Decidability of Bi-trace Consistency

Jeremy E. Dawson and Alwen Tiu

Logic and Computation Group
College of Engineering and Computer Science
Australian National University
Canberra ACT 0200, Australia
http://users.rsise.anu.edu.au/~jeremy/
http://users.rsise.anu.edu.au/~tiu/

Abstract. YET TO CHANGE FROM TPHOLS 2009 PAPER We consider a formalisation of a notion of observer (or intruder) theories, commonly used in symbolic analysis of security protocols. An observer theory describes the knowledge and capabilities of an observer, and can be given a formal account using deductive systems, such as those used in various "environment-sensitive" bisimulation for process calculi, e.g., the spi-calculus. Two notions are critical to the correctness of such formalisations and the effectiveness of symbolic techniques based on them: decidability of message deduction by the observer and consistency of a given observer theory. We consider a formalisation, in Isabelle/HOL, of both notions based on an encoding of observer theories as pairs of symbolic traces. This encoding has recently been used in a theory of open bisimulation for the spi-calculus. We machine-checked some important properties, including decidability of observer deduction and consistency, and some key steps which are crucial to the automation of open bisimulation checking for the spi-calculus, and highlight some novelty in our Isabelle/HOL formalisations of decidability proofs.

1 Introduction

YET TO CHANGE FROM TPHOLS 2009 PAPER In most symbolic techniques for reasoning about security protocols, certain assumptions are often made concerning the capability of an intruder that tries to compromise the protocols. A well-known model of intruder is the so-called Dolev-Yao model [10], which assumes perfect crytography. We consider here a formal account of Dolev-Yao intruder model, formalised as some sort of deduction system. This deductive formulation is used in formalisations of various "environment-sensitive" bisimulations (see e.g., [6]) for process calculi designed for modeling security protocols, such as the spi-calculus [3]. An environment-sensitive bisimulation is a bisimulation relation which is indexed by a structure representing the intruder's knowledge, which we call an *observer theory.*

An important line of work related to the spi-calculus, or process calculi in general, is that of automating bisimulation checking. The transition semantics of these calculi often involve processes with infinite branching (e.g., transitions for input-prefixed processes in the π-calculus [12]), and therefore a *symbolic method* is needed to deal with potential infinite branches lazily. The resulting bisimulation, called *symbolic bisimulation*, has been developed for the spi-calculus [7]. The work reported in [7] is, however, only aimed at finding an effective approximation of environment-sensitive bisimulation, and there has been no metatheory developed for this symbolic bisimulation so far. A recent work by the second author [14] attempts just that: to establish a symbolic bisimulation that has good metatheory, in particular, a symbolic bisimulation which is also a congruence. The latter is also called *open bisimulation* [13]. One important part of the formulation of open bisimulation for the spi-calculus is a symbolic representation of observer theories, which needs to satisfy certain consistency properties, in addition to closure under a certain notion of "respectful substitutions", as typical in formulations of open bisimulation.

A large part of the work on open bisimulation in [14] deals with establishing properties of observer theories and their symbolic counterparts. This paper is essentially about formally verifying the results of [14] concerning properties of (symbolic) observer theories in Isabelle/HOL. In particular, it is concerned with proving decidability of the deduction system for observer theory, correctness of a finite characterisation of consistency of observer theories (hence decidability of consistency of observer theories), and preservation of consistency under respectful substitutions. Additionally, we also verify some key steps towards a decision procedure for checking consistency of symbolic observer theories, which is needed in automation of open bisimulation. A substantial formalisation work described here concerns decidability proofs. Such proofs are difficult to formalise in Isabelle/HOL, as noted in [17], due to the fact that Isabelle/HOL is based on classical logic. We essentially follow [17] in that decidability in this case can be inferred straightforwardly by inspection on the way we define total functions corresponding to the decidability problems in question. That is, we show, by meta-level inspection, that the definitions of the functions do not introduce any infinite aspect and are therefore are finitely computable.

There is a recent work [11] in formalising the spi-calculus and a notion of environment-sensitive bisimulation (called the *hedged bisimulation* [8]) in Isabelle/HOL. However, this notion of bisimulation is a "concrete" bisimulation (as opposed to symbolic), which means that the structure of observer theories is less involved and much easier to deal with compared to its symbolic counterpart. Our work on observer theories is mostly orthogonal to their work, and it can eventually be integrated into their formalisation to provide a completely formalised open bisimulation for the spi-calculus. Such an integration may not be too difficult, given that much of their work, e.g., formalisation of the operational semantics of the spi-calculus, can be reused without modifications.

NEW TO THIS PAPER

In [**?**] we discussed respectful substitutions for bi-traces, defined as follows. A *bi-trace* is a *list* of message pairs marked with i (indicating input) or o (output), with the restriction that any variable appears first in an input pair. A substitution pair $\vec{\theta} = (\theta_1, \theta_2)$ replaces free names $x \in \mathcal{N}$ by messages, using substitutions $\theta_1(\theta_2)$ for the first (second) component of each pair. For a bi-trace h, $\vec{\theta}$ *respects* h, or is h-respectful [15, Definition 34], if for every free name x in an input pair $(M, N)^i$, $\{h'\}\vec{\theta} \vdash x\theta_1 \leftrightarrow x\theta_2$, where h' is the part of h preceding $(M, N)^i$. This is expressed in Isabelle by $h \in$ `bt_resp` $\vec{\theta}$.

Since a bitrace is a sequence of pairs of messages (marked as input or output), we projected this notion to a similar trace of single messages (which we will henceforth call a *trace*). This gave the notion of a "respectful substitution" for a trace. We reproduce [**?**, Theorem 20] here.

