Enforcing Opacity of Regular Predicates on Modal Transition Systems

Philippe Darondeau, Hervé Marchand, Laurie S. L. Ricker

Noname manuscript No.
(will be inserted by the editor)

Enforcing Opacity of Regular Predicates on Modal Transition Systems

Philippe Darondeau
Hervé Marchand · Laurie Ricker

Received: date / Accepted: date

Abstract Given a labelled transition system G partially observed by an attacker, and a regular predicate Sec over the runs of G, enforcing opacity of the secret Sec in G means computing a supervisory controller K such that an attacker who observes a run of the controlled system K/G cannot ascertain that the trace of this run belongs to Sec based on the knowledge of G and K. We lift the problem from a single labelled transition system G to the class of all labelled transition systems specified by a *Modal Transition System* M. The lifted problem is to compute the maximally permissive controller K such that Sec is opaque in K/G for every labelled transition system G which is a model of M. The situations of the attacker and of the controller are asymmetric: at run time, the attacker may fully know G and K whereas the controller knows only M and the sequence of actions executed so far by the unknown G. We address the problem in two cases. Let Σ_a denote the set of actions that can be observed by the attacker, and let Σ_c and Σ_o denote the sets of actions that can be controlled and observed by the controller, respectively. We provide optimal and regular controllers that enforce the opacity of regular secrets when $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma_a = \Sigma$. We provide optimal and regular controllers that enforce the opacity of regular upper-closed secrets ($Sec = Sec.\Sigma^*$) under the following assumptions: (i) $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_o = \Sigma$ or (ii) $\Sigma_a, \Sigma_c \subseteq \Sigma_o = \Sigma$ and $w\sigma \in Sec \Rightarrow w \in Sec$ for all $\sigma \in \Sigma \setminus \Sigma_c$.

Keywords Partial Observation · Opacity · Modal Automata · Supervisory Control.

Ph. Darondeau
INRIA Rennes - Bretagne Atlantique, France.

H. Marchand
INRIA Rennes - Bretagne Atlantique, France. {herve.marchand}@inria.fr

L. Ricker
Department of Mathematics and Computer Science, Mount Allison University, Sackville NB, Canada. lricker@mta.ca

1 Introduction

The concept of opacity, first introduced in the context of sessions of security protocols [13, 12], was later extended to transition systems [3]. A predicate over the runs of a transition system is opaque w.r.t. an observation function if every observation produced by a run that satisfies the predicate is also produced by some run that does not satisfy the predicate. The concept of opacity is very flexible as it depends both on the class of predicates and on the observation function. By adjusting these two parameters, many common security properties (e.g., confidentiality, anonymity) can be rephrased in terms of opacity [3, 10]. In general, opacity is undecidable but this property may be checked effectively when it is applied to regular predicates on runs of finite transition systems and with observation functions induced by projection operators. Algorithms for checking opacity in Discrete Event Systems (DES) are presented together with applications in [18, 20, 10].

An active and hot topic at the frontier of the theories of security and DES is the search for supervisory controllers that enforce the opacity of a predicate on a given transition system. As noted in [8], long-term motivation for such work may be found in the need to protect SCADA systems and networks of sensors and actuators from interference with malicious agents through TCP/IP. If the sequences of messages sent from the SCADA system allows an attacker to know that the system, e.g., a power grid, is close to becoming unstable, then a sudden increase of load, e.g., a surge of power demand, can lead to disastrous consequences. However, until now the literature has dealt exclusively with finite transition systems. Approaches differ by considering either state-based opacity, e.g., initial-state opacity [18, 17] or current-state opacity [8], or language-based opacity [1, 2, 5, 6, 10, 20, 21]. With state opacity, the secret predicate bears either upon the initial state, or upon the current state, or upon the set of all states that have been visited from the beginning of a run. With language opacity, the secret predicate is a set of sequences of actions that label transitions. Language opacity and current-state opacity are mutually reducible. Approaches also differ upon whether synthesis algorithms or closed formulas or both are provided for maximally permissive controllers enforcing opacity. Closed formulas are proposed in [2, 21, 20]. *In fine*, all approaches rely on Ramadge and Wonham's basic theory of supervisory control for DES [15, 14, 16]. Significant adaptations must, however, be brought to the basic theory, because opacity objectives do not reduce to safety and liveness. In fact, opacity objectives are not concerned with individual runs but with sets of indiscernible runs from the perspective of the attacker. Classes of indiscernible runs may be captured by estimators, as is usually done for the purpose of diagnosis. Closely related to the concept of opacity is the concept of *secrecy* (both a predicate and the complement of this predicate have to be kept secret). In [20] conditions under which secrecy can be ensured are provided. In [19], controller synthesis algorithms are proposed for initial state-opacity and infinite-step opacity. Finally, although not specifically concerned with con-

trol, the aim of [22] is to enforce opacity by inserting additional observable events in the system's output behavior.

In this paper, we lift the opacity enforcing control problem from finite transition systems to families of finite transition systems specified by modal transition systems. Modal transition systems were introduced in [9] as tuples $(S, \Sigma, \rightarrow_\Box, \rightarrow_\Diamond, s_0)$ with two modal transition relations $\rightarrow_\Box$ (the *strong* or *must* transition relation) and $\rightarrow_\Diamond$ (the *weak* or *may* transition relation), both included in $S \times \Sigma \times S$ and subject to the inclusion constraint $\rightarrow_\Box \subseteq \rightarrow_\Diamond$. A Modal Transition System (MTS) should be understood as a logical formula, with labelled transition systems as models. Modal transition systems are indeed a well-identified fragment of the modal μ-calculus [7]. Intuitively, a Labelled Transition System (LTS) is a model of an MTS if there exists a relation $\models$ between their respective sets of states Q and S such that $q_0 \models s_0$ holds for the initial states and whenever $q \models s$, for $q \in Q$ and $s \in S$, all *must* transitions from s are simulated by transitions from q, all transitions from q are simulated by *may* transitions from s and $\models$ is preserved under simulation of transitions in both directions.

Example 1 The modal transition systems M_1 and M_2 depicted in Figure 1, where the relations $\rightarrow_\Box$ and $\rightarrow_\Diamond$ are represented with solid lines and dashed lines, respectively, state that the presence of the first transition a is mandatory in any model of M_2 while it is optional in models of M_1. The second transition a is optional for both MTS, and any model of M_1 or M_2 must be able to perform a b after the sequence $a.a$ has been executed. The presence of a second transition b (returning to the initial state of M_1 or M_2) is optional in models of M_1 or M_2.
The two LTS depicted in Figure 1(c) and 1(d) are models of M_1, whereas they are not models of M_2. Indeed, after the initial transition a, M_2 requires a transition b, which is missing in G_1 and G_2. $\diamond$

We frequently use systems without an exact knowledge of their behaviour. This is generally the case when the system belongs to a range of products with many versions, such as applications for smart phones or software, and even more so for software with automatic updates. This is also the case when the system is a web service orchestration, selected on request by a broker to match operating guidelines specified in the request [11]. For instance, when you use a web service orchestration to buy goods online and find yourself short of funds to complete your transaction, you may want this information to be opaque to the orchestrator. Opacity may be obtained, e.g., by letting the online payment service interact with the orchestrator by the same return event when an incorrect PIN has been entered or when a bank account balance is too low for payment. In such situations, modal transition systems may serve to represent the partial knowledge of the user on the possible behaviours of the system (modal transition systems with final states, introduced in [4], are in fact a restricted form of the operating guidelines of [11]). Enforcing opacity of regular predicates on modal transition systems may then serve to prevent con-

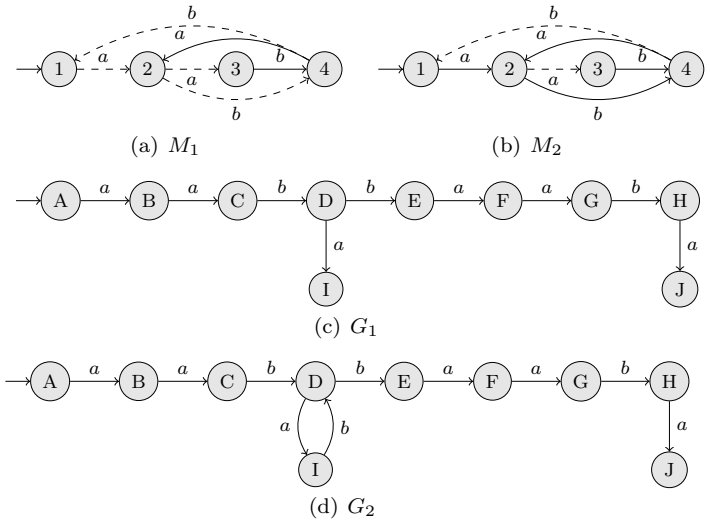


Fig. 1 Two modal transition systems $M_1,\ M_2$ and two labelled transition systems $G_1,\ G_2$.

fidential user information from being leaked by the partially-unknown system which they actually use.

