathsf{solved}})$, we have by Proposition 7 that there exist

$$g = \Big(\mathsf{k}_{u'}(R',t') \Leftarrow B'_1, \ldots, B'_m\Big) \in K_{\mathsf{solved}},$$

a substitution σ' grounding for g such that $B'_1\sigma', \ldots, B'_m\sigma' \in \mathcal{H}(K_{\mathsf{solved}})$, $\mathsf{k}_{u'}(R',t')\sigma' = \mathsf{k}_u(X_j,t_j)\sigma$ for some prefix $u \sqsubseteq w_j$ of w_j and $\mathcal{S}(B_j\sigma, K_{\mathsf{solved}}) > \sum_{i\in\{1,\ldots,m\}} \mathcal{S}(B'_i\sigma', K_{\mathsf{solved}})$.

As $\sigma \cup \sigma'$ is a unifier of $\mathsf{k}_u(X_j,t_j)$ and $\mathsf{k}_{u'}(R',t')$, it follows that $\tau = \mathrm{mgu}(\mathsf{k}_u(X_j,t_j), \mathsf{k}_{u'}(R',t'))$ exists. Let $\sigma \cup \sigma'$ must be an instance of the most general unifier, let ω be a substitution such that $\sigma \cup \sigma' = \tau\omega$.

As K is saturated, it follows that the RESOLUTIONsaturation rule was applied to f and g. Let h be the resulting statement:

$$h = \Big(\mathsf{k}_w(R,t) \Leftarrow B_1, \ldots, B_{j-1}, B_{j+1}, \ldots, B_n, B'_1, \ldots, B'_m\Big)\tau.$$

We distinguish two cases:

1. if h is not solved we have that $h \in K$ by the update function (as K is saturated).
   We can therefore apply the induction hypothesis on h and on the substitution ω to immediately conclude.
2. if h is solved, we distinguish two cases:
   (a) either $h\Downarrow \in K$, in which case we conclude by applying Lemma 17 to h and ω.
   (b) or $h\Downarrow = \Big(\mathsf{k}_w(R'', t) \Leftarrow C_1, \ldots, C_k\Big)$ and

$$h' = \Big(\mathsf{k}_w(R''', t) \Leftarrow C_1, \ldots, C_k\Big) \in \mathbf{conseq}(K_{\mathsf{solved}})$$

   and

$$h'' = \Big(\mathsf{i}_w(R''', R'') \Leftarrow C_1, \ldots, C_k\Big) \in K_{\mathsf{solved}}$$

   for some R''', in which case we conclude by applying Lemma 18.

Proposition 8. *If* $\mathsf{k}_u(R, t) \in \mathcal{H}_{\mathsf{e}}(K)$ *then* $\mathsf{k}_{uv}(R, t) \in \mathcal{H}_{\mathsf{e}}(K)$.

Proof. As $\mathsf{k}_u(R, t) \in \mathcal{H}_{\mathsf{e}}(K)$, it follows that $\mathsf{k}_u(R', t) \in \mathcal{H}(K)$ and $\mathsf{i}_u(R', R) \in \mathcal{H}_{\mathsf{e}}(K)$ for some R'. By the ExtendK rule, we have that $\mathsf{k}_{uv}(R', t) \in \mathcal{H}(K)$ and by the Extend rule, we have that $\mathsf{i}_{uv}(R', R) \in \mathcal{H}_{\mathsf{e}}(K)$. We conclude by rule Equational Consequence that $\mathsf{k}_{uv}(R, t) \in \mathcal{H}_{\mathsf{e}}(K)$, which is what we had to show.

Lemma 20. *Let S be a set of seed statements and let K be the saturation of the knowledge base obtained by updating the empty knowledge base by each statement in S. Then $\mathcal{H}(S) \subseteq \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$*

Proof. Let $H \in \mathcal{H}(S)$. We will prove by induction on the proof tree of $H \in \mathcal{H}(S)$ that each node of the tree is in $\mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$. We proceed by case distinction on the last rule that has been applied to derive H.

1. ExtendK: we have that $H = \mathsf{k}_w(R, t)$ and $\mathsf{k}_u(R, t) \in \mathcal{H}(S)$ for some prefix u of w, in which case by the induction hypothesis we have that $\mathsf{k}_u(R, t) \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$ and we conclude by Proposition 8.
2. Simple Consequence: there is a statement

$$f = \Big(H' \Leftarrow B'_1, \ldots, B'_n\Big) \in S$$

   and a substitution σ grounding for f such that $H = H'\sigma$ and $B'_i\sigma \in \mathcal{H}(S)$.
   By the induction hypothesis, we have that $B'_i\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$. W.l.o.g. assume that $B'_i = \mathsf{k}_{w'_i}(X_i, t'_i)$. As $B'_i\sigma \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$, we have by definition of $\mathcal{H}_{\mathsf{e}}$ that there exist R'_i such that

$$\mathsf{k}_{w'_i\sigma}(R'_i, t_i\sigma) \in \mathcal{H}(K_{\mathsf{solved}}), \tag{7}$$

$$\mathsf{i}_{w'_i\sigma}(R'_i, X_i\sigma) \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}}) \tag{8}$$

   for all $1 \leq i \leq n$.
   But $w'_i\sigma$ is a prefix of w, where w is such that $H = \mathit{predicate}_w(\ldots)$ with $\mathit{predicate} \in \{\mathsf{r}, \mathsf{k}\}$. Note that as S is a set of *seed* statements, $\mathit{predicate} \notin \{\mathsf{i}, \mathsf{ri}\}$. By applying the Extend rule to Equation (8), we obtain

$$\mathsf{i}_w(R'_i, X_i\sigma) \in \mathcal{H}_{\mathsf{e}}(S_{\mathsf{solved}}). \tag{9}$$

   Let σ' be the substitution defined to be σ except that it maps X_i to R'_i for all $1 \leq i \leq n$.
   We will show that $H'\sigma' \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$. As K was updated by f, there are three cases:
   (a) if $f \in K$, we conclude by Lemma 19 or Lemma 14 (depending on the predicate). Moreover, when $\mathit{predicate} = \mathsf{r}$, we use the fact that $\mathcal{H}(K_{\mathsf{solved}}) \subseteq \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$.
   (b) if $f\Downarrow \in K$ and $f \notin K$, in which case f must be a solved deduction statement. In this case, by Lemma 17, we obtain that $H'\sigma' \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$.

(c) if $f\Downarrow = \Big(\mathsf{k}_w(R,t) \Leftarrow C_1,\ldots,C_m\Big)$ and there exists R' such that

$$\Big(\mathsf{k}_w(R',t) \Leftarrow C_1,\ldots,C_m\Big) \in \mathbf{conseq}(K_{\mathsf{solved}})$$

and such that

$$\Big(\mathsf{i}_w(R,R') \Leftarrow C_1,\ldots,C_m\Big) \in K_{\mathsf{solved}}.$$

In this case, we have that $H'\sigma' \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$ by Lemma 18.

We have shown that $H'\sigma' \in_{\mathsf{E}'} \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$. We distinguish several cases depending on *predicate*:

- *predicate* = r: In such a case, we have that $H'\sigma' = H'\sigma = H$ and we easily conclude.
- *predicate* = k: In such a case, we have that $H'\sigma' = \mathsf{k}_w(S\sigma', t\sigma')$. Relying on Equation 9 and applying the rules Cong and Refl, we deduce that: $\mathsf{i}_w(S\sigma, S\sigma') \in \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$. Since $\mathsf{k}_w(S\sigma', t\sigma') \in_{\mathsf{E}'} \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$ and $t\sigma = t\sigma'$, using Proposition 6, we conclude that $\mathsf{k}_w(S\sigma, t\sigma) \in_{\mathsf{E}'} \mathcal{H}_{\mathsf{e}}(K_{\mathsf{solved}})$.

D Effectiveness of the procedure

D.1 Proof of Lemma 1

We let $\Rightarrow$ denote the saturation relation. We let $\Rightarrow^{=}$ denote the reflexive closure of $\Rightarrow$.

Lemma 21. *Let K be a knowledge base and $\mathcal{M}_0 \subseteq \mathcal{M}$ a set of public names such that $names(K) \cap \mathcal{M}_0 = \emptyset$. Let $K_1 \subseteq K_{\mathcal{M}_0,R}$ where R is the set of solved reach statements in K. If h is a statement such that $names(h) \cap \mathcal{M}_0 = \emptyset$, then*

$$(K \uplus K_1) \oplus h = (K \oplus h) \uplus K_1.$$

Proof. If h is not solved or if it is not a deduction statement, we have that $(K \uplus K_1) \oplus h = (K \uplus K_1) \cup \{h\} = (K \cup \{h\}) \uplus K_1 = (K \oplus h) \uplus K_1$. If h is a solved deduction statement, let

$$h\Downarrow = \mathsf{k}_{\ell_1,\dots,\ell_k}(R,t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n).$$

We distinguish two cases:

1. either $\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \notin \mathbf{conseq}((K \uplus K_1)_{\mathsf{solved}})$ for any R', in which case
$$(K \uplus K_1) \oplus h = (K \uplus K_1) \cup \{h\Downarrow\} = (K \cup \{h\Downarrow\}) \uplus K_1.$$
It follows that $\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \notin \mathbf{conseq}(K_{\mathsf{solved}})$ for any R' either (since $K \subseteq K \uplus K_1$). Therefore $K \oplus h = K \cup \{h\Downarrow\}$ and we immediately conclude by replacing $K \cup \{h\Downarrow\}$ by $K \oplus h$ in the equation above.
2. or $\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \in \mathbf{conseq}((K \uplus K_1)_{\mathsf{solved}})$ for some R'. In this case, $(K \uplus K_1) \oplus h = (K \uplus K_1) \cup \{f\}$ where
$$f = \Big(i_{\ell_1,\dots,\ell_k}(R,R') \Leftarrow \{\mathsf{k}_{\ell_1,\dots,\ell_{i_j}}(X_j,x_j)\}_{j\in\{1,\dots,n\}}\Big)\!\downarrow\!.$$
To conclude we show the following claim.

   If $names(t) \cap \mathcal{M}_0 = \emptyset$ and
   $$\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \in \mathbf{conseq}((K \uplus K_1)_{\mathsf{solved}})$$
   then
   $$\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \in \mathbf{conseq}(K_{\mathsf{solved}})$$
   To proof this claim we proceed by induction on the size of the proof tree of
   $$\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1,x_1),\dots,\mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n,x_n) \in \mathbf{conseq}((K \uplus K_1)_{\mathsf{solved}}).$$
   Base case: we need to consider two cases according to which rule has been applied.
   - Axiom: the rule does not depend on the knowledge base and we trivially conclude.
   - Res: we have that $n = 0$, i.e., $H \Leftarrow \in (K \cup K_1)_{\mathsf{solved}}$ and $H\sigma = \mathsf{k}_{\ell_1,\dots,\ell_k}(R',t)$. As $names(t) \cap \mathcal{M}_0 = \emptyset$ we have that $H \Leftarrow \in K_{\mathsf{solved}}$. Hence, $\mathsf{k}_{\ell_1,\dots,\ell_k}(R',t) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

Inductive case: We suppose that the proof ends with an application of the RES rule. We have that $H \Leftarrow B_1, \ldots, B_m \ \in (K \cup K_1)_{\mathsf{solved}}$, $B_i\sigma \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}((K \uplus K_1)_{\mathsf{solved}})$ and $H\sigma = \mathsf{k}_{\ell_1,\ldots,\ell_k}(R', t)$. Let $H = \mathsf{k}_u(S, t')$ and $B_i = \mathsf{k}_{u_i}(Y_i, y_i)$. As $H\sigma = \mathsf{k}_{\ell_1,\ldots,\ell_k}(R', t)$ and $names(t) \cap \mathcal{M}_0 = \emptyset$, by inspection of the statements in K_1, it must be that $H \Leftarrow B_1, \ldots, B_m \ \in K_{\mathsf{solved}}$. Moreover, as $t'\sigma = t$ we have by hypothesis that $t'\sigma \cap M_0 = \emptyset$ and hence $t' \cap M_0 = \emptyset$. As $y_i \in vars(t')$ we have that $y_i\sigma \cap M_0 = \emptyset$ and we can apply our induction hypothesis to conclude that $B_i\sigma \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}(K_{\mathsf{solved}})$ for $1 \leq i \leq n$. Hence, as

$$H \Leftarrow B_1, \ldots, B_m \ \in K_{\mathsf{solved}}$$

and

$$B_i\sigma \Leftarrow \mathsf{k}_{\ell_1,\ldots,\ell_{i_1}}(X_1, x_1), \ldots, \mathsf{k}_{\ell_1,\ldots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}(K_{\mathsf{solved}})$$

for $1 \leq i \leq n$ we conclude that $\mathsf{k}_{\ell_1,\ldots,\ell_k}(R', t) \in \mathbf{conseq}(K)$.

Lemma 22. *Let K be a knowledge base and $\mathcal{M}_0 \subseteq \mathcal{M}$ a set of public names such that $names(K) \cap \mathcal{M}_0 = \emptyset$. Let $K_1 \subseteq K_{\mathcal{M}_0,R}$ where R is the set of solved reach statements in K. If*

$$K \uplus K_1 \Rightarrow K''$$

then $K'' = K' \uplus K_2$ with $K \Rightarrow^{=} K'$, $K_2 \subseteq K_{\mathcal{M}_0,R'}$ where R' is the set of solved reach statements in K' and $names(K') \cap \mathcal{M}_0 = \emptyset$.

Proof. We perform a case distinction depending on which saturation rule triggered:

1. if rule RESOLUTION triggered, we will show that $f, g \in K$.
   Indeed, no statement $\left(\mathsf{k}(m, m) \Leftarrow\right) \in K_1$ can play the role of g in the RESOLUTION saturation rule since $t' = m$ must unify with $t \notin X$. Therefore t must be m, but $m \notin names(K)$ by hypothesis and therefore t cannot be m.
   No statement in K_1 can play the role of f in the RESOLUTION saturation rule since they have no antecedents.
   Therefore $f, g \in K$ and $names(h) \notin \mathcal{M}_0$. We choose $K' = K \oplus h$, $K_2 = K_1$ and we conclude by Lemma 21.
2. if rule EQUATION triggered, we distinguish three cases:
   (a) if a statement $\left(\mathsf{k}(m, m) \Leftarrow\right) \in K_1$ plays the role of f in the EQUATION saturation rule, we have that $t = m$. As t' unifies with m, we have that either $t' = m$ or that t' is a variable. The second case is not possible since g must be well-formed. Therefore $t' = m$. As $m \notin names(K)$ by hypothesis it follows that $g \in K_1$ and therefore $g = \mathsf{k}(m, m)$. Therefore the resulting statement is $\mathsf{i}(m, m)$. We choose $K_2 = K_1 \cup \{\mathsf{i}(m, m)\}$, $K' = K$ to conclude.
   (b) if a statement $\left(\mathsf{k}(m, m) \Leftarrow\right) \in K_1$ plays the role of g, the reasoning is analogous to the case above
   (c) otherwise $f, g \in K$. Therefore $names(h) \cap \mathcal{M}_0 = \emptyset$. We choose $K' = K \oplus h$ and $K_2 = K_1$ to conclude.
3. if rule TEST triggered, we distinguish two cases:
   (a) if $\left(\mathsf{i}(m, m) \Leftarrow\right) \in K_1$ plays the role of f, then $g = \mathsf{r}_u \Leftarrow B_1, \ldots, B_n \in K$. We choose $K' = K$ and $K_2 = K_1 \cup \{\mathsf{ri}_u(m, m) \Leftarrow B_1, \ldots B_n\}$ to conclude.
   (b) otherwise $f \in K$. The statement g must also be in K since g is a reachability statement and K_1 does not contain reachability statements. We choose $K' = K \oplus h$ and $K_2 = K_1$ to conclude.

From the above lemma we can immediately conclude that if $\mathcal{M}_0 = \mathcal{M} \setminus names(K)$ and

$$K \cup \{\mathsf{k}(m, m)\}_{m \in \mathcal{M}_0} \Rightarrow^* K'$$

and K' is saturated, then

$$K \Rightarrow^* K''$$

with K'' saturated and $K' = K'' \cup K_{\mathcal{M}_0,R''}$ where R'' is the set of solved reach statements in K''. This means that there is no need to keep track of all (an infinite number of) names during the saturation process.