Theorem 1 (`subst_exists, subst_unique`)**.** *Given a consistent bi-trace h whose projections to a single message trace are $s_1 = \pi_1(h)$ and $s_2 = \pi_2(h)$, and a substitution θ_1 which respects s_1, there exists θ_2 such that $\vec{\theta} = (\theta_1, \theta_2)$ respects h, and θ_2 is "unique" in the sense that any two such θ_2 act the same on names in s_2*

We also reproduce (in part) the definition of a *consistent* bi-trace.

Definition 2. *[15, Definition 35] The set of* consistent bi-traces *are defined inductively (on the length of bi-traces) as follows:*

(a) The empty bi-trace is consistent.
(b) If h is a consistent bi-trace then $h.(M, N)^i$ is also a consistent bi-trace, provided that $h \vdash M \leftrightarrow N$.
(c) If h is a consistent bi-trace, then $h' = h.(M, N)^o$ is a consistent bi-trace, provided that for every h-respectful substitution pair $\vec{\theta}$, $\{h'\vec{\theta}\}$ is a consistent theory.

Thus, to determine whether $h' = h.(M, N)^o$ is a consistent bi-trace, we need to consider *all* h-respectful substitution pairs $\vec{\theta}$. By Theorem 1 it is enough to consider all respectful substitutions of a trace s_1; for each such substitution we can get the respectful substitution of the bi-trace.

Now we could produce a finite set of substitutions Θ_1 such that each respectful substitution θ_1 of $s_1 = \pi_1(h)$ is of the form $\theta'_1 \circ \rho_1$ for some $\theta'_1 \in \Theta_1$. Then θ'_1 gives rise to a "unique" h-respectful substitution θ. We would then need to check whether $\{h(\theta' \circ \rho)\}$ [1] is consistent for all ρ.

Now we can check whether $\{h\theta'\}$ is a consistent theory, using the finitely checkable criterion for a consistent theory, given by Tiu [15, Proposition 20] see also [**?**, Proposition 11, Theorem 13]. But this does not give us consistency of $\{h(\theta' \circ \rho)\}$ for all ρ.

[1] Here, $(\theta' \circ \rho)$ means θ' followed by ρ.

Instead, from the criteria for a consistent theory we can get a finite set Σ of most general substitutions σ which may make $\{h\theta\}$ inconsistent. That is, if we can substitute in $\{h\}$ to get an inconsistent theory, it is of the form $\{h\}\sigma\theta$ for some θ, and some $\sigma \in \Sigma$. Then, for each such σ we can find the finite set of most general substitutions which can be composed with σ_1 or σ_2 to get a respectful substitution, as described above. Then we can test each of these $\sigma_i \circ \theta$ to see if they do in fact lead to an inconsistent theory. In this case, if any less general substitution $\sigma_i \circ \theta \circ \rho$ gives an inconsistent theory then so does $\sigma_i \circ \theta$.

Note that in our results and proofs we either consider only substitutions which do not touch variables which do not appear, or, as in Theorem 1, regards substitutions as the same when they differ only on variables which are not present. END OF NEW STUFF

We assume that the reader is familar with Isabelle proof assistant, its object logic HOL and logical frameworks in general. In the remainder of this section we briefly describe relevant Isabelle notations used throughout the paper. In Section **??**, we give an overview of observer theories and an intuition behind them. We also give a brief description of two problems that will be the focus of subsequent sections, namely, those that concern decidability of consistency checking for (symbolic) observer theories. In Section **??** we consider formalisation of a notion of theory reduction and decidability of consistency checking for observer theories. In Section **??** we discuss a symbolic representation of observer theories using pairs of symbolic traces [5], called *bi-traces*, their consistency requirements and a notion of *respectful substitutions*. We prove a key lemma which relates a symbolic technique for trace refinement [5] to bi-traces, and discuss how this may lead to a decision procedure for testing bi-trace consistency. Section 4 concludes.

Isabelle notation The Isabelle codes for the results of this paper can be found at `http://users.rsise.anu.edu.au/~jeremy/isabelle/2005/spi/`. In the statement of lemma or theorem, a name given in `typewriter` font indicates the name of the relevant theorem in our Isabelle development. We show selected theorems and definitions in the text, and more in the Appendix. A version of the paper, including the Appendix, is in `http://users.rsise.anu.edu.au/~jeremy/pubs/spi/fotesb/`. So now we indicate some key points of the Isabelle notation.

- A name preceded by `?` indicates a variable: other names are entities which have been defined as part of the theory
- Conclusion β depending on assumptions α_i is `[|` α_1`;`α_2`;` $\ldots$ `;`α_n `|] ==>` β
- $\forall$, $\exists$ are written as `ALL`, `EX`
- $\subseteq$, $\supseteq$, $\in$ are written as `<=`, `>=`, `:`

2 A finite set of most general respectful substitutions

A common requirement of our substitutions will be (as it would ordinarily be expressed) that the variables in the domain (that is, those variables which are substituted) do not appear in any of the messages in the range (that is, the messages which are subsituted for the variables in the domain). In fact, since we

$$\frac{x \in \mathcal{N}}{\Sigma \vdash x}\ (var) \qquad \frac{}{\Sigma, M \vdash M}\ (id) \qquad \frac{\Sigma \vdash M \quad \Sigma \vdash N}{\Sigma \vdash \langle M, N\rangle}\ (pr)$$

$$\frac{\Sigma \vdash M \quad \Sigma \vdash N}{\Sigma \vdash \{M\}_N}\ (er) \qquad \frac{\Sigma, M, N \vdash R}{\Sigma, \langle M, N\rangle \vdash R}\ (pl) \qquad \frac{\Sigma \vdash N \quad \Sigma, M, N \vdash R}{\Sigma, \{M\}_N \vdash R}\ (el)$$

Fig. 1. A proof system for message synthesis

formalise a substitution in Isabelle as a total function, we formalise the domain of a substitution as those variables which are not simply mapped to themselves. This requirement is expressed by the predicate `dr_disj`.

