HAL
archives-ouvertes.fr

Concurrent Secrets

Eric Badouel, Marek Bednarczyk, Andrzej Borzyszkowski, Benoit Caillaud, Philippe Darondeau

▶ **To cite this version:**

Eric Badouel, Marek Bednarczyk, Andrzej Borzyszkowski, Benoit Caillaud, Philippe Darondeau. Concurrent Secrets. [Research Report] RR-5771, INRIA. 2005, pp.22. <inria-00070249>

HAL Id: inria-00070249
https://hal.inria.fr/inria-00070249

Submitted on 19 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Concurrent Secrets

Eric Badouel and Marek Bednarczyk and Andrzej Borzyszkowski and

Benoît Caillaud and Philippe Darondeau

N°5771

Novembre 2005

Systèmes communicants

ISSN 0249-6399 ISRN INRIA/RR--5771--FR+ENG

Concurrent Secrets

Eric Badouel * and Marek Bednarczyk † and Andrzej Borzyszkowski ‡ and Benoît Caillaud § and Philippe Darondeau ¶

Systèmes communicants
Projets S4

Rapport de recherche n ° 5771 — Novembre 2005 — 22 pages

Abstract: Given a finite state system with partial observers and for each observer, a regular set of trajectories which we call a secret, we consider the question whether the observers can ever find out that the trajectory of the system belongs to some secret. We search for a regular control on the system, enforcing the specified secrets on the observers, even though they have full knowledge of this control. We show that an optimal control always exists although it is generally not regular. We state sufficient conditions for computing a finite and optimal control of the system enforcing the concurrent secret as desired. We briefly point to applications.

Key-words: secrets, opacity, finite control

(Résumé : tsvp)

Research supported by CATALYSIS, a program within CNRS/PAN cooperation framework

* Eric.Badouel@irisa.fr
† m.bednarczyk@ipipan.gda.pl
‡ a.borzyszkowski@math.univ.gda.pl
§ Benoit.Caillaud@irisa.fr
¶ Philippe.Darondeau@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Secrets concurrents

Résumé : Etant donné un système d'états fini, un ensemble d'observateurs partiels, et pour chacun d'eux, un ensemble régulier de trajectoires du système appelé un secret, nous considérons la question suivante: les observateurs peuvent ils savoir que la trajectoire courante du système appartient a l'un de ces secrets? Nous cherchons à construire un contrôle régulier du système, capable de protéger les secrets contre des observateurs connaissant ce contrôle. Nous montrons qu'il existe toujours un contrôle optimal mais qu'il n'est pas nécessairement régulier. Nous donnons des conditions suffisantes pour le calcul de contrôleurs finis optimaux protégeant les secrets concurrents. Nous indiquons brièvement les applications.

Mots-clé : secrets, opacité, contrôle fini

1 Introduction

This work is an attempt to import supervisory control into the area of computer security. Given an automaton, or plant, and given specifications of the desired behaviour of the plant, Ramadge and Wonham's theory presented in [5] [6] yields a finite, non blocking, and maximal permissive control of the plant enforcing this behaviour in the normal case where the unobservable events are uncontrolable. Controller synthesis is a desirable complement to model checking, for it allows curing the problems that a model checker can only reveal. Supervisory control has found applications in manufacturing systems, in embedded systems, and more generally in safety critical systems. We feel it could find applications as well in computer security. We shall strive here to put forward some elements in support of this thesis.

With the above goal in mind, we have searched for a class of security problems likely to be dealt with as control problems. We model an interactive computer system and its users as a closed entity in which the users observe their own interactions with the system. The closed entity is represented with a finite automaton over some alphabet Σ. The synchronous interactions between each user i and the system are figured by the elements of a corresponding sub-alphabet $\Sigma_i \subseteq \Sigma$ (users may synchronize when their sub-alphabets intersect). Usually in supervisory control, the control objective is a predicate on the runs of the plant, specifying some combination of safety and liveness properties, and the observers act as sensors, i.e. they supply informations on the status of the plant, used by the controller to produce an adequate feedback, enabling or disabling events in the plant. Here, the game is different: the observers are not on the side of the controller but they are now opponents. As for the control objective, there is still a family of predicates S_i on the runs of the system, but the interpretation is again different: an observer i should never find out that the actual trajectory of the system belongs to secret S_i he has been assigned.

One reason why we believe the model sketched above is worth investigating is that, in the case of a single observer, it has already been introduced independently in [4] and studied further in [3]. What we call *secrets* here was called there *opaque predicates*, albeit with larger families of predicates (sets of runs) and observation functions, referring possibly to the (initial, or last...) states of the system visited in a run. It was shown in [3] that anonymity problems and non-interference problems may be reduced to opacity problems, using suitable observation functions. It was shown ibidem that model-checking a system for opacity is undecidable in the general case where an opaque predicate may refer to the visited states, or it may be any recursive predicate on sequences of transition labels. Nonetheless, techniques based on abstract interpretation were proposed in [3] for checking opacity in the framework of (unbounded) Petri nets.

In this paper, we limit ourselves to deal with finite state systems and with regular predicates defined on sequences of transition labels. We have thus all cards in hands to decide opacity, even though several pairs *(observer, secret)* are simultaneously taken into account. Now differing from [3], we want to be able to enforce opacity by supervisory control when the result of the decision is negative. In other terms, we want to disable the least possible family of trajectories such that no observer can ever find out that the system's

actual trajectory belongs to some secret. At first sight, this is a simple problem, all the more when it is assumed that all events are controllable as we do in this paper (we leave the uncontrollable events to further consideration). The problem is in fact not that simple, for the observers have full knowledge of the system, hence any supervisory control device that may be added to the system is known to them. We will nevertheless show that there always exists an optimal control for enforcing the concurrent secrets on opponents, fully aware of this control. We will also provide techniques for computing this optimal control under assumptions that seem to fit with some applications.

The rest of the paper is organized as follows. The notation and the problem are introduced in section 2. Section 3 shows that a unique optimal solution always exists, but it is generally not regular. Relying on the fixpoint characterization of the optimal control, proofs of the control enabledness of trajectories are defined as infinite trees in section 4, and specific conditions on proof trees ensuring the regularity of the optimal control are also stated there. Section 5 produces connected conditions on concurrent secrets and a full example. A more realistic application is indicated in section 6, where we also suggest some directions for further work.

2 Secrets, concurrent secrets, and the control problem

To begin with, let us fix the notation. Σ is a (non-empty) finite alphabet, Σ^* is the free monoid generated by Σ, and $Rat(\Sigma^*)$ is the family of the rational subsets of Σ^* i.e. the family of the regular languages over Σ. Let uv denote the concatenation product of the words u and v, thus u is a prefix of uv and the empty word ε is a prefix of every word. The length of u is denoted by $|u|$, for $l \leq |u|$, $u[l]$ denotes the prefix of u with the length l, and for $0 < l \leq |u|$, $u(l)$ denotes the l^{th} letter occurring in u. For any sub-alphabet $\Sigma_i \subseteq \Sigma$, let $\pi_i : \Sigma^* \to \Sigma_i^*$ be the unique monoid morphism extending the map $\pi_i(\sigma) = \sigma$ if $\sigma \in \Sigma_i$ else ε (letters $\sigma \in \Sigma$ are mapped to words by the usual embedding of Σ into Σ^*). For $u, v \in \Sigma^*$, let $u \simeq_i v$ if $\pi_i(u) = \pi_i(v)$. Throughout the paper, L is a non-empty prefix-closed language in $Rat(\Sigma^*)$ and for all $i \in \{1, \cdots, n\}$, $\Sigma_i \subseteq \Sigma$, $S_i \in Rat(\Sigma^*)$, and $S_i \subseteq L$.

The language L represents the behaviour of a system with n users. For $i \in \{1, \cdots, n\}$, the sub-alphabet Σ_i represents the set of the interactions that may take place between the system and the user i. Users observe the system by interacting with it. If the system's trajectory is represented by $w \in L$, then the induced observation for the user i is $\pi_i(w)$. Two users can communicate only by jointly interacting with the system, *e.g.* $\sigma \in \Sigma_i \cap \Sigma_j$ is an interaction of the system with the users i and j. If it went differently, two users with the respective alphabets Σ_i and Σ_j would have the same power of observation as a single user with the alphabet $\Sigma_i \cup \Sigma_j$, and they could have been replaced with the latter.

For each $i \in \{1, \cdots, n\}$, the membership of the actual system's trajectory to the subset $S_i \subseteq L$ is intended to be kept *secret* from the user i. In the terminology of [4] and [3], the predicate S_i should be *opaque* w.r.t. the observation function π_i and language L.