The purpose of this paper differs from the purpose of our earlier paper [4]. In [4], the goal was to enforce specifications of service, expressed by modal transition systems, on service providers, modelled by LTS. Here the goal is to enforce the opacity of a secret predicate on all models LTS of a modal transition system MTS.

The rest of the paper is organized as follows. First, we recall briefly the background of modal transition systems and supervisory control for opacity, and we state the opacity enforcement problem for modal transition systems. The parameters of the problem are the secret predicate, the subset of actions Σ_a that the attacker can observe, and the subsets of actions Σ_o and Σ_c that the controller can observe and control, respectively. Then, we address the opacity enforcement problem for regular secrets in the most straightforward case $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma_a = \Sigma$. In the main and last technical section of the paper, we provide optimal and regular controllers that enforce the opacity of regular upper-closed secrets ($Sec = Sec.\Sigma^*$) for modal transition systems under the following assumptions:

(i) $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_o = \Sigma$ or
(ii) $\Sigma_a, \Sigma_c \subseteq \Sigma_o = \Sigma$ and $w\sigma \in Sec \Rightarrow w \in Sec$ for all $\sigma \in \Sigma \setminus \Sigma_c$.

2 Transition Systems and opacity

In this section we recall the background of labelled transition systems and opacity.

2.1 Labelled Transition Systems and their languages

A (finite) deterministic *labelled transition system* over Σ is a 4-tuple $G = (Q, \Sigma, \delta, q_0)$ where Q is a finite set of states, $q_0 \in Q$ is an *initial state*, Σ is the alphabet of actions, and δ is a partial map from $Q \times \Sigma$ to Q, called the *labelled transition map*. This map is extended inductively to $\delta : Q \times \Sigma^* \to Q$ by letting $\delta(q, \varepsilon) = q$ (where ε is the empty word) and $\delta(q, w.\sigma) = \delta(\delta(q, w), \sigma)$ for all $q \in Q$, $w \in \Sigma^*$ and $\sigma \in \Sigma$ ($w.\sigma$ denotes the word acquired by appending σ to w. In the sequel, we let $w.w'$ denote the concatenation of the words w and w' and we let $w.L' = \{w.w' \,|\, w' \in L'\}$ and $L.L' = \{w.w' \,|\, w \in L \ \wedge \ w' \in L'\}$. A state $q \in Q$ is *reachable* (from q_0) if $\delta(q_0, w) = q$ for some word $w \in \Sigma^*$. G is *finite* if Q and Σ are finite; it is *reduced* if all states in Q are reachable and every event $\sigma \in \Sigma$ is enabled at some state q, i.e., $\delta(q, \sigma)$ is defined in this state. In the sequel, we consider only finite and reduced labelled transition systems. G is said to be complete if $\delta(q, \sigma)$ is defined for all $q \in Q$ and $\sigma \in \Sigma$.

The *language* of $G = (Q, \Sigma, \delta, q_0)$ is the set of words

$$\mathcal{L}(G) = \{w \in \Sigma^* \,|\, \delta(q_0, w) \text{ defined}\}.$$

For $q \in Q$, we let $\mathcal{L}(G, q)$, the language generated by G from state q, be defined as:

$$\mathcal{L}(G, q) = \{w \in \Sigma^* \,|\, \delta(q, w) \text{ defined}\}.$$

For $F \subseteq Q$, we let $\mathcal{L}_F(G)$, the set of sequences recognized by states in F, be defined as:

$$\mathcal{L}_F(G) = \{w \in \Sigma^* \,|\, \delta(q_o, w) \in F\}$$

Given a language $L \subseteq \Sigma^*$ and a sub-alphabet $\Sigma_a \subseteq \Sigma$, the *natural projection* of L on Σ_a^* is the language $\pi_a(L) \subseteq \Sigma_a^*$ equal to $\{\pi_a(w) \,|\, w \in L\}$ where π_a is the operation from Σ^* to Σ_a^* that erases in words of Σ^* all events not in Σ_a. Formally, π_a is defined inductively by:

- $\pi_a(\varepsilon) = \varepsilon$ (the empty word),
- $\pi_a(w.\sigma) = \pi_a(w).\sigma$ for $w \in \Sigma^*$ and $\sigma \in \Sigma_a$,
- $\pi_a(w.\sigma) = \pi_a(w)$ for $w \in \Sigma^*$ and $\sigma \notin \Sigma_a$.

For $w,\ w' \in \Sigma^*$, we let $w \sim_a w'$ be an abbreviation for $\pi_a(w) = \pi_a(w')$.

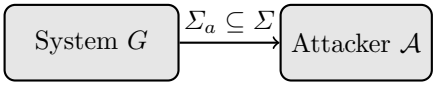
Finally, given transition systems $G = (Q, \Sigma, \delta, q_0)$ and $G' = (Q', \Sigma, \delta', q_0')$ labelled over the same alphabet Σ, their *product* is the (reachable restriction of the) labelled transition system $G \times G' = (Q \times Q', \Sigma, \delta \times \delta', (q_0, q_0'))$ where

$$(\delta \times \delta')((q, q'), \sigma) = (\delta(q, \sigma), \delta'(q', \sigma)).$$

2.2 Opacity

Consider an LTS $G = (Q, \Sigma, \delta, q_0)$ over Σ and $\Sigma_a \subseteq \Sigma$. The alphabet $\Sigma_a \subseteq \Sigma$ defines the set of actions that the *attacker* can observe. Let $Sec \subseteq \Sigma^*$ be a regular predicate, called the *secret*, that represents confidential information

on the execution of G. To catch this confidential information, the attacker is armed with full knowledge on the structure of G but only partial knowledge of its dynamic behavior, namely the information afforded by the natural projection on Σ_a^* of the actual sequence in Σ^* generated by G. In this framework, we assume that the attacker passively observes the system and hence does not interact with it, i.e., the flow information is as follows:



For secret predicates given as regular languages, the definition of opacity introduced in [3] may be adapted as follows.

Definition 1 *Given a secret $Sec \in Reg(\Sigma^*)$, Sec is* opaque *in G w.r.t. Σ_a if, $\forall w \in \mathcal{L}(G) \cap Sec$, $\exists w' \in \mathcal{L}(G) \setminus Sec$ such that $w \sim_a w'$.*

Intuitively, a secret is opaque if every secret word w is observationally equivalent to at least one non secret word $w'$of the system. If Sec is not opaque then there is an *information flow* from the system to the attacker.

Definition 2 Given a word $w \in Sec \cap \mathcal{L}(G)$, *$w$ discloses the secret Sec* if

$$w \sim_a w' \Rightarrow w' \in Sec \text{ for all words } w' \in \mathcal{L}(G).$$

Example 2 Let G be the LTS of Figure 2, where $\Sigma = \{h, p, a, b\}$, $\Sigma_a = \{a, b\}$.

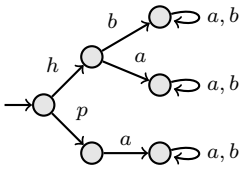


Fig. 2 Breaking the secrecy of h.

The secret that should not be revealed is the occurrence of the (unobservable) action h (namely, $Sec = \Sigma^*.h.\Sigma^*$). As $h.b \in Sec$ is the sole word in $\mathcal{L}(G)$ compatible with the partial observation b, $h.b$ discloses the secret Sec, hence Sec is not opaque w.r.t. G and Σ_a. ◇

2.3 Supervisory Control for Opacity

Given a Secret Sec on the system G, the goal of supervisory control is to enforce the secrecy of Sec on G by pairing this system with a *supervisory controller* modeled by an LTS that observes a subset Σ_o of the actions in Σ and controls a subset Σ_c of the actions in Σ following the scheme of Figure 3. Thus, *enforcing* the opacity of the secret Sec w.r.t. Σ_a in G means computing

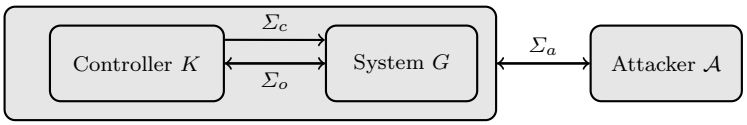


Fig. 3 Control Architecture

a supervisory controller K such that Sec is opaque w.r.t. Σ_a in the product $G \times K$[1], called the *controlled system* and usually written K/G.

In Ramadge and Wonham's setting for supervisory control [15, 14, 16], an *admissible* controller K may be seen as an LTS $K = (X, \Sigma, \delta_K, x_0)$, subject to constraints parametric on two subsets of actions Σ_c and Σ_o.