Another requirement is that no new variables appear in the domain or in any of the range messages. We assume a definition of "most general unifier" which produces substitutions satisfying this criterion.

Then we call a substitution *null* if it is the identity function on messages, equivalently, if its domain is empty.

Given the form of the definition of a respectful substitution, we can express it as a number of constraints of the form $\Sigma \vdash M$, meaning that to be respectful, a substitution θ must satisfy $\Sigma\theta \vdash M\theta$.

Figure 1 shows the proof rules for deriving $\Sigma \vdash M$. Note that we abbreviate $\Sigma \cup \{M, N\}$ as Σ, M, N.

2.1 Satisfying a single constraint for a respectful substitution

We look at how we get the ways of satisfying a given single constraint of the form $\Sigma \vdash M$.

Given a constraint $\Sigma \vdash M$ we define possible ways of "reducing it" to (usually simpler) constraint(s) which have a suitable property. In general a constraint $C = \Sigma \vdash M$ can be reduced in several ways, but each reduction produces one "main" constraint $C' = \Sigma' \vdash M$ and a set of "right" constraints $\mathcal{C}'_R$. Repeated reductions act on the "main" constraint and accumulate an ever-increasing set of "right" constraints. Thus we could say $C \to (C', \mathcal{C}'_R)$ (and, for convenience, we may write this), but since all the reductions leave M untouched in the main constraint, our formalisation uses a reduction $\Sigma \to (\Sigma', \mathcal{C}'_R)$. Then, to express the accumulation of "right" constraints, we define $\longrightarrow$. The second formulation shows the notation we may sometimes use, where $C = \Sigma \vdash M$ and $C' = \Sigma' \vdash M$, but the first follows our Isabelle formalisation.

$$\frac{\Sigma \to (\Sigma', \mathcal{C}'_R)}{(\Sigma, \mathcal{C}_R) \longrightarrow (\Sigma', \mathcal{C}_R \cup \mathcal{C}'_R)} \qquad \frac{C \to (C', \mathcal{C}'_R)}{(C, \mathcal{C}_R) \longrightarrow (C', \mathcal{C}_R \cup \mathcal{C}'_R)}$$

This idea appears in a different form in Tiu[**?**], and the reduction rules also have an obvious resemblance to the observer theory reduction of [**?**, ??].

$$\Sigma, \langle M_a, M_b\rangle \to ((\Sigma, M_a, M_b), \{\})$$
$$\Sigma, \{M_p\}_{M_k} \to ((\Sigma, M_p, M_k), \{\Sigma, \{M_p\}_{M_k} \vdash M_k\})$$

We assume that Γ does not contain $\langle M_a, M_b\rangle$and $\{M_p\}_{M_k}$respectively (otherwise reduction would not terminate).

In our Isabelle formalisation, $\rightarrow$ is `orkvs` and $\longrightarrow$ is `orkvss`.

For such a constraint set $(C, \mathcal{C}_R)$, we say it is satisfied if the constraint C holds, and the constraints in $\mathcal{C}_R$ can be derived only using *right* derivations. A right derivation is essentially one in which the right-hand side can be built up from messages on the left-hand side using the pairing and encryption operators. More formally, it is a derivation which uses only the rules (var), (id), (pr) and (er) of Figure 1. We write $\Sigma \vdash_R M$ to indicate right derivability. Now it is decidable whether or not a constraint $\Sigma \vdash_R M$ is derivable (see [**?**, ??]). This follows from the fact that in a backwards proof using only these rules, the right-hand side gets smaller at each rule application.

Proposition 3. *(a)* (`subst_orkvss_if`) *If* $(C, \mathcal{C}_R) \longrightarrow (C', \mathcal{C}'_R)$*, then, for any substitution* θ*, if* $(C', \mathcal{C}'_R)$ *are satisfiable after substitution by* θ*, then so are* $(C, \mathcal{C}_R)$*.*

(b) (`idv_orkvs`) *If* C *is derivable but not right derivable, then there exists* $(C', \mathcal{C}'_R)$*, which are derivable, such that* $C \rightarrow (C', \mathcal{C}'_R)$*.*

(c) $\longrightarrow$ *is well-founded (as the main constraint gets smaller at each step).*

(d) (`orkvss_iff`) C *is derivable if and only if there exists* $(C', \mathcal{C}'_R)$*, where* C' *as well as* $\mathcal{C}'_R$ *are right derivable, such that* $C \longrightarrow^* (C', \mathcal{C}'_R)$*.*

The reduction relation is also finitely branching, as there is at most one reduction for each member of Σ (recall that we assume Σ is finite: it is a set of messages which have actually been sent). Therefore this gives a decision procedure for $C = \Sigma \vdash M$: obtain all $(C', \mathcal{C}'_R)$, and test whether, for any of them, C' and $\mathcal{C}'_R$ are all right derivable.