Definition 1 *S_i is* opaque *w.r.t. π_i (and L) if* $(\forall w \in S_i)\,(\exists w' \in L \setminus S_i)\ w \simeq_i w'$

As we explained in the introduction, we use here a strongly restricted form of the original definition where the observation functions may be state-dependent and history-dependent. On the other hand, we consider a concurrent version of opacity.

Definition 2 $(S_i)_i$ *is* opaque *w.r.t.* $(\pi_i)_i$ *if for all* i, S_i *is opaque w.r.t.* π_i.

Dealing with concurrent opacity does not make a big change for checking opacity, which is extremely easy in our case since we concern ourselves exclusively with regular systems and secrets.

Proposition 1 *It is decidable whether* $(S_i)_i$ *is opaque w.r.t.* $(\pi_i)_i$.

Proof: By definition, it suffices to decide for each $i \in \{1, \ldots, n\}$ whether S_i is opaque w.r.t. π_i. The considered property holds if and only if $\pi_i(S_i) \subseteq \pi_i(L \setminus S_i)$. As L and S_i are regular, $L \setminus S_i$ is regular, and since morphic images of regular languages are regular, this relation of inclusion can be decided. □

Example 1 *Let* $\Sigma = \{a, b, c\}$ *and* L *be the set of prefixes of words in* $(a + b)\, c$. *Let* $\Sigma_1 = \Sigma_2 = \{c\}$, *and let* S_1 *and* S_2 *be the intersections of* L *with* $\Sigma^* a \Sigma^*$ *and* $\Sigma^* b \Sigma^*$, *respectively. The concurrent secret* (S_1, S_2) *is opaque. From the observation of the event* c, *one is indeed unable to infer whether it was preceded with an* a *or with a* b.

Our purpose is to study the enforcement of (concurrent) opacity by supervisory control. Given L and $S_i \subseteq L$ (for $i \in \{1, \ldots, n\}$) one searches for the largest non-empty prefix-closed sub-language L' of L such that the induced restriction $(S'_i)_i$ of $(S_i)_i$ defined with $S'_i = S_i \cap L'$ is opaque w.r.t. $(\pi_i)_i$ and L'. Naturally $L' = L$ when $(S_i)_i$ is opaque w.r.t. $(\pi_i)_i$ and L. Enforcing concurrent opacity ($n > 1$) requires as we shall see significantly more efforts than enforcing opacity. We assume here that all events $\sigma \in \Sigma$ are controllable events, therefore we do not impose any specific constraint on admissible behaviours of controllers *i.e.* on languages $L' \in \Sigma^*$. We leave the consideration of uncontrollable events to further study.

3 A fixpoint characterization of the maximum permissive control enforcing concurrent opacity

In this section, we show that the concurrent opacity control problem has a unique maximal solution that we characterize as a greatest fixpoint. We propose two counter-examples in which this maximum permissive control either is not regular or cannot be computed within a finite number of fixpoint iterations.

In the sequel, $\mathcal{S} = \{(\Sigma_1, S_1), \ldots, (\Sigma_n, S_n)\}$ denotes a concurrent secret upon a fixed language $L \subseteq \Sigma^*$ ($\Sigma_i \subseteq \Sigma$ and $S_i \subseteq L \subseteq \Sigma^*$ for all i). We say that $\mathcal{S}$ is opaque if $(S_i)_i$ is opaque w.r.t. $(\pi_i)_i$. A control is any non-empty prefix-closed language $L' \subseteq L$. We say that $\mathcal{S}$ is opaque under the control $L' \subseteq L$ if the induced secret $(S'_i)_i$ defined with $S'_i = S_i \cap L'$ is opaque w.r.t. $(\pi_i)_i$ and L'.

Definition 3 *For any prefix-closed subset L' of L, the* safe kernel *of L' w.r.t. the secret $\mathcal{S}$, notation $K(L',\mathcal{S})$, is the largest prefix-closed subset of L' such that $uv \in K(L',\mathcal{S}) \Rightarrow (\forall i)(\exists u' \in L' \setminus S_i)\ u \simeq_i u'$.*

Thus, $\mathcal{S}$ is opaque under the control $L' \subseteq L$ if and only if $L' = K(L',\mathcal{S})$, *i.e.* L' is a fixpoint of $K(\bullet,\mathcal{S})$. It is immediately observed that $K(L',\mathcal{S})$ is monotonic in the first argument. As the prefix-closed subsets of L form a complete sub-lattice of $\mathcal{P}(\Sigma^*)$, it follows from Knaster-Tarski's theorem [7] that $K(\bullet,\mathcal{S})$ has a greatest fixpoint in this sub-lattice.

Definition 4 *Let $C(L,\mathcal{S})$ be the greatest fixed point of the operator $K(\bullet,\mathcal{S})$.*

Proposition 2 *If $C(L,\mathcal{S}) \neq \emptyset$, then it is the maximal permissive control enforcing the opacity of $\mathcal{S}$, otherwise no such control can exist.*

Proof: This is a direct application of the Knaster-Tarski's fixpoint theorem. □

Warning 1 *The condition $L' \subseteq C(L,\mathcal{S})$ is necessary* but not sufficient *for some non-empty control L' to enforce the opacity of $\mathcal{S}$. For instance, in Example 1, $C(L,\mathcal{S}) = L$, but the secret S_1 is not opaque w.r.t. $L' = \varepsilon + a + ac$.*

The fixpoint characterization of the optimal control enforcing opacity does not show that $C(L,\mathcal{S})$ can be computed, nor that the control can be implemented with a finite device. When $n = 1$, *i.e.* when $\mathcal{S} = \{(\Sigma_1, S_1)\}$, this is not a problem because in this particular case, $C(L,\mathcal{S})$ is equal to $K(L,\mathcal{S})$ and it may be shown that the latter is the set of words with all prefixes in $L \cap \pi_1^{-1}(L \setminus S_1)$. Therefore, $C(L,\mathcal{S}) = \Sigma^* \setminus ((\Sigma^* \setminus (L \cap \pi_1^{-1}(L \setminus S_1))\,\Sigma^*)$ which is clearly a regular language since L and S_1 are regular. When $n > 1$, two situations contrast. The nice situation is when $C(L,\mathcal{S})$ can be computed from L by a finite number of iterated applications of the operator $K(\bullet,\mathcal{S})$. It can actually be shown that when L' is regular subset of L, the same holds for $K(L',\mathcal{S})$, hence in the considered case $C(L,\mathcal{S})$ is regular. The rest of the section illustrates the converse situation.

3.1 A case where the closure ordinal of $K(\bullet,\mathcal{S})$ is transfinite

Let $\Sigma = \{a,b,c,d,e,f\}$ and let L be the prefix-closed language accepted by the finite automaton of Fig.1 (where all states are accepting states). Define $\mathcal{S} = \{(\Sigma_1, S_1), (\Sigma_2, S_2)\}$ with $\Sigma_1 = \{c, f\}$, $S_1 = \Sigma^* afc\,(\Sigma \setminus \{c\})^*$ (this secret is safe if, by observing only c and f, one cannot find out in any run that the last occurrence of c was preceded by af), and $\Sigma_2 = \{b, e\}$, $S_2 = \Sigma^* deb\,(\Sigma \setminus \{b\})^*$ (this secret is safe if, by observing only b and e, one cannot find out in any run that the last occurrence of b was preceded by de). Let $L_1 = K(L,\mathcal{S})$ be the first language encountered in the greatest fixpoint iteration converging to $C(L,\mathcal{S})$, then $L_1 = L \setminus afc\Sigma^*$ (the run afc reveals the secret S_1 and the runs in $afd\Sigma^*$ reveal nothing). The second item $L_2 = K(L_1,\mathcal{S})$ is the language $L_1 \setminus afdeb\Sigma^*$ (relatively to L_1, the run $afdeb$ reveals the secret S_2, and the runs in $afdea\Sigma^*$ reveal nothing). After afc and $afdeb$

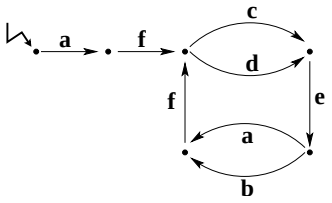

Figure 1: An automaton

have been eliminated, the initial situation reproduces up to the prefix $afdea$. Therefore, the fixpoint iteration produces a strictly decreasing and infinite sequence of languages L_j. The limit $C(L,\mathcal{S})$ of this decreasing chain is the set of all prefixes of words in the regular set $L_\omega = (afde)^*$, hence it is regular. The optimal control enforcing the opacity of $\mathcal{S}$ may be implemented by any finite automaton recognizing L_ω.