D.2 Proof of Lemma 2

Proof. The definition of $\mathbf{conseq}(K)$ yields a direct recursive algorithm which moreover computes R:

- (Axiom) Check whether $t = x_j$ for $1 \leq j \leq n$. If this is the case return (yes, X_j).
- (Res) Otherwise, guess a (solved) statement $\mathsf{k}_u(R', t') \Leftarrow \mathsf{k}_{u_1}(Y_1, y_1), \dots \mathsf{k}_{u_k}(Y_k, y_k) \in K$ and compute substitution σ such that $\mathsf{k}_{\ell_1,\dots,\ell_k}(R', t) = \mathsf{k}_u(R', t')\sigma$. Check recursively whether

$$\exists R_i.\mathsf{k}_{u_i}(R_i, y_i)\sigma \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1, x_1), \dots, \mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}(K)$$

  for $1 \leq i \leq k$. In that case return (yes, $R'[Y_i \mapsto R_i]_{1 \leq i \leq n}$). Otherwise return no.

Termination is ensured because the size of t when checking whether

$$\exists R.\mathsf{k}_{\ell_1,\dots,\ell_k}(R, t) \Leftarrow \mathsf{k}_{\ell_1,\dots,\ell_{i_1}}(X_1, x_1), \dots, \mathsf{k}_{\ell_1,\dots,\ell_{i_n}}(X_n, x_n) \in \mathbf{conseq}(K)$$

strictly decreases in each recursive call. Indeed, when $\mathsf{k}_u(R', t') \Leftarrow \mathsf{k}_{u_1}(Y_1, y_1), \dots \mathsf{k}_{u_k}(Y_k, y_k) \in K$ we have that $t' \notin X$ because it is well-formed and $y_i \in vars(t')$ by definition of a statement. Hence, $|y_i\sigma| < |t'\sigma| = |t|$.

D.3 Termination of the saturation

Throughout this section we suppose that all statements have distinct variables. This can be supposed w.l.o.g. because all variables are universally quantified.

Basic properties

Lemma 23. *Let K be a knowledge base. We have:*

1. *for all substitutions σ, for all $f \in \mathbf{conseq}(K)$, $f\sigma \in \mathbf{conseq}(K)$*
2. *for all $\big(\mathsf{k}_u(R, t) \Leftarrow B_1, \dots, B_n\big) \in \mathbf{conseq}(K)$, for all symbolic run v, $\big(\mathsf{k}_{uv}(R, t) \Leftarrow B_1, \dots, B_n\big) \in \mathbf{conseq}(K)$*
3. *for all $\big(H \Leftarrow \mathsf{k}_{u_1 v_1}(R_1, t_1), \dots, \mathsf{k}_{u_n v_n}(R_n, t_n)\big) \in \mathbf{conseq}(K)$, for all predicates $\mathsf{k}_{w_1}(S_1, r_1), \dots, \mathsf{k}_{w_k}(S_k, r_k)$, if for all $i \in \{1, \dots, n\}$, $\big(\mathsf{k}_{u_i}(R_i, t_i) \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \dots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$ then $\big(H \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \dots, \mathsf{k}_{w_k}(S_k, r_k)\big)$*

Proof. We prove the three properties separately.

Property (1): We prove the property by induction on the length of the derivation of $f \in \mathbf{conseq}(K)$.

Base case, size of the derivation is 1: We have

- either $f = \big(H \Leftarrow \mathsf{k}_u(R, t), B_1, \dots, B_m\big)$ which implies that $f\sigma = \big(H\sigma \Leftarrow \mathsf{k}_{u\sigma}(R\sigma, t\sigma), B_1\sigma, \dots, B_m\sigma\big)$ and so $f\sigma \in \mathbf{conseq}(K)$ by rule Axiom.
- or $f = H\gamma \Leftarrow B_1, \dots, B_m\big)$ for some $\big(\mathsf{k}_u(R, t) \Leftarrow \big) \in K$ and substitution γ. Since $f\sigma = \big(H\gamma\sigma \Leftarrow B_1\sigma, \dots, B_m\sigma\big)$, we directly have that $f\sigma \in \mathbf{conseq}(K)$ by rule Res.

Inductive step, size of the derivation bigger than 1: We have that $f = \big(\mathsf{k}_{uv}(R, t)\gamma \Leftarrow C_1, \dots, C_m\big)$ for some $\big(\mathsf{k}_u(R, t) \Leftarrow B_1, \dots, B_n\big) \in K$ and substitution γ such that for all $i \in \{1, \dots, n\}$, $\big(B_i\gamma \Leftarrow C_1, \dots, C_m\big) \in \mathbf{conseq}(K)$. By induction hypothesis, we deduce that for all $i \in \{1, \dots, n\}$, $\big(B_i\gamma\sigma \Leftarrow C_1\sigma, \dots, C_m\sigma\big) \in \mathbf{conseq}(K)$. Hence we can apply the rule Res which allows us to conclude that $f\sigma \in \mathbf{conseq}(K)$.

Property (2): Let v be a symbolic run and $f = \big(\mathsf{k}_u(R, t) \Leftarrow B_1, \dots, B_n\big)$. We proceed by case distinction on the last rule applied in the derivation of $f \in \mathbf{conseq}(K)$.

Rule Axiom*:* There exist two symbolic runs u_1, u_2 such that $f = \big(\mathsf{k}_{u_1 u_2}(R, t) \Leftarrow \mathsf{k}_{u_1}(R, t), B_1, \dots, B_m\big)$ which trivially implies that $\big(\mathsf{k}_{u_1 u_2 v}(R, t) \Leftarrow \mathsf{k}_{u_1}(R, t), B_1, \dots, B_m\big) \in \mathbf{conseq}(K)$.

Rule Res*:* There exist two symbolic runs u_1, u_2, a substitution γ and $g = \big(\mathsf{k}_{u_1}(R', t') \Leftarrow C_1, \dots, C_m\big) \in K$ such that $\mathsf{k}_{u_1 u_2}(R', t')\gamma = \mathsf{k}_u(R, t)$ and for all $i \in \{1, \dots, m\}$, $\big(C_i\gamma \Leftarrow B_1, \dots, B_n\big) \in \mathbf{conseq}(K)$. But the

variables of g being distinct from the one of f, we can assume w.l.o.g that the variables of g are distinct from the one of v. Hence $v\gamma = v$ which allows us to deduce that $\big(\mathsf{k}_{u_1u_2v}(R', t')\gamma \Leftarrow B_1, \ldots, B_n\big) \in \mathbf{conseq}(K)$.

Property (3): We do a proof by induction on the length of the derivation of $f = \big(H \Leftarrow \mathsf{k}_{u_1v_1}(R_1, t_1), \ldots, \mathsf{k}_{u_nv_n}(R_n, t_n)\big) \in \mathbf{conseq}(K)$.

Base case, size of the derivation is 1*:* In such a case, one of the following cases holds:

- Case $f = \big(\mathsf{k}_{uv}(R, t) \Leftarrow \mathsf{k}_u(R, t), B_1, \ldots, B_{n-1}\big)$: By hypothesis, we know that there exists $u' \sqsubseteq u$ such that $\big(\mathsf{k}_{u'}(R, t) \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \ldots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$. Thanks to the second property of this lemma, we can deduce that $\big(\mathsf{k}_{uv}(R, t) \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \ldots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$. Hence the result holds.
- Case $H = \mathsf{k}_{uv}(R, t)\gamma$ for some $\big(\mathsf{k}_u(R, t) \Leftarrow\big) \in K$ and substitution γ: We can apply the rule Res with different hypotheses, that is $\big(\mathsf{k}_{uv}(R, t)\gamma\mathsf{k}_{w_1}(S_1, r_1), \ldots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$.

Inductive step, size of the derivation bigger than 1: In such a case, we have that $H = \mathsf{k}_{uv}(R, t)\gamma$ for some $\big(\mathsf{k}_u(R, t) \Leftarrow B_1, \ldots, B_m\big) \in K$ and substitution γ such that for all $i \in \{1, \ldots, n\}$, $\big(B_i\gamma \Leftarrow \mathsf{k}_{u_1v_1}(R_1, t_1), \ldots, \mathsf{k}_{u_nv_n}(R_n, t_n)\big) \in \mathbf{conseq}(K)$. By inductive hypothesis, we deduce that for all $i \in \{1, \ldots, n\}$, $\big(B_i\gamma \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \ldots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$. Hence by application of rule Res, we can conclude that $\mathsf{k}_{uv}(R, t)\gamma \Leftarrow \mathsf{k}_{w_1}(S_1, r_1), \ldots, \mathsf{k}_{w_k}(S_k, r_k)\big) \in \mathbf{conseq}(K)$.

In the following we will caracterize the shape of the knowledge base built by applying saturation rules.

Definition 23. *We say that a symbolic run $\ell_1 \ldots \ell_n$ is initial if for all $i \in \{1, \ldots, n\}$, $\ell_i = \mathbf{out}(c_i)$ or $\ell_i = \mathbf{in}(c_i, x_i)$ with $x_i \in \mathcal{X}$. Moreover, for all $i, j \in \{1, \ldots, n\}$, $i \neq j$ implies $vars(\ell_i) \cap vars(\ell_j) = \emptyset$.*

Definition 24. *Let $f = \big(\mathsf{k}_w(R, t) \Leftarrow B_1, \ldots, B_n\big)$ be a statement. We say that f satisfies origination whenever there exists u, v such that $w = uv$ and*

- *v is initial and for all $x \in vars(v)$, $x \notin vars(u)$ and for all $\mathsf{k}_w(X, r) \in \{B_1, \ldots, B_n\}$, $x \notin vars(r)$;*
- *for all $x \in vars(u)$, there exists $\mathsf{k}_w(X, r) \in \{B_1, \ldots, B_n\}$ such that $x \in vars(r)$ and $x \notin vars(w)$.*

Given a clause f with a knowledge predicate as head, we denote by $\mathsf{inst}(f) = u$ and $\mathsf{init}(f) = v$, where u is chosen to be maximal (in size). We say that a knowledge base satisfies origination when all its clauses with a knowledge predicate as head satisfy origination.

We first prove that any set of seed statements satisfies origination.

Lemma 24. *Let T be a ground trace of size n and $\mathcal{M}_0$ a set of public names. The set $\mathsf{seed}(T, \mathcal{M}_0)$ satisfies origination.*

Proof. Let us use the notations of Section 4.1. Among $\mathsf{seed}(T, \mathcal{M}_0)$, there are three kinds of horn clauses with a knowledge predicate as head:

- $h = \big(\mathsf{k}_{\ell_1\sigma\tau\downarrow, \ldots, \ell_m\sigma\tau\downarrow}(w_{|S(m)|}, t_m\sigma\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1\sigma\tau\downarrow, \ldots, \ell_{j-1}\sigma\tau\downarrow}(X_j, x_j\sigma\tau\downarrow)\}_{j \in R(m)}\big)$ where $m \in S(n)$, $\sigma \in \mathrm{mgu}_{\mathsf{R}}(\{s_k = t_k\}_{k \in T(m)})$ and $\tau \in \mathrm{variants}(\ell_1\sigma, \ldots, \ell_m\sigma, t_m\sigma)$. By definition of $R(m)$, we deduce that for all $x \in vars(\ell_1\sigma\tau\downarrow, \ldots, \ell_m\sigma\tau\downarrow)$, there exists $j \in R(m)$ such that $x \in vars(x_j\sigma\tau\downarrow)$. Moreover, if we consider j_0 the smallest j such that $x \in vars(x_{j_0}\sigma\tau\downarrow)$, we obtain that $x \notin vars(\ell_1\sigma\tau\downarrow, \ldots, \ell_{j_0-1}\sigma\tau\downarrow)$. We can conclude that h satisfies origination with $\mathsf{inst}(f) = \ell_1\sigma\tau\downarrow, \ldots, \ell_m\sigma\tau\downarrow$ and $\mathsf{init}(f) = \varepsilon$.
- $h = \big(\mathsf{k}(c, c) \Leftarrow\big)$ with $c \in \mathcal{M}_0$. In such a case, we trivially have that h satisfies origination with $\mathsf{inst}(f) = \varepsilon$ and $\mathsf{init}(f) = \varepsilon$.
- $h = \big(\mathsf{k}_{\ell_1, \ldots, \ell_m}(f(Y_1, \ldots, Y_k), f(y_1, \ldots, y_k)\tau\downarrow) \Leftarrow \{\mathsf{k}_{\ell_1, \ldots, \ell_m}(Y_j, y_j\tau\downarrow)\}_{j \in \{1, \ldots, k\}}\big)$ where $m \in \{0, \ldots, n\}$, f is a symbol function of arity k and $\tau \in \mathrm{variants}(f(y_1, \ldots, y_k))$. Since $\ell_1, \ldots, \ell_m$ is initial, we directly have that h satisfies origination with $\mathsf{inst}(h) = \varepsilon$ and $\mathsf{init}(h) = \ell_1, \ldots, \ell_m$. □

Moreover origination is preserved by application of the resolution rule.

Lemma 25. *Let K be a knowledge base satisfying origination. Let $f \in K$ and $g \in K_{\mathsf{solved}}$ and let K' be the knowledge base obtained by applying the rule* Resolution *on f and g. We have that K' satisfies origination.*

Proof. On the one hand, if the head of f is not an intruder knowledge predicate then the result trivially holds since in such a case, K' is K added with a clause whose head is not an intruder knowledge predicate. On the other hand, let us consider f with an intruder knowledge predicate as head. Since we apply the rule Resolution on f and g, we know that:

- $f = \big(H \Leftarrow \mathsf{k}_{uv}(X,t), B_1, \ldots, B_n\big)$
- $g = \big(\mathsf{k}_w(R',t') \Leftarrow B_{n+1}, \ldots, B_m\big)$
- $\theta = \mathrm{mgu}(\mathsf{k}_u(X,t), \mathsf{k}_w(R',t'))$
- $h = \big(H \Leftarrow B_1, \ldots, B_m\big)\theta$
- $K = K \oplus h$

We now show that h satisfies origination with $\mathsf{inst}(h) = \mathsf{inst}(f)\theta$ if $\mathsf{inst}(g)\theta \sqsubset \mathsf{inst}(f)\theta$, otherwise $\mathsf{inst}(h) = \mathsf{inst}(g)\theta$. Moreover, $\mathsf{init}(h)$ is such that $\mathsf{inst}(f)\mathsf{init}(f)\theta = \mathsf{inst}(h)\mathsf{init}(h)$.

- Let us focus on the unification of u with w. We know that $w = \mathsf{inst}(g)\mathsf{init}(g)$. Moreover, $u \sqsubseteq \mathsf{inst}(f)\mathsf{init}(f)$ and so $\mathsf{inst}(g)\mathsf{init}(g)\theta \sqsubseteq \mathsf{inst}(h)\mathsf{init}(h)$ where $\mathsf{init}(h)$ is a suffix of $\mathsf{init}(f)\theta$. Since $\mathsf{init}(g)$ and $\mathsf{init}(f)$ are both initial, we have w.l.o.g. that $\mathsf{init}(h)$ is a suffix of $\mathsf{init}(f)$ (typically, we assume that the variables of $\mathsf{init}(g)$ are in $dom(\theta)$). Since $\mathsf{init}(f)$ is initial then so is $\mathsf{init}(h)$. Lastly, since the variables of $\mathsf{init}(f)$ were not occurring in $\mathsf{inst}(f)$ nor in the rest of the clause f, and since the variables of $\mathsf{init}(h)$ do not appear in the image of θ, we deduce that for all $x \in vars(\mathsf{init}(h))$, $x \notin vars(\mathsf{inst}(h))$ and for all $\mathsf{k}_s(Y,p) \in \{B_1\theta, \ldots, B_m\theta\}$, $x \notin vars(p)$.
- Let $x \in vars(\mathsf{inst}(h))$. There exists $y \in vars(\mathsf{inst}(f), \mathsf{inst}(g))$ and $x \in vars(y\theta)$ (note that y might be x). If $y \in vars(\mathsf{inst}(f))$ then we deduce that there exists $\mathsf{k}_s(Y,p) \in \{\mathsf{k}_{uv}(X,t), B_1, \ldots, B_n\}$ such that $y \in vars(p)$ and $y \notin vars(s)$. Moreover, if $y \in vars(\mathsf{inst}(g))$ then we deduce that there exists $\mathsf{k}_s(Y,p) \in \{B_{n+1}, \ldots, B_m\}$ such that $y \in vars(p)$ and $y \notin vars(s)$. Hence, there exists $\mathsf{k}_s(Y,p) \in \{\mathsf{k}_{uv}(X,t), B_1, \ldots, B_m\}$ such that $y \in vars(p)$ and $y \notin vars(s)$. W.l.o.g. let us assume that there is no other variable z and $\mathsf{k}_{s'}(Y',p') \in \{\mathsf{k}_{uv}(X,t), B_1, \ldots, B_m\}$ such that $x \in vars(z\theta)$, $z \in vars(p')$, $z \notin vars(s')$ and $s' \sqsubset s$. Hence $y \in vars(p)$ and $y \notin vars(s)$ imply that $x \in vars(p\theta)$ and $x \notin vars(s\theta)$. If $\mathsf{k}_s(Y,p) \in \{B_1, \ldots, B_m\}$ then the result holds. Hence it remains to consider the case where $\mathsf{k}_s(Y,p) = \mathsf{k}_{uv}(X,t)$. In such a case, $x \in vars(t\theta)$ and $x \notin vars(uv\theta)$. But we know that $x \in vars(t\theta)$ implies that there exists $\mathsf{k}_{s'}(Y',p') \in \{B_{n+1}, \ldots, B_m\}$ such that $x \in vars(p'\theta)$. Moreover, we know that $u\theta = w\theta$ and by definition of clause, $s'\theta \sqsubseteq w\theta$. Hence we can deduce that $x \notin vars(s'\theta)$ and so the result holds.