Note that many of our Isabelle results are for observer theories (sets of *pairs* of messages) rather than for message sets, and for pairs of substitution functions; we use such results for this work by putting a pair of identical messages in place of a single message.

```
subst_orkvss_if : "[| (?Tvs, ?Svs) : orkvss;
    satvss (subst_vs ?fg ?Tvs) ?MN |] ==>
  satvss (subst_vs ?fg ?Svs) ?MN"),

idv_orkvs : "[| (?G, ?MN) : indist; ~ is_der_virt (?G, ?MN) |] ==>
  EX G' Cs. ((G', Cs), ?G) : orkvs & satvss (G', Cs) ?MN"

orkvss_iff : "finite ?S -->
  satvss (?S, ?Us) ?MN = (EX S' Us'. is_der_virt (S', ?MN) &
    ((S', Us'), ?S, ?Us) : orkvss^* & satvss (S', Us') ?MN)"
```

In fact we defined a function `idv_ss_sets`, where `idv_ss_sets` M is a set of sets of messages. We defined `idv_ss_sets` in a way makes it clear that it is finitely computable (see the discussion on this topic in [**?**, ??]).

Lemma 4 (idv_ss_sets)**.** *$\Sigma \vdash_R M$ if and only if Σ contains a member S of* `idv_ss_sets` *M.*

```
idv_ss_sets : "idv_ss (?S, ?M) = (EX sp:idv_ss_sets ?M. sp <= ?S)"
```

At this point we have a decision procedure for $C = \Sigma \vdash M$:

(a) Find all $(C', \mathcal{C}'_R)$ such that $C \longrightarrow^* (C', \mathcal{C}'_R)$, and test whether *any* of them satisfies the following
(b) Test whether *all* constraints $\Sigma' \vdash M'$ in $\{C'\} \cup \mathcal{C}'_R$ satisfy the following
(c) Calculate `idv_ss_sets` M' (which is a set of sets of messages) and see if among them there is *any* one, S, which satisfies the following
(d) Find all functions f from S into Σ', and test whether *any* of them satisfies the following
(e) Test whether for *all* $s \in S$, $s = f(s)$

Now the last two items here are just a long-winded way of saying $S \subseteq \Sigma'$. But here our task is to find substitutions θ such that $\Sigma\theta \vdash M\theta$ is satisfied. The last step of the decision procedure could be replaced by: let θ be the most general unifier of $\{(s, f(s)) \mid s \in S\}$. This leaves us with the problem that the earlier steps of the decision procedure given were performed with respect to the constraint $\Sigma \vdash M$, not $\Sigma\theta \vdash M\theta$. We now have to address this issue.

Firstly, we look at the effect of calculating `idv_ss_sets` M *before* applying θ to M, when we are interested in $\Sigma\theta \vdash M\theta$. (We will use this lemma with $\Pi = \Sigma\theta$.) we get two results which are not converses — note that the conclusion of (b) uses $\vdash$, not $\vdash_R$ — but which will be sufficient for deriving an equivalence later, in Theorem 9.

Lemma 5. *(a) (idv_sets_sub_idv) If $\Pi \vdash_R M\theta$ then there exists $S \in$ idv_ss_sets M such that $S\theta \subseteq \Pi$*
(b) (idv_sets_sub_indist) If $S \in$ idv_ss_sets M and Σ satisfies the respectfulness condition, and the variables in M are all in Σ and $S\theta \subseteq \Sigma\theta$, then $\Sigma\theta \vdash M\theta$

```
idv_sets_sub_idv : "is_der_virt (?G, subst_pr ?fg (?a, ?a)) ==>
  EX sp:idv_sets (?a, ?a). subst_pr ?fg ‘ sp <= ?G"

idv_sets_sub_indist : "[| ?sp : idv_sets ?MN; thy_strl_resp ?fg ?ns ?G;
    fn_oth (insert ?MN ?G) <= Union (set ?ns);
    subst_pr ?fg ‘ ?sp <= subst_pr ?fg ‘ ?G |] ==>
  (subst_pr ?fg ‘ ?G, subst_pr ?fg ?MN) : indist"
```

Note that Lemma 5(a), unlike Lemma 4, is not an equivalence, but gives necessary conditions for a constraint to be satisfied. That is, it gives us *some*, if not *all*, pairs of messages which need to be identified by a proposed substitution. For the converse result, Lemma 5(b), we need to consider that there may be variables in M which are affected by θ: here the respectfulness condition ensures that in such a case $\Sigma\theta \vdash x\theta$. This is the key to the outcome we arrive at below, which is that we need to obtain and compose several substitutions until we get a respectful one. It is also the reason why the results we describe next are formulated in terms of two substitutions: one already determined and one yet to be determined.

Secondly we look at the effect of calculating $\longrightarrow^*$ *before* applying θ to M. The problem here is that we may find a variable x among the messages in Σ, where $x\theta$ is a message formed by the pairing or the encryption operators, providing a reduction for $\Sigma\theta$ where there is no corresponding reduction for Σ.

Here we rely on the fact that we are looking for a respectful substitution θ, so, under this assumption we get $\Sigma\theta \vdash x\theta$.

We define an inference system for deriving $\Sigma\theta \vdash M\theta$ without ever using any $x\theta$ as the principal formula for any of the left rules. It also gives a *normal* derivation (no left rule above a right rule or above the left premise of (el)) without redundant steps. This inference system is shown in Figure 2. Clearly $\Sigma \vdash_\theta M$ implies $\Sigma\theta \vdash M$.

$$\frac{(\Sigma \setminus \mathcal{N})\ \theta \vdash_R M}{\Sigma \vdash_\theta M}\ virt$$

$$\frac{\Sigma, M, N \vdash_\theta R}{\Sigma, \langle M, N\rangle \vdash_\theta R}\ pl \quad \text{if } \langle M, N\rangle \notin \Sigma$$

$$\frac{(\Sigma \setminus \mathcal{N})\ \theta \vdash_R N\theta \quad \Sigma, M, N \vdash_\theta R}{\Sigma, \{M\}_N \vdash_\theta R}\ el \quad \text{if } \{M\}_N \notin \Sigma$$

Fig. 2. A proof system for substituted message synthesis

Now the relation $\rightarrow$ (on the constraints before substitution) is naturally related to this proof system, similarly to Proposition 3(d), although Lemma 6 is not an equivalence. Let $\mathcal{N}$ be the set of all message variables.