1. *Controllability constraint*: the first set Σ_c is comprised of the *controllable* actions that the controller can block or *control*. For any uncontrollable action $\sigma \in \Sigma \setminus \Sigma_c = \Sigma_{uc}$ and for any word w, if $\delta_K(x_0, w) = x$ and $w\sigma \in \mathcal{L}(G)$ then $\delta_K(x, \sigma)$ must be defined. In other words, we should have $\mathcal{L}(K).\Sigma_{uc} \cap \mathcal{L}(G) \subseteq \mathcal{L}(K)$.
2. *Observability constraint*: the second set Σ_o is comprised of the actions that the controller can *observe*. For any action $\sigma \notin \Sigma_o$ and for any state x in which $\delta_K(x, \sigma)$ is defined, it is required that $\delta_K(x, \sigma) = x$.

$K^\dagger$ is said to be *maximally permissive* among the controllers that enforce the opacity of Sec in G w.r.t. Σ_a if $\mathcal{L}(K/G) \subseteq \mathcal{L}(K^\dagger/G)$ for all such controllers K.

Next, we illustrate the (intuition behind the) computation of opacity enforcing controllers through a simple example.

Example 3 The system to be controlled is given in Figure 4 with $\Sigma = \{a, c_1, c_2, b, d, e, h, p\}$. *We assume that* $\Sigma_a = \{a, b, d, e\}$, $\Sigma_o = \{a, c_1, c_2, b, d, e\}$, *and* $\Sigma_c = \{b, c_1, c_2, e\}$ *(thus* $\Sigma_{uc} = \{a, d, e, h, p\}$*). The secret is given by the regular language* $Sec = \Sigma^*.h.\Sigma^*$. *When observing* d, *the attacker knows that* h

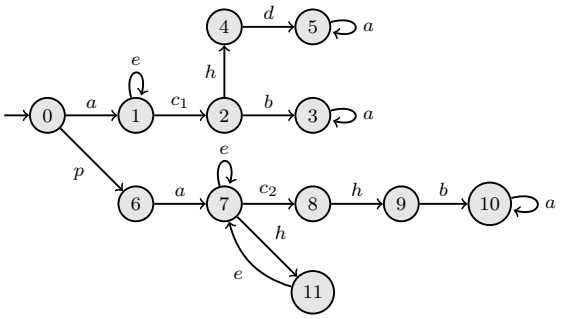
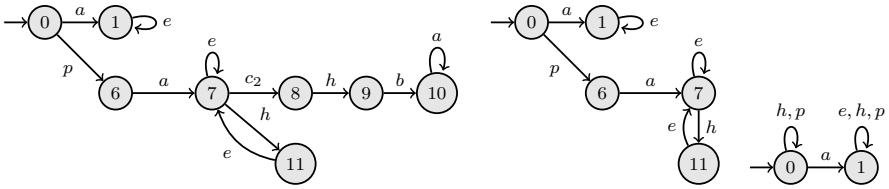


Fig. 4 Original system G.

[1] The language generated by $G \times K$ is given by $\mathcal{L}(K) \cap \mathcal{L}(G)$.

occurred and the secret is revealed (in state 5). By control, action c_1 has to be disabled, thus avoiding the triggering of the subsequent uncontrollable sequence h.d, and the LTS depicted in Figure 5(a) is obtained.



(a) Step 1 (b) Step 2 (c) Controller K

Fig. 5 Controller K Computation

However, after the observation of the action b, the secret is now revealed to the attacker who knows the control law. By control, action c_2 has to be disabled, giving the LTS of Figure 5(b). The secret is now opaque with respect to this LTS. The resulting controller K is depicted in Figure 5(c). ◇

The previous example suggests that to compute the controller K, it is sufficient to remove from G, using the standard Ramadge & Wonham algorithm, the sequences that reveal the secret, i.e. the language $\mathcal{L}(G) \setminus \pi_a^{-1} \circ \pi_a(\mathcal{L}(G) \setminus Sec)$ and to iterate the process until Sec is opaque w.r.t. the resulting LTS (iteration is mandatory as removing some behaviors of G by control might create new information flow). However, it has been shown in [5] that this iteration might not terminate, even though the maximally permissive controller $K^\dagger$ exists. Furthermore, knowing whether the maximal controller is regular under the sole assumption that $\Sigma_c \subseteq \Sigma_o$ is still an open question. Meanwhile, in [5], it was shown that there exists a maximally permissive and regular controller $K^\dagger$ in all cases where $\Sigma_c \subseteq \Sigma_o$ and Σ_a compares with Σ_c and Σ_o. In [6], more elaborate constructions were presented for computing $K^\dagger$ in the case where $\Sigma_c \subseteq \Sigma_o$ and $\Sigma_a \subseteq \Sigma_o$.

Alternatively, [21] provides a method for computing the supremal controlled system ensuring the opacity of a secret Sec when $\Sigma_o = \Sigma$ and $(\forall w, w' \in \mathcal{L}(G))(\forall \sigma \in \Sigma_{uc} \cap \Sigma_a)\, \pi_a(w) = \pi_a(w') \wedge w\sigma \in \mathcal{L}(G) \Rightarrow w'\sigma \in \mathcal{L}(G)$.

Note that [5] and [6] provide effective synthesis algorithms solving the opacity control problem (i.e., they compute the maximally permissive controller $K^\dagger$), whereas [21] provides closed formulae characterizing the behavior $K^\dagger/G$ without explicitly computing $K^\dagger$.

Remark 1 Closely related to the concept of opacity is the concept of secrecy *(both a predicate and the complement of this predicate have to be kept secret). In [20] conditions under which secrecy can be ensured are provided. Similarly, [19] provides algorithms to compute controllers ensuring initial-state opacity (preventing the attacker from inferring one of the initial states of the system after some observation) as well as infinite-step opacity (preventing the attacker*

from inferring that the system was in the secret in the past after some fixed number of steps).

3 Modal Transition Systems and opacity control problem

In this paper, we make a first step towards extending the results of section 2.3 to systems modeled by modal transition systems (MTS). We first introduce the formal definition of MTS and then describe the opacity control problem in this setting.

3.1 Modal Transition Systems [9]

Definition 3 *A deterministic* modal transition system *(or MTS) over Σ is a 5-tuple $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$ where*

- *S is a finite set of* logical states *with s_0 the initial state,*
- *Σ is the alphabet of actions*
- *$\delta^{\Box} : S \times \Sigma \to S$ and $\delta^{\Diamond} : S \times \Sigma \to S$ are two partial maps, called the* strong *and the* weak *labelled transition maps, respectively, subject to the constraint $\delta^{\Box} \subseteq \delta^{\Diamond}$.*

The maps $\delta^{\Box}$ and $\delta^{\Diamond}$ are extended inductively to words as was done with the transition maps of labelled transition systems. For any modal transition system M, we let $\mathcal{L}(M) = \mathcal{L}(\overline{M})$ where $\overline{M} = (S, \Sigma, \delta^{\Diamond}, s_0)$, thus $\overline{M}$ denotes the LTS whose transition map is the weak transition map of M. Similarly, $\underline{M} = (S, \Sigma, \delta^{\Box}, s_0)$ denotes the LTS whose transition map is the strong transition map of M.

A modal transition system M determines a family of labelled transition systems G called its *models* (notation: $G \models MTS$).

Definition 4 *A labelled transition system $G = (Q, \Sigma, \delta, q_0)$ is a model of $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$ if there exists a relation $\models \subseteq Q \times S$ such that $q_0 \models s_0$ and for all $q \in Q$ and $s \in S$, $q \models s$ entails the following for all $\sigma \in \Sigma$:*

- *if $\delta(q, \sigma)$ is defined then $\delta^{\Diamond}(s, \sigma)$ is defined and $\delta(q, \sigma) \models \delta^{\Diamond}(s, \sigma)$,*
- *if $\delta^{\Box}(s, \sigma)$ is defined then $\delta(q, \sigma)$ is defined and $\delta(q, \sigma) \models \delta^{\Box}(s, \sigma)$.*

Example 4 In example 1, $G_1 \models M_1$ as we can build a relation $\models \subseteq Q \times S$ equal to $\{(A,1),(B,2),(C,3),(D,4),(E,1),(F,2),(G,3),(H,4),(I,2),(J,2)))\}$ that fulfills the conditions of Definition 4. ◇

With these definitions, it can be shown that

$$\underline{M} \models M,\ \overline{M} \models M,\ G \models M \Rightarrow \mathcal{L}(\underline{M}) \subseteq \mathcal{L}(G) \subseteq \mathcal{L}(\overline{M}).$$

Therefore,

$$\mathcal{L}(\underline{M}) = \bigcap \{\mathcal{L}(G) \mid G \models M\} \text{ and } \mathcal{L}(\overline{M}) = \bigcup \{\mathcal{L}(G) \mid G \models M\}.$$

$\mathcal{L}(\underline{M})$ and $\mathcal{L}(\overline{M})$ are called the infimum and supremum of $\mathcal{L}(G)$, respectively, for all $G \models M$ However, $\underline{M}$ and $\overline{M}$ are not the unique models of M with minimal or maximal language, respectively, since there may exist other LTS with the same language.