Let us now extend the concurrent secret into $\mathcal{S} = \{(\Sigma_1, S_1), (\Sigma_2, S_2), (\Sigma_3, S_3)\}$ with (Σ_1, S_1) and (Σ_2, S_2) as above, $\Sigma_3 = \emptyset$ and $S_3 = \Sigma^* \setminus (\Sigma^* c \Sigma^*)$. Then, the closure ordinal of $K(\bullet, \mathcal{S})$ increases from ω to $\omega + 1$. To see this observe that, since Σ_3 is empty, the secret S_3 is safe relatively to any language $L' \subseteq L$ containing at least one word containing at least one occurrence of c. The greatest fixpoint iteration for $C(L,\mathcal{S})$ starts with the same decreasing sequence L_j as before, but $K(L_\omega, \mathcal{S})$ differs now from L_ω because L_ω contains no word containing c (differing in that form all L_j). In fact, $L_{\omega+1} = K(L_\omega, \mathcal{S}) = \emptyset$ and this is a fixpoint. Opacity can therefore not be enforced.

3.2 A case where $C(L,\mathcal{S})$ is not regular

Let $\Sigma = \{a, b, x, y\}$ and L be the set of prefixes of words in $(ax)^* (\varepsilon + ab) (yb)^*$. Define $\mathcal{S} = \{(\Sigma_i, S_i) \mid 1 \leq i \leq 3\}$ as follows (letting $\complement L' = L \setminus L'$ for $L' \subseteq L$):

1. $\Sigma_1 = \{a, b\}$, $\complement S_1 = \varepsilon + (ax)^* ab (yb)^* + (\Sigma \setminus \{b\})^*$
2. $\Sigma_2 = \{x, y\}$, $\complement S_2 = (ax)^* (yb)^*$
3. $\Sigma_3 = \{a, b, x, y\}$, $\complement S_3 = \varepsilon + a\Sigma^*$

We claim that $C(L,\mathcal{S})$ is not a regular language and worse, the family of regular controls enforcing the opacity of $\mathcal{S}$ has no largest element. In order to establish the first part of the claim, we will show that $C(L,\mathcal{S})$ is equal to the set L' of all prefixes of words in the non regular language $\cup_{n\in\mathbb{N}} (ax)^n (\varepsilon + ab) (yb)^n$. Recalling that the subset of maximal words in a regular language is a regular language, it is then clear that $C(L,\mathcal{S})$ is not regular. We show that $C(L,\mathcal{S}) = L'$ by proving the reciprocal inclusion of the two sets.

For $i \in \{1,2,3\}$ and for all $w \in L$, define $w(\!|i|\!) = \{w' \in L' \mid w \simeq_i w' \wedge w' \in \complement S_i\}$. Since $C(L,\mathcal{S})$ is the maximal permissive control enforcing the opacity of $\mathcal{S}$, it suffices to show that

$\mathcal{S}$ is opaque w.r.t. L' in order to establish $L' \subseteq C(L,\mathcal{S})$. Therefore, as $L' \subseteq L$, it suffices to prove the assertion:
i) for all $w \in L'$ and $i \in \{1,2,3\}$, the set $w(\!|i|\!)$ is non-empty.
The relation $C(L,\mathcal{S}) \subseteq L'$ may be reduced to a similar assertion. For $\tau \in \{1,2,3\}^*$ and for all $w \in L$, let $w(\!|\tau|\!)$ be inductively defined with $w(\!|\varepsilon|\!) = \{w\}$ and $w(\!|\tau i|\!) = \cup\{w'(\!|i|\!) \mid w' \in w(\!|\tau|\!)\}$. Whenever, for some $w \in L$, $w(\!|\tau|\!) = \emptyset$ for some τ, $w \notin C(L,\mathcal{S})$. Actually, if τ has length j, then w does not belong to the j^{th} language L_j encountered in the greatest fixpoint iteration for $C(L,\mathcal{S})$, *i.e.* $L_0 = L$, $L_1 = K(L_0,\mathcal{S})$, and so on. Therefore, $C(L,\mathcal{S}) \subseteq L'$ follows from the assertion:
ii) for all $w \in L \setminus L'$, the set $w(\!|\tau|\!)$ is empty for some $\tau \in \{1,2,3\}^*$.
Note that for any $w \in L$, if $w \in \varepsilon + a\Sigma^*$ then $w(\!|3|\!) = \{w\}$ else $w(\!|3|\!) = \emptyset$, hence $C(L,\mathcal{S})$ or any stronger opacity enforcing control cannot contain any word starting with a letter different from a (this was the point in introducing the secret S_3).

Let us now prove *(i)* and *(ii)*. For this purpose, one may classify the words $w \in L$ into categories as follows (singleton sets of words are represented as words).

1. $w = (ax)^n ab\,(yb)^m$:
   $w \in w(\!|1|\!)$, $w(\!|2|\!) = (ax)^n (yb)^m$, $w(\!|3|\!) = w$
2. $w = (ax)^n (yb)^m$:
   (a) $n, m \geq 1 \Rightarrow w(\!|1|\!) = (ax)^{n-1} ab\,(yb)^{m-1}$, $w \in w(\!|2|\!)$, $w(\!|3|\!) = w$
   (b) $m = 0 \Rightarrow w \in w(\!|1|\!) \cap w(\!|2|\!)$, $w(\!|3|\!) = w$
   (c) $n = 0, m \geq 1 \Rightarrow w(\!|1|\!) = \emptyset$, $w \in w(\!|2|\!)$, $w(\!|3|\!) = \emptyset$
3. $w = (ax)^n ab\,(yb)^m y$:
   $w(\!|2|\!) = (ax)^n (yb)^{m+1}$, $(ax)^n ab\,(yb)^m \in w(\!|1|\!)$, $w(\!|3|\!) = w$
4. $w = (ax)^n a$:
   $w \in w(\!|1|\!)$, $w(\!|2|\!) = (ax)^n$, $w(\!|3|\!) = w$
5. $w = (ax)^n (yb)^m y$:
   (a) $n, m \geq 1 \Rightarrow w(\!|1|\!) = (ax)^{n-1}ab(yb)^{m-1}$, $(ax)^n(yb)^{m+1} \in w(\!|2|\!)$, $w(\!|3|\!) = w$
   (b) $n \geq 1, m = 0 \Rightarrow w \in w(\!|1|\!)$, $(ax)^n yb \in w(\!|2|\!)$, $w(\!|3|\!) = w$
   (c) $n = 0, m \geq 1 \Rightarrow w(\!|1|\!) = \emptyset$, $(yb)^{m+1} \in w(\!|2|\!)$, $w(\!|3|\!) = \emptyset$
   (d) $n, m = 0 \Rightarrow \varepsilon \in w(\!|1|\!)$, $yb \in w(\!|2|\!)$, $w(\!|3|\!) = \emptyset$

The words $w' \in L'$ may be classified similarly into five categories as follows:

1. $w' = (ax)^n ab (yb)^m$ with $n \geq m$,
2. $w' = (ax)^n (yb)^m$ with $n \geq m$,
3. $w' = (ax)^n ab (yb)^m y$ with $n \geq m+1$,
4. $w' = (ax)^n a$, or
5. $w' = (ax)^n (yb)^m y$ with $n \geq m+1$.

It results from a comparison of the two classifications that the property *(i)* holds. Finally, the property *(ii)* also holds, since all words $w \in L \setminus L'$ necessarily meet one the five cases below.

1. $w = (ax)^n ab (yb)^m$ with $n < m$: then $w(\!|(2 \cdot 1)^{n+1}|\!) = \emptyset$
2. $w = (ax)^n (yb)^m$ with $n < m$: then $w(\!|(1 \cdot 2)^n \cdot 1|\!) = \emptyset$
3. $w = (ax)^n ab (yb)^m y$ with $n \leq m$: then $w(\!|1 \cdot (2 \cdot 1)^{n+1}|\!) = \emptyset$
4. $w = (ax)^n (yb)^m y$ with $0 < n \leq m$: then $w(\!|(1 \cdot 2)^n \cdot 1|\!) = \emptyset$
5. $w = (yb)^m y$: then $w(\!|3|\!) = \emptyset$

We have thus proved that $C(L, \mathcal{S})$ is not a regular language. It remains to show that the family of regular controls enforcing the opacity of $\mathcal{S}$ has no largest element. Suppose the contrary, and let R be the largest prefix-closed regular subset of L such that $\mathcal{S}$ is opaque w.r.t. R. Necessarily, $(ax)^n (yb)^n \notin R$ for some n. If it were otherwise, because $w(\!|1|\!) = (ax)^{n-1} (yb)^{n-1}$ for $w = (ax)^n (yb)^n$, R would coincide with L', which is not possible (L' is not regular). Let n be the least integer such that $(ax)^n (yb)^n \notin R$, and let R' be R augmented with all prefixes of words in $\{(ax)^n (yb)^n, (ax)^{n-1} ab (yb)^{n-1}\}$ not already in R. The language R' is prefix-closed and regular, and using the classification of L produced earlier, one can verify that $\mathcal{S}$ is opaque w.r.t. R'. Thus, a contradiction has been reached.