Lemma 6 (`indist_nsv_ex_orkvss`). *If $\Sigma \vdash_\theta M\theta$ and C is $\Sigma \vdash M$ then there exists $(C', \mathcal{C}'_R)$ such that $C \rightarrow^* (C', \mathcal{C}'_R)$ and, for all the constraints $\Sigma' \vdash M'$ in $(C', \mathcal{C}'_R)$, we have $(\Sigma' \setminus \mathcal{N})\theta \vdash_R M'\theta$.*

Now since respectfulness of θ essentially means that the (remainder of) $\Sigma\theta$ entails $x\theta$, we can get that omitting $\mathcal{N}$ from Σ before applying θ does not matter. Or, precisely:

Lemma 7 (`indist_ne_nsv_excl`). *If $\Sigma\theta \vdash M$ and θ satisfies the appropriate respectfulness condition for Σ, then $\Sigma \vdash_\theta M$.*

The "appropriate respectfulness condition" uses `thy_strl_resp`, and is described in detail in [**?**, Appendix A.19]. Briefly, from the notion of a respectful substitution of a bi-trace (or a trace) we get a "respectful" substitution of an observer theory (or message set) with respect to an ordered list of sets of variables. Then, a respectful substitution of a trace becomes a respectful substitution of the underlying message set, and also of the sets that we see after applying the reduction $\longrightarrow$.

We tried to improve Lemma 6 to a stronger result of the form "Given Σ and θ, there exists $(C', \mathcal{C}'_R)$ such that for all M, if $C = \Sigma \vdash_\theta M$ then ...". Taking advantage of this would make a decision procedure more efficient. We did not succeed in this. However we did succeed in improving in this way the combination of Lemmas 6 and 7. Thus we know that there is one of the finite set of possibilities for $(C', \mathcal{C}'_R)$ which applies for all M.

Lemma 8. *If θ satisfies the appropriate respectfulness condition for Σ, then, given Σ, there exist Σ' and $\mathcal{C}'_R$ such that, for any M, if C is $\Sigma \vdash M$ and C' is $\Sigma' \vdash M$, then $C \rightarrow^* (C', \mathcal{C}'_R)$ and, for all the constraints $\Sigma'' \vdash M'$ in $\mathcal{C}'_R$, we have $(\Sigma'' \setminus \mathcal{N})\theta \vdash_R M'\theta$. Also, if $\Sigma \vdash M$ is derivable, then we have $(\Sigma' \setminus \mathcal{N})\theta \vdash_R M\theta$.*

```
indist_nsv_ex_orkvss : "(?G, ?MN) : indist_nsv_excl ?fg ==>
  EX H Vs. ((H, Vs), ?G, {}) : orkvss^* &
  is_der_virt (subst_pr ?fg ` (H - Ne), ?MN) &
  (ALL (K, X):Vs. is_der_virt (subst_seq ?fg (K - Ne, X)))"

indist_ne_nsv_excl : "[| finite ?G; (?H, ?MN) : indist_norm_excl;
  ?H = subst_pr ?fg ` ?G; ?G <= range pair; thy_strl_resp ?fg ?ns ?G;
  fn_oth ?G <= Union (set ?ns) |] ==> (?G, ?MN) : indist_nsv_excl ?fg"

indist_resp_ex_orkvss : "[| finite ?G; ?G <= range pair;
    thy_strl_resp ?fg ?ns ?G; fn_oth ?G <= Union (set ?ns) |] ==>
   EX H Vs.  ((H, Vs), ?G, ?Us) : orkvss^* &
    ((ALL (K, X):?Us. is_der_virt (subst_seq ?fg (K - Ne, X))) -->
      (ALL (K, X):Vs. is_der_virt (subst_seq ?fg (K - Ne, X)))) &
    (ALL MN. (subst_pr ?fg ` ?G, MN) : indist -->
      is_der_virt (subst_pr ?fg ` (H - Ne), MN))"
```

We can now combine Lemmas 7, 6 and 5. To find a substitution θ satisfying $\Sigma \vdash M$ (one of the respectfulness constraints!), where θ is respectful (so we can use Lemma 7), we select one $(C', \mathcal{C}'_R)$ as in Lemma 6.

Let us index the constraints in the selected $(C', \mathcal{C}'_R)$ by $i \in I$. Since Lemma (a) gives a necessary condition for a substitution θ to make each of the constraints $\Sigma_i \vdash_R M_i$ in $(C', \mathcal{C}'_R)$ to be satisfied. We attempt to satisfy all those conditions. Each condition is of the form: there exists $S \in$ `idv_ss_sets` M_i such that $S\theta \subseteq \Sigma_i\theta$. Since $S\theta \subseteq \Sigma_i\theta$ is equivalent to: there exists a function f_i from S into Σ_i such that θ unifies the set of pairs $\mathcal{P}_i = \{(s, f_i(s)) \mid s \in S\}$. So for each way of choosing a function f_i for each $i \in I$, we take the union $\mathcal{P} = \bigcup_{i \in I} \mathcal{P}_i$. (We would then want θ to unify all the pairs in $\mathcal{P}$.)