Finally, we easily see that obtaining the canonical form of h preserves the origination property and conclude that K' satisfies origination.

In the following we say that S is a set of seed statements if $S = \mathsf{seed}(()T)$ for some ground trace T.

The following corollary is a direct consequence of Lemmas 24 and 25. Moreover. we say that K is built from S if all the clauses of K can be obtained by applying saturation rules from Figure 3 to $K_i(S)$.

Corollary 3. *Let S be a set of seed statements and K a knowledge base built form S. K satisfies origination.*

Initial substitution We first introduce a few notations.

- Given a satement $f = \big(\mathsf{k}_u(R,t) \Leftarrow B_1, \ldots, B_n\big)$, we denote by $\mathsf{w}(f)$ the symbolic run u.
- Given a set of seed statements S, we denote by $\mathsf{IPC}(S)$ the subset of S corresponding to the protocol clauses and the public name clauses.
- Given a symbolic run w and an integer n, we denote by $w|_n$ the symbolic run prefix of w of size n.

Definition 25. *Let S be a set of seed statements. We define an initial substitution and initial subterms respectively as a pair (w,σ) and a set $\mathsf{st}_{\mathcal{IS}}(S,\sigma)$ such that:*

– w *is an initial symbolic run; and*
– $f_1, \dots f_n \in \mathsf{IPC}(S), \forall i \in \{1, \dots, n\}, |\mathsf{w}(f_i)| \leq |w|$*; and*
– $\sigma = \mathrm{mgu}(\{(w|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (w|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n))\} \cup T)$*; and*
– $T \subseteq \mathsf{st}(f_1, \dots, f_n) \times \mathsf{st}(f_1, \dots, f_n)$*; and*
– $vars(img(\sigma)) \cap vars(w) = \emptyset$.
– $\mathsf{st}_{\mathcal{IS}}(S, \sigma) = \mathsf{st}(f_1, \dots, f_n) \cup \bigcup_{f \in S, vars(f) = \emptyset} \mathsf{st}(f)$

We denote by $\mathcal{IS}(S)$ *the set of all initial substitutions for* S.

We note that in the definition of $\mathsf{st}_{\mathcal{IS}}(S, \sigma)$ the clauses $f_1, \dots, f_n$ are uniquely identified by the domain of σ as we suppose that all clauses have distinct variables. Moreover, adding all ground clauses to $\mathsf{st}_{\mathcal{IS}}(S, \sigma)$ guarantees that $\mathsf{st}_{\mathcal{IS}}(S, \sigma)$ is uniquely defined.

Definition 26. *Let* S *be a set seed statement,* $(w, \sigma) \in \mathcal{IS}(S)$ *and* γ *a substitution. We say that* (w, σ) *is maximal for* γ *in* S *if for all* $(w, \sigma') \in \mathcal{IS}(S)$*, for all substitution* γ'*, such that* $w\sigma\gamma = w\sigma'\gamma'$ *there exists* θ *such that* $\sigma = \sigma'\theta$.

Lemma 26. *Let* S *be a set of seed statements and* $(w, \sigma) \in \mathcal{IS}(S)$*. For all* $t \in \mathsf{st}(img(\sigma))$*, there exists* $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma)$ *such that* $u\sigma = t$.

Proof. By definition of $\mathcal{IS}(S)$, we have that

$$\sigma = \mathrm{mgu}(\{(w|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (w|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n))\} \cup T)$$

with $f_1, \dots f_n \in \mathsf{IPC}(S), T \subseteq \mathsf{st}(f_1, \dots, f_n) \times \mathsf{st}(f_1, \dots, f_n)$ and $vars(img(\sigma)) \cap vars(w) = \emptyset$. Let $i \in \{1, \dots, n\}$ such that for all $j \in \{1, \dots, n\}$, $|\mathsf{w}(f_i)| \geq |\mathsf{w}(f_j)|$. We deduce that

$$\sigma = \sigma'_0 \,\mathrm{mgu}(\{(\mathsf{w}(f_i)|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (\mathsf{w}(f_i)|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n))\} \cup T)$$

where $dom(\sigma'_0) \subseteq vars(w)$ and $img(\sigma'_0) \subseteq \mathsf{st}(f_i)$. This allows us to conclude that $\sigma = \sigma'_0 \,\mathrm{mgu}(U_0)$ with $U_0 \subseteq \mathsf{st}(f_1, \dots, f_n)\sigma'_0 \times \mathsf{st}(f_1, \dots, f_n)\sigma'_0$, and for all $t \in \mathsf{st}(img(\sigma'_0))$, $t \in \mathsf{st}(f_i)$ and $t\sigma'_0 = t$.

It remains to prove that for all U and σ', if $\mathrm{mgu}(U)$ exists and for all $t \in \mathsf{st}(img(\sigma'))$, there exists $u \in \mathsf{st}(f_1, \dots, f_n)$ such that $u\sigma' = t$ and $U \subseteq \mathsf{st}(f_1, \dots, f_n)\sigma' \times \mathsf{st}(f_1, \dots, f_n)\sigma'$, then for all $t \in \mathsf{st}(img(\sigma' \,\mathrm{mgu}(U)))$, there exists $u \in \mathsf{st}(f_1, \dots, f_n)$ such that $u\sigma' \,\mathrm{mgu}(U) = t$.

We prove this result by induction on $\mathsf{m}(U)$ defined as follows:

$$\mathsf{m}(U) = (|vars(U)|, \{\{|t_1| + |t_2| \mid (t_1, t_2) \in U\}\})$$

Base case $\mathsf{m}(U) = (0, \emptyset)$*:* In such a case, $U = \emptyset$. Thus we have that $\mathrm{mgu}(U) = Id$ and so the result trivially hold since, by hypothesis, we have for all $t \in \mathsf{st}(img(\sigma'))$, there exists $u \in \mathsf{st}(f_1, \dots, f_n)$ such that $u\sigma' = t$.

Inductive step: Otherwise, since $\mathrm{mgu}(U)$ exists, we have that either (a) $U = \{f(u_1, \dots, u_m), f(v_1, \dots, v_m)\} \cup U'$ or (b) $U = \{x, u\} \cup U'$.

In case (a), $\mathrm{mgu}(U) = \mathrm{mgu}(U'')$ with $U'' = \{(u_1, v_1); \dots; (u_m, v_m)\} \cup U'$. But $U \subseteq \mathsf{st}(f_1, \dots, f_n)\sigma' \times \mathsf{st}(f_1, \dots, f_n)\sigma'$ implies that $U'' \subseteq \mathsf{st}(f_1, \dots, f_n)\sigma' \times \mathsf{st}(f_1, \dots, f_n)\sigma'$. Moreover, $\mathsf{m}(U) > \mathsf{m}(U'')$ hence we can apply our inductive hypothesis on U'' and σ'. Since $\mathrm{mgu}(U) = \mathrm{mgu}(U'')$, the result holds.

In case (b), $\sigma' \,\mathrm{mgu}(U) = \sigma'\sigma'' \,\mathrm{mgu}(U'\sigma'')$ with $\sigma'' = \{x \to u\}$. Let $t \in \mathsf{st}(img(\sigma'\sigma''))$, we have that either there exists $v \in \mathsf{st}(img(\sigma'))$ such that $t = v\sigma''$ or else $t \in \mathsf{st}(u)$.

If the first case, we know by hypothesis on σ' that there exists $v' \in \mathsf{st}(f_1, \dots, f_n)$ such that $v'\sigma' = v$ hence $v'\sigma'\sigma'' = t$.

In the later case, we know that $u \in \mathsf{st}(f_1, \dots, f_n)\sigma'$ hence $t \in \mathsf{st}(f_1, \dots, f_n)\sigma'$. Thus either there exists $t' \in \mathsf{st}(f_1, \dots, f_n)$ such that $t = t'\sigma'$ or $t \in \mathsf{st}(img(\sigma'))$. Once again by hypothesis on σ', we deduce that in both cases, there exists $t' \in \mathsf{st}(f_1, \dots, f_n)$ such that $t = t'\sigma'$. Moreover, $\mathrm{mgu}(U)$ exists also implies that $x \notin \mathsf{st}(u)$ and so $x \notin \mathsf{st}(t)$. Therefore, $t\sigma'' = t$ which allows us to deduce that $t = t'\sigma'\sigma''$.

With the fact that $\mathrm{mgu}(U) > \mathrm{mgu}(U'\sigma'')$, we satisfy all the conditions to apply our inductive hypothesis on $U'\sigma''$ and $\sigma'\sigma''$, and so the result holds.

Corollary 4. *Let S be a set of seed statements and $(w,\sigma) \in \mathcal{IS}(S)$. For all $v \in \mathsf{st}_{\mathcal{IS}}(S,\sigma)$, for all $t \in \mathsf{st}(v\sigma)$, there exists $u \in \mathsf{st}_{\mathcal{IS}}(S,\sigma)$ such that $u\sigma = t$.*

Lemma 27. *Let S be a set of seed statements and K be a knowledge base built from S. Let $(w,\sigma),(w,\sigma') \in \mathcal{IS}(K)$. Let γ,γ' be two substitutions. If $w\sigma\gamma = w\sigma'\gamma'$ then there exists σ'',γ'', α,α' such that $(w,\sigma'') \in \mathcal{IS}(K)$, $w\sigma''\gamma'' = w\sigma\gamma$, $\sigma'' = \sigma\alpha$ and $\sigma'' = \sigma'\alpha'$. Moreover, for all $x \in dom(\gamma'')$, either $x \in dom(\gamma)$ and $x\gamma'' = x\gamma$ or $x \in dom(\gamma')$ and $x\gamma'' = x\gamma'$.*

Proof. By definition, $(w,\sigma),(w,\sigma') \in \mathcal{IS}(K)$ implies that there exist $f_1,\dots f_n, g_1,\dots,g_m \in \mathsf{IPC}(S)$, $T \subseteq \mathsf{st}(f_1,\dots,f_n) \times \mathsf{st}(f_1,\dots,f_n)$ and $R \subseteq \mathsf{st}(g_1,\dots,g_m) \times \mathsf{st}(g_1,\dots,g_m)$ such that

- $\forall i \in \{1,\dots,n\}, |\mathsf{w}(f_i)| \leq |w|$
- $\forall i \in \{1,\dots,m\}, |\mathsf{w}(g_i)| \leq |w|$
- $\sigma = \mathrm{mgu}((w|_{|\mathsf{w}(f_1)|},\mathsf{w}(f_1));\dots;(w|_{|\mathsf{w}(f_n)|},\mathsf{w}(f_n));T)$
- $\sigma' = \mathrm{mgu}((w|_{|\mathsf{w}(g_1)|},\mathsf{w}(g_1));\dots;(w|_{|\mathsf{w}(g_m)|},\mathsf{w}(g_m));R)$

We know that all clauses have distinct variables but some clauses used to generate σ may have been used to generated σ'. Hence let us define F,E,G the following sets:

- $F = \{f_i | i \in \{1,\dots,n\} \text{ and } \forall j \in \{1,\dots,m\}, f_i \neq g_j\}$
- $G = \{g_j | j \in \{1,\dots,m\} \text{ and } \forall i \in \{1,\dots,n\}, f_i \neq g_j\}$
- $E = \{f_i | i \in \{1,\dots,n\} \text{ and } \exists j \in \{1,\dots,m\}, f_i = g_j\}$

Since $w\sigma\gamma = w\sigma'\gamma'$, we have that for all $f \in E$, $\mathsf{w}(f)\sigma\gamma = \mathsf{w}(f)\sigma'\gamma'$. Moreover, for all $f \in E$, $t \in \mathsf{st}(f)$, $vars(t) \subseteq vars(\mathsf{w}(f))$ hence $t\sigma\gamma = t\sigma'\gamma'$. Hence, let us build θ such that for all $f \in E$, $\theta[vars(f)] = \sigma\gamma[vars(f)]$; for all $f \in F$, $\theta[vars(f)] = \sigma\gamma[vars(f)]$; and for all $g \in G$, $\theta[vars(g)] = \sigma'\gamma'[vars(g)]$. In such a case, we deduce that:

- for all $i \in \{1,\dots,n\}$, $(w|_{|\mathsf{w}(f_i)|}\theta = \mathsf{w}(f_i)\theta$
- for all $j \in \{1,\dots,m\}$, $(w|_{|\mathsf{w}(g_i)|}\theta = \mathsf{w}(g_i)\theta$
- for all $(u,v) \in T \cup R$, $u\theta = v\theta$.

Therefore, there exists σ'' such that:

$$\sigma'' = \mathrm{mgu}\begin{pmatrix}(w|_{|\mathsf{w}(f_1)|},\mathsf{w}(f_1));\dots;(w|_{|\mathsf{w}(f_n)|},\mathsf{w}(f_n));\\(w|_{|\mathsf{w}(g_1)|},\mathsf{w}(g_1));\dots;(w|_{|\mathsf{w}(g_m)|},\mathsf{w}(g_m));\\T \cup R\end{pmatrix}$$

Hence $(w,\sigma'') \in \mathcal{IS}(K)$. Moreover, since θ is also a unifier, there exists γ'' such that $\theta = \sigma''\gamma''$ hence $w\sigma''\gamma'' = w\sigma\gamma$.

By definition of a most general unifier, we also have that $\sigma'' = \sigma\,\mathrm{mgu}(\cup_{x\in dom(\sigma')}\{x\sigma, x\sigma'\sigma\}) = \sigma'\,\mathrm{mgu}(\cup_{x\in dom(\sigma)}\{x\sigma', x\sigma\sigma'$ Hence we deduce the existence of α,α' such that $\sigma'' = \sigma\alpha$ and $\sigma'' = \sigma'\alpha'$. Moreover, let us consider the variables in the image of σ''. We know that $\sigma'' = \sigma\,\mathrm{mgu}(\cup_{x\in dom(\sigma')}\{x\sigma, x\sigma'\sigma\})$. We can split $dom(\sigma')$ into two sets of variables depending if they are also in $dom(\sigma)$ or not. Hence, we deduce that:

$$\sigma'' = \mathrm{mgu}(\cup_{x\in dom(\sigma')\setminus dom(\sigma)}\{x, x\sigma'\sigma\}; \cup_{x\in dom(\sigma')\cap dom(\sigma)}\{x\sigma, x\sigma'\sigma\})$$

By definition, for all $x \in dom(\sigma')$, $vars(x\sigma'\sigma) \subseteq range(\sigma) \cup range(\sigma')$ hence $vars(x\sigma'\sigma) \subseteq dom(\gamma) \cup dom(\gamma')$. Moreover, for all $x \in dom(\sigma)$, $vars(x\sigma) \subseteq dom(\gamma)$. Lastly, if $x \in dom(\sigma') \setminus dom(\sigma)$ then we deduce that $x \in dom(\sigma'')$ and so $x \notin range(\sigma'')$. Since the most general unifier preserves the variables, we can conclude that $range(\sigma'') \subseteq dom(\gamma) \cup dom(\gamma')$.

Let us now determine $x\gamma''$ for all $x \in dom(\gamma')$. We know that $w\sigma''\gamma'' = w\sigma\gamma = w\sigma'\gamma'$. Moreover, $\sigma'' = \sigma\alpha = \sigma'\alpha'$. Hence, we deduce that $w\sigma\alpha\gamma'' = w\sigma\gamma$ which implies that for all $x \in dom(\gamma)$, $x\gamma = x\alpha\gamma''$. If $x \in dom(\alpha)$ then we deduce that $x \notin dom(\gamma'')$. Hence if $x \in dom(\gamma'') \cap dom(\gamma)$ then $x\alpha = x$ and so $x\gamma = x\gamma''$. Similarly, we have that $x \in dom(\gamma'') \cap dom(\gamma')$, $x\gamma' = x\gamma''$.