4 Control enabling and ω-trees

Warning 2 *From now on, $S_i \Sigma^* \subseteq S_i$ is imposed on all sets S_i in $\{(\Sigma_1, S_1), \ldots, (\Sigma_n, S_n)\}$.*

This section serves as a bridge between the general problem and the practical solutions to this problem that we shall propose in specific cases. The working assumption that secrets are suffix-closed is motivated by its convenience (if not its necessity) for enforcing opacity with finite control. Although this working assumption was not satisfied in the two examples from sections 3.1 and 3.2, it is not unnatural for the properties of a system to be expressed by Scott open sets of trajectories (w.r.t. the prefix order). We give below a simpler definition

of the operator $K(\bullet, \mathcal{S})$, which is equivalent to the earlier definition when secrets are suffix-closed. Then we consider ω-trees that may be seen as proofs of the membership of words to the greatest fixpoint of this operator, and hence as proofs of the control enabledness of the associated trajectories. Finally, we propose structural conditions on sets of proof trees strong enough to entail the regularity of $C(L, \mathcal{S})$, thus paving the way for section 5.

Definition 5 (modified form of Def. 3) *For any prefix-closed subset L' of L, the* safe kernel *of L' w.r.t. the secret $\mathcal{S}$, notation $K(L', \mathcal{S})$, is the largest subset of L' such that $w \in K(L', \mathcal{S}) \Rightarrow (\forall i)(\exists w' \in L' \setminus S_i)\ w \simeq_i w'$.*

Proposition 3 *Definitions 3 and 5 are equivalent.*

Proof: In order to make the distinction for the duration of this proof, let $K(\bullet, \mathcal{S})$ and $K'(\bullet, \mathcal{S})$ be the two operators from Def. 3 and Def. 5, respectively. Clearly, $K(L', \mathcal{S}) \subseteq K'(L', \mathcal{S})$ for any L'. We show hereafter the converse relation. Consider any word $w \in K'(L', \mathcal{S})$ and let $w = uv$ be any decomposition of this word into two factors. We should prove that for all $i \in \{1, \ldots, n\}$, $u \simeq_i u'$ for some $u' \in L' \setminus S_i$. As $w \in K'(L', \mathcal{S})$ and by definition, $w \simeq_i w'$ for some $w' \in L' \setminus S_i$. Now $w' \simeq_i uv$, hence there exists at least one decomposition $w' = u'v'$ such that $u \simeq_i u'$. Finally, $u' \in L'$ by prefix-closedness of L', and $u' \notin S_i$ by suffix-closedness of S_i. Therefore, $w \in K(L', \mathcal{S})$. □

Definition 6 *Given $w \in L$, a* proof *of the* control enabledness *of w is a map $f : \{1, \ldots, n\}^* \to L$ such that $f(\varepsilon) = w$ and for all $\tau \in \{1, \ldots, n\}^*$ and $j \in \{1, \ldots, n\}$, $f(\tau) \simeq_j f(\tau j)$ and $f(\tau j) \notin S_j$.*

The map f in the above definition is just a complete n-ary ordered tree labelled on nodes, thus in particular it is an infinite tree. The next proposition follows immediately from the co-inductive definition of $C(L, \mathcal{S})$.

Proposition 4 *For any $w \in L$, $w \in C(L, \mathcal{S})$ if and only if there exists a proof of the control enabledness of w.*

A nice situation is when the control enabledness of a trajectory may be proved with a regular tree. Let us recall the definition.

Definition 7 *Let $f : \{1, \ldots, n\}^* \to L$ be a (complete n-ary ordered labelled) tree. For any $\tau \in \{1, \ldots, n\}^*$, the sub-tree of f rooted at τ, in notation f/τ, is the (complete n-ary ordered labelled) tree defined with $(f/\tau)(\tau') = f(\tau\tau')$ for all $\tau' \in \{1, \ldots, n\}^*$. The tree f is regular if it has a finite number of sub-trees f/τ.*

Any regular tree f may be folded to a finite rooted graph which unfolds to f and which therefore represents this regular tree. When the control enabledness of the (good) trajectories may be proved using regular trees exclusively, this predicate is recursively enumerable.

This condition is necessary and sufficient for enforcing control, but not very efficiently. For this reason, the main focus in supervisory control is on finite state controllers. In the rest of the section, we search for additional conditions, stronger than the regularity of proof trees, entailing the regularity of the control $C(L,\mathcal{S})$.

A first attempt towards this goal is to impose an upper bound on the number of (different) subtrees of a regular proof tree. Equivalently, one may require that all proof trees conform to a finite collection of finite patterns as follows.

Definition 8 *A* finite pattern for proofs *(of control enabledness of trajectories) is a finite deterministic and complete automaton* $(Q,\{1,\ldots,n\},q_0)$ *(thus* $q_0\in Q$ *and each* $i\in\{1,\ldots,n\}$ *maps* Q *to itself). A proof tree* $f:\{1,\ldots,n\}^*\to L$ conforms to *a finite pattern if there exists a labelling map* $\lambda:Q\to L$ *such that* $f(\tau)=\lambda(q_0\cdot\tau)$ *for all* $\tau\in\{1,\ldots,n\}^*$ *letting* $q\cdot\tau$ *be defined inductively with* $q\cdot\varepsilon=q$ *and* $q\cdot(\tau_1\tau_2)=(q\cdot\tau_1)\cdot\tau_2$ *for all* $q\in Q$.

The idea behind this definition is that proof trees contain bounded information up to the choice of a bounded number of words in L.

Example 2 *Let* $\Sigma=\{a,b\}$ *and* $L=\Sigma^*$. *Let* $\mathcal{S}=\{(\Sigma_1,S_1),(\Sigma_2,S_2)\}$ *with* $\Sigma_1=\{a\}$, $\complement S_1=b^*a^*$ *and* $\Sigma_2=\{b\}$, $\complement S_2=a^*b^*$. *The finite pattern shown on Fig. 2 supplies proofs*

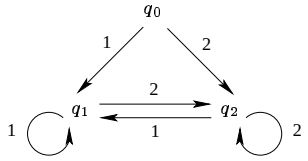

Figure 2:

of control enabledness for all trajectories. For any word w *with* n *occurrences of* a *and* m *occurrences of* b, *the labelling map defined with* $\lambda(q_0)=w$, $\lambda(q_1)=b^ma^n$, *and* $\lambda(q_2)=a^nb^m$ *induces in fact an* ω*-tree showing that* $w\in C(L,\mathcal{S})$.

There are two sources of problems with the proof patterns from Def. 8. A first difficulty is that, given L, $\mathcal{S}$ and $(Q,\{1,\ldots,n\},q_0)$, the set of the labelling maps $\lambda:Q\to L$ considered in this definition is generally not regular, *i.e.* it cannot be defined with a finite automaton on $(\Sigma^*)^n$. For instance, if the labelling maps considered in the example 2 did form a regular set, then the set of all pairs (b^ma^n,a^nb^m) would be regular, but the iteration lemma for rational sets [2] entails the opposite (supposing the set is regular and according to the lemma, for some $N>1$ and for large enough n and m, (b^ma^n,a^nb^m) may be written as $(x,x')(y,y')(z,z')$ such that $(x,x')(y,y')^*(z,z')$ is included in the set, $0<|y|+|y'|$, and $|x|+|x'|+|y|+|y'|\leq N$). A second difficulty is that, given L, $\mathcal{S}$ and $(Q,\{1,\ldots,n\},q_0)$, the set of values taken at $q=q_0$ by the labelling maps from Def. 8 is sometimes not regular. An example is shown hereafter.

Example 3 *Let* $\Sigma = \{a, b\}$ *and* $L = \Sigma^*$. *Let* $\mathcal{S} = \{(\Sigma_1, S_1), (\Sigma_2, S_2)\}$ *where* $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$, *and* $\complement S_1 = \complement S_2 = (\varepsilon + b)(ab)^*(\varepsilon + a)$. *Consider the set of all maps labelling the finite proof pattern from Fig. 3 in an adequate way. The set of values taken by these maps at* $q = q_0$ *is the set of all words in which the numbers of occurrences* a *and* b *differ by at most one, hence it is not regular.*

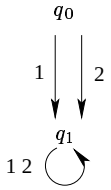

Figure 3:

Note that in both examples above, $C(L, \mathcal{S}) = \Sigma^*$, and proofs of control enabledness may be obtained for all $w \in \Sigma^*$ by labelling the finite proof pattern shown in Fig. 4.

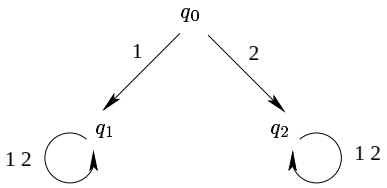

Figure 4:

In order to dodge these problems, one may concentrate on restricted proof patterns as follows.