The function `sat_unifs` does this. The theorem `sat_unifs_iff` expresses its relevant properties, but its definition is considerably more complicated. Thus `sat_unifs` ... obtains a set of possible $\mathcal{P}$, one of which should be unified by θ. Then `unif_ch1` takes the union of these over all possible choices of $(C', \mathcal{C}'_R)$.

```
sat_unifs_iff : "(EX unif':sat_unifs ?F ?G ?smcs. unif' <= ?unif) =
  (ALL ssn:?smcs. EX A:?F ssn. EX M<=?unif. fst ‘ M = A & snd ‘ M <= ?G ssn)"

unif_ch1_def : "unif_ch1 ?f ?G ?M ==
  UN (H, Vs):{x. (x, ?G, {}) : orkvss^*}.
  sat_unifs (idv_sets o subst_pr ?f o snd)
  (%(K, X). subst_pr ?f ‘ (K - Ne)) (insert (H, ?M, ?M) Vs)"
```

Theorem 9 (smder_unif_ch1). *If $\gamma \circ \delta$ is a respectful substitution (for the trace whose message set is Σ) and the variables in M are all in Σ then $\Sigma\gamma\delta \vdash M\gamma\delta$ is derivable iff there exists a set $\mathcal{P}$ of pairs in `unif_ch1` γ Σ M such that for each $(M', M'') \in \mathcal{P}$, $M'\delta = M''\delta$.*

```
smder_unif_ch1 : "[| validst ?st; ?st : sm_resp (comp_subst ?g ?f);
         fn_msg ?M <= (UN mm:set ?st. fn_msg (rem_io mm)) |] ==>
   ((subst_msg (comp_subst ?g ?f) ‘ rem_io ‘ set ?st,
     subst_msg (comp_subst ?g ?f) ?M) : derrec smpsc {}) =
   (EX unif:unif_ch1_ss ?f (rem_io ‘ set ?st) ?M.
      ALL (a, b):unif. subst_msg ?g a = subst_msg ?g b)"
```

Thus, given Σ and M, if we are looking for a respectful substitution θ such that $\Sigma\theta \vdash M\theta$, the first step is to apply Theorem 9 with $\gamma = id$, that is, calculate `unif_ch1` id Σ M, and take the most general unifier of each resulting set of pairs. This gives a finite set of substitutions γ_i, $i \in I$, with the property that (although they need not be respectful), any respectful substitution θ such that $\Sigma\theta \vdash M\theta$ is a specialisation of some γ_i. We now therefore need to look at specialising each such γ_i to a respectful substitution.

2.2 Moving towards a respectful substitution: the function umgrs1

We define a function `umgrs1` which proceeds towards finding a respectful substitution. For a trace s and a substitution γ, `umgrs1` s γ is a finite set of substitutions δ which could be composed with γ to be the next step towards a respectful substitution.

Some of the Isabelle code relates to pairs of messages (as found in an observer theory) rather than single messages. In part, we reuse this code, simply applying it to identical pairs of messages in place of single messages. Thus `unif_ch` applies to pairs of messages just as `unif_ch_st` applies to single messages. This accounts for the definition of `unif_ch_st` in terms of `unif_ch`, where $(a, b) \in$ `slice_unif` r iff $((a, a), (b, b)) \in r$, and `pair` $x = (x, x)$.

The function `bt_constrs` gives the list of constraints for a substitution to be h-respectful for a bi-trace h. Each constraint is, in general, of the form $\Gamma \vdash M \leftrightarrow N$, but here we will have $M = N$ and Γ consists of identical pairs.

We omit the definition of `UN_choices`, but show the theorem, `UN_choices_iff`, which characterises it.

For each constraint, `unif_ch1` gives a set of sets of pairs: for the constraint to be satisfied by a substitution, one of the sets of pairs must be unified by the substitution. With several constraints, each one may be satisfied by unifying one of several sets of pairs: `UN_choices` takes all possible ways of making this choice for each constraint, and, for each such way, takes the union of the sets of pairs which are to be unified.

```
unif_ch_st_def : "unif_ch_st ?g ?st ==
  slice_unif ` unif_ch ?g (map (map_io pair) ?st)"

unif_ch_def : "unif_ch ?g ?bt ==
  UN_choices (bt_constrs ?bt) (%(ss, m, n). unif_ch1 (?g, ?g) ss n)"

UN_choices_iff : "(EX U':UN_choices ?A ?B. U' <= ?U) =
              (ALL a:?A. EX b:?B a. b <= ?U)"
```

2.3 Using `umgrs1` in a complete search for respectful substitutions

`unif_ch_st` γ s gives a "necessary" set of sets of pairs of messages; that is, to get a s-respectful substitution ζ which is a further substitution after γ, we must compose γ with a substitution which unifies all message pairs in one of the sets given by `unif_ch_st` γ s.

Lemma 10 (`sm_resp_unif_ch`). *If $\gamma \circ \delta$ respects s then δ unifies one of the sets of pairs in `unif_ch_st` γ s.*

```
sm_resp_unif_ch :
  "[| validst ?st; ?st : sm_resp (comp_subst ?h ?g) |] ==>
  EX unif:unif_ch_st ?g ?st.
    ALL (a, b):unif. subst_msg ?h a = subst_msg ?h b")]

umgrs1_def : "umgrs1 ?st ?g == Some -` mgu ` unif_ch_st ?g ?st"
```

Now we get the same result expressed in terms of the most general unifiers of the several sets of pairs.

Lemma 11 (`ex_umgrs1`). *If $\gamma \circ \delta$ is an s-respectful substitution, then there exists $\sigma \in$ `umgrs1` s γ and ρ such that $\delta = \sigma \circ \rho$*

```
ex_umgrs1 : "[| validst ?st; ?st : sm_resp (comp_subst ?h ?g) |] ==>
  EX k:umgrs1 ?st ?g. EX j. ?h = comp_subst j k"
```

We find that if `umgrs1` s γ contains the null substitution, we are done: we do not need that `umgrs1` s γ consists of only the null substitution.