Lemma 28. *Let S be a set of seed statements and $(w,\sigma) \in \mathcal{IS}(K)$. Let γ be a substitution and $u, v \in \bigcup_{t\in \mathsf{st}_{\mathcal{IS}}(S,\sigma)} \mathsf{st}(t\sigma)$ such that $u\gamma = v\gamma$. There exists σ', α, γ' such that $(w,\sigma') \in \mathcal{IS}(K)$, $\sigma' = \sigma\alpha$, $\sigma\gamma = \sigma'\gamma'$, $u\alpha = v\alpha$ and $\gamma' = \gamma[dom(\gamma')]$.*

Proof. By definition, $(w,\sigma) \in \mathcal{IS}(K)$ implies that there exist $f_1, \dots f_n \in \mathsf{IPC}(S)$, $T \subseteq \mathsf{st}(f_1,\dots,f_n) \times \mathsf{st}(f_1,\dots,f_n)$ such that

- $\forall i \in \{1,\dots,n\}, |\mathsf{w}(f_i)| \leq |w|$
- $\sigma = \mathrm{mgu}((w|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (w|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n)); T)$

But by Corollary 4, $u, v \in \bigcup_{t\in \mathsf{st}_{\mathcal{IS}}(S,\sigma)} \mathsf{st}(t\sigma)$ implies that there exists $u', v' \in \mathsf{st}_{\mathcal{IS}}(S,\sigma)$ such that $u = u'\sigma$ and $v = v'\sigma$. Since $u\gamma = v\gamma$, we deduce that $((w|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (w|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n)); T; (u', v'))$ are unifiable by $\sigma\gamma$. Let us define σ' such that $\sigma' = \mathrm{mgu}((w|_{|\mathsf{w}(f_1)|}, \mathsf{w}(f_1)); \dots; (w|_{|\mathsf{w}(f_n)|}, \mathsf{w}(f_n)); T; (u', v'))$. It implies that there exist γ' such that $\sigma\gamma = \sigma'\gamma'$. Moreover, by definition of a most general unifier, we have that $\sigma' = \sigma\,\mathrm{mgu}(u'\sigma, v'\sigma)$. Hence we deduce that there exists $\alpha = \mathrm{mgu}(u'\sigma, v'\sigma)$ such that $\sigma' = \sigma\alpha$. Moreover, $vars(u', v') \subseteq range(\sigma) = dom(\gamma)$ hence since a most general unifier preserves the variables, we deduce that $range(\sigma') \subseteq dom(\gamma)$. Consider $x \in dom(\gamma')$. We know that $x\alpha = x$ since $dom(\gamma') = range(\sigma')$. Moreover, we trivially have that $x \notin dom(\sigma)$. Therefore, since $\sigma\gamma = \sigma\alpha\gamma'$, we deduce that $x\gamma = x\gamma'$.

Corollary 5. *Let S be a set of seed statements. Let $(w,\sigma) \in \mathcal{IS}(S)$ and let γ be a substitution such that $dom(\gamma) \subseteq vars(w\sigma)$. There exists σ', γ', α such that:*

- *$(w,\sigma') \in \mathcal{IS}(S)$ and is maximal for γ' in S; and*
- *$\sigma' = \sigma\alpha$ and $w\sigma'\gamma' = w\sigma\gamma$; and*
- *$dom(\gamma') = vars(w\sigma')$ and $\gamma' = \gamma[dom(\gamma')]$; and*
- *for all $u, v \in \mathsf{st}_{\mathcal{IS}}(S,\sigma)$, $u\gamma' = v\gamma'$ implies $u = v$.*

Characterisation of the form of a knowledge base

Definition 27. *Let S be a set of statements and K be a knowledge base built from S. Let $f = \big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_{w_1}(X_1,t_1), \dots, \mathsf{k}_{w_n}(X_n,t_n)\big) \in K$. Consider the initial substitution $(w_0,\sigma_0) \in \mathcal{IS}(S)$ such that $dom(\gamma) = vars(w_0\sigma_0)$, $w = w_0\sigma_0\gamma$ and (w_0,σ_0) is maximal for γ in K. We say that a term u is well-formed in f if there exist $u_1, \dots, u_m \in \mathsf{st}_{\mathcal{IS}}(S,\sigma_0)$, $i_1, \dots, i_k \in \{1,\dots,n\}$ and a context C built only of function symbols such that:*

- *for all $j \in \{1,\dots,k\}$, $t_{i_j} \in \mathcal{X}$; and*
- *$u = C[t_{i_1}, \dots, t_{i_k}, u_1\sigma_0\gamma, \dots, u_m\sigma_0\gamma]$; and*
- *for all position p of C, there exists T such that $\big(\mathsf{k}_w(T, u|_p) \Leftarrow \mathsf{k}_{w_1}(X_1,t_1), \dots, \mathsf{k}_{w_n}(X_n,t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.*

Definition 28. *Let S be a set of seed statements and K be a knowledge base built from S. We say that K is well formed if for all $f = \big(\mathsf{k}_w(R,t) \Leftarrow \mathsf{k}_{w_1}(X_1,t_1), \dots, \mathsf{k}_{w_n}(X_n,t_n)\big) \in K$, there exists $(w_0,\sigma_0) \in \mathcal{IS}(K)$ and a substitution γ such that $dom(\gamma) = vars(w_0\sigma_0)$ and the following properties hold:*

1. *(w_0,σ_0) is maximal for γ in K.*
2. *$w = w_0\sigma_0\gamma$*
3. *for all $x \in dom(\gamma) \setminus vars(w_0)$, $vars(x\gamma) \subseteq vars(t_1,\dots,t_n)$.*
4. *for all $x \in dom(\gamma) \cap vars(w_0)$, $x\gamma \in \mathcal{X}$, $x\gamma \notin vars(t, t_1, \dots, t_n)$ and $x\gamma$ occurs only once in w.*
5. *for all $x \in dom(\gamma)$, $x\gamma \notin \mathcal{X}$ implies that there exists T such that $\big(\mathsf{k}_w(T, x\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1,t_1), \dots, \mathsf{k}_{w_n}(X_n,t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.*
6. *one of the two following properties holds:*
   a) *there exists $u \in \mathsf{st}_{\mathcal{IS}}(S,\sigma_0)$ such that $t = u\sigma_0\gamma$ and for all $v \in \{t_1,\dots,t_n\}$, v is well formed in f.*
   b) *there exist u and T such that $t \in \mathsf{st}(u)$, $\big(\mathsf{k}_w(T,u) \Leftarrow \mathsf{k}_{w_1}(X_1,t_1), \dots, \mathsf{k}_{w_n}(X_n,t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$ and for all $x \in dom(\gamma) \setminus vars(w_0)$, $x\gamma \in \mathcal{X}$ implies $x\gamma \in \{t_1,\dots,t_n\}$.*
7. *if $f \in K_{\mathsf{solved}}$ then one of the two following properties holds:*

a) there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ *such that* $t = u\sigma_0\gamma$ *and* $u\sigma_0 \notin \mathcal{X}$.
b) $dom(\gamma) = vars(w_0)$ *and* $t = \mathsf{f}(t_1, \ldots, t_n)$ *for some function symbol* f *and* $w = w_i$ $(1 \leq i \leq n)$.

Note that when $f \in K_{\mathsf{solved}}$, Property 7.a (resp. 7.b) implies Property 6.a (resp. 6.b).

Lemma 29. *Let* S *be a set of seed statements.* $K_i(S)$ *is well formed.*

Proof. Let us use the notations of Section 4.1. Let $S = \mathsf{seed}(T, \mathcal{M}_0)$ and $f = \big(\mathsf{k}_w(R, t) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in K_i(S)$. We distinguish two cases according to whether $f \in \mathsf{IPC}(S)$ or not.

$f \in \mathsf{IPC}(S)$ In this case, there exists an initial w_0 such that $\sigma_0 = \mathrm{mgu}(w_0, w)$ exists, and $w = w_0\sigma_0$. Moreover, $(w_0, \sigma_0) \in \mathcal{IS}(S)$ and $\mathsf{st}(f) \subseteq \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$. Taking γ to be the identity substitution, we have that $w = w_0\sigma_0\gamma$. Moreover, $w = w_0\sigma_0$ implies that (w_0, σ_0) is maximal for γ in S. Properties 3, 4 and 5 are trivially satisfied since γ is the identity substitution. Property 6.a also holds by choosing $u = t$. Indeed, since $\mathsf{st}(f) \subseteq \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ we have that $t \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$. Moreover as $dom(\sigma_0) \subseteq vars(w_0)$ and $vars(w_0) \cap vars(t) = \emptyset$ we have that $t = t\sigma_0 = t\sigma_0\gamma$. For the same reason, we directly have that $t_1, \ldots, t_n$ are well formed in f. Lastly, if f is solved, we have by definition of a knowledge base that $t \notin \mathcal{X}$.

$f \notin \mathsf{IPC}(S)$ In this case, we know that w is initial and $vars(t, t_1, \ldots, t_n) \cap vars(w) = \emptyset$. Defining σ_0 to be the identity substitution, there exists an initial w_0 and a variable renaming γ from $vars(w_0)$ to $vars(w)$ such that $(w_0, \sigma_0) \in \mathcal{IS}(S)$ and $w = w_0\gamma = w_0\sigma_0\gamma$. Property 3 trivially holds since $dom(\gamma) \setminus vars(w_0) = \emptyset$. Property 4 also directly holds since $vars(t, t_1, \ldots, t_n) \cap vars(w) = \emptyset$ and w is initial. Property 5 holds since for all $x \in dom(\gamma)$, $x\gamma \in \mathcal{X}$. As we focus on subterm convergent rewriting system, we know that either f is not solved and there exists $i \in \{1, \ldots, n\}$ such that $t \in \mathsf{st}(t_i)$ or else f is solved and $t = \mathsf{f}(t_1, \ldots, t_n)$ for some function symbol f. Thus Property 6.b holds respectively with $u = t_i$ or $u = t$. Lastly, when f is solved, we already showed that $t = \mathsf{f}(t_1, \ldots, t_n)$ and therefore Property 7.b holds.

Lemma 30. *Let* S *be a set of seed statements and* K *be a well formed knowledge base built from* S. *Let* $(w_0, \sigma_0) \in \mathcal{IS}(S)$ *and let* γ *be a substitution such that* $dom(\gamma) = vars(w_0\sigma_0)$ *and* (w_0, σ_0) *is maximal for* γ *in* S. *Assume that* $\big(\mathsf{k}_{w_0\sigma_0\gamma}(R, u) \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. *There exist* $u_1, \ldots, u_m \in \mathsf{st}_{\mathcal{IS}}(S, \sigma)$, $i_1, \ldots, i_k \in \{1, \ldots, n\}$ *and a context* C *built only on function symbols such that:*

- $u = C[t_{i_1}, \ldots, t_{i_k}, u_1\sigma_0\gamma, \ldots, u_m\sigma_0\gamma]$; *and*
- *for all* $i \in \{1, \ldots, m\}$, $u_i\sigma_0 \notin \mathcal{X}$; *and*
- *for all* p *position of* C, *there exists* T *such that* $\big(\mathsf{k}_{w_0\sigma\gamma}(T, u|_p) \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

Proof. We proceed by induction on the size N of the derivation of $f = \big(\mathsf{k}_{w_0\sigma\gamma}(R, u) \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

Base case $N = 0$*:* either there exists $i \in \{1, \ldots, n\}$ such that $u = t_i$ (rule Axiom) and we trivially conclude choosing $C = \_$ or there exists $\big(\mathsf{k}_w(R', t') \Leftarrow\big) \in K_{\mathsf{solved}}$ and a substitution α such that $w\alpha \sqsubseteq w_0\sigma_0\gamma$, $u = t'$ and $R = R'\alpha$. In fact $vars(t') = \emptyset$. But K is well formed and in particular, Properties 1,2 and 7.a hold. Hence there exist σ_0', γ' and a term $u_0 \in \mathsf{st}(S, \sigma_0')$ such that $(w_0', \sigma_0') \in \mathcal{IS}(S)$, $w_0'\sigma_0'\gamma' = w$, $u_0\sigma_0'\gamma' = t'$ and $u_0\sigma_0' \notin \mathcal{X}$ for some $w_0'w_0'' = w_0$.

Let us define δ such that $dom(\delta) = vars(w_0'') \cup dom(\gamma')$, $\delta[vars(w_0'')] = \sigma_0\gamma[vars(w_0'')]$ and $\delta[dom(\gamma')] = \gamma'\alpha[dom(\gamma')]$. In such a case, we deduce that $w_0\sigma_0'\delta = w_0\sigma_0\gamma$. But (w, σ_0) is maximal for γ in K hence there exists θ such that $\sigma_0 = \sigma_0'\theta$. Therefore, we have that $u_0\sigma_0\gamma = u_0\sigma_0'\theta\gamma$ and $u_0\sigma_0 \notin \mathcal{X}$. But $w_0\sigma_0'\delta = w_0\sigma_0'\theta\gamma$ and $vars(u_0\sigma_0') \subseteq vars(w_0\sigma_0')$ hence we deduce that $u_0\sigma_0'\theta\gamma = u_0\sigma_0'\delta$ and so $u_0\sigma_0\gamma = u_0\sigma_0'\gamma'\alpha = u$.

Inductive step $N > 0$*:* there exists $g = \big(\mathsf{k}_w(R', t') \Leftarrow B_1, \ldots, B_m\big) \in K_{\mathsf{solved}}$ and a substitution α such that $w\alpha \sqsubseteq w_0\sigma\gamma$, $u = t'\alpha$, $R = R'\alpha$ and for all $i \in \{1, \ldots, m\}$, $\big(B_i\alpha \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Since K is well formed there exist σ_0', γ' such that $(w_0', \sigma_0') \in \mathcal{IS}(K)$, $w_0'\sigma_0'\gamma' = w$ with $w_0'w_0'' = w_0$. Moreover, either Property 7.a holds and so there exists $u_0 \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0')$ such that $u_0\sigma_0'\gamma' = t'$

and $u_0\sigma_0' \notin \mathcal{X}$ or else Property 7.b holds and so $dom(\gamma') = vars(w_0)$ and $t = \mathsf{f}(y_1, \ldots, y_m)$ for some function symbol f where for all $i \in \{1, \ldots, m\}$, y_i is the variable of B_i.

In the case Property 7.a holds, we do the same reasoning as in the base case of the induction which allows us to conclude. Let us focus on the case where Property 7.b holds. Since for all $i \in \{1, \ldots, m\}$, $\big(B_i\alpha \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$, by induction hypothesis, we deduce that for all $i \in \{1, \ldots, m\}$, there exist $u_1^i, \ldots, u_{m^i}^i \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$, $j_1^i, \ldots, j_{k^i}^i \in \{1, \ldots, n\}$ and C_i built only on function symbol such that:

- $y_i\alpha = C_i[t_{j_1^i}, \ldots, t_{j_{k^i}^i}, u_1^i\sigma_0\gamma, \ldots, u_{m^i}^i\sigma_0\gamma]$; and
- for all $\ell \in \{1, \ldots, m^i\}$, $u_\ell^i\sigma_0 \notin \mathcal{X}$; and
- for all p position of C_i, there exist T_i such that $\big(\mathsf{k}_{w_0\sigma_0\gamma}(R_i', y_i\alpha|_p) \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$

This allows us to first deduce that $t'\alpha = u$ has the expected form with a context $\mathsf{f}(C_1, \ldots, C_m)$. Secondly, by combining our induction hypotheses, we conclude that for all p position of $\mathsf{f}(C_1, \ldots, C_m)$, there exists T' such that $\big(\mathsf{k}_{w_0\sigma_0\gamma}(T', u|_p) \Leftarrow \mathsf{k}_{w_1}(R_1, t_1), \ldots, \mathsf{k}_{w_n}(R_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

Lemma 31. *Let S be a set of seed statements and K a well formed knowledge base built from S. Consider $f \in K$ and $g \in K_{\mathsf{solved}}$ such that:*

- $f = \big(\mathsf{k}_w(R, t) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big)$*; and*
- $g = \big(\mathsf{k}_{w'}(R', t') \Leftarrow \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)$*; and*
- *there exists $u \sqsubseteq w_1$ such that $\theta = \mathrm{mgu}(\mathsf{k}_u(X_1, t_1), \mathsf{k}_{w'}(R', t'))$.*

Let $h = \big(\mathsf{k}_w(R, t) \Leftarrow \mathsf{k}_{w_2}(X_2, t_2), \ldots, \mathsf{k}_{w_n}(X_n, t_n), \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)\theta$. We have that $K' = K \oplus h$ is well formed.