Definition 9 *A* skeleton of proof *(of enabledness of trajectories) is a finite proof pattern* $(Q, \{1, \ldots, n\}, q_0)$ *with a prefix-closed subset* $T \subseteq \{1, \ldots, n\}^*$ *such that:*
$(\forall q \in Q)\,(\exists!\, \tau \in T)\,(q = q_0 \cdot \tau)$, *and for any map* $\lambda : Q \to L$,
$(\forall \tau)\,(\forall j)\ \ (\tau j \in T \wedge \lambda(q_0 \cdot \tau) \simeq_j \lambda(q_0 \cdot \tau j) \wedge \lambda(q_0 \cdot \tau j) \notin S_j)$ *entails*
$(\forall q)\,(\forall j)\ \ (\lambda(q) \simeq_j \lambda(q \cdot j) \wedge \lambda(q \cdot j) \notin S_j)$ *where we let* τ *and* j *range over* $\{1, \ldots, n\}^*$ *and* $\{1, \ldots, n\}$, *respectively.*

The set T in this definition induces a (finite) tree, rooted at q_0, that spans the (underlying graph of) the automaton $(Q, \{1, \ldots, n\}, q_0)$. The point is that for any map $\lambda : Q \to L$, if the property $(\lambda(q) \simeq_j \lambda(q \cdot j) \wedge \lambda(q \cdot j) \notin S_j)$ holds for all arcs $(q, q \cdot j)$ in the spanning tree, then it holds also for all chords, *i.e.* for the remaining edges of (the underlying graph of) $(Q, \{1, \ldots, n\}, q_0)$.

Proposition 5 *If there exists a finite family of proof skeletons such that all trajectories $w \in C(L,\mathcal{S})$ have proofs of enabledness conform to some of the corresponding patterns, then $C(L,\mathcal{S})$ is a regular language.*

Proof: It suffices to show that when one skeleton $(Q,\{1,\ldots,n\},q_0,T)$ has been fixed, the set of trajectories $w \in L$ with proofs of enabledness conform to the pattern $(Q,\{1,\ldots,n\},q_0)$ is regular. In view of the definitions 8 and 9, a word w belongs to the considered set if and only if $\lambda(q_0)=w$ for some map $\lambda : Q \to L$ satisfying $\lambda(q_0\cdot\tau) \simeq_j \lambda(q_0\cdot\tau j)$ and $\lambda(q_0\cdot\tau j) \notin S_j$ whenever $\tau j \in T$ and $j \in \{1,\ldots,n\}$. In order to show that this is a regular set, we construct the Arnold-Nivat product [1] of a family of automata $\mathcal{A}_\tau$ indexed with $\tau \in T$, as follows. Let $\mathcal{A}_\varepsilon$ be a (finite deterministic) partial automaton recognizing L, and for each sequence τj in T with $j \in \{1,\ldots,n\}$, let $\mathcal{A}_{\tau j}$ be a (finite deterministic) partial automaton recognizing $L \setminus S_j$. This defines the components of the product. As for the synchronizations, let $\mathcal{V}$ be the set of T-vectors $\vec{v} : T \to (\Sigma \cup \{\varepsilon\})$ such that $\vec{v}(\tau) \in \Sigma_j \Leftrightarrow \vec{v}(\tau j) \in \Sigma_j$ and $\vec{v}(\tau) \in \Sigma_j \Rightarrow \vec{v}(\tau) = \vec{v}(\tau j)$ whenever τj in T and $j \in \{1,\ldots,n\}$. The induced product is a (finite deterministic) partial automaton $\mathcal{A} = (\mathcal{Q},\mathcal{V},\vec{q_0})$ as follows:
- the set of states $\mathcal{Q}$ is a set of T-vectors,
- for each $\tau \in T$, $\vec{q_0}(\tau)$ is the initial state of $\mathcal{A}_\tau$,
- for all $\vec{q} \in \mathcal{Q}$ and $\tau \in T$, $\vec{q}(\tau)$ is a state of $\mathcal{A}_\tau$,
- for all $\vec{q} \in \mathcal{Q}$, $\vec{v} \in \mathcal{V}$ and $\tau \in T$, $(\vec{q}\cdot\vec{v})(\tau) = \vec{q}(\tau)\cdot\vec{v}(\tau)$.

Therefore, $\vec{q}\cdot\vec{v}$ is defined if and only if $\vec{q}(\tau)\cdot\vec{v}(\tau) = \vec{q'}(\tau)$ is defined for all τ.

Let $\vec{v}_1 \ldots \vec{v}_m$ be a word over $\mathcal{V}$ accepted by $\mathcal{A}$. An associated T-vector $\vec{w} : T \to L$ may be defined by setting $\vec{w}(\tau) = \vec{v}_1(\tau)\ldots\vec{v}_m(\tau)$ for all $\tau \in T$. It follows directly from the construction that the map $\lambda : Q \to L$ defined with $\lambda(q_0\cdot\tau) = \vec{w}(\tau)$ for all $\tau \in T$ satisfies $\lambda(q_0\cdot\tau) \simeq_j \lambda(q_0\cdot\tau j)$ and $\lambda(q_0\cdot\tau j) \notin S_j$ for $\tau j \in T$ and $j \in \{1,\ldots,n\}$, hence $\vec{w}(\varepsilon) \in C(L,\mathcal{S})$.

As $\mathcal{A}$ is a finite automaton, the projection of the language of $\mathcal{A}$ along ε is a regular language. In order to complete the proof, it suffices therefore to show that for any map $\lambda : Q \to L$ satisfying $\lambda(q_0\cdot\tau) \simeq_j \lambda(q_0\cdot\tau j)$ and $\lambda(q_0\cdot\tau j) \notin S_j$ for all $\tau j \in T$, the vector $\vec{w} : T \to L$ defined with $\vec{w}(\tau) = \lambda(q_0\cdot\tau)$ for all $\tau \in T$ may be written as a word $\vec{v}_1 \ldots \vec{v}_m$ recognized by $\mathcal{A}$. Given the construction of this automaton, it suffices to exhibit a sequence $\vec{v}_1 \ldots \vec{v}_m \in \mathcal{V}^*$ such that $\vec{w}(\tau) = \vec{v}_1(\tau)\ldots\vec{v}_m(\tau)$ for all $\tau \in T$. This is the contribution of the lemma 1 (see herafter). □

Lemma 1 *Let $\vec{w} : T \to \Sigma^*$ where T is a prefix-closed subset of $\{1,\ldots,n\}^*$ and $\vec{w}(\tau) \simeq_j \vec{w}(\tau j)$ for all $\tau j \in T$ with $j \in \{1,\ldots,n\}$. Then $\vec{w}(\tau) = \vec{v}_1(\tau)\ldots\vec{v}_m(\tau)$ for all $\tau \in T$ for some sequence of vectors $\vec{v}_k : T \to \Sigma \cup \{\varepsilon\}$ such that for all $\tau j \in T$, $(\vec{v}_k(\tau) \in \Sigma_j \vee \vec{v}_k(\tau j) \in \Sigma_j) \Rightarrow \vec{v}_k(\tau) = \vec{v}_k(\tau j)$.*

Proof: Let $\mathcal{E}$ be the set of all pairs (τ,i) such that $\tau \in T$ and $0 < i \leq |\vec{w}(\tau)|$. Let $<$ be the partial order on $\mathcal{E}$ defined with $(\tau,i) < (\tau',i')$ if $\tau = \tau'$ and $i < i'$. For each $j \in \{1,\ldots,n\}$, let $(\tau,i) \Vdash_j (\tau j,k)$ if $\pi_j(\vec{w}(\tau)[i])) = \pi_j(\vec{w}(\tau j)[k]))$, $\vec{w}(\tau)(i) = \vec{w}(\tau j)(k)$, and this letter is in Σ_j. Let $\equiv$ denote the equivalence on $\mathcal{E}$ generated from the union of the relations $\Vdash_j$. We

claim that this equivalence does not intersect and is compatible with the partial order $<$. Let us establish this double claim.

i) Suppose for a contradiction that $(\tau, i) < (\tau, i')$ and $(\tau, i) \equiv (\tau, i')$. Then, by definition of $\equiv$ and the relations $\Vdash_j$, the words $\vec{w}(\tau)[i]$ and $\vec{w}(\tau)[i']$ end with a common letter $\vec{w}(\tau)(i) = \vec{w}(\tau)(i')$, and this letter occurs the same number of times in both words. As $i < i'$, this is clearly not possible.

ii) Suppose for a contradiction that $(\tau, i) < (\tau, i')$ and $(\tau', j) < (\tau', j')$ while $(\tau, i) \equiv (\tau', j')$ and $(\tau, i') \equiv (\tau', j)$. Then, by definition of $\equiv$ and the relations $\Vdash_j$, $\vec{w}(\tau)(i) = \vec{w}(\tau')(j')$ and this letter σ occurs the same number of times in both words $\vec{w}(\tau)[i]$ and $\vec{w}(\tau')[j']$. In the same way, $\vec{w}(\tau)(i') = \vec{w}(\tau')(j)$ and this letter σ' occurs the same number of times in both words $\vec{w}(\tau)[i']$ and $\vec{w}(\tau')[j]$. Since $i < i'$ and $j < j'$, it follows that σ and σ' are different letters (see Fig. 5).