A central property of modal transition systems is stated by the following relation:

$$G_1 \models M \wedge G_2 \models M \Rightarrow G_1 \times G_2 \models M.$$

Furthermore, it can be shown that given two LTS G_1 and G_2 such that $\mathcal{L}(G_1) = \mathcal{L}(G_2)$ and an MTS M, if $G_1 \models M$ then $G_2 \models M$. This result holds because we only consider deterministic labelled transition systems.

We refer the reader to [9, 7] for more information on the theory of the modal transition systems.

In addition to these reminders, we introduce now a specific construction used in later proofs. Given a modal transition system M, for each word $w \in \mathcal{L}(M)$ we want to construct a labelled transition system $w \circ M$ such that $\mathcal{L}(w \circ M)$ is the infimum of $\mathcal{L}(G)$ for all labelled transition systems G satisfying $G \models M$ and $w \in \mathcal{L}(G)$.

Definition 5 *Given $w = \sigma_1 \dots \sigma_n \in \mathcal{L}(M)$ where $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$, let $w \circ M$ denote the LTS produced by the following procedure, where $s_i = \delta^{\Diamond}(s_0, \sigma_1 \dots \sigma_i)$ for $1 \leq i \leq n$:*

- *make $n+1$ separate copies of the set of states S with elements (s, i), $s \in S$ and $0 \leq i \leq n$,*
- *for $1 \leq i \leq n$, let $\delta((s_{i-1}, i-1), \sigma_i) = (s_i, i)$,*
- *for $0 \leq i \leq n$ and for all pairs $(s, \sigma) \in S \times \Sigma$ such that $i = n$ or ($s \neq s_i$ or $\sigma \neq \sigma_{i+1}$), let $\delta((s, i), \sigma) = (\delta^{\Box}(s, \sigma), i)$,*
- *let $(s_0, 0)$ be the initial state and δ be the partial transition map,*
- *remove all unreachable states.*

Example 5 To illustrate this construction, let us consider the word a.a. Then $(a.a) \circ M_1$ is given by the following LTS for which only the reachable part is kept. ◇

Lemma 1 *For $w = \varepsilon$ (the empty word), $\varepsilon \circ M$ is isomorphic to $\underline{M}$. For any other word w, the LTS $w \circ M$ enjoys the properties $w \circ M \models M$ and $w \in \mathcal{L}(w \circ M)$.*

Lemma 2 *For any word $w.\sigma \in \mathcal{L}(M)$ with $w \in \Sigma^*$, $\sigma \in \Sigma$ and $\delta^{\Diamond}(s_0, w.\sigma) = s$, the language of the labelled transition system $(w.\sigma) \circ M$ is equal to $\mathcal{L}(w \circ M) \cup w.\sigma.\mathcal{L}(\underline{M}, s)$.*

Proposition 1 $\mathcal{L}(w \circ M) = \bigcap\{\mathcal{L}(G) \mid G \models M \wedge w \in \mathcal{L}(G)\}$.

Proof. In view of Lemma 1, it suffices to show that $G \models M \wedge w \in \mathcal{L}(G) \Rightarrow \mathcal{L}(w \circ M) \subseteq \mathcal{L}(G)$. The proof is by induction on w. For $w = \varepsilon$, this holds since $\mathcal{L}(\varepsilon \circ M) = \mathcal{L}(\underline{M})$. For any other word $w.\sigma \in \mathcal{L}(M)$ with $\sigma \in \Sigma$, by Lemma 2, $\mathcal{L}((w.\sigma) \circ M) = \mathcal{L}(w \circ M) \cup w.\sigma.\mathcal{L}(\underline{M}, s)$. By induction, $\mathcal{L}(w \circ M) \subseteq$

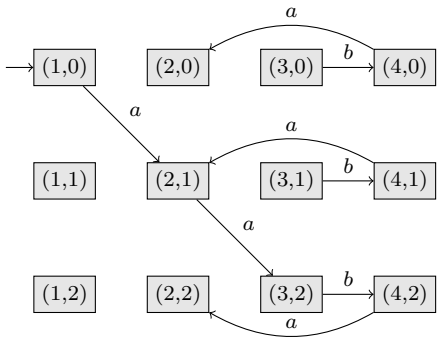


Fig. 6 $(a.a) \circ M_1$

$\bigcap\{\mathcal{L}(G) \mid G \models M \wedge w \in \mathcal{L}(G)\} \subseteq \bigcap\{\mathcal{L}(G) \mid G \models M \wedge w.\sigma \in \mathcal{L}(G)\}$. By definition of the relation $\models$, $G \models M \wedge w.\sigma \in \mathcal{L}(G) \Rightarrow w.\sigma.\mathcal{L}(\underline{M}, s) \subseteq \mathcal{L}(G)$ for any labelled transition system G. Hence $\mathcal{L}((w.\sigma) \circ M) \subseteq \mathcal{L}(G)$ and the proof is complete. □

Based on Proposition 1, we can state the following lemma:

Lemma 3 *Given $Sec \in Reg(\Sigma^*)$ and $w \in Sec$, w discloses the secret Sec in some model G of M if and only if w discloses the secret Sec in $w \circ M$.*

3.2 The Opacity Control Problem for Modal Transition Systems

From now on, $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$ is a fixed modal transition system, and Sec is a fixed regular subset of Σ^*, called the secret. Let Σ_a be the subset of actions in Σ that can be observed by the attacker. Let Σ_o and Σ_c be the subsets of actions in Σ that may be observed or blocked by the controller, respectively.

Definition 6 *$K = (X, \Sigma, \delta_K, x_0)$ enforces the opacity of Sec in M w.r.t. Σ_a if for every labelled transition system G over Σ such that $G \models M$,*

1. *K is an admissible controller of G (w.r.t. Σ_o and Σ_c) and,*
2. *Sec is opaque in K/G (w.r.t. Σ_a).*

As for permissivity, it would not make any sense to require that $K^\dagger$ be maximally permissive for every model G of M (among the controllers K that enforce the opacity of Sec in G w.r.t. Σ_a). In the framework of opacity control for modal transition systems, we shall consider the following definition:

Definition 7 *$K^\dagger$ is* maximally permissive *if $\mathcal{L}(K/G) \subseteq \mathcal{L}(K^\dagger/G)$ for every controller K that enforces the opacity of Sec in M (w.r.t. Σ_a) and for every model G of M.*

With regard to the above definitions, the opacity control problem can be stated as follows:

Problem 1 *Given a modal transition system M, a regular secret $Sec \subseteq \Sigma^*$, $\Sigma_a \subseteq \Sigma$ (the actions observed by the attacker), $\Sigma_o \subseteq \Sigma$ and $\Sigma_c \subseteq \Sigma$ (the observable and controllable actions of the controller, respectively), build a maximally permissive controller $K^\dagger$ enforcing the opacity of Sec in M w.r.t. Σ_a.*

Let us illustrate the opacity control problem and the notion of permissivity through an example:

Example 6 Consider the MTS M_1 of Example 1 and assume that the secret is given by the regular language $Sec = \Sigma^*.a.a.b.b.\Sigma^*$ and that $\Sigma_a = \Sigma_c = \{a\}$. Sec is opaque w.r.t. G_2 and Σ_a but not opaque w.r.t. G_1 and Σ_a since by observing $a.a.a.a$, the attacker knows that the actual word executed is either $a.a.b.b.a.a$ or $a.a.b.b.a.a.b$, both of which are in Sec. Now let us consider the controller K depicted in Figure 7. It can be shown that K is an admissible

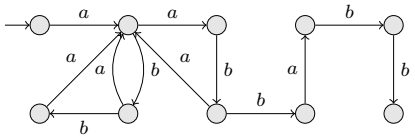
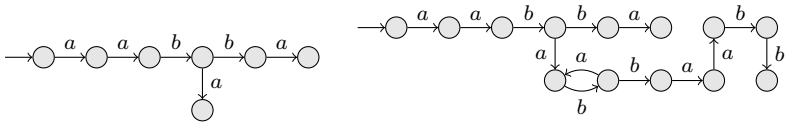


Fig. 7 Admissible controller K

controller ensuring the opacity of Sec in G_1 and G_2 (we will show in Section 4 that K is indeed the maximally permissive controller enforcing the opacity of Sec in M_1). The resulting controlled LTS K/G_1 and K/G_2 are depicted in Figure 8. Intuitively, in G_1, K disables the fourth occurrence of a in $a.a.b.b.a.a$



(a) K/G_1 (b) K/G_2

Fig. 8 Controlled LTS

since this occurrence of a would leak the secret. The controller K is maximally permissive for enforcing the opacity of Sec in G_1 (observing three occurrences of a does not reveal Sec). In G_2, K disables the second a in $b.a.a.b$ if $b.a.b$ has not be observed so far. However, the controller K is not the maximally permissive controller enforcing the opacity of Sec in G_2, as Sec was already opaque for G_2. Overall, as the controller does not know whether the actual plant is G_1 or G_2 (although the attacker knows it), the controller has to always disable the fourth a in the word $a.a.b.b.a.a$. ◇

The previous example illustrates the fact that if the controller does not know which model of M is actually implemented whereas the attacker does know, then the controller must enforce opacity of Sec uniformly in all such models of M.