Proof. We only focus on the case where $K \oplus h \neq K$ otherwise the result trivially holds. According to Definition 28, to show that K' is well formed, we only need to prove some properties on h. Since K is well formed, we deduce that there exists $(w_0, \sigma_0) \in \mathcal{IS}(K)$ and a substitution γ such that $dom(\gamma) = vars(w_0\sigma_0)$ and the following properties hold:

f.1) (w_0, σ_0) is maximal for γ in K.
f.2) $w = w_0\sigma_0\gamma$.
f.3) for all $x \in dom(\gamma) \setminus vars(w_0)$, $vars(x\gamma) \subseteq vars(t_1, \ldots, t_n)$.
f.4) for all $x \in dom(\gamma) \cap vars(w_0)$, $x\gamma \notin vars(t, t_1, \ldots, t_n)$ and $x\gamma$ occurs only once in w.
f.5) for all $x \in dom(\gamma)$, $x\gamma \notin \mathcal{X}$ implies that there exists T such that $\big(\mathsf{k}_w(T, x\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.
f.6) one of the two following properties holds:
  a) there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ such that $t = u\sigma_0\gamma$ and for all $v \in \{t_1, \ldots, t_n\}$, v is well formed in f.
  b) there exist u and T such that $t \in \mathsf{st}(u)$, $\big(\mathsf{k}_w(T, u) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$ and for all $x \in dom(\gamma) \setminus vars(w_0)$, $x\gamma \in \mathcal{X}$ implies $x\gamma \in \{t_1, \ldots, t_n\}$.

Moreover, we also deduce that there exists $(w_0', \sigma_0') \in \mathcal{IS}(K)$ and a substitution γ' such that $dom(\gamma') = vars(w_0', \sigma_0')$ and the following properties hold:

g.1) $w_0' \sqsubseteq w_0$ and (w_0', σ_0') is maximal for γ' in K.
g.2) $w' = w_0'\sigma_0'\gamma'$.
g.3) for all $x \in dom(\gamma') \setminus vars(w_0')$, $vars(x\gamma') \subseteq vars(x_1', \ldots, x_m')$.
g.4) for all $x \in dom(\gamma') \cap vars(w_0')$, $x\gamma \notin vars(t', t_1', \ldots, t_n')$ and $x\gamma$ occurs only once in w'.
g.5) for all $x \in dom(\gamma')$, $x\gamma' \notin \mathcal{X}$ implies that there exists T such that $\big(\mathsf{k}_{w'}(T, x\gamma') \Leftarrow \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.
g.6) one of the two following properties holds:
  a) there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0')$ such that $t' = u\sigma_0'\gamma'$ and $u\sigma_0' \notin \mathcal{X}$.
  b) $dom(\gamma') = vars(w_0')$, $t' = \mathsf{f}(x_1', \ldots, x_m')$ for some function symbol f and $w' = w_i'$ $(1 \leq i \leq m)$

Since $(w_0', \sigma_0') \in \mathcal{IS}(K)$ and $w_0' \sqsubseteq w_0$ (Prop. g.1), we deduce that $(w_0, \sigma_0') \in \mathcal{IS}(K)$. Since $w_0' \sqsubseteq w_0$, there exists w_0'' such that $w_0 = w_0' w_0''$. Let us define δ' such that $dom(\delta') = dom(\gamma') \cup vars(w_0'')$, $\delta'[dom(\gamma')] = \gamma'\theta[dom(\gamma')]$ and $\delta'[vars(w_0'')] = \sigma_0\gamma\theta[vars(w_0'')]$. Moreover, let us denote $\delta = \gamma\theta[dom(\gamma)]$. We know that $\theta = \mathrm{mgu}(\mathsf{k}_u(X_1, t_1), \mathsf{k}_{w'}(R', t'))$ with $u \sqsubseteq w_1$ hence $u\theta = w'\theta$ which implies, by Prop. g.2 and f.2, that $w_0'\sigma_0\gamma\theta = w_0'\sigma_0'\gamma'\theta = w_0'\sigma_0'\delta'$. Moreover, since $dom(\sigma_0') \cap vars(w_0'') = \emptyset$, we have $w_0''\sigma_0'\delta' = w_0''\delta' = w_0''\sigma_0\gamma\theta$. Hence, we can conclude that $w_0\sigma_0\gamma\theta = w_0'\sigma_0'\delta'$.

By Lemma 27, there exist σ_0'', γ_0'', α, α' such that $(w_0, \sigma_0'') \in \mathcal{IS}(K)$, $w_0\sigma_0''\gamma'' = w_0\sigma\delta = w_0\sigma'\delta'$, $\sigma_0'' = \sigma_0\alpha = \sigma_0'\alpha'$ and for all $x \in dom(\gamma'')$, either $x \in dom(\delta)$ and $x\gamma'' = x\delta$ or $x \in dom(\delta')$ and $x\gamma'' = x\delta'$. Moreover, by Corollary 5, there exists $\sigma_0''', \gamma''', \alpha''$ such that the following properties hold:

s.1) $(w_0, \sigma_0''') \in \mathcal{IS}(K)$ and is maximal for γ''' in K.
s.2) $\sigma_0''' = \sigma_0''\alpha''$ hence $\sigma_0''' = \sigma_0\alpha\alpha'' = \sigma_0'\alpha'\alpha''$.
s.3) $w_0\sigma_0'''\gamma''' = w_0\sigma_0''\gamma'' = w_0\sigma_0\delta = w_0\sigma_0'\delta'$.
s.4) $dom(\gamma''') = vars(w_0\sigma_0''')$ and $\gamma''' = \gamma''[dom(\gamma''')]$. Hence, for all $x \in dom(\gamma''')$, either $x \in dom(\delta)$ and $x\gamma''' = x\delta$ or $x \in dom(\delta')$ and $x\gamma''' = x\delta'$.
s.5) for all $u, v \in \mathsf{st}_{\mathcal{IS}}(S, \sigma''')$, $u\sigma'''\gamma''' = v\sigma'''\gamma'''$ implies that $u\sigma''' = v\sigma'''$.

Let us now prove the different properties required for h with $(w_0, \sigma_0''') \in \mathcal{IS}(K)$ and γ'''. We already proved that (w_0, σ_0''') is maximal for γ''' in K (Property s.1). Moreover, we know that $w = w_0\sigma_0\gamma$ hence $w\theta = w_0\sigma_0\gamma\theta = w_0\sigma_0\delta = w_0\sigma_0''\gamma'''$ (Property s.3). Let us denote $Side = \mathsf{k}_{w_2\theta}(X_2, t_2\theta), \ldots, \mathsf{k}_{w_n\theta}(X_n, t_n\theta), \mathsf{k}_{w_1'\theta}(X_1', x_1'\theta), \ldots, \mathsf{k}_{w_m'\theta}(X$ and let us denote $T_B = \{t_2\theta, \ldots, t_n\theta, x_1'\theta, \ldots, x_m'\theta\}$. It remains to prove the following properties:

h.3) for all $x \in dom(\gamma''') \setminus vars(w_0)$, $vars(x\gamma''') \subseteq vars(T_B)$.
h.4) for all $x \in dom(\gamma''') \cap vars(w_0)$, $x\gamma'''$ is a variable that occurs only once in h.
h.5) for all $x \in dom(\gamma''')$, $x\gamma''' \notin \mathcal{X}$ implies that there exists T such that $\big(\mathsf{k}_{w\theta}(T, x\gamma''') \Leftarrow Side\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.
h.6) one of the two following properties holds:
  a) there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $t\theta = u\sigma_0'''\gamma'''$ and for all $v \in T_B$, v is well formed in h.
  b) there exist u and T such that $t\theta \in \mathsf{st}(u)$, $\big(\mathsf{k}_{w\theta}(T, u) \Leftarrow Side\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$ and for all $x \in dom(\gamma''') \setminus vars(w_0)$, $x\gamma''' \in \mathcal{X}$ implies $x\gamma''' \in T_B$.
h.7) if h is solved then there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $t\theta = u\sigma_0'''\gamma'''$ and $u\sigma_0''' \notin \mathcal{X}$.

Before proving properties h.3–h.7 we will show two other useful properties.

Sub-property 1: Assume that Property f.6.a is satisfied. For all $x \in dom(\theta)$, if $x\theta \notin \mathcal{X}$ then one of the following property holds:

- there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $u\sigma_0''' \notin \mathcal{X}$ and $x\theta = u\sigma_0'''\gamma'''$.
- $x\theta$ is well formed for h.

To prove this sub-property, we do an induction on the size of $|x\theta|$ with $x \in dom(\theta)$. Let us assume that $x\theta \notin \mathcal{X}$. We need to consider θ, that is $\theta = \mathrm{mgu}(\mathsf{k}_{w_0'\sigma_0\gamma}(X_1, t_1), \mathsf{k}_{w_0'\sigma_0'\gamma'}(R', t'))$. By the properties of the most general unifier, we deduce that one of the following property holds:

1. there exists $r \in \mathsf{st}(w_0'\sigma_0)$ such that $r \notin \mathcal{X}$ and $r\gamma\theta = x\theta$:
   Therefore, there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ such that $u\sigma_0\gamma\theta = x\theta$. But $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ implies $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ and by Property s.3, $w_0\sigma_0'''\gamma''' = w_0\sigma_0\delta$. Thus, $x\theta = u\sigma_0'''\gamma'''$.
2. there exists $r \in \mathsf{st}(w_0'\sigma_0')$ such that $r \notin \mathcal{X}$ and $r\gamma'\theta = x\theta$:
   Using the same reasoning as above, we deduce that there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $x\theta = u\sigma_0'''\gamma'''$.
3. there exists $y \in dom(\gamma)$ and $r \in \mathsf{st}(y\gamma)$ such that $r\theta = x\theta$ and $y\gamma \notin \mathcal{X}$:
   By Property f.5, we know that there exists T such that $\mathsf{k}_w(T, y\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. But by Lemma 30, there exist $u_1, \ldots, u_k \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$, $i_1, \ldots, i_\ell \in \{1, \ldots, n\}$ and a context C built on function symbols such that $y\gamma = C[t_{i_1}, \ldots, t_{i_\ell}, u_1\sigma_0\gamma, \ldots, u_k\sigma_0\gamma]$ and for all position p in C, there exists T such that $\big(\mathsf{k}_w(T, y\gamma|_p) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. By

Property f.6.a we have that for all $j \in \{1,\dots,\ell\}$, t_{i_j} is well formed in f. This allows us to deduce that $y\gamma$ is also well formed in f. In particular, there exist $u'_1,\dots,u'_{k'} \in \mathsf{st}_{\mathcal{IS}}(S,\sigma_0)$, $i'_1,\dots,i'_{\ell'} \in \{1,\dots,n\}$ and a context C' such that $y\gamma = C'[t_{i'_1},\dots,t_{i'_{\ell'}}, u'_1\sigma_0\gamma,\dots,u'_{k'}\sigma_0\gamma]$. W.l.o.g., we can assume that there exists a position p of C such that $r = y\gamma|_p$ (otherwise we refer to previous cases). With the fact that $y\gamma$ is well formed in f, it allows us to deduce that r is well formed in f. Furthermore, by applying our induction hypothesis on each $t_{i'_1},\dots,t_{i'_{\ell'}}$ and by application of Lemma 23, we deduce that $r\theta = x\theta$ is well formed in h.

4. there exist $y \in dom(\gamma')$ and $r \in \mathsf{st}(y\gamma')$ such that $r\theta = x\theta$ and $y\gamma' \notin \mathcal{X}$:
   This case is similar to Case 3. In fact by Property g.5, we know that there exists T such that $\mathsf{k}_{w'}(T, y\gamma) \Leftarrow \mathsf{k}_{w'_1}(X'_1,x'_1),\dots,\mathsf{k}_{w'_m}(X'_m,x'_m)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. But by Lemma 30 and the fact that all $x'_1,\dots,x'_m$ are variables, we obtain that $y\gamma$ is well formed in g. Then we apply the same reasoning as Case 3.
5. there exists $r \in \mathsf{st}(t_1)$ such that $r\theta = x\theta$ with $r \notin \mathcal{X}$:
   Since Property f.6.a holds, we know that t_1 is well formed in f. Hence we can apply the same reasoning as in Case 3 (we had $y\gamma$ well formed in f and $r \in \mathsf{st}(y\gamma)$).
6. there exists $r \in \mathsf{st}(t')$ such that $r\theta = x\theta$ with $r \notin \mathcal{X}$:
   If Property g.6.a holds then we have that $r \in \mathsf{st}(u\sigma'_0\gamma')$ for some $u \in \mathsf{st}_{\mathcal{IS}}(S,\sigma'_0)$ such that $u\sigma'_0\gamma' = t'$. In such a case, either there exists $r' \in \mathsf{st}(w'_0\sigma'_0)$ such that $r' \notin \mathcal{X}$ and $r'\gamma' = r$ or else there exists $y \in dom(\gamma')$ such that $r \in \mathsf{st}(y\gamma')$ and $y\gamma' \notin \mathcal{X}$. We can respectively apply the same reasoning applied in Case 2 and 4.
   If Property g.6.b holds then $t = \mathsf{f}(t_1,\dots,t_n)$ for some function symbol f. By considering $C = \mathsf{f}(\_,\dots,\_)$, we also have that for all position p of C, there exists T such that $\mathsf{k}_{w'}(T,t'|_p) \Leftarrow \mathsf{k}_{w'_1}(X'_1,x'_1),\dots,\mathsf{k}_{w'_m}(X'_m,x'_m)$. Thus, t' is well formed in g. Hence we can apply the same reasoning as in Case 3 (we had $y\gamma$ well formed in f and $r \in \mathsf{st}(y\gamma)$).

Sub-property 2: Assume that Properties h.5 and h.6.b are satisfied and h is solved. Let u and T such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T,u) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Let $v \in \mathsf{st}(u)$. If there is not T' such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T,v) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$ then there exists $v_0 \in \mathsf{st}_{\mathcal{IS}}(S,\sigma'''_0)$ such that $v = v_0\sigma'''_0\gamma'''$ and $v_0\sigma'''_0 \notin \mathcal{X}$. We prove this sub-property by induction on $|u|$:

Base case $|u| = 1$*:* In such a case, by Lemma 30, we deduce that either $u \in T_B$ or there exists $r \in \mathsf{st}_{\mathcal{IS}}(S,\sigma'''_0)$ such that $r\sigma'''_0\gamma''' = u$ and $r\sigma'''_0 \notin \mathcal{X}$. But $v \in \mathsf{st}(u)$ and $|u| = 1$ imply that $v = u$. Thus, $u \in T_B$ is in contradiction with the fact that there is not T' such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T,v) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Hence $r\sigma'''_0\gamma''' = u$ and $r\sigma'''_0 \notin \mathcal{X}$ which means that the result holds.

Inductive Step $|u| > 1$: By Lemma 30, we deduce that there exists $u_1,\dots,u_k \in \mathsf{st}_{\mathcal{IS}}(S,\sigma'''_0)$, $v_1,\dots,v_\ell \in T_B$ and a context C built only on function symbols such that:

- $u = C[v_1,\dots,v_\ell, u_1\sigma'''_0\gamma''',\dots,u_k\sigma'''_0\gamma''']$; and
- for all $j \in \{1,\dots,k\}$, $u_j\sigma'''_0 \notin \mathcal{X}$; and
- for all p position of C, there exists T such that $\big(\mathsf{k}_{w_0\sigma'''\gamma'''}(T,u|_p) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

But h is solved meaning that $T_B \subseteq \mathcal{X}$. But $v \in \mathsf{st}(u)$ and there is not T' such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T,v) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Hence we deduce that there exists $i \in\in \{1,\dots,k\}$ such that v is a strict subterm of $u_i\sigma'''_0\gamma'''$. In such a case, either there exists $r \in \mathsf{st}_{\mathcal{IS}}(S,\sigma'''_0)$ such that $r\sigma'''_0 \in \mathsf{st}(v_i\sigma'''_0)$, $r\sigma'''_0\gamma''' = v$ and $r\sigma'''_0 \notin \mathcal{X}$, or else there exists $y \in dom(\gamma''')$ such that $v \in \mathsf{st}(y\gamma''')$. In the first case, the result directly holds. In the later case, Property h.6.b and $v \in \mathsf{st}(y\gamma''')$ indicates that if $y\gamma''' \in \mathcal{X}$ then $v \in T_B$ which is in contradiction with the fact that there is not T' such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T,v) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Therefore, we deduce that $y\gamma''' \notin \mathcal{X}$ hence by Property h.5, we deduce that there exists T' such that $\big(\mathsf{k}_{w_0\sigma'''_0\gamma'''}(T',y\gamma''') \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. But $u_i\sigma'''_0 \in \mathcal{X}$ implies that $|y\gamma'''| < |u|$. Hence by application of our inductive hypothesis on $y\gamma'''$, we conclude that the result holds.