Now let $\tau = \rho x_1 \dots x_k$ and $\tau' = \rho y_1 \dots y_l$ where ρ is the longest common prefix of τ and τ' and $x_h, y_h \in \{1, \dots, n\}$. Then by definition of $\equiv$ and the relations $\Vdash_j$, $(\tau, i) \equiv (\tau', j')$ and $(\tau, i') \equiv (\tau', j)$ entail that σ and σ' belong jointly to all the alphabets Σ_{x_h} $(1 \leq h \leq k)$ and Σ_{y_h} $(1 \leq h \leq l)$. On the other hand, by the i part of the proof, $(\tau, i) \equiv (\tau', j')$ entails that necessarily $\vec{w}(\tau)[i] \Vdash_{x_k}^{-1} \circ \dots \circ \Vdash_{x_1}^{-1} \circ \Vdash_{y_1} \circ \dots \circ \Vdash_{y_l} \vec{w}(\tau')[j']$. Therefore the words $\vec{w}(\tau)[i]$ and $\vec{w}(\tau')[j']$ must have the same number of occurrences of the letter σ', which is obviously not possible.

Let $\mathcal{C} = (\mathcal{E}/\equiv)$. Since $<$ is compatible and does not intersect with $\equiv$, the binary relation $(< \cup \equiv)^*/\equiv$ is a strict partial order on $\mathcal{C}$. Let $C_1 \dots C_m$ be an enumeration of $\mathcal{C}$ compatible with this order. Each equivalence class $C \in \mathcal{C}$ induces naturally a vector $\vec{v} \in \mathcal{V}$, *viz.* $\vec{v}(\tau) = \vec{w}(\tau)(i)$ if $(\tau, i) \in C$ for some i, or ε otherwise. Let $\vec{v}_1 \dots \vec{v}_m$ be the vectors associated with $C_1 \dots C_m$, respectively. Then for any $\tau \in T$, $\vec{w}(\tau) = \vec{v}_1(\tau) \dots \vec{v}_m(\tau)$ as desired. □

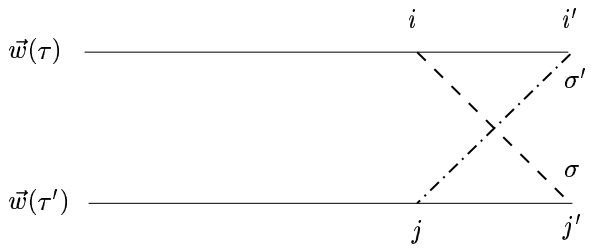

Figure 5:

Proposition 5 opens the way to the practical synthesis of maximal permissive supervisory control for concurrent opacity. The conditions for the application of this proposition are examined further in section 5.

5 Concurrent secrets with a regular opacity control

We propose here conditions on concurrent secrets $\mathcal{S} = \{(\Sigma_1, S_1), \ldots, (\Sigma_n, S_n)\}$ ensuring that the maximal permissive opacity control $C(L, \mathcal{S})$ is the language of a finite automaton, that may effectively be constructed from finite automata accepting the language L and the secrets S_i. We examine first the case where the alphabets Σ_i form a chain for the inclusion, second the case where the secrets S_i form a chain for the inclusion, third the case where any secret S_i is saturated by any equivalence $\simeq_j$ such that $i \neq j$ (a set is *saturated* by an equivalence if it is a union of equivalence classes). We consider finally the combinations of the three cases for the different pairs (i, j).

Proposition 6 *If the alphabets Σ_i form a chain for the inclusion, then the enabledness of all trajectories $w \in C(L, \mathcal{S})$ may be shown with a single proof skeleton $SK1$.*

Proof: Given the chain $\Sigma_1 \subseteq \Sigma_2 \subseteq \ldots \subseteq \Sigma_n$, we construct a proof skeleton $SK1 = (Q, \{1, \ldots, n\}, q_0, T)$ as follows. T is the set of strictly increasing sequences of numbers in $\{1, \ldots, n\}$ (T is drawn with solid arcs in Fig. 6), $Q = T$ and $q_0 = \varepsilon$. For any τ in T and $i \in \{1, \ldots, n\}$, $\tau \cdot i = \tau' i$ where τ' is the largest prefix of τ formed of integers strictly smaller than i (see again Fig. 6). As $(\simeq_j \circ \simeq_i) \subseteq \simeq_i$ for $i \leq j$, $SK1$ conforms to Def. 9. Finally, for any $w \in C(L, \mathcal{S})$, by Prop. 5, there must exist a map $\lambda : Q \to L$, *i.e.* $\lambda : T \to L$, such that $\lambda(\varepsilon) = w$ and for all $\tau j \in T$, $\lambda(\tau) \simeq_j \lambda(\tau j) \wedge \lambda(\tau j) \notin S_j$. □

Proposition 7 *If the secrets S_i form a chain for the inclusion, then the enabledness of all trajectories $w \in C(L, \mathcal{S})$ may be shown with a single proof skeleton $SK2$.*

Proof: Given the chain $S_1 \subseteq S_2 \subseteq \ldots \subseteq S_n$, we construct a proof skeleton $SK2 = (Q, \{1, \ldots, n\}, q_0, T)$ as follows. T is the set of strictly increasing sequences of numbers in $\{1, \ldots, n\}$ (T is drawn with solid arcs in Fig. 7), $Q = T$ and $q_0 = \varepsilon$. For any τ in T and $i \in \{1, \ldots, n\}$, $\tau \cdot i = \tau i$ if $\tau i \in T$ and $\tau \cdot i = \tau$ otherwise (see again Fig. 7). If $i \leq j$, then for any τj in T ($= Q$), and for any map $\lambda : Q \to L$, $\lambda(\tau j) \notin S_j \Rightarrow \lambda(\tau j \cdot i) \notin S_i$ since $S_i \subseteq S_j$. Therefore, $SK2$ conforms to Def. 9, and the desired conclusion follows from Prop. 5. □

Proposition 8 *If for all distinct $i, j \in \{1, \ldots, n\}$, the secret S_i is saturated by the equivalence relation $\simeq_j$, then the enabledness of all trajectories $w \in C(L, \mathcal{S})$ may be shown with a single proof skeleton $SK3$.*

Proof: We construct a proof skeleton $SK3 = (Q, \{1, \ldots, n\}, q_0, T)$ as follows. T is the set of sequences in $\{1, \ldots, n\}^*$ with at most one occurrence of each number (T is drawn with solid arcs in Fig. 8), $Q = T$ and $q_0 = \varepsilon$. For any τ in T and $i \in \{1, \ldots, n\}$, $\tau \cdot i = \tau i$ if $\tau i \in T$ and $\tau \cdot i = \tau$ otherwise (see again Fig. 8). Let $\lambda : Q \to L$ be any map such that $\lambda(\tau) \simeq_j \lambda(\tau j) \wedge \lambda(\tau j) \notin S_j)$ whenever $\tau j \in T$. One may show by induction on τ that $\lambda(\tau) \notin S_i$ for any $i \in \{1, \ldots, n\}$ occurring in τ. Indeed, if this property holds for τ, it

must hold for τj because $\lambda(\tau) \simeq_j \lambda(\tau j)$ and $\simeq_j$ saturates S_i and $L \setminus S_i$ for all i occurring in τ. Therefore, $SK3$ conforms to Def. 9, and the desired conclusion follows from Prop. 5. □

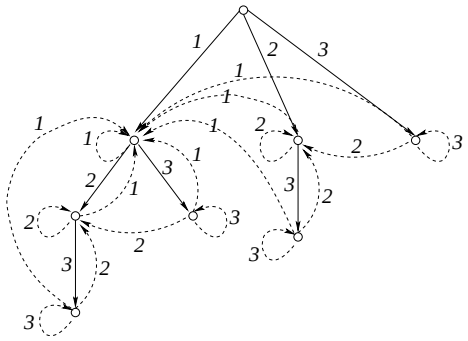

Figure 6: $SK1$ for $n = 3$

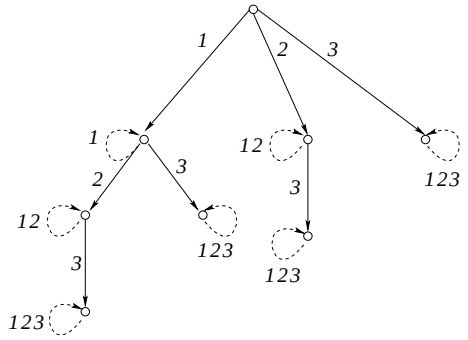

Figure 7: $SK2$ for $n = 3$

One can deal similarly with many other situations where $\Sigma_i \subseteq \Sigma_j$ or $S_i \subseteq S_j$ or $\simeq_i$ saturates S_j, or conversely with i and j, for all distinct $i, j \in \{1, \ldots, n\}$.