Lemma 12 (`umgrs1_id`). *If any $\delta \in$ `umgrs1` s γ is the identity substitution, then γ is s-respectful.*

```
umgrs1_id : "[| ?d : umgrs1 ?st ?g; validst ?st;
  subst_msg ?d = id |] ==> ?st : sm_resp ?g"
```

Let us sum up to this point: given a substitution γ, every s-respectful substitution of the form $\gamma \circ \ldots$ is also of the form $\gamma \circ \delta \circ \ldots$, for some $\delta \in$ `umgrs1` $s\ \gamma$. So the procedure is to proceed towards a s-respectful substitution by calculating and composing more substitutions δ. Also, we know that we are done when `umgrs1` $s\ \gamma$ gives a null substitution.

So now we specify this accumulation of substitutions δ.

We define a result of a single step: for non-null $\delta \in$ `umgrs1` $s\ \gamma$, our next attempt towards a respectful substitution is $\gamma \circ \delta$, and then `umgrs1cp` s is the set of all such $(\gamma \circ \delta, \gamma)$. Here, although `umgrs1cp` is not recursive, it is defined in Isabelle as an inductively defined set.

```
umgrs1cp.intros : "[| ?d : umgrs1 ?st ?g; domain ?d ~= {};
    ?h = comp_subst ?d ?g |] ==> (?h, ?g) : umgrs1cp ?st"
```

Then `umgrs_resp` $s\ \gamma$ gives the result of continuing this process as far as it will go: we keep on producing $\gamma \circ \delta$ from γ, as above, until (by Lemma 12) we have a respectful substitution.

```
umgrs_resp_def : "umgrs_resp ?st ?g ==
  {h. (h, ?g) : (umgrs1cp ?st)^* & (EX k:umgrs1 ?st h. domain k = {})}"
```

To show that this procedure actually reaches all desired respectful substitutions, we need that it terminates. From this it also follows that the set of these substitutions is finite.

Lemma 13. *Assuming γ satisfies `dr_disj` γ and involves only variables in s,*

(a) (`umgrs_wfp`) *The relation `umgrs1cp` s is well-founded.*
(b) (`finite_umgrs_resp`) *`finite_umgrs_resp` $s\ \gamma$ contains only finitely many substitutions.*

Proof. At each step, the (finite) set of variables in s but not involved in the substitution gets smaller. This uses some ancillary results mentioned below. □

```
umgrs_wfp :
  "[| domain ?g Un range_names ?g <= UNION (rem_io ` set ?st) fn_msg;
             dr_disj ?g |] ==> ?g : wfp (umgrs1cp ?st)"

finite_umgrs_resp :
  "[| domain ?g Un range_names ?g <= UNION (rem_io ` set ?st) fn_msg;
     dr_disj ?g |] ==> finite (umgrs_resp ?st ?g)"
```

This gives us our desired result, showing how to construct a finite set of most general respectful substitutions which specialise a given substitution γ.

Theorem 14 (`ex_umgrs_resp`)**.**

(a) (`ex_umgrs_resp`) *If $\gamma \circ \delta$ is an s-respectful substitution, then there exists $\sigma \in$ `umgrs_resp` $s\ \gamma$ and ρ such that $\gamma \circ \delta = \sigma \circ \rho$*

(b) (umgrs_resp_id) *If $\sigma \in$ `umgrs_resp` $s\ \gamma$ then σ is an s-respectful substitution*

```
ex_umgrs_resp : "[| validst ?st; dr_disj ?g;
  domain ?g Un range_names ?g <= UNION (rem_io ` set ?st) fn_msg;
  ?st : sm_resp (comp_subst ?h ?g) |] ==>
  EX k:umgrs_resp ?st ?g. EX j. comp_subst ?h ?g = comp_subst j k"

umgrs_resp_id : "[| ?h : umgrs_resp ?st ?g; validst ?st |] ==>
  ?st : sm_resp ?h"
```

Recall that Theorem 9 gives a set of substitutions γ_i, $i \in I$, and we needed to find all respectful substitutions specialising some γ_i. Now Theorem 14 gives, for each γ_i, a set of substitutions σ_{ij}, $j \in J$, such that each respectful substitution specialising γ_i is a specialisation of some σ_{ij}, and each σ_{ij} is respectful.

Therefore, putting Theorems 9 and 14 together, we get, given Σ and M, a set σ_{ij} of respectful substitutions such that

(a) $\Sigma\sigma_{ij} \vdash M\sigma_{ij}$
(b) if θ is respectful and $\Sigma\theta \vdash M\theta$ the θ is a specialisation of some σ_{ij}

Theorem 15 **(**smder_sub_resp**).** *Let Θ be the set of respectful substitutions θ (for the trace whose message set is Σ) such that $\Sigma\theta \vdash M\theta$ is derivable. Let $\Phi = \bigcup\{$`umgrs_resp` $s\ \gamma \mid \gamma \in$ `unif_ch1` $id\ \Sigma\ M\}$. Then*

(a) $\Phi \subseteq \Theta$
(b) if $\theta \in \Theta$ then there exists $\sigma \in \Phi$ and ρ such that $\theta = \sigma \circ \rho$

We now mention some of the more minor technical results used above.

Lemma 16. *Assuming `dr_disj` γ then for each step $(\zeta, \gamma) \in$ `umgrs1cp` s, `domain` $\zeta \supset$ `domain` γ*

```
dom_umgrs1c "[| dr_disj ?g; (?h, ?g) : umgrs1cp ?st |] ==>
  domain ?g < domain ?h"
```

This is combined with results such as

Lemma 17. *for $(\zeta, \gamma) \in$ `umgrs1cp` s,*

(a) (drs_umgrs1cp) *if γ involves only variables which are in s, then so does ζ*
(b) (dr_disj_1cp_u) *if `dr_disj` γ then `dr_disj` ζ*

to show that, starting from a given γ and continuing to obtain substitutions in this way, (every branch of) this process terminates, and we obtain only finitely many substitutions. That is, it gives Lemma 13.