Now that we proved the sub-properties we need, we will prove Properties h.3 to h.7.

Property h.3 Let $x \in dom(\gamma''') \setminus vars(w_0)$. By Property s.4, we know that either $x \in dom(\delta)$ and $x\gamma''' = x\delta$ or $x \in dom(\delta')$ and $x\gamma''' = x\delta'$. But $vars(w_0') \subseteq vars(w_0)$ hence we can apply Properties f.3, g.3, and so we obtain that $vars(x\gamma''') \subset T_B \cup \{t_1\theta\}$. Since $t_1\theta = t'\theta$ with $vars(t') \subseteq \{x_1', \ldots, x_m'\}$ we conclude that $vars(x\gamma''') \subset T_B$. Therefore Property h.3 holds.

Properties h.4 and h.5 Let $x \in dom(\gamma''')$. Once again by Property s.4, We know that either $x \in dom(\delta)$ and $x\gamma''' = x\delta$ or $x \in dom(\delta')$ and $x\gamma''' = x\delta'$.

- Assume first that $x \in dom(\delta')$ and so $x\gamma''' = x\delta'$. In such a case, either $x \in dom(\gamma')$ or else $x \in vars(w_0'')$. We know that for all $dom(\gamma''') \cap dom(\sigma_0''') = \emptyset$. But $\sigma_0''' = \sigma_0\alpha\alpha'$. Hence $dom(\gamma''') \cap dom(\sigma_0) = \emptyset$. Therefore, if $x \in vars(w_0'')$ then $x \in vars(w_0)$ and so $x \in vars(w_0\sigma_0) = dom(\gamma)$. It allows us to deduce that $x \in dom(\gamma) \cap vars(w_0) \setminus dom(\sigma_0)$. By hypothesis on f, we can deduce that $x\gamma$ is a variable that occurs only once in f. In particular, $x\gamma$ does not occur in $\mathsf{k}_{w_1}(X_1, t_1)$ which implies that $x \notin vars(\theta)$. Hence we can conclude that $x\gamma\theta = x\gamma'''$ is a variable that occurs only once in h.

  Let us now consider the case $x \in dom(\gamma')$. In such a case, $x\delta' = x\gamma'\theta$. Once again, we need to distinguish wether $x \in vars(w_0')$ or not. The case $x \in vars(w_0')$ implies $x \in vars(w_0)$ and since $x \notin dom(\sigma_0''')$, which means $x \notin dom(\sigma_0)$, we deduce that $x \in dom(\gamma) \cap vars(w_0)$. Thus by hypothesis f.4, we deduce that $x\gamma$ is a variable of w that occurs only once in f. Thus, we deduce that either $x\gamma'\theta = x\gamma$ or else $x\gamma\theta = x\gamma'$. Moreover, since $x\gamma$ (resp. $x\gamma'$) is only occurring once in f (resp. g), we can conclude that $x\gamma'''$ is a variable that occurs only once in h.
  It remains to consider the case where $x \notin vars(w_0')$. In such a case, if $x\gamma' \in \mathcal{X}$ then by Corollary 3, there exists $i \in \{1, \ldots, m\}$ such that $x\gamma' = x_i'$. Otherwise, when $x\gamma' \notin \mathcal{X}$, by Property g.5, we deduce that there exists T such that $\big(\mathsf{k}_{w'}(T, x\gamma') \Leftarrow \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)$. Therefore, in both cases, we deduce that there exists T such that $\big(\mathsf{k}_{w'}(T, x\gamma') \Leftarrow \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)$. By Lemma 23 and since $x\gamma''' = x\gamma'\theta$, we conclude that $\big(\mathsf{k}_{w\theta}(T, x\gamma''') \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.
  This allows us to conclude that when $x \in dom(\delta')$, Properties h.4 and h.5 hold.
- Assume now that $x \in dom(\delta)$ and so $x\gamma''' = x\delta = x\gamma\theta$. Let first assume that $x \in vars(w_0) \setminus dom(\sigma_0''')$. In such a case, using similar reasoning as above, we deduce that $x\gamma'''$ is a variable that occurs only once in h, proving Property h.4. Secondly, assume that $x\gamma''' \notin \mathcal{X}$. In such a case, we do a case analysis on whether $x\gamma \in \mathcal{X}$ or not:

  Case $x\gamma \notin \mathcal{X}$: In such a case, by Property f.5, we know that there exist R_0 such that $\big(\mathsf{k}_w(R_0, x\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Hence, by relying on Lemma 23, we deduce that $\big(\mathsf{k}_{w\theta}(R_0\theta, x\gamma''') \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

  Case $x\gamma \in \mathcal{X}$: Since $x\gamma''' \notin \mathcal{X}$, we deduce that $x\gamma \in dom(\theta)$. Let us first assume that Property f.6.a is satisfied. In such a case, by Sub-property 1, one of the following properties holds:
  - there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $u\sigma_0''' \notin \mathcal{X}$ and $x\gamma\theta = u\sigma_0'''\gamma'''$: This case is in fact impossible. Indeed, we already know that $x = x\sigma_0'''$ hence $x\gamma\theta = x\sigma_0'''\gamma''' = u\sigma_0'''\gamma'''$. Therefore by Property s.5, we deduce that $x = x\sigma_0''' = u\sigma_0'''$ which is a contradiction with $u\sigma_0''' \notin \mathcal{X}$.
  - $x\gamma\theta$ is well formed in h: By definition of $x\gamma'''\theta$ being well formed, we deduce that $\big(\mathsf{k}_{w\theta}(T, x\gamma''') \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$

  Let us now assume that Property f.6.b is satisfied. In such a case, we know that $x\gamma \in \{t_2, \ldots, t_n\}$. Hence, we trivially deduce that $\big(\mathsf{k}_{w\theta}(T, w\gamma\theta) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$.

Property h.6 We know that either Property f.6.a or f.6.b holds. Let us assume that Property f.6.a holds. In such a case, we know that there exits $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ such that $t = u\sigma_0\gamma$ and for all $i \in \{1, \ldots, n\}$, t_i is well formed in f. But by Property s.2, $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ implies that $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$. Moreover, by Property s.3, we deduce that $u\sigma_0\gamma\theta = u\sigma_0'''\gamma'''$.

Let $x \in \{x_1, \ldots, x_m\} \cup \{t_2, \ldots, t_n\} \cap \mathcal{X}$. If $x \notin dom(\theta)$ then $x \in T_B$ and trivially well formed in h. If $x \in dom(\theta)$ then by Sub-property 1 and the fact that $x\theta \in T_B$, we deduce that $x\theta$ is well formed in

h. Let $v \in \{t_2, \ldots, t_n\} \setminus \mathcal{X}$. By Property f.6.a, we deduce that v is well formed in f that is there exist $u_1, \ldots, u_k \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$, $i_1, \ldots, i_\ell \in \{1, \ldots, n\}$ and a context C built on symbol functions such that $v = C[t_{i_1}, \ldots, t_{i_\ell}, u_1\sigma_0\gamma, \ldots, u_k\sigma_0\gamma]$, for all $j \in \{1, \ldots, \ell\}$, $t_{i_j} \in \mathcal{X}$ and for all position p of C, there exists T such that $\big(\mathsf{k}_w(T, u|_p) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Thus, $v\theta = C[t_{i_1}\theta, \ldots, t_{i_\ell}\theta, u_1\sigma_0'''\gamma''', \ldots, u_k\sigma_0'''\gamma''']$. By applying Sub-Property 1 on each $t_{i_j}\theta$, $j \in \{1, \ldots, \ell\}$, we can conclude that $v\theta$ is well formed in h and so Property h.6.a holds.

Assume now that Property f.6.b holds. First, there exists u and T such that $t \in \mathsf{st}(u)$ and $\big(\mathsf{k}_w(T, u) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Therefore, we directly have by Lemma 23 that $\big(\mathsf{k}_{w\theta}(T\theta, u\theta) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Second, let $x \in dom(\gamma''') \setminus vars(w_0)$ such that $x\gamma''' \in \mathcal{X}$. We know that either $x\gamma''' = x\gamma\theta$ or $x\gamma''' = x\gamma'\theta$ (Property s.4). If $x\gamma''' = x\gamma\theta$ (resp $x\gamma'\theta$) then it implies that $x\gamma \in \mathcal{X}$ (resp. $x\gamma' \in \mathcal{X}$). But by Property f.6.b (resp. Property g.6), we deduce that $x\gamma \in \{t_2, \ldots, t_n\}$ (resp. $x\gamma' \in \{x'_1, \ldots, x'_m\}$) hence so $x\gamma''' \in T_B$.

Property h.7 Suppose $h\Downarrow$ is solved. With $h\Downarrow$ being solved and $K' = K \oplus h \neq K$, we deduce that there is no T such that $\big(\mathsf{k}_{w\theta}(T, t\theta) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. If Property h.6.a holds then there exits $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $t\theta = u\sigma_0'''\gamma'''$. Thus, we only need to show that $u\sigma_0''' \notin \mathcal{X}$. We show it by contradiction: $u\sigma_0''' \in \mathcal{X}$ implies that $u\sigma_0''' \in dom(\gamma''')$. But if $u\sigma_0'''\gamma''' \in \mathcal{X}$ then $t\theta = u\sigma_0'''\gamma''' \in T_B$, and if $t\theta = u\sigma_0'''\gamma''' \notin \mathcal{X}$, Property h.5 holds. Therefore, we deduce that there exists T such that $\big(\mathsf{k}_{w\theta}(T, t\theta) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$, which is a contradiction with our hypothesis. Therefore, when Property h.6.a holds, Property h.7.a holds.

Let us now focus on the case where Property h.6.b holds which implies that there exists u and T such that $t\theta \in |(u)$ and $\big(\mathsf{k}_{w\theta}(T, u) \Leftarrow B\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Therefore, by applying Sub-property 2, we deduce that there exists $v_0 \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0''')$ such that $t\theta = v_0\sigma_0'''\gamma'''$ and $v_0\sigma_0''' \notin \mathcal{X}$. Hence we can conclude that Property h.7 holds.

Combining Lemmas 29 and 31 we obtain the following corollary

Corollary 6. *Let S be a set of seed statements and K a knowledge based built from S. K is well-formed.*

The measures

Definition 29. *Let S be a set of seed statements and K be a well formed knowledge base built from S. Let $N = |\mathsf{st}(\mathsf{IPC}(S))|$. Let $f = \big(\mathsf{k}_w(R, t) \Leftarrow B_1, \ldots, B_n\big) \in K$. Let $(w_0, \sigma_0) \in \mathcal{IS}(K)$ and γ be a substitution such that $dom(\gamma) = vars(w_0\sigma_0)$, (w_0, σ_0) is maximal for γ in K and $w = w_0\sigma_0\gamma$. (Existence of (w_0, σ_0) and γ is guaranteed since K is well formed). We define the measure*

$$\mathsf{m}_C(f, K) = N - \Big|\{u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0) \mid \exists T.\big(\mathsf{k}_w(T, u\sigma_0\gamma) \Leftarrow B_1, \ldots, B_n\big) \in \mathbf{conseq}(K_{\mathsf{solved}})\}\Big|$$

Lemma 32. *Let S be a set of seed statements and K be a well formed knowledge base built from S. Consider $f \in K \setminus K_{\mathsf{solved}}$ and $g \in K_{\mathsf{solved}}$ such that:*

- *$f = \big(\mathsf{k}_w(R, t) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big)$; and*
- *$g = \big(\mathsf{k}_{w'}(R', t') \Leftarrow \mathsf{k}_{w'_1}(X'_1, x'_1), \ldots, \mathsf{k}_{w'_m}(X'_m, x'_m)\big)$; and*
- *there exists $u \sqsubseteq w_1$ such that $\theta = \mathrm{mgu}(\mathsf{k}_u(X_1, t_1), \mathsf{k}_{w'}(R', t'))$.*

Let $h = \big(\mathsf{k}_w(R, t) \Leftarrow \mathsf{k}_{w_2}(X_2, t_2), \ldots, \mathsf{k}_{w_n}(X_n, t_n), \mathsf{k}_{w'_1}(X'_1, x'_1), \ldots, \mathsf{k}_{w'_m}(X'_m, x'_m)\big)\theta$. We have that $\mathsf{m}_C(h, K) \leq \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g, K))$. Then we have that

- *for all $f' \in K$, $\mathsf{m}_C(f', K \oplus h) \leq \mathsf{m}_C(f', K)$*
- *if h is solved and $K \oplus h \neq K$ then $\mathsf{m}_C(h, K \oplus h) < \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g, K))$*

Proof. Since K is well formed, we know that there exist $w_0, \sigma_0, \sigma_0'', \gamma, \gamma''$ such that $(w_0, \sigma_0), (w_0, \sigma_0'') \in \mathcal{IS}(K)$ and:

- (w_0, σ_0) (resp. (w_0, σ_0'')) is maximal for γ (resp. γ'') in K

– $dom(\gamma) = vars(w_0\sigma_0)$ and $dom(\gamma'') = vars(w_0\sigma_0'')$
– $w = w_0\sigma_0\gamma$ and $w\theta = w_0\sigma_0''\gamma''$.

Since (w_0, σ_0'') is maximal for γ'', we deduce that there exists α such that $\sigma_0'' = \sigma_0\alpha$. Therefore, we deduce that $\gamma\theta[dom(\gamma)] = \alpha\gamma''[dom(\gamma)]$.

Let $u \in \mathcal{IS}(S, \sigma_0)$ and T such that $\big(\mathsf{k}_w(T, u\sigma_0\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. Since $\sigma_0'' = \sigma_0\alpha$, we deduce that $u \in \mathcal{IS}(S, \sigma_0'')$. Moreover, by Lemma 23, $\big(\mathsf{k}_w(T, u\sigma_0\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$ implies that $\big(\mathsf{k}_{w\theta}(T, u\sigma_0\gamma\theta) \Leftarrow \mathsf{k}_{w_2\theta}(X_2, t_2\theta), \ldots, \mathsf{k}_{w_n\theta}(X_n, t_n\theta), \mathsf{k}_{w_1'\theta}(X_1', x_1'\theta), \ldots, \mathsf{k}_{w_m'\theta}(X_m', x_m'\theta)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})$. But $u\sigma_0\gamma = u\sigma_0\alpha\gamma'' = u\sigma_0''\gamma''$. Hence, we can conclude that $\mathsf{m}_C(h, K) \leq \mathsf{m}_C(f, K)$. By applying similar reasoning, we deduce that $\mathsf{m}_C(h, K) \leq \mathsf{m}_C(g, K)$. Therefore, we conclude that $\mathsf{m}_C(h, K) \leq \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g, K))$.

Let $f' \in K$. By definition of **conseq**, we directly have that $\mathbf{conseq}(K_{\mathsf{solved}}) \subseteq \mathbf{conseq}((K \oplus h)_{\mathsf{solved}})$. Therefore, we deduce that $\mathsf{m}_C(f', K \oplus h) \leq \mathsf{m}_C(f', K)$.

Moreover, if h is solved and $K \oplus h \neq K$ then there is no T such that $\big(\mathsf{k}_w(T, t) \Leftarrow \mathsf{k}_{w_2}(X_2, t_2), \ldots, \mathsf{k}_{w_n}(X_n, t_n), \mathsf{k}_{w_1'}(X_1', x_1'), \ldots$ $\mathbf{conseq}(K_{\mathsf{solved}})$. Since K is well formed, we know that there exists $u_0 \in \mathcal{IS}(S, \sigma_0''')$ such that $t\theta = u_0\sigma_0'''\gamma'''$. Therefore, we deduce that $u_0 \notin \{u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0) | \exists R \text{ s.t. } \big(\mathsf{k}_w(R, u\sigma_0\gamma) \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in \mathbf{conseq}(K_{\mathsf{solved}})\}$. It implies that $\mathsf{m}_C(h, K \oplus h) < \mathsf{m}_C(f, K \oplus h)$. By using a similar reasoning, we deduce that $\mathsf{m}_C(h, K \oplus h) < \mathsf{m}_C(g, K \oplus h)$ and so we can conclude that $\mathsf{m}_C(h, K \oplus h) < \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g, K))$.