In the following section, we address the case in which a maximally permissive and regular controller $K^\dagger$ enforcing the opacity of Sec in M can be constructed.

4 Computing $K^\dagger$ for regular upper-closed secrets

There is a somewhat trivial case to consider when the attacker has full observation of the system (i.e., $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma_a = \Sigma$). In this case we simply compute, in accordance with Ramadge and Wonham's theory [15, 14, 16], the maximally permissive $K^\dagger$ for $\overline{M}$ and the expected behaviour $\mathcal{L}(\overline{M}) \setminus Sec$.

In this section, we assume that the secret Sec is upper-closed w.r.t. the prefix-order on words, i.e., $Sec = Sec.\Sigma^*$. That is, once the secret is disclosed, it is disclosed forever. This assumption, also made in [1], implies that the goal of the opacity game is not to permit the attacker to ascertain that some *prefix* of the partially observed run of the LTS *was* in the secret. We make other two alternative working assumptions.

(1) $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_o = \Sigma$, or
(2) $\Sigma_a, \Sigma_c \subseteq \Sigma_o = \Sigma$ and $w \notin Sec \wedge w\sigma \in Sec \Rightarrow \sigma \in \Sigma_c$.

Under these assumptions, the attacker has partial observation, whereas the controller has full observation and can block every action that (1) reveals (i.e., all controllable events are observable) or (2) enters the secret (i.e., an event that would allow access to the secret can always be disabled). This gives a strong advantage to the controller over the attacker, but remember that the controller ignores which LTS is executing among all models of the given MTS, whereas the attacker knows.

4.1 Incorporating the secret with the modal transition system

As a first step towards computing controllers, we incorporate the predicate Sec with the modal transition system M. To do so, we combine the modal transition system $M = (S, \Sigma, \delta^\Box, \delta^\Diamond, s_0)$ and the secret Sec into a modal transition system $M_\#$, with distinguished logical states representing the intersection of $\mathcal{L}(M)$ and the complement of Sec.

First, one constructs a *complete* deterministic LTS $A = (Y, \Sigma, \delta_A, y_0)$ and a subset of states $Y_F \subseteq Y$ recognizing Sec from the initial state y_0, i.e., $\mathcal{L}_{Y_F}(A) = Sec$. Note that $y \in Y_F \Rightarrow (\forall \sigma \in \Sigma)\delta_A(y, \sigma) \in Y_F$ because Sec is upper-closed w.r.t. the prefix-order on words.

Next, one computes the product $M_\#$ of M and A. The initial state of $M_\#$ is the pair (s_0, y_0). The set of states $S_\# \subseteq S \times Y$ of $M_\#$ and the weak

transition map $\delta^{\Diamond}_{\#}$ are jointly and inductively defined by setting $\delta^{\Diamond}_{\#}((s,y),\sigma) = (s',y')$ and $(s',y') \in S_{\#}$ when $\delta^{\Diamond}(s,\sigma) = s'$ and $\delta_A(y,\sigma) = y'$. The strong transition map $\delta^{\Box}_{\#}$ is defined similarly, but replacing $\delta^{\Diamond}(s,\sigma)$ with $\delta^{\Box}(s,\sigma)$. The distinguished logical states $S^F_{\#}$ of $M_{\#}$ are the pairs $(s,y) \in S_{\#}$ such that $y \in Y_F$. Then the following property holds:

$$\forall w \in \mathcal{L}(M_{\#}),\ w \in Sec \text{ if and only if } \delta^{\Diamond}_{\#}(s_0,w) \in S^F_{\#}$$

Furthermore, since the automaton A is complete,

$$\mathcal{L}(M) = \mathcal{L}(M_{\#}) \text{ and } G \models M \Leftrightarrow G \models M_{\#} \text{ for all } G \text{ (over } \Sigma)$$

Example 7 Consider the MTS M_1 of Example 1 and $Sec = \Sigma^*.a.a.b.b.\Sigma^*$. The complete deterministic automaton A that recognizes Sec is given in Figure 9 with $Y_F = \{5\}$,

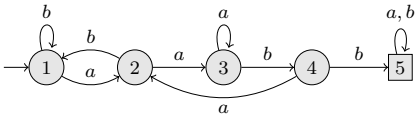


Fig. 9 *A* recognizing *Sec*

whereas, $M_{\#}$ is depicted in Figure 10 with $S^F_{\#} = \{6,7,8,9\}$. ◇

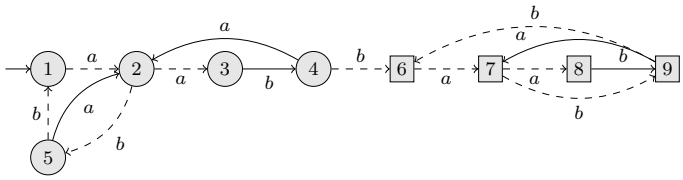


Fig. 10 $M_{\#}$

In the remainder of Section 4, we assume without loss of generality, and for simplicity of notation, that $M = M_{\#}$ and we let $S^F = S^F_{\#}$, thus $Sec = \mathcal{L}_{S^F}(M)$. Moreover, as the predicate Sec is upper-closed, $s \in S^F \Rightarrow \delta^{\Diamond}(s,\sigma) \in S^F$ if the latter is defined.

4.2 The general schema

In this section, we sketch the intuitions under the methods that will be employed for solving Problem 1. Following Definitions 6 and 7, our aim is to

compute the maximally permissive controller K such that the secret is opaque with respect to K/G and Σ_a for every LTS $G \models M$.

As $\Sigma_o = \Sigma$ and $\mathcal{L}(M) = \mathcal{L}(\overline{M})$ is the supremum of $\mathcal{L}(G)$ for all labelled transition systems $G \models M$, in order that a controller K may be admissible for *every* model G of M, it is necessary and sufficient that $\mathcal{L}(K)$ satisfies the controllability constraint w.r.t. $\mathcal{L}(M)$ and Σ_c (Section 2.3), i.e., $w.\sigma \in \mathcal{L}(M) \Rightarrow w.\sigma \in \mathcal{L}(K)$ for any word $w \in \mathcal{L}(M) \cap \mathcal{L}(K)$ and for any uncontrollable action $\sigma \in \Sigma \setminus \Sigma_c$. Note that, as the controller has full observation ($\Sigma_o = \Sigma$), the observability constraint (Section 2.3) is necessary satisfied. When the controllability constraint is satisfied, we say that K is an *admissible* controller of M (w.r.t. Σ_c and $\Sigma_o = \Sigma$).

Among the admissible controllers of M, we should search for controllers K ensuring the opacity of the secret, i.e., such that the following condition holds for *every* labelled transition system $G \models M$ (recall that K/G denotes the product of G and K):

$$\forall w \in \mathcal{L}(K/G),\ \exists w' \in \mathcal{L}(K/G),\quad w \sim_a w' \ \wedge\ \delta^{\Diamond}(s_0, w') \notin S^F.$$

We want to compute the maximally permissive controller K satisfying this condition. We proceed in two steps.

- In a first step, we derive from M an LTS H such that $\mathcal{L}(H) = \mathcal{L}(M)$. The set of states of H is included in $S \times \mathcal{P}(S)$, where $\mathcal{P}(S)$ denotes the power set of S. The meaning of these states is as follows. If $\delta^{\Diamond}(s_0, w) = s$ in M, then w leads in H to the state (s, E) defined by

$$E = \{s' \in S |\, \exists w' \in \mathcal{L}(w \circ M) : w \sim_a w' \wedge \delta^{\Diamond}(s_0, w') = s'\}$$

- In a second step, we trim H according to Ramadge and Wonham's procedure to avoid reaching any state (s, E), where $E \subseteq S^F$.

We will show that the labelled transition system $K^{\dagger}$ obtained in this way is the maximally permissive controller that enforces the opacity of Sec in M.