3 Substitutions which may give a consistent theory

We now discuss the criteria for a theory to be consistent, and show how to express these in a way that is relevant to the results of §2.

```
reduce_subs_Ne_indist_thy : "[| finite ?oth;
    thy_strl_resp ?f ?ns ?oth; fn_oth ?oth <= Union (set ?ns) |] ==>
  ((subst_oth ?f ?oth, ?MN) : indist) =
    ((subst_oth ?f (reduce ?oth - Ne), ?MN) : indist)"

subst_reduce_Ne_tc : "[| validbt ?bt; ?bt : bt_resp ?f |] ==>
  thy_cons (subst_oth ?f (oth_of ?bt)) =
    thy_cons (subst_oth ?f (reduce (oth_of ?bt) - Ne))"
```

4 Conclusions and Further Work

YET TO CHANGE FROM TPHOLS 2009 PAPER

We have modelled observer theories and bi-traces in the Isabelle theorem prover, and have confirmed, by proofs in Isabelle, the results of a considerable part of [14]. This work constitutes a significant step formalising open bisimulation for the spi-calculus in Isabelle/HOL, and ultimately towards a logical framework for proving process equivalence.

We discussed the issue of showing finite computability in Isabelle/HOL, using a mixed formal/informal argument, and building upon the discussion in Urban et al [17]. We defined a function `reduce` in Isabelle, and showed that it computes $\Gamma\Downarrow$. Isabelle required us to show that the function terminates. We asserted, with relevant discussion, that inspection shows that the definition does not introduce any infinite aspect into the computation and so asserted that therefore the function is finitely computable. Similarly, we provided a finitely computable function `is_der_virt` and proved that it tests $\Gamma \vdash M \leftrightarrow N$ for a reduced theory Γ.

We then considered bi-traces and bi-trace consistency. The problem here is that, to test bi-trace consistency, it is necessary to test whether $\Gamma\theta$ is consistent for all θ satisfying certain conditions. We proved a number of lemmas which simplify this task, and appear to lead to a finitely computable algorithm for this. In particular, our result on the unique completion of respectful substitutions that relates symbolic trace and bi-trace opens up the possibility to use symbolic trace refinement algorithm [5] to compute a notion of *bi-trace refinement*, which will be useful for bi-trace consistency checking.

Another approach to representating observer theories is to use equational theories, instead of deduction rules, e.g., as in the applied-pi calculus [1]. In this setting, the notion of consistency of a theory is replaced by the notion of *static equivalence* between knowledge of observers [1]. Baudet has shown that static equivalence between two symbolic theories is decidable [4], for a class of theories called subterm-convergent theories (which subsumes the Dolev-Yao model of intruder). It will be interesting to work out the precise correspondence between static equivalence and our notion of bi-trace consistency, as such correspondence may transfer proof techniques from one approach to the other.

Acknowledgment We thank the anonymous referees for their comments on an earlier draft. This work is supported by the Australian Research Council through the Discovery Projects funding scheme (project number DP0880549).

References

1. Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *POPL*, pages 104–115, 2001.
2. Martín Abadi and Andrew D. Gordon. A bisimulation method for cryptographic protocols. *Nord. J. Comput.*, 5(4):267–303, 1998.
3. Martin Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 99.
4. Mathieu Baudet. *Sécurité des protocoles cryptographiques: aspects logiques et calculatoires.* PhD thesis, École Normale Supérieure de Cachan, France, 2007.
5. Michele Boreale. Symbolic trace analysis of cryptographic protocols. In *Proceedings of ICALP 2001*, volume 2076 of *LNCS*, pages 667 – 681. Springer-Verlag, 2001.
6. Michele Boreale, Rocco De Nicola, and Rosario Pugliese. Proof techniques for cryptographic processes. *SIAM J. Comput.*, 31(3):947–986, 2001.
7. Johannes Borgström, Sébastien Briais, and Uwe Nestmann. Symbolic bisimulation in the spi calculus. In *CONCUR*, vol. 3170 of *LNCS*, pp 161–176. Springer, 2004.
8. Johannes Borgström and Uwe Nestmann. On bisimulations for the spi calculus. *Mathematical Structures in Computer Science*, 15(3):487–552, 2005.
9. J. E. Dawson and R. Goré. Formalising cut-admissibility for provability logic. Submitted, 2009.
10. D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 2(29), 1983.
11. Temesghen Kahsai and Marino Miculan. Implementing spi calculus using nominal techniques. In *CiE*, volume 5028 of *LNCS*, pages 294–305. Springer, 2008.
12. Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, Part II. *Information and Computation*, pages 41–77, 1992.
13. Davide Sangiorgi. A theory of bisimulation for the pi-calculus. *Acta Inf.*, 33(1):69–97, 1996.
14. Alwen Tiu. A trace based bisimulation for the spi calculus: An extended abstract. In *Proc. APLAS*, vol. 4807 of *LNCS*, pages 367–382. Springer, 2007.
15. Alwen Tiu. A trace based bisimulation for the spi calculus. Preprint, available from `http://arxiv.org/pdf/0901.2166v1`, 2009.
16. Alwen Tiu and Rajeev Goré. A proof theoretic analysis of intruder theories. In *Proceedings of RTA 2009*, 2009. To appear.
17. Christian Urban, James Cheney, and Stefan Berghofer. Mechanizing the metatheory of LF. In *LICS*, pages 45–56. IEEE Computer Society, 2008.