Definition 30. *Let S be a set of seed statements and K be a well formed knowledge base built from S. Let $f \in K \setminus K_{\mathsf{solved}}$ and $g_1, \ldots, g_n \in K_{\mathsf{solved}}$. We denote by $\mathsf{S}_{\text{RES}}(f, [g_1, \ldots, g_n])$ the set of clauses such that for all $h \in \mathsf{S}_{\text{RES}}(f, [g_1, \ldots, g_n])$, there exist clauses $h_0, \ldots, h_n$ such that $h_0 = f$, $h = h_n$ and for all $i \in \{1, \ldots, n\}$, h_i is the result of an application of the rule* RESOLUTION *on h_{i-1} and g_i.*

Moreover we define the measure

$$\mathsf{m}_A(K, f, [g_1, \ldots, g_n]) = |\mathsf{S}_{\text{RES}}(f, [g_1, \ldots, g_n]) \setminus K|$$

Corollary 7. *Let S be a set of seed statements and K be a well formed knowledge base built from S. Let $f \in K \setminus K_{\mathsf{solved}}$. Let $g_1, \ldots, g_n \in K_{\mathsf{solved}}$. For all $h \in \mathsf{S}_{\text{RES}}(f, [g_1, \ldots, g_n])$, for all $f' \in K$,*

– $\mathsf{m}_C(h, K \oplus h) \leq \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K))$
– $\mathsf{m}_C(f', K \oplus h) \leq \mathsf{m}_C(f', K)$
– *if h is solved and $K \oplus h \neq K$ then* $\mathsf{m}_C(h, K \oplus h) < \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K))$

Lemma 33. *Let S be a set of seed statements and K a well formed knowledge base built from S. For all $f \in K$, there exists $N \in \mathbb{N}$ such that for all $M > N$, for all $g_1, \ldots, g_M \in K_{\mathsf{solved}}$, $\mathsf{S}_{\text{RES}}(f, [g_1, \ldots, g_M]) = \emptyset$.*

Proof. For all $f = \big(H \Leftarrow \mathsf{k}_{w_1}(X_1, t_1), \ldots, \mathsf{k}_{w_n}(X_n, t_n)\big) \in K$, we define the multiset

$$\mathsf{m}(f) = \{\{(|w_i|, |t_i|) \mid i \in \{1, \ldots, n\}\}\}$$

Consider now $g \in K_{\mathsf{solved}}$ such that:

– $g = \big(\mathsf{k}_{w'}(R', t') \Leftarrow \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)$;
– $t_1 \notin \mathcal{X}$ and
– there exists $u \sqsubseteq w_1$ such that $\theta = \text{mgu}(\mathsf{k}_u(X_1, t_1), \mathsf{k}_{w'}(R', t'))$.

Let $h = \big(H \Leftarrow \mathsf{k}_{w_2}(X_2, t_2), \ldots, \mathsf{k}_{w_n}(X_n, t_n), \mathsf{k}_{w_1'}(X_1', x_1'), \ldots, \mathsf{k}_{w_m'}(X_m', x_m')\big)\theta$.

We know that K is well formed. In particular, there exist $(w_0, \sigma_0) \in \mathcal{IS}(K)$ and a substitution γ such that $dom(\gamma) = vars(w_0\sigma_0)$ and $w' = w_0\sigma_0\gamma$. Moreover, either (a) $t' = \mathsf{f}(x_1', \ldots, x_m')$ for some function symbol f and $w' = w_1' = \ldots = w_m'$; or else (b) there exists $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ such that $t' = u\sigma_0\gamma$. But $u \in \mathsf{st}_{\mathcal{IS}}(S, \sigma_0)$ implies that $vars(u\sigma_0) \subseteq vars(w_0\sigma_0)$. Therefore, $vars(t') \subseteq vars(w_0\sigma_0\gamma) = vars(w')$ and so by Corollary 3, we deduce that for all $i \in \{1, \ldots, m\}$, $|w_i'| < |w'|$.

In case (a), since $t_1 \notin \mathcal{X}$, we deduce that for all $i \in \{1, \ldots, n\}$, $t_i\theta = t_i$ and for all $j \in \{1, \ldots, m\}$, $|x_j\theta| < |t_1\theta|$. Therefore, whether it is Case (a) or (b), $\mathsf{m}(h)$ is the multi set $\mathsf{m}(f)$ where we replace the element $(|w_1|, |t_1|)$ by several elements $(|w'\theta|, |x_i'\theta|)$, $i \in \{1, \ldots, m\}$ strictly smaller than $(|w_1|, |t_1|)$. We can conclude that $\mathsf{m}(h) < \mathsf{m}(f)$ and so the result holds.

Definition 31. *Let S be a set of seed statements and K be a well formed knowledge base built from S. We denote by $\mathsf{m}_F(K)$ the following multiset:*

$$\left\{\!\!\left\{ \begin{pmatrix} \min(\mathsf{m}_C(f,K), \mathsf{m}_C(g_1,K), \ldots, \mathsf{m}_C(g_n,K)), \\ \mathsf{m}_A(K, f, [g_1, \ldots, g_n]) \end{pmatrix} \middle| \begin{array}{l} f \in K_i(S) \setminus K_i(S)_{\mathsf{solved}}, \\ g_1, \ldots, g_n \in K_{\mathsf{solved}}, \\ \mathsf{S}_{\mathrm{RES}}(f, [g_1, \ldots, g_n]) \neq \emptyset \end{array} \right\}\!\!\right\}$$

We use the natural ordering on multiset with the lexicographic ordering for the elements of $\mathsf{m}_F(K)$.

Lemma 34. *Let S be a set of seed statements and K be a well formed knowledge base built from S. Let $f \in K_i(S) \setminus K_i(S)_{\mathsf{solved}}$. Let $g_1, \ldots, g_n \in K_{\mathsf{solved}}$. Let $h \in \mathsf{S}_{\mathrm{RES}}(f, [g_1, \ldots, g_n])$ such that $h \notin K$ and $K \oplus h \neq K$. We have that $\mathsf{m}_F(K \oplus h) < \mathsf{m}_F(K)$.*

Proof. Let $f' \in K_i(S) \setminus K_i(S)_{\mathsf{solved}}$ and $g'_1, \ldots, g'_m \in K_{\mathsf{solved}}$ such that $\mathsf{S}_{\mathrm{RES}}(f', [g'_1, \ldots, g'_m]) \neq \emptyset$. By Corollary 7, we know that $\mathsf{m}_C(f', K \oplus h) \leq \mathsf{m}_C(f', K)$ and for all $i \in \{1, \ldots, m\}$, $\mathsf{m}_C(g'_i, K \oplus h) \leq \mathsf{m}_C(g'_i, K)$. Moreover by definition, we trivially have that $\mathsf{m}_A(K \oplus h, f', [g'_1, \ldots, g'_m]) \leq \mathsf{m}_A(K, f', [g'_1, \ldots, g'_m])$. Since $h \notin K$, $h \in \mathsf{S}_{\mathrm{RES}}(f, [g_1, \ldots, g_n])$ and $h \in K \oplus h$, we also deduce that $\mathsf{m}_A(K \oplus h, f, [g_1, \ldots, g_n]) < \mathsf{m}_A(K, f, [g_1, \ldots, g_n]])$.

Let us first consider the case where h is not solved. In such a case, $K_{\mathsf{solved}} = (K \oplus h)_{\mathsf{solved}}$. Moreover, we just showed that:

- $\min(\mathsf{m}_C(f', K \oplus h), \mathsf{m}_C(g'_1, K \oplus h), \ldots, \mathsf{m}_C(g'_m, K \oplus h))$ $\leq \min(\mathsf{m}_C(f', K), \mathsf{m}_C(g'_1, K), \ldots, \mathsf{m}_C(g'_m, K))$; and
- $\mathsf{m}_A(K \oplus h, f', [g'_1, \ldots, g'_m]) \leq \mathsf{m}_A(K, f', [g'_1, \ldots, g'_m])$; and
- $\min(\mathsf{m}_C(f, K \oplus h), \mathsf{m}_C(g_1, K \oplus h), \ldots, \mathsf{m}_C(g_n, K \oplus h))$ $\leq \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K))$; and
- $\mathsf{m}_A(K \oplus h, f, [g_1, \ldots, g_n]) < \mathsf{m}_A(K, f, [g_1, \ldots, g_n]])$.

This allows us to deduce that $\mathsf{m}_F(K \oplus h) < \mathsf{m}_F(K)$.

In the case where h is solved, we need to consider more elements for $\mathsf{m}_F(K \oplus h)$ since $(K \oplus h)_{\mathsf{solved}} = K_{\mathsf{solved}} \cup \{h\}$. By Corollary 7, $\mathsf{m}_C(h, K \oplus h) < \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K))$. Therefore, for all $f'' \in K_i(S) \setminus K_i(S)_{\mathsf{solved}}$, for all $g''_1, \ldots, g''_k \in K_{\mathsf{solved}} \cup \{h\}$, if $h \in \{g''_1, \ldots, g''_k\}$ then $\min(\mathsf{m}_C(f'', K \oplus h), \mathsf{m}_C(g''_1, K \oplus h), \ldots, \mathsf{m}_C(g''_k, K \oplus h)) < \min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K))$. Hence, $\mathsf{m}_F(K \oplus h)$ is the multiset $\mathsf{m}_F(K)$ where we replaced at least one element, i.e.,

$$(\min(\mathsf{m}_C(f, K), \mathsf{m}_C(g_1, K), \ldots, \mathsf{m}_C(g_n, K)), \mathsf{m}_A(K, f, [g_1, \ldots, g_n]]))$$

by several strictly smaller elements:

- $(\min(\mathsf{m}_C(f, K \oplus h), \mathsf{m}_C(g_1, K \oplus h), \ldots, \mathsf{m}_C(g_n, K \oplus h)), \mathsf{m}_A(K \oplus h, f, [g_1, \ldots, g_n]))$
- $(\min(\mathsf{m}_C(f'', K \oplus h), \mathsf{m}_C(g''_1, K \oplus h), \ldots, \mathsf{m}_C(g''_k, K \oplus h)), \mathsf{m}_A(K \oplus h, f'', [g''_1, \ldots, g''_n]))$ for all $f'' \in K_i(S) \setminus K_i(S)_{\mathsf{solved}}$ and for all $g''_1, \ldots, g''_k \in K_{\mathsf{solved}} \cup \{h\}$ such that $h \in \{g''_1, \ldots, g''_k\}$.

This allows us to conclude that $\mathsf{m}_F(K \oplus h) < \mathsf{m}_F(K)$.

Theorem 4. *Let T be a ground trace and $S = \mathsf{seed}(T)$. For a subterm convergent rewrite system the computation of $\mathsf{sat}(K_i(S))$ terminates in a finite number of steps.*

Proof. We have by Corollary 6 that $\mathsf{sat}(K_i(S))$ is well-formed. Hence, by Lemma 33 the number of elements in $\mathsf{m}_F(K)$ is finite. Moreover, by Lemma 34, m_F strictly decreases when applying rule Resolution on a statement with a knowledge predicate as head. Moreover, by Corollary 6 and Lemma 33 the measure on the resulting knowledge base also contains a finite number of elements. Hence, the Resolution rule only generates a finite number of statements in $\mathsf{sat}(K_i(S))$ with a knowledge predicate as head. As a direct consequence the rule Equation also generates a finite number of statements. Lastly, by Lemma 33, we can deduce that Resolution and Test generate only a finite number of statements, whatever the head predicate.

E Proof of the algorithm

In order to prove Theorem 5 we need the following technical lemmas.

Lemma 35. *Let T be a trace and let K be a saturated knowledge base associated to T. Then for any statement $f \in K$, we have that:*

1. *if $f = \left(\mathsf{r}_{l_1,\ldots,l_n} \Leftarrow \{\mathsf{k}_{w_i}(X_i, t_i)\}_{i\in\{1,\ldots,m\}}\right)$ and $x \in vars(l_k)$ then there exists $w_j = l_1, \ldots, l_{k'}$ with $k' < k$ such that $x \in vars(t_j)$.*
2. *if $f = \left(\mathsf{k}_{l_1,\ldots,l_n}(R, t) \Leftarrow \{\mathsf{k}_{w_i}(X_i, t_i)\}_{i\in\{1,\ldots,m\}}\right)$ and $x \in vars(t)$ then $x \in vars(t_1, \ldots, t_m)$.*

Proof. The seed knowledge base satisfies the above properties and they are preserved by canonicalization, update and saturation.

Lemma 36. *Let T_0 be a trace, $\varphi_0 = \emptyset$ the empty frame, and $\{c_1, \ldots, c_k\}$ names such that $c_i \notin names(T_0)$ for all $1 \leq i \leq k$.*
If

$$(T_0, \varphi_0) \stackrel{L_1}{\Longrightarrow} (T_1, \varphi_1) \stackrel{L_2}{\Longrightarrow} \ldots \stackrel{L_n}{\Longrightarrow} (T_n, \varphi_n)$$

and $\forall 1 \leq i \leq k$

- *either $c_i \notin names(L_1, \ldots, L_n)$*
- *or $\varphi_{idx(c_i)-1} \vdash^{R_i} t_i$ for some t_i where $idx(c_i) = \min\{j \mid c_i \in names(L_j)\}$*

then

$$(T_0, \varphi_0) \stackrel{L_1\pi'}{\Longrightarrow} (T_1\pi, \varphi_1\pi) \stackrel{L_2\pi'}{\Longrightarrow} \ldots \stackrel{L_n\pi'}{\Longrightarrow} (T_n\pi, \varphi_n\pi),$$

where $\pi' = \{c_i \mapsto R_i\}_{i\in\{1,\ldots,k\}}$ and $\pi = \{c_i \mapsto t_i\}_{i\in\{1,\ldots,k\}}$.

Proof. By induction on n, the same operational steps will take place with the new labels.

Lemma 37. *Let T be a trace, let $\{c_1, \ldots, c_k\}$ be public names not appearing in T and let $\pi : \{c_1, \ldots, c_k\} \to \mathsf{Messages}$ and $\pi' : \{c_1, \ldots, c_k\} \to \mathsf{Recipes}$ be mappings from names to terms. If $T \models \mathsf{k}_w(R, t)$ and $T \models \mathsf{k}_{w\pi}(c_i\pi', c_i\pi)$ then $T \models \mathsf{k}_{w\pi}(R\pi', t\pi)$.*

Proof. By induction on R.

Lemma 38. *Let T be a trace and φ a frame such that $(T, \varphi) \stackrel{L}{\Rightarrow} (T', \varphi')$ and such that*

1. *either $M = L$,*
2. *or $L = \boldsymbol{in}(d, R)$ and $M = \boldsymbol{in}(d, R')$ such that $(R = R')\varphi$.*

Then we have that $(T, \varphi) \stackrel{M}{\Rightarrow} (T', \varphi')$.

Proof. If $M = L$ then the result is obvious. Otherwise, R and R' are recipes for the same term in φ and therefore the transition still holds. □

Theorem 5. *Let T be a ground trace, P a ground process and $K = (\mathsf{sat}(K_i(T)))_{\mathsf{solved}}$. We have that*

- *if $T \sqsubseteq_{ct} P$ then* Reachability*(K, P) and* Identity*(K, P) hold.*
- *if P is determinate and* Reachability*(K, P) and* Identity*(K, P) hold then $T \sqsubseteq_{ct} P$.*

Proof. We first prove that if any of the tests fail then $T \not\sqsubseteq P$.

– If the Reachability test fails, we have that $\left(\mathsf{r}_{l_1,\ldots,l_n} \Leftarrow \{\mathsf{k}_{w_i}(X_i, x_i)\}_{i\in\{1,\ldots,m\}}\right) \in K$ and for all T', φ we have that $P \overset{M_1,\ldots,M_n}{\not\Longrightarrow} (T', \varphi)$. By Theorem 3 (soundness of K), we have that there exists T'', φ'' such that $(T, \emptyset) \xrightarrow{M_1,\ldots,M_n} (T'', \varphi'')$. Hence, $T \not\sqsubseteq P$.
– If the Identity test fails, we have that $\left(\mathsf{ri}_{l_1,\ldots,l_n}(R, R') \Leftarrow \{\mathsf{k}_{w_i}(X_i, x_i)\}_{i\in\{1,\ldots,m\}}\right) \in K$ and:

1. either $(P, \emptyset) \overset{M_1,\ldots,M_n}{\not\Longrightarrow} (T', \varphi)$ for all T', φ. However, by Theorem 3 (soundness of K), we have that there exists T'', φ'' such that $(T, \emptyset) \xrightarrow{M_1,\ldots,M_n} (T'', \varphi'')$. Hence, $T \not\sqsubseteq P$.
2. or for any T', φ such that $(P, \emptyset) \overset{M_1,\ldots,M_n}{\Longrightarrow} (T', \varphi)$ we have $(R\pi \neq R'\pi)\varphi$. By Theorem 3, we have however that that there exists T'', φ'' such that $(T, \emptyset) \xrightarrow{M_1,\ldots,M_n} (T'', \varphi'')$ and $(R\pi = R'\pi)\varphi''$. Hence, $T \not\sqsubseteq P$.