Example 4 *Let $n = 3$, and suppose $S_1 \subseteq S_2$, $\Sigma_3 \subseteq \Sigma_2$, and $\simeq_1$ saturates S_3. The enabledness of all trajectories $w \in C(L, \mathcal{S})$ may be shown using the proof skeleton $SK4$ (see Fig. 9).*

Unfortunately, we cannot extend propositions 6,7, and 8 into a general proposition, for we do not know whether $C(L, \mathcal{S})$ is regular in the following cases (the reader may verify that $C(L, \mathcal{S})$ is regular if none of these cases applies up to a permutation of indices):

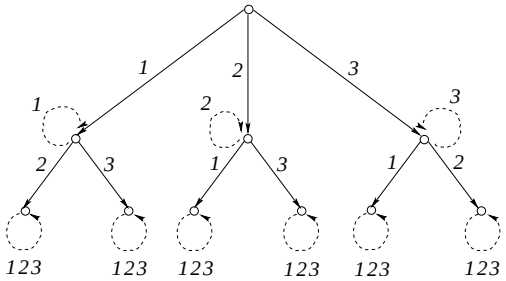

Figure 8: $SK3$ for $n = 3$

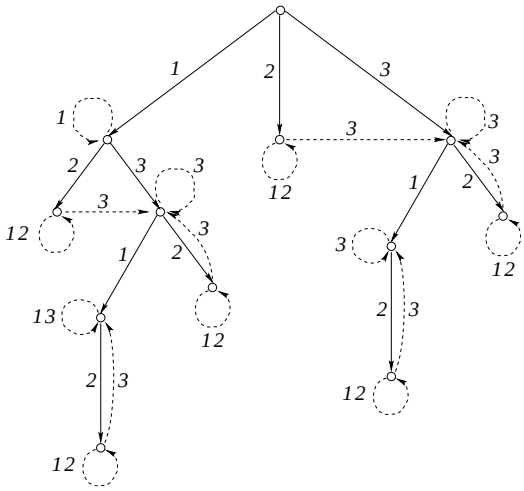

Figure 9: $SK4$

- $S_1 \subseteq S_2$, $\Sigma_2 \subseteq \Sigma_3$, and $\simeq_1$ saturates S_3,
- $S_1 \subseteq S_2$, $\simeq_2$ saturates S_3, and $\Sigma_3 \subseteq \Sigma_1$,
- $\simeq_1$ saturates S_2, $\simeq_2$ saturates S_3, and $\simeq_3$ saturates S_1.

The best we can do is therefore to propose an algorithm that constructs a unique skeleton for all proofs of enabledness in all cases where we know this is possible. In this perspective, we introduce three rewrite rules on labelled graphs. In each rule, one vertex of the left member is dropped and the edges that were incident to this vertex are redirected to other vertices. The vertices and edges present on both sides of a rule serve as an application context (indicated by the labels put on the concerned vertices). The rewrite rules are displayed in the following Fig. 10 (where $i \neq j$ and *sat* is an abbreviation for "saturates").

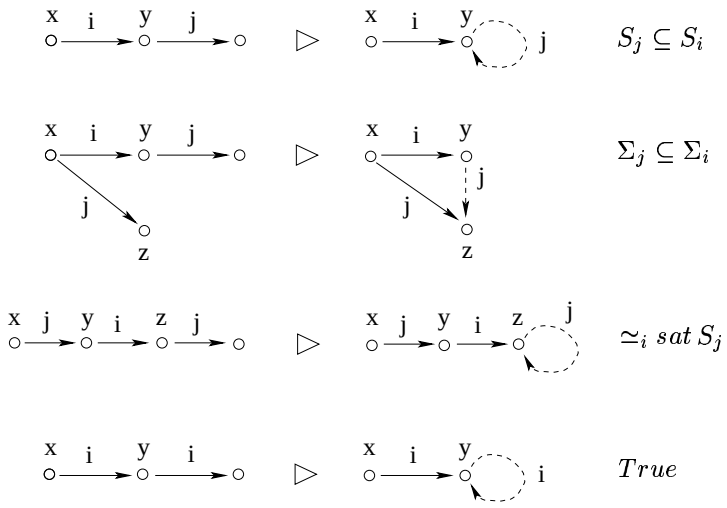

Figure 10: Four rules

Proposition 9 *Given $\mathcal{S} = \{(\Sigma_1, S_1), \ldots, (\Sigma_n, S_n)\}$, let $\mathcal{R}$ be the set of the rewrite rules that correspond to predicates true in $\mathcal{S}$. Whenever the complete n-ary tree rewrites to some finite graph, any such graph yields a uniform skeleton for proving the enabledness of all trajectories. The spanning tree of the skeleton is the subset of edges of the complete n-ary tree that have been preserved by the rewriting.*

Proof: In view of Def. 9 it is enough to show, for each graph G on the right hand side of a rewrite rule (see Fig. 10), that any map $\lambda : \{x, y\} \to L$ or $\lambda : \{x, y, z\} \to L$ compatible with the rigid edges of G is compatible also with the dashed edge of G, where λ is compatible with $x \xrightarrow{i} y$ if $\lambda(x) \simeq_i \lambda(y)$ and $\lambda(y) \notin S_i$. Considering the predicates defining the application conditions of the rewrite rules, this verification is immediate. □

Let us remark that $\mathcal{R}$ is neither confluent nor terminating. For instance, if $S_j \subseteq S_i$ and $\Sigma_j \subseteq \Sigma_i$, then the two rules $S_j \subseteq S_i$ and $\Sigma_j \subseteq \Sigma_i$ may be applied forever in alternation. Moreover, distinct derivatives are reached in a fixed number of steps according to the rule which is applied first. It may occur too that no finite graph may be derived from the complete n-ary tree, even though $S_j \subseteq S_i$ or $\Sigma_j \subseteq \Sigma_i$ or $\simeq_i sat S_j$ for all pairs (i,j). For instance, the infinite branch of the complete n-ary tree labelled with $(1\,2\,3)^\omega$ cannot be reduced under the assumptions $S_1 \subseteq S_2$, $\Sigma_2 \subseteq \Sigma_3$, and $\simeq_1 sat S_3$. Nevertheless, in any case where finite graphs may be derived from the complete n-ary tree, their spanning trees (pictured by plain edges) have a unique minimum. This follows from observing that the rule $\Sigma_j \subseteq \Sigma_i$ can still be applied to contexts where $x = z$ and j loops on x.

When proposition 9 can be applied, the construction proposed in the proof of proposition 5 may be used to produce a finite automaton realizing the maximal permissive opacity control, but Prop. 9 is not immediately effective. We remedy this deficiency in the end of the section.

Proposition 10 *It is decidable whether the complete n-ary tree $\mathcal{R}$-rewrites to some finite graph and such graphs can be computed.*

Proof: As a preliminary remark, note that $\mathcal{R}$ is not necessarily confluent, hence the finite graph we compute is just one among a set of possible proof skeletons.

Let $I = \{1, \ldots, n\}$ and let $F \subseteq I^*$ be the set of all words ii, or ij, or iji such that $True$, or $(S_j \subseteq S_i \vee \Sigma_j \subseteq \Sigma_i)$, or $\simeq_i sat S_j$, respectively. If the words in F are considered as forbidden factors for words in I^*, the remaining words form a regular language $T = I^* \setminus (I^* F I^*)$.

If T is infinite, the rewrite system $\mathcal{R}$ cannot terminate on the complete n-ary tree and it cannot produce any finite graph. If T is finite, let $(Q, \{1, \ldots, n\}, q_0)$ be the partial automaton defined with $Q = T$, $q_0 = \varepsilon$, and $\tau \cdot i = \tau i$ for τi in T.