4.3 A preliminary construction

In the sequel, $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$, $S^F \subseteq S$, $Sec = \mathcal{L}_{S^F}(M)$, and the set $\Sigma_{ua} = \Sigma \setminus \Sigma_a$ denotes the set of actions which are unobservable from the perspective of the attacker. For all transition maps δ, for all sets of states E and for all $L \subseteq \Sigma^*$, we let $\delta(E, \sigma) = \{\delta(s, \sigma) \,|\, s \in E\}$, $\delta(s, L) = \{\delta(s, w) \,|\, w \in L\}$, and $\delta(E, L) = \{\delta(s, w) \,|\, s \in E \wedge w \in L\}$.

Definition 8 *Let $H = (\Theta, \Sigma, \delta_H, \theta_0)$ be the LTS with the set of states $\Theta \subseteq S \times \mathcal{P}(S)$ (where $\mathcal{P}(S)$ denotes the powerset of S) and the labelled transition map δ_H jointly and inductively defined as follows:*

- *let $\theta_0 = (s_0, \delta^{\Box}(s_0, \Sigma_{ua}^*))$ and $\theta_0 \in \Theta$,*

• *inductively, for each state* $(s, E) \in \Theta$ *and for each action* $\sigma \in \Sigma$ *such that* $\delta^{\Diamond}(s, \sigma)$ *is defined, let* $\delta_H((s, E), \sigma) = (s', E')$ *and* $(s', E') \in \Theta$ *where* $s' = \delta^{\Diamond}(s, \sigma)$ *and the set of states* E' *is given according to the case by:*

– $\sigma \notin \Sigma_a$: $E' = E \cup \delta^{\Box}(s', \Sigma_{ua}^*)$,
– $\sigma \in \Sigma_a$: $E' = \delta^{\Box}(E, \sigma.\Sigma_{ua}^*) \cup \delta^{\Box}(s', \Sigma_{ua}^*)$.

Obviously, $\mathcal{L}(H) = \mathcal{L}(M)$ (because we inductively define $s' = \delta^{\Diamond}(s, \sigma)$ when computing δ_H), and $s \in E$ for every state $(s, E) \in \Theta$.

Example 8 To illustrate the construction of H*, let us turn our attention to the MTS depicted in Figure 10 with* $\Sigma_a = \{a\}$*. The LTS derived from this MTS according to Definition 8 is depicted in Figure 11, where states* $(s, \{s_1, s_2, \cdots, s_n\})$ *are represented as pairs* $s, s_1.s_2 \cdots s_n$*.*

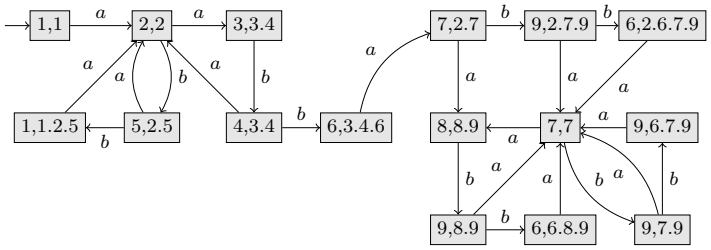


Fig. 11 H computed from $M_\sharp$ (Figure 10)

◇

The following lemma, which is a bit technical, shows that the above construction achieves the goals announced in section 4.2.

Lemma 4 *For any* $w \in \mathcal{L}(M)$, $\delta_H(\theta_0, w) = (s, E) \Rightarrow s = \delta^{\Diamond}(s_0, w)$ *and*

$$E = \{s' \in S \mid \exists w' \in \mathcal{L}(w \circ M) : w' \sim_a w \wedge \delta^{\Diamond}(s_0, w') = s'\}.$$

Proof. The proof is by induction on w. The base of the induction is given by the case $w = \varepsilon$. Then $\delta_H(\theta_0, \varepsilon) = \theta_0 = (s_0, \delta^{\Box}(s_0, \Sigma_{ua}^*))$ by Def. 8. Clearly, $s_0 = \delta^{\Diamond}(s_0, \varepsilon)$. For $w' \in \Sigma^*$, $w' \in \Sigma_{ua}^* \Leftrightarrow w' \sim_a \varepsilon$, and $\delta^{\Box}(s_0, w')$ is defined if and only if $w' \in \mathcal{L}(\underline{M}) = \mathcal{L}(\varepsilon \circ M)$ (Lemma 1). As $\delta^{\Diamond}(s_0, w') = \delta^{\Box}(s_0, w')$ if the latter is defined, the lemma holds for $w = \varepsilon$.

Assume now that the lemma holds for $w = \sigma_1 \ldots \sigma_{n-1}$ (by convention, $n = 1$ means $w = \varepsilon$), and consider $w.\sigma_n \in \mathcal{L}(M)$ with $\sigma_n \in \Sigma$. Let $\delta_H(\theta_0, \sigma_1 \ldots \sigma_i) = (s_i, E_i)$ for $1 \leq i \leq n$. As $s_n = \delta^{\Diamond}(s_{n-1}, \sigma_n)$ (by Def. 8) and $s_{n-1} = \delta^{\Diamond}(s_0, w)$ (by the induction hypothesis), $s_n = \delta^{\Diamond}(s_0, w.\sigma_n)$. To simplify the notation, let $\sigma = \sigma_n$ and $s = s_n$, hence $s = \delta^{\Diamond}(s_0, w.\sigma)$.

We prove $E_n = \{s' \in S \mid \exists w' \in \mathcal{L}((w.\sigma) \circ M) : w' \sim_a w.\sigma \wedge \delta^{\Diamond}(s_0, w') = s'\}$ by case analysis.

Case $\sigma \notin \Sigma_a$. By Def. 8, $E_n = E_{n-1} \cup \delta^{\Box}(s, \Sigma_{ua}^*)$, and by induction, $E_{n-1} = \{s' \in S \mid \exists w' \in \mathcal{L}(w \circ M) : w' \sim_a w \wedge \delta^{\Diamond}(s_0, w') = s'\}$. As $w \sim_a w.\sigma$ and $\mathcal{L}((w.\sigma) \circ M) = \mathcal{L}(w \circ M) \cup w.\sigma.\mathcal{L}(\underline{M}, s)$ (Lemma 2), it suffices to prove:

$$\delta^{\Box}(s, \Sigma_{ua}^*) = \{s' \in S \mid \exists w' \in w.\sigma.\mathcal{L}(\underline{M}, s) : w' \sim_a w.\sigma \wedge \delta^{\Diamond}(s_0, w') = s'\}.$$

Now, $s' \in \delta^{\Box}(s, \Sigma_{ua}^*)$ is and only if $s' = \delta^{\Box}(s, v')$ for some $v' \in \Sigma_{ua}^*$, and $\delta^{\Box}(s, v')$ is defined and equal to $\delta^{\Diamond}(s, v')$, if and only if $v' \in \mathcal{L}(\underline{M}, s)$. As $s = \delta^{\Diamond}(s_0, w.\sigma)$, it follows that $s' \in \delta^{\Box}(s, \Sigma_{ua}^*)$ if and only if $s' \in \delta^{\Diamond}(s, w.\sigma.v')$ for some $v' \in \Sigma_{ua}^* \cap \mathcal{L}(\underline{M}, s)$, if and only if $s' \in \delta^{\Diamond}(s_o, w')$ for some $w' \in w.\sigma.\mathcal{L}(\underline{M}, s)$ satisfying $w' \sim_a w.\sigma$. Therefore, the lemma holds in this case.