Next, we prove that if $T \not\sqsubseteq P$, then at least one test fails. We assume by contradiction that $T \not\sqsubseteq P$, that all tests pass and we derive a contradiction. As $T \not\sqsubseteq P$, it follows that there exist $L_1, \ldots, L_n, \varphi$ such that

– either $(T, \emptyset) \xrightarrow{L_1,\ldots,L_n} (T', \varphi)$ and $\forall S \in P, S', \psi.\ (S, \emptyset) \overset{L_1,\ldots,L_n}{\not\Longrightarrow} (S', \psi)$.
– or, $(T, \emptyset) \xrightarrow{L_1,\ldots,L_n} (T', \varphi)$ and $(R = R')\varphi$ and $\forall S \in P, S', \psi$ if $(S, \emptyset) \overset{L_1,\ldots,L_n}{\Longrightarrow} (S', \psi)$ then $(R \neq R')\psi$.

Let n be the smallest index such that one of the above holds. We then have that:

$$(T, \emptyset) \xrightarrow{L_1} (T_1, \varphi_1) \xrightarrow{L_2} \ldots \xrightarrow{L_{n-1}} (T_{n-1}, \varphi_{n-1}) \xrightarrow{L_n} (T_n, \varphi_n)$$

and for all R, R' and i, such that $1 \leq i \leq n-1$ and$(R = R')\varphi_i$ there exists $S \in P$ such that

$$(S, \emptyset) \overset{L_1}{\Longrightarrow} (S_1, \psi_1) \overset{L_2}{\Longrightarrow} \ldots \overset{L_i}{\Longrightarrow} (S_i, \psi_i),$$

and $(R = R')\psi_i$ and

1. either for all $U \in P, V$ we have $(U, \emptyset) \overset{L_1,\ldots,L_n}{\not\Longrightarrow} (V, \psi)$
2. or there exist recipes R, R' such that for all $U \in P, V$ such that $(U, \emptyset) \overset{L_1,\ldots,L_n}{\Longrightarrow} (V, \psi)$ we have $(R \neq R')\psi$.

We consider each of these two cases separately:

1. As $(T, \emptyset) \xrightarrow{L_1,\ldots,L_n} (T_n, \varphi_n)$, we have by Theorem 3 (completeness) that $\mathsf{r}_{L_1\varphi_n\downarrow,\ldots,L_n\varphi_n\downarrow} \in \mathcal{H}_{\mathsf{e}}(K)$. By the definition of $\mathcal{H}_{\mathsf{e}}$, we have that it contains no reachability statements in addition to those in $\mathcal{H}$. Therefore $\mathsf{r}_{L_1\varphi_n\downarrow,\ldots,L_n\varphi_n\downarrow} \in \mathcal{H}(K)$. Hence there exist a statement $f = \left(\mathsf{r}_{l_1,\ldots,l_n} \Leftarrow \mathsf{k}_{w_i}(X_i, x_i)_{i\in\{1,\ldots,m\}}\right) \in K$ and a substitution τ grounding for f such that $l_i\tau = L_i\varphi_n\downarrow$ (for all $1 \leq i \leq n$) and such that $\mathsf{k}_{w_i\tau}(X_i\tau, x_i\tau) \in \mathcal{H}(K)$.
   Let $c_1, \ldots, c_k$ be fresh public names and let $\sigma : vars(l_1, \ldots, l_n) \to \{c_1, \ldots, c_k\}$ be a bijection. For all $1 \leq j \leq k$ we have that $\left(\mathsf{k}(c_j, c_j) \Leftarrow\right) \in K_i(T)$. By definition of $\mathcal{H}$ we have that $\mathsf{k}_{w_i\sigma}(X_i\sigma', x_i\sigma) \in \mathcal{H}(K)$ for all $1 \leq i \leq m$ where $dom(\sigma') = \{X_1, \ldots, X_m\}$ and $\sigma'(X_i) = x_i\sigma$ for all $1 \leq i \leq m$.
   Instantiating f with $\sigma \cup \sigma'$, we obtain that $\mathsf{r}_{l_1\sigma,\ldots,l_n\sigma} \in \mathcal{H}(K)$. By Theorem 3 (soundness), it follows that $T \models \mathsf{r}_{l_1\sigma,\ldots,l_n\sigma}$. Therefore, there exist recipes R'_i (for all $1 \leq i \leq n$ such that $l_i = \mathbf{in}(d_i, t_i)$) such that $T \models \mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R'_i, t_i\sigma)$. By Theorem 3 (completeness) and definition of $\mathcal{H}_{\mathsf{e}}$ there exist recipes R_i such that $\mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R_i, t_i\sigma) \in \mathcal{H}(K)$.
   Let $M_i = l_i$ if $l_i \in \{\mathbf{test}, \mathbf{out}(c) \mid c \in \mathcal{C}\}$ and let $M_i = \mathbf{in}(d_i, R_i)$ if $l_i = \mathbf{in}(d_i, t_i)$ for all $1 \leq i \leq n$. As Reachability(K, P) holds there exists $S'_0 \in P$ such that, if we let $\psi'_0 = \emptyset$, we have

$$(S'_0, \psi'_0) \overset{M_1}{\Longrightarrow} (S'_1, \psi'_1) \overset{M_2}{\Longrightarrow} \ldots \overset{M_n}{\Longrightarrow} (S'_n, \psi'_n).$$

Let i be such that $l_i = \mathbf{in}(d_i, t_i)$. Applying Lemma 35 to f we have that for all $x \in vars(t_i)$ there exists w_j such that $|w_j| < i$ and $x = x_j$. We already have that $\mathsf{k}_{w_j\tau}(X_j\tau, x_j\tau) \in \mathcal{H}(K)$ by choice of f and of τ. By Theorem 3 (soundness), we obtain that $T \models \mathsf{k}_{w_j\tau}(X_j\tau, x_j\tau)$. Hence, as $|w_j| < i$, $T \models \mathsf{k}_{l_1\tau,\ldots,l_{i-1}\tau}(X_j\tau, x_j\tau)$.
Let $\pi : \{c_1, \ldots, c_k\} \to \mathsf{Messages}$ and $\pi' : \{c_1, \ldots, c_k\} \to \mathsf{Recipes}$ be two mappings such that $\pi(c_l) = x_j\tau$ and $\pi'(c_l) = X_j\tau$ when $\sigma(x_j) = c_l$. As $X_j\tau = c_l\pi'$, $x_j\tau = c_l\pi$ and $l_1\tau, \ldots, l_{i-1}\tau = l_1\sigma\pi, \ldots, l_{i-1}\sigma\pi$ therefore we have that $T \models \mathsf{k}_{l_1\sigma\pi,\ldots,l_{i-1}\sigma\pi}(c_l\pi', c_l\pi)$. We already established that $\mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R_i, t_i\sigma) \in \mathcal{H}(K)$. By Theorem 3 (soundness) we have that $T \models \mathsf{k}_{l_1\sigma,\ldots,l_{i-1}\sigma}(R_i, t_i\sigma)$. We apply Lemma 37 to obtain that $T \models \mathsf{k}_{l_1\sigma\pi,\ldots,l_{i-1}\sigma\pi}(R_i\pi', t_i\sigma\pi)$. But $t_i\sigma\pi = t_i\tau$ and $l_1\sigma\pi, \ldots, l_{i-1}\sigma\pi = l_1\tau, \ldots, l_{i-1}\tau$ and therefore we have that $T \models \mathsf{k}_{l_1\tau,\ldots,l_{i-1}\tau}(R_i\pi', t_i\tau)$. From Lemma 36, we obtain that

$$(S'_0, \psi'_0) = (S'_0\pi, \psi_0\pi) \xRightarrow{M_1\pi'} (S'_1\pi, \psi'_1\pi) \xRightarrow{M_2\pi'} \ldots \xRightarrow{M_n\pi'} (S'_n\pi, \psi'_n\pi).$$

We will show by induction on n that

$$(S'_0\pi, \psi_0\pi) \xRightarrow{L_1} (S'_1\pi, \psi'_1\pi) \xRightarrow{L_2} \ldots \xRightarrow{L_n} (S'_n\pi, \psi'_n\pi).$$

We assume by the induction hypothesis that

$$(S'_0\pi, \psi_0\pi) \xRightarrow{L_1} (S'_1\pi, \psi'_1\pi) \xRightarrow{L_2} \ldots \xRightarrow{L_{i-1}} (S'_{i-1}\pi, \psi'_{i-1}\pi)$$

and we show that

$$(S'_{i-1}\pi, \psi'_{i-1}\pi) \xRightarrow{L_i} (S'_i\pi, \psi'_i\pi).$$

We will show that L_i and $M_i\pi'$ satisfy the conditions of Lemma 38 which allows us to conclude. Indeed, either $L_i = M_i\pi'$ (in the case of a **test** or **out** action), or $L_i = \mathbf{in}(d_i, R''_i)$ and $M_i\pi' = \mathbf{in}(d_i, R_i\pi')$ (in the case of a **in** action). In the second case, we need to show that $(R_i\pi' = R''_i)\psi_{i'-1}$. By the definition of $\models$, we have that $T \models \mathsf{k}_{l_1\tau,\ldots,l_{i-1}\tau}(R''_i, t_i\tau)$. We have previously shown that $T \models \mathsf{k}_{l_1\tau,\ldots,l_{i-1}\tau}(R_i\pi', t_i\tau)$ and therefore $T \models \mathsf{i}_{l_1\tau,\ldots,l_{i-1}\tau}(R_i\pi', R''_i)$, or, equivalently, $(R_i\pi' = R''_i)\varphi_{i-1}$. By the hypothesis, we have that there exists $S \in P$ such that

$$(S, \emptyset) \xRightarrow{L_1} (S_1, \psi_1) \xRightarrow{L_2} \ldots \xRightarrow{L_{i-1}} (S_{i-1}, \psi_{i-1}),$$

and $(R_i\pi' = R''_i)\psi_{i-1}$. By determinacy of P it follows that $\psi_{i-1}\pi \approx_s \psi'_{i-1}\pi$ and therefore $(R_i\pi' = R''_i)\psi'_{i-1}$ as well. As the hypothesis of Lemma 38 are satisfied, we can conclude.

We have shown that $(S'_0, \emptyset) \xRightarrow{L_1,\ldots,L_n} (S'_n, \psi'_n)$, therefore obtaining a contradiction. Hence Item 1 cannot hold.

2. We assume that for all $U \in P, V, \psi_U$ such that $(U, \emptyset) \xRightarrow{L_1,\ldots,L_n} (U', \psi_U)$ we have $(R \neq_E R')\psi_U$ to obtain a contradiction. Since P is determinate, we can fix one such U.
As $(T, \emptyset) \xRightarrow{L_1,\ldots,L_n} (T_n, \varphi_n)$ and $(R =_E R')\varphi_n$, by completeness, we have that $\mathsf{i}_{L_1\varphi_n\downarrow,\ldots,L_n\varphi_n\downarrow}(R, R') \in \mathcal{H}_{\mathsf{e}}(K)$. Note, we also have that $(R \neq_E R')\psi_U$. From the fact that $\mathsf{i}_{L_1\varphi_n\downarrow,\ldots,L_n\varphi_n\downarrow}(R, R') \in \mathcal{H}_{\mathsf{e}}(K)$ and $(R \neq_E R')\psi_U$, we can show that
   - there exist recipes Q, Q' and $k \leq n$ such that $\mathsf{i}_{L_1\varphi_n\downarrow,\ldots,L_k\varphi_n\downarrow}(Q, Q') \in \mathcal{H}(K)$ but $(Q \neq_E Q')\psi_U$.

   We fix recipes Q, Q' such that they satisfy the above condition and such that the sum of the sizes of the recipes Q and Q' is the smallest one.
   As $\mathsf{r}_{L_1\varphi_n\downarrow,\ldots,L_n\varphi_n\downarrow} \in \mathcal{H}(K)$, we have by Lemma 15 that $\mathsf{ri}_{L_1\varphi\downarrow,\ldots,L_n\varphi\downarrow}(Q, Q') \in \mathcal{H}(K)$. Therefore there exists a statement

$$f = \left(\mathsf{ri}_{l_1,\ldots,l_n}(P, P') \Leftarrow \{\mathsf{k}_{w_i}(X_i, x_i)\}_{i\in\{1,\ldots,m\}}\right) \in K$$

   and a substitution τ grounding for f such that $\mathsf{k}_{w_i\tau}(X_i\tau, x_i\tau) \in \mathcal{H}(K)$ (for all $1 \leq i \leq m$), $l_1\tau, \ldots, l_n\tau = L_1\varphi_n\downarrow, \ldots, L_n\varphi_n\downarrow$, $P\tau = Q$ and $P'\tau = Q'$. We make the following observations:
   (a) Note first that neither P nor P' are recipe variables. This can be shown by proving that none of the symbolic recipes in a head of a statement in K can be a recipe variable. The latter can be shown by a straightforward induction the number of steps used in the saturation procedure.

(b) This implies that if recipe variables X_i and X_j for $1 \leq i, j \leq m$ belong to the union of subterms of P and of P', then the sum of sizes of $X_i\tau$ and $X_j\tau$ is less than the sum of the sizes of $P\tau = Q$ and $P'\tau = Q'$.

(c) If $x_i = x_j$ for $1 \leq i, j \leq m$ then, by Lemma 16, we have that $\mathsf{i}_{i_1\varphi_n\downarrow,\ldots,L_k\varphi_n\downarrow}(X_i\tau, X_j\tau) \in \mathcal{H}(K)$ for some $k \leq n$ and hence $(X_i\tau =_E X_j\tau)\varphi_k$. If $k < n$ we must have that $(X_i\tau =_E X_j\tau)\psi_U$ as we would otherwise contradict minimality of n. If $k = n$, by our construction of Q and Q' and observation (b) above, if recipe variables X_i and X_j belong to the union of subterms of P and of P', then we must have $(X_i\tau =_E X_j\tau)\psi_U$. Otherwise we would contradict minimality of Q, Q'.

(d) For each $1 \leq i \leq m$, consider the set $comp_i = \{\, j \mid x_j = x_i \text{ and } X_j \text{ is a subterm of } P \text{ or } P' \,\}$. Consider the map $least : \{\, 1 \leq i \leq m \,\} \to \{\, 1 \leq i \leq m \,\}$ defined as follows. $least(i) = x_r$ where r is the least element of the set $comp_i$ if $comp_i$ is nonempty and is the least element of the set $\{\, j \mid x_j = x_i \,\}$ otherwise.

Let τ_0 be the substitution such that for $\tau_0(X_i) = \tau(X_{least(i)})$ for $1 \leq i \leq m$, and $\tau_0(x) = \tau(x)$ for all message variables. By observation (c) above, we have that

- $(P\tau =_E P\tau_0)\psi_U$ and hence $(Q =_E P\tau_0)\psi_U$.
- $(P'\tau =_E P'\tau_0)\psi_U$ and hence $(Q' =_E P'\tau_0)\psi_U$.
- $(P\tau_0 =_E P'\tau_0)\varphi_n$.

We proceed exactly as for Item 1 and show that there exists $S'_0 \in P$ such that

$$(S'_0, \psi'_0) = (S'_0\pi, \psi_0\pi) \stackrel{L_1}{\Longrightarrow} (S'_1\pi, \psi'_1\pi) \stackrel{L_2}{\Longrightarrow} \ldots \stackrel{L_n}{\Longrightarrow} (S'_n\pi, \psi'_n\pi)$$

where $\pi(c) = \tau(x)$ if $\sigma(c) = x$. Let $\pi'(c_i) = \tau_0(X_i)$ if $\sigma(c_i) = x_i$.

Furthermore, as $\textsc{Identity}(K, P)$ holds it follows that $(P\omega = P'\omega)\psi'_n$, where $\omega = \{X_i \mapsto x_i\sigma\}$. As the equational theory is stable by substitution of terms for names, we have that $(P\omega\pi' =_E P'\omega\pi')\psi'_n\pi$. But $P\omega\pi' =_E P\tau_0$ and $P'\omega\pi' = P'\tau_0$, which means that $(P\tau_0 =_E P'\tau_0)\psi'_n\pi$. Now, thanks to determinacy, we have that $(P\tau_0 =_E P'\tau_0)\psi_U$. By observation (d) above, we get $(Q =_E Q')\psi_U$, thus obtaining a contradiction.

As both cases yield a contradiction, it follows that if $T \not\sqsubseteq P$ then $\textsc{Reachability}(K, P)$ or $\textsc{Identity}(K, P)$ fail.