To obtain a proof skeleton $(Q, \{1, \ldots, n\}, q_0, T)$ conforming Def. 9, it suffices now to complete the partial automaton $(Q, \{1, \ldots, n\}, q_0)$ as follows: for all words τ in T, and by increasing lengths of words τ,
- set $\tau i \cdot j = \tau i$ if $\tau i \cdot j$ is undefined and $S_j \subseteq S_i$ or $\simeq_i sat S_j$ and $\tau = \tau' \cdot j$,
- set $\tau i \cdot j = \tau \cdot j$ if $\tau i \cdot j$ is still undefined and $\Sigma_j \subseteq \Sigma_i$. □

Example 5 *Let $\Sigma = \{a, b, c\}$ and let $L = \downarrow (0 \leq \#c - \#a - \#b \leq 3)$ where $\downarrow P$ is the set of words with all prefixes in P, and $\#\sigma$ counts the occurrences of σ. Define $\mathcal{S} = \{(\Sigma_1, S_1), \ldots, (\Sigma_n, S_n)\}$ with $\Sigma_1 = \{a, c\}$, $\Sigma_2 = \{b, c\}$, $\Sigma_3 = \{b\}$, and $S_1 = \downarrow (\#c - \#a - \#b = 3)$, $S_2 = \downarrow (\#c - \#a \geq 3)$, $S_3 = \downarrow (\#a > 0)$. Note that $S_i = S_i \Sigma^*$ for all $i \in \{1, \ldots, 3\}$. The conditions of the example 4 are satisfied, and the construction sketched in the proof of proposition 10 yields the skeleton $SK4$ and the spanning tree T displayed in Fig. 9.*

$C(L, \mathcal{S})$ may be computed by stages following the structure of T. We compute first $C(L, \mathcal{S}) \setminus S_3$, using the skeleton that appears in $SK4$ at the end of both paths 13 and 3. Next, we compute $C(L, \mathcal{S}) \setminus S_1$ from $C(L, \mathcal{S}) \setminus S_3$, using the skeleton at the end of the path 1 in $SK4$. Finally, we compute $C(L, \mathcal{S})$ from $C(L, \mathcal{S}) \setminus S_1$ and $C(L, \mathcal{S}) \setminus S_3$.

By Prop. 9, $C(L,\mathcal{S}) \setminus S_3 =\downarrow P$ *where* P *is the predicate on* $\#a$, $\#b$, $\#c$ *shown in figure 11 (where* e.g. *the predicate on the node reached by the path* 1 *reads as* $(\exists b')((0 \leq \#c - \#a - b' \leq 2) \wedge (\exists a')((0 \leq \#c - a' - b' \leq 3) \wedge (\#c - a' \leq 2)))$. *This predicate is equivalent to* $\#a = 0 \wedge (0 \leq \#c - \#b \leq 3) \wedge \#c \leq 4 \wedge \#b \leq 2$. *Similarly,* $C(L,\mathcal{S}) \setminus S_1 =\downarrow Q$ *where* Q *is the predicate shown in figure 12. This predicate is equivalent to* $(0 \leq \#c - \#a - \#b \leq 2) \wedge \#b \leq 2$.

In view of Fig. 9, a word w *in* L *is in* $C(L,\mathcal{S})$ *if* $w \simeq_1 w_1 \in C(L,\mathcal{S}) \setminus S_1 \wedge w \simeq_2 w_2 \in (L \setminus S_2) \wedge w \simeq_1 w_3 \in C(L,\mathcal{S}) \setminus S_3$ *for some* w_1, w_2, w_3. *As* $\Sigma_1 = \{a, c\}$, $\Sigma_2 = \{b, c\}$, *and* $\Sigma_3 = \{b\}$, *the specified conditions hold if and only if* $0 \leq \#c - \#a \leq 4$, $0 \leq \#c - \#a \leq 2$, *and* $\#b \leq 2$, *respectively. Therefore,* $C(L,\mathcal{S}) =\downarrow R$ *where* $R \equiv (0 \leq \#c - \#a - \#b \leq 3) \wedge (\#c - \#a \leq 2) \wedge (\#b \leq 2)$. *A Petri net implementation of this control is shown in Fig. 13.*

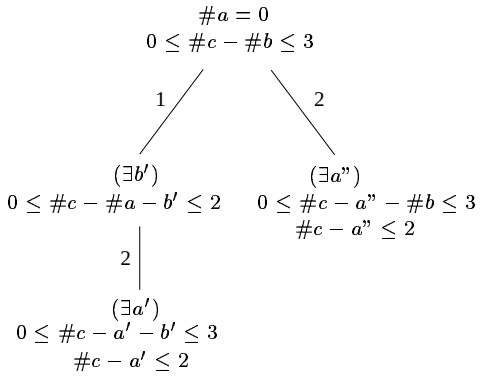

Figure 11: Predicate P

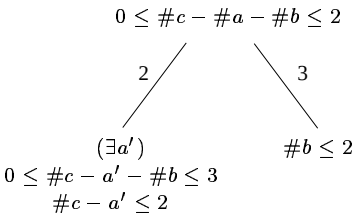

Figure 12: Predicate Q

6 Conclusion

As the example which was dealt with in section 6 has more technical than practical interest, we shall try first in this section to better illustrate the possible applications of the work we

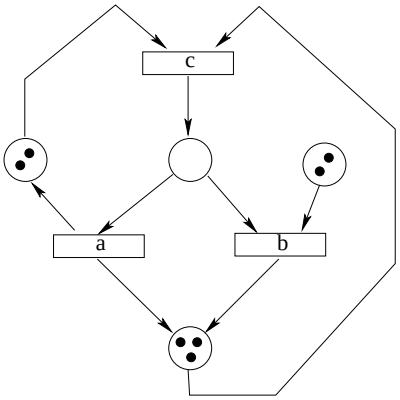

Figure 13: The Petri net controller

have presented. Consider a computer system that provides services to n users $U1, \ldots, Un$ with disjoint alphabets $\Sigma_1, \ldots, \Sigma_n$. Let $L \subseteq \Sigma^*$ be the language of the system, where $\Sigma \supseteq \Sigma_i$ for all i. One wants to give every user Ui the guarantee that no coalition of other users can ever know that he has started working. The problem is therefore to enforce the opacity of the concurrent secret $\mathcal{S} = \{(\Sigma'_1, S_1), \ldots, (\Sigma_n, S'_n)\}$ where for each i, $S_i = L \cap \Sigma^* \Sigma_i \Sigma^*$ and $\Sigma'_i = \cup_{j \neq i} \Sigma_j$. As $\simeq_j$ saturates S_i for $j \neq i$, one can construct a finite automaton accepting $C(L, \mathcal{S})$. We feel this example is typical of many practical security problems.

Some limitations of this work are voluntary, *e.g.* we restricted ourselves on purpose to regular languages and to regular control, but some other limitations could hopefully be lifted in continuations of this work. A list follows.

From the beginning of section 4, we worked with open secrets, *i.e.* secrets S_i such that $S_i \Sigma^* \subseteq S_i$. The goal was to make the definition Def. 3 equivalent to the simpler definition Def. 5. Another way to obtain this equivalence is to impose on each secret S_i the following condition, where $\leq$ is the order prefix:
$(\forall w \in L \setminus S_i)\ \ \pi_i(w) = u\sigma \Rightarrow (\exists v \in L \setminus S_i)\ \ (v \leq w \wedge \pi_i(v) = u)$. Such secrets may *e.g.* carry the information that some system process *is* in a critical section.

As regards the control objective, we focussed our efforts on opacity, but we did not take the deadlock freeness or the liveness of the controlled system into consideration and this is a shortcoming. Another valuable extension would be to work with boolean combinations of opacity predicates, *e.g.* if S_1 is opaque w.r.t. Σ_1 then S_2 is not opaque w.r.t. Σ_2.

We end with a few words on observability and controlability. On the side of the observation functions, we have restricted our attention to projections on subalphabets, but it would be more adequate to accomodate also all alphabetic morphisms. As regards control, we dealt with all events as controlable events, but it would be more realistic to accomodate also incontrolable events.

References

[1] A. Arnold and M. Nivat, Comportements de processus. In *Actes du Colloque AFCET "Les mathématiques de l'informatique"*, 1982, pp.35-68.

[2] J. Berstel, *Transductions and Context-Free Languages*. Teubner Verlag, 1978.

[3] J.W. Bryans, M. Koutny, L. Mazaré and P.Y.A. Ryan, Opacity Generalised to Transition Systems. In *Proceedings of the Workshop on Formal Aspects in Security and Trust (FAST 2005)*, 2005.

[4] L. Mazaré, Using unification for opacity properties. In *Proceedings of the Workshop on Issues in the Theory of Security (WITS'04)*, 2004.

[5] P.J. Ramadge and W.M. Wonham, Supervisory Control of a Class of Discrete Event Processes, *SIAM Journal of Control and Optimization*, vol. 25, 1987, pp 206-230.

[6] P.J. Ramadge and W.M. Wonham, On the Supremal Controllable Language of a Given Language, *SIAM Journal of Control and Optimization*, vol. 25, 1987, pp 637-659.

[7] A. Tarski, A lattice-theoretical fixpoint theorem and its applications, *Pacific Journal of Mathematics*, vol. 5, 1955, pp. 285-309.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399