Case $\sigma \in \Sigma_a$. By Def. 8, $E_n = \delta^{\Box}(E_{n-1}, \sigma.\Sigma_{ua}^*) \cup \delta^{\Box}(s, \Sigma_{ua}^*)$ and by induction, $E_{n-1} = \{s' \in S \mid \exists w' \in \mathcal{L}(w \circ M) : w' \sim_a w \wedge \delta^{\Diamond}(s_0, w') = s'\}$. Accordingly, $\delta^{\Box}(E_{n-1}, \sigma.\Sigma_{ua}^*) = \{s'' \in S \mid \exists s' \in S\, \exists w' \in \mathcal{L}(w \circ M)\, \exists v' \in \Sigma_{ua}^* : w' \sim_a w \wedge \delta^{\Diamond}(s_0, w') = s' \wedge \delta^{\Box}(s', \sigma.v') = s''\}$. For s', w' and v' as above, let $w'' = w'.\sigma.v'$. As $w' \in \mathcal{L}(w \circ M)$ and $\delta^{\Diamond}(s_0, w') = s'$, $\delta^{\Box}(s', \sigma.v')$ is defined if and only if $w'' \in \mathcal{L}(w \circ M)$, and then $\delta^{\Diamond}(s_0, w") = \delta^{\Box}(s', \sigma.v')$. Moreover, $w'' \sim_a w'.\sigma \sim_a w.\sigma$. Conversely, any $w'' \in \mathcal{L}(w \circ M)$ such that $w'' \sim_a w.\sigma$ decomposes as $w'' = w'.\sigma.v'$ with $w' \in \mathcal{L}(w \circ M), w' \sim_a w$, and $v' \in \Sigma_{ua}^*$. The above relation simplifies therefore to $\delta^{\Box}(E_{n-1}, \sigma.\Sigma_{ua}^*) = \{s'' \in S \mid \exists w'' \in \mathcal{L}(w \circ M) : w'' \sim_a w.\sigma \wedge \delta^{\Diamond}(s_0, w'') = s''\}$. As $\mathcal{L}((w.\sigma) \circ M) = \mathcal{L}(w \circ M) \cup w.\sigma.\mathcal{L}(\underline{M}, s)$ (Lemma 2), in order to complete the proof, it suffices to show that $\delta^{\Box}(s, \Sigma_{ua}^*) = \{s'' \in S \mid \exists v' \in \mathcal{L}(\underline{M}, s) : w.\sigma.v' \sim_a w.\sigma \wedge \delta^{\Diamond}(s_0, w.\sigma.v') = s''\}$. This follows easily because $w.\sigma.v' \sim_a w.\sigma$ if and only if $v' \in \Sigma_{ua}^*$ and $\delta^{\Diamond}(s_0, w.\sigma.v') = \delta^{\Box}(s, v')$ if the latter is defined. □

4.4 The construction of $K^{\dagger}$

As in the previous section, $M = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$, $S^F \subseteq S$, $Sec = \mathcal{L}_{S^F}(M)$ and $H = (\Theta, \Sigma, \delta_H, \theta_0)$. Since $\Theta \subseteq S \times \mathcal{P}(S)$ where S is the set of logical states of M, H is a finite LTS, with the language $\mathcal{L}(H) = \mathcal{L}(M) = \cup\{\mathcal{L}(G) \mid G \models M\}$. Our goal is to produce $K^{\dagger}$ from H by removing all words $w \in \mathcal{L}(H)$ that disclose the secret Sec in some model G of M.

As $\mathcal{L}(w \circ M)$ is the infimum of $\mathcal{L}(G)$ for all G such that $G \models M$ and $w \in \mathcal{L}(G)$ (Proposition 1), a word $w \in \mathcal{L}(H)$ discloses the secret Sec in some model of M if and only if it discloses the secret Sec in $w \circ M$. By Lemma 4, a word $w \in \mathcal{L}(H)$ discloses the secret Sec in $w \circ M$ if and only if $\delta_H(\theta_0, w) \in Bad^0$ where we let

$$Bad^0 = \{(s, E) \in \Theta \mid E \subseteq S^F\}.$$

Enforcing the opacity of the secret Sec in all models of M amounts therefore to barring access to bad states of H, i.e., to states in Bad^0.

As $\mathcal{L}(H) = \cup\{\mathcal{L}(G) \mid G \models M\}$, the controllability constraint $\mathcal{L}(K).\Sigma_{uc} \cap \mathcal{L}(G) \subseteq \mathcal{L}(K)$ holds for all models G of M is and only if it holds for $\mathcal{L}(H)$, hence a controller K is an admissible controller of all models G of M if and only if it is an admissible controller of H.

So, in order for $K = (X, \Sigma, \delta_K, x_0)$ to enforce the opacity of the secret Sec in *every* model G of M (w.r.t. Σ_a) and to be an admissible controller of G (w.r.t. Σ_c), it is necessary that the following two conditions hold:

- no state (θ, x) with $\theta \in Bad^0$ can be reached from (θ_0, x_0) in K/H, (C_1)
- K is an admissible controller of H (w.r.t. Σ_c). (C_2)

According to Ramadge and Wonham's theory of state-based supervision, the maximally permissive controller $K^\dagger$ for which both conditions hold is obtained by pruning H iteratively. Throughout the iteration, one maintains a partition $\{Good, Bad\}$ of the set of states $X = \Theta$ and a partial transition map $\delta_X : \Theta \times \Sigma \to \Theta$. $Good$ decreases and Bad increases while δ_X gets less and less defined at each step. The algorithm is at follows:

- Initially, $Good^0 = X \setminus Bad^0$ and $\delta_X = \delta_H$.
- at step $n+1$ in the iteration:
  - initially, one lets $Bad^{n+1} = Bad^n$,
  - for all pairs of arguments $\theta \in Good^n$ and $\sigma \in \Sigma$ such that $\delta_X(\theta, \sigma) \in Bad^n$:
    - one removes (θ, σ) from the domain of definition of δ_X,
    - if σ is uncontrollable ($\sigma \notin \Sigma_c$), then one adds the considered state θ to the set Bad^{n+1},
  - finally, one lets $Good^{n+1} = X \setminus Bad^{n+1}$,
- the global iteration stops when $\delta_X(\theta, \sigma) \in Bad^n$ for no pair of arguments $\theta \in Good^n$ and $\sigma \in \Sigma$,

At termination of the algorithm, let $Good = Good^n$ and $Bad = X \setminus Good^n$, then $K^\dagger$ is the induced restriction of the LTS $(Good, \Sigma, \delta_X, \theta_0)$ on the states reachable from θ_0.

- If $\theta_0 \notin Good$, then no controller can prevent Bad states from being reached (hence no controller can enforce the opacity of the secret in all models of MTS).
- If $\theta_0 \in Good$, then $K^\dagger$ is the maximally permissive controller preventing states in Bad^0 from being reached in H.

However, this does not entail directly that $K^\dagger$ enforces the opacity of the secret in all models of MTS, since (C_1) and (C_2) were only necessary conditions for achieving this goal. The following lemma is crucial to prove that $K^\dagger$ enforces indeed the opacity of the secret in all models of MTS.

Lemma 5 *If the iterative procedure defined above is applied to the LTS H specified by Definition 8 and to the set $Bad^0 = \{(s, E) \in \Theta \mid E \subseteq S^F\}$, then $s \in S^F$ for every state (s, E) of H that is eventually turned to Bad.*

Proof. – We consider first the case where $\Sigma_a \subseteq \Sigma_c$. We show that in this case, the set Bad stays equal to Bad^0. At step $n+1$ in the iteration, assume by induction on n that $Bad^n = Bad^0$, and let $\delta_H((s, E), \sigma) = (s', E')$ for some $\sigma \in \Sigma \setminus \Sigma_c$ and $(s', E') \in Bad^n$. As $\Sigma_a \subseteq \Sigma_c$, $\sigma \in \Sigma \setminus \Sigma_a$. By Def. 8, $E \subseteq E'$. As $E' \subseteq S^F$, $E \subseteq S^F$ and $(s, E) \in Bad^0$, hence $Bad^{n+1} = Bad^0$. Moreover, $s \in E \subseteq S^F$ entails $s \in S^F$ for every state $(s, E) \in Bad = Bad^0$.

– We consider next the case where $\Sigma_a, \Sigma_c \subseteq \Sigma_o = \Sigma$ and $w\sigma \in Sec \Rightarrow w \in Sec$ for all $\sigma \in \Sigma \setminus \Sigma_c$, i.e., $\delta^{\Diamond}(s,\sigma) \in S^F \Rightarrow s \in S^F$ for all $\sigma \in \Sigma \setminus \Sigma_c$. At step $n+1$ in the iteration, let $\delta_H((s,E),\sigma) = (s',E')$ for some $\sigma \in \Sigma \setminus \Sigma_c$ and $(s',E') \in Bad^n$. By induction on n, assume that $s' \in S^F$. As $\delta^{\Diamond}(s,\sigma) = s'$ and $\sigma \in \Sigma \setminus \Sigma_c$, necessarily, $s \in S^F$. Therefore, the proof of the lemma is complete.

□

Remark 2 If $\delta_H((s,E),\sigma) = (s',E')$ for $\sigma \in \Sigma_c \setminus \Sigma_a$ then $E \subseteq E'$ by Def. 8, hence $(s',E') \in Bad^0 \Rightarrow (s,E) \in Bad^0$. Therefore, actions σ in $\Sigma_c \setminus \Sigma_a$ are never blocked by $K^{\dagger}$.

Remark 3 It may occur, for some state (s,E) of $K^{\dagger}$ and for some $s' \in E$ that $K^{\dagger}$ has no state of the form (s',E'). In such a case, some state (s',E') of H must have been turned to *Bad*, hence $s' \in S^F$ by Lemma 5. Therefore, $E \cap (S \setminus S^F) = (E \setminus \{s'\}) \cap (S \setminus S^F)$. This explains why we do not need to update the second component of a state (s,E) during Ramadge and Wonham's iterative cleaning process.

Proposition 2 *$K^{\dagger}$ enforces the opacity of the secret Sec in all models G of M.*

Proof. Let $G \models M$ and $w \in \mathcal{L}(K^{\dagger}/G)$. We must show that there exists $w' \in \mathcal{L}(K^{\dagger}/G)$ such that $w \sim_a w'$ and $\delta^{\Diamond}(s_0,w') \notin S^F$ (in M). As $w \in \mathcal{L}(K^{\dagger})$, $\delta_H(\theta_0,w)$ must be a *Good* state, hence $\delta_H(\theta_0,w) \notin Bad^0 \subseteq Bad$. By Lemma 4 and the definition of the set Bad^0, $w \sim_a w'$ and $\delta^{\Diamond}(s_0,w') \notin S^F$ for some $w' \in \mathcal{L}(w \circ M)$. As $G \models M$ and $w \in \mathcal{L}(G)$, by proposition 1, $w' \in \mathcal{L}(G)$. As the secret *Sec* is an upper-closed set, $\delta^{\Diamond}(s_0,w') \notin S^F$ entails $\delta^{\Diamond}(s_0,v') \notin S^F$ for every prefix v' of w'. By Lemma 5, $\delta_H(s_0,v')$ is never turned from *Good* to *Bad* for any prefix v' of w'. Therefore, $w' \in \mathcal{L}(K^{\dagger})$, and $w' \in \mathcal{L}(K^{\dagger}/G)$.

□

Theorem 1 *$K^{\dagger}$ is maximally permissive among all admissible controllers enforcing the opacity of the secret in all models of M.*

Proof. $K^{\dagger}$ is maximally permissive among the controllers of H that satisfy the two necessary conditions (C_1) and (C_2). Proposition 2 completes the proof of the theorem.

Example 9 In H (Figure 11) computed from $M_\sharp$ (Figure 10), Bad^0 is the set of states (s,E) in which $E \subseteq S^F = \{6,7,8,9\}$. Ramadge & Wonham's iteration converges in one step, producing the controller $K^{\dagger}$ depicted in Figure 12 and Figure 7. This maximally permissive controller enforces the opacity of the secret $\Sigma^*.a.a.b.b.\Sigma^*$ in all models of M_1 (Figure 1(a)) and in particular in G_1 (Figure 1(c)) and G_2 (Figure 1(d)). The controlled systems $K^{\dagger}/G_1$ and $K^{\dagger}/G_2$ are depicted in Figure 8. ◇

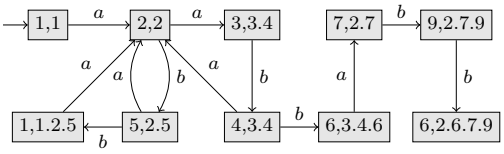


Fig. 12 $K^\dagger$ for M_1

Example 10 Consider the modal transition system M_2 (Figure 1(b)). Given the secret $Sec = \Sigma^*.a.a.b.b.\Sigma^*$ recognized by the automaton depicted in Figure 9, the corresponding $M_\sharp$ is like that in Figure 10, except that transitions $1 \xrightarrow{a} 2$, $2 \xrightarrow{b} 5$, $6 \xrightarrow{a} 7$ and $7 \xrightarrow{b} 9$ are now strong transitions. As before, $S^F = \{6, 7, 8, 9\}$. The transition system H computed from M_2 and the secret Sec is shown in Figure 13. As Bad^0 is the set of states (s, E) such that $E \subseteq S^F$,

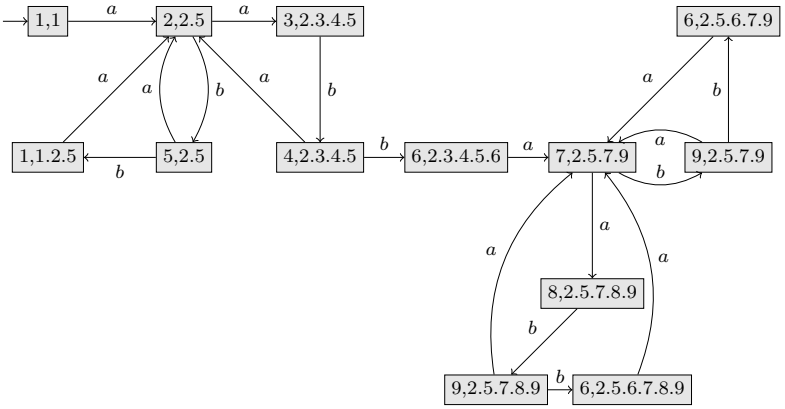


Fig. 13 H for M_2

$Bad^0 = \emptyset$. Therefore $K^\dagger = H$. As $\mathcal{L}(H) = \mathcal{L}(M_2)$, this means that the secret Sec is opaque in every model G of M_2 and no additional control is needed. $\diamond$

References

1. E. Badouel, A. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau. Concurrent secrets. *Discrete Event Dynamic Systems*, 17:425–446, December 2007.
2. M. Ben-Kalefa and F. Lin. Supervisory control for opacity of discrete event systems. In *Proc. 45th Annual Allerton Conference on Communication, Control, and Computing*, pages 1113–1119, Allerton House, 2011.

3. J.W. Bryans, M. Koutny, L. Mazaré, and P.Y.A. Ryan. Opacity generalized to transition systems. *Int. Journal of Computer Security*, 7(6):421–435, 2008.
4. Ph. Darondeau, J. Dubreil, and H. Marchand. Supervisory control for modal specifications of services. In *Workshop on Discrete Event Systems, WODES'10*, pages 428–435, Berlin, Germany, August 2010.
5. J. Dubreil, Ph. Darondeau, and H. Marchand. Opacity enforcing control synthesis. In *Workshop on Discrete Event Systems, WODES'08*, pages 28–35, Gothenburg, Sweden, March 2008.
6. J. Dubreil, Ph. Darondeau, and H. Marchand. Supervisory control for opacity. *IEEE Transactions on Automatic Control*, 55(5):1089–1100, May 2010.
7. G. Feuillade and S. Pinchinat. Modal specifications for the control theory of discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 17:211–232, 2007.
8. C.N. Hadjicostis. Supervisory control strategies for enhancing system security and privacy. In *Proc. 48th Annual Allerton Conference on Communication, Control, and Computing*, pages 1622–1627, Allerton House, IL, 2010.
9. K.G. Larsen. Modal specifications. *Automatic Verification Methods for Finite State Systems*, 407:232–246, 1990.
10. F. Lin. Opacity of discrete event systems and its applications. *Automatica*, 47(4):496–503, 2011.
11. N. Lohmann, P. Massuth, and K. Wolf. Operating guidelines for finite-state services. In *Proc. ICATPN*, volume 4546 of *LNCS*, pages 321–341, 2007.
12. L. Mazaré. Decidability of opacity with non-atomic keys. In *Proc. FAST'04*, pages 71–84, 2004.
13. L. Mazaré. Using unification for opacity properties. In *Proceedings of the 4th IFIP WG1.7 Workshop on Issues in the Theory of Security (WITS'04)*, pages 165–176, Barcelona (Spain), 2004.
14. P.J. Ramadge and W.M. Wonham. On the supremal controllable language of a given language. *SIAM J. of Control and Optimization*, 25:637–659, 1987.
15. P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event processes. *SIAM J. of Control and Optimization*, 25(1):206–230, January 1987.
16. P.J. Ramadge and W.M. Wonham. The control of discrete event systems. *Proceedings of the IEEE; Special issue on Dynamics of Discrete Event Systems*, 77(1):81–98, 1989.
17. A. Saboori and C. Hadjicostis. Opacity-enforcing supervisory strategies for secure discrete event systems. In *Proc. 47th IEEE Conference on Decision and Control*, pages 889–894, Cancun, Mexico, December 2008.
18. A. Saboori and C. Hadjicostis. Verification of initial-state opacity in security applications of DES. In *9th International Workshop on Discrete Event Systems, 2008. WODES 2008.*, pages 328–333, May 2008.

19. A. Saboori and C. Hadjicostis. Opacity-enforcing supervisory strategies via state estimator constructions. *IEEE Transactions on Automatic Control*, 57(5):1155–1165, 2012.
20. S. Takai and R. Kumar. Verification and synthesis for secrecy in discrete-event systems. In *Proc. American Control Conference*, pages 4741–4746, 2009.
21. S. Takai and Y. Oka. A formula for the supremal controllable and opaque sublanguage arising in supervisory control. *SICE Journal of Control, Measurement, and System Integration*, 1(4):307–312, March 2008.
22. Y. Wu and S. Lafortune. Enforcement of opacity properties using insertion functions. In *51st IEEE Conference on Decision and Control*, pages 6722–6728, Maui, Hawaii, USA, December 2012.