

# Supervisory Control for Opacity

Jérémy Dubreil, Philippe Darondeau, and Hervé Marchand

**Abstract**—In the field of computer security, a problem that received little attention so far is the enforcement of confidentiality properties by supervisory control. Given a critical system  $G$  that may leak confidential information, the problem consists in designing a controller  $C$ , possibly disabling occurrences of a fixed subset of events of  $G$ , so that the closed-loop system  $G/C$  does not leak confidential information. We consider this problem in the case where  $G$  is a finite transition system with set of events  $\Sigma$  and an inquisitive user, called the adversary, observes a subset  $\Sigma_a$  of  $\Sigma$ . The confidential information is the fact (when it is true) that the trace of the execution of  $G$  on  $\Sigma^*$  belongs to a regular set  $S \subseteq \Sigma^*$ , called the secret. The secret  $S$  is said to be opaque w.r.t.  $G$  (respectively,  $G/C$ ) and  $\Sigma_a$  if the adversary cannot safely infer this fact from the trace of the execution of  $G$  (respectively,  $G/C$ ) on  $\Sigma_a^*$ . In the converse case, the secret can be disclosed. We present an effective algorithm for computing the most permissive controller  $C$  such that  $S$  is opaque w.r.t.  $G/C$  and  $\Sigma_a$ . This algorithm subsumes two earlier algorithms working under the strong assumption that the alphabet  $\Sigma_a$  of the adversary and the set of events that the controller can disable are comparable.

**Index Terms**—Confidentiality, control, discrete event systems (DES), opacity, partial observation, security.

## I. INTRODUCTION

THE development of infrastructures like the Internet or the mobile phone networks has led to the emergence of sophisticated on-line services providing information access or decision taking facilities. Such networks are open by nature and therefore vulnerable to malicious users. All the same, security and confidence in security are essential to distant services such as e-voting, on-line payment, or medical information storage. Such services handle indeed critical information that should be neither erased nor corrupted nor leaked to unauthorized users. Confidence in the security of a service relies on and requires some certification of security. Manual validation is expensive, may be impossible for large systems, and is permeable to mistakes. The development of automatic tools serving to analyze or to ensure the security of services has become crucial to discover and avoid security breaches. In this context, there has been growing interest in the formal verification of security properties [1]–[3] and in their model-based testing [4]–[8].<sup>1</sup>

Manuscript received January 13, 2009; revised June 19, 2009 and August 24, 2009. First published February 02, 2010; current version published May 12, 2010. This work was supported in part by the Politess RNRT project. Recommended by Associate Editor S. Reveliotis.

The authors are with the INRIA, Centre Rennes-Bretagne Atlantique, Rennes, France (e-mail: jeremy.dubreil@inria.fr; jeremy.dubreil@aitb.org; darondeau@inria.fr; hmarchan@inria.fr).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TAC.2010.2042008

<sup>1</sup>Not mention works on cryptography that fall out of the scope of this paper.

Security properties are generally classified into three different categories [9]:

- availability (a user can always perform legal actions);
- integrity (a user can never perform illegal actions);
- confidentiality (a user cannot discover or infer the secret information).

Consider the case of an e-voting system. Ensuring that the votes cannot be modified by a third party is a concern of integrity. Ensuring that every elector can vote is a concern of availability. Ensuring that is not possible for a third party to discover the vote of an elector is a concern of confidentiality.

In this paper, we focus on confidentiality and especially on opacity, a central notion that was introduced in [10] and adapted to transition systems in [11]. Given a transition system, consider an observation map that projects runs to observations, and a specific subset of runs called the secret, with the meaning that the observed behavior of the system should never disclose when its actual behavior belongs to this set. The secret is said to be opaque if the projection of every run in this set coincides with the projection of some run outside the set. Then, an adversary who observes the run of the system cannot safely infer from this (partial) observation that the run belongs to the secret. More specific notions of opacity, like initial opacity or  $K$ -step opacity have been introduced in [12], [13]. At the same time, the notion of opacity defined in [10] is general enough to allow other notions like anonymity (strong or weak as defined in [14]) and strong non-deterministic non-interference [15] to be expressed as opacity for suitable secrets and observation maps [11]. Note that opacity is more or less dual to diagnosability [16], [17], that consists in deciding whether the actual run of the system belongs to the secret from a bounded extension of this run.

Our purpose in this paper is not to model-check transition systems for opacity properties but to enforce such properties on transition systems by supervisory control. According to Ramadge and Wonham's theory presented in [18], [19], the aim of supervisory control is to enforce a safety property on a transition system. This is achieved by defining a map that determines after each incomplete run of the system the set of actions which may be taken without compromising the safety property. All uncontrollable actions must be in this set. This control map is generally expected to be as permissible as possible, i.e. no unnecessary restriction should be imposed on the system. If successful termination of runs is taken into account, the control map should moreover be non-blocking, meaning that it should not prevent the system from eventually reaching some final state. Within Ramadge and Wonham's framework, the computational aspects of supervisory control have been investigated mainly for finite transition systems and regular safety properties. In this case, the control map can be computed in a regular form, yielding a finite state supervisor. The expected controlled system is then the product  $G/C$  of the uncontrolled system  $G$  and supervisor  $C$ .

Applying supervisory control to enforce confidentiality properties is an emerging field of research. In [20], the author adapts the decentralized supervisory control theory in order to ensure the Chinese Wall Policy, whereas in [21], the authors focus on (bisimulation-based) strong non-deterministic non-interference properties. Attempts to adapt supervisory control to opacity have been made in [22]–[24]. In these works, one considers a finite and deterministic labeled transition system  $G$  over an alphabet  $\Sigma$ , a regular set  $S \subseteq \Sigma^*$  (the secret) and a subset  $\Sigma_a \subseteq \Sigma$  (the alphabet of the adversary), and one searches for a finite state supervisor  $C$  enforcing the opacity of  $S$  w.r.t. the natural projection from  $\Sigma^*$  to  $\Sigma_a^*$  (by the opacity of  $S$  we mean the opacity of the set of runs with traces in  $S$ ). In this setting,  $G$  represents a system running in the scope of an inquisitive adversary. The adversary observes all actions in  $\Sigma_a$  and tries to infer from these partial observations the knowledge that the trace of the run of  $G$  is in the secret set  $S$ . For instance,  $G$  may be the driver of a semi-autonomous agent that travels through a finite network, and  $S$  is the set of incoming paths to a region of the network that should not be identified as the current region of the agent without compromising efficiency (the task of the agent may be to keep watch on a shopping center with two types of stores, specialized either in jewels or in candies, and thieves should not know when jewels stores are without supervision) or continuity of the service (the agent should not fall in an ambush). For another illustration,  $G$  may be a voting system with a finite set of voters, numbered from 1 to  $m$ , where the sequence of voters is random and may be observed (votes are not observable). Suppose that voting may be stopped at any time when it is certain that further voting would not change the result, which is then published. Let elementary votes be represented as pairs  $i : \text{yes}$  or  $i : \text{no}$ . Let  $S_y$ , respectively,  $S_n$  be the sets of sequences in which the last vote is *yes*, respectively, *no*. Enforcing the joint opacity of  $S_y$  and  $S_n$  prevents  $G$  from stopping whenever this would reveal some vote  $i : \text{yes}$  or  $i : \text{no}$ .

One subtlety lays in the additional assumption that the adversary knows exactly the system  $G$  and the supervisor  $C$ . This means that new confidential information may be inferred by the adversary from the knowledge of  $C$  and the partial observation of the run of the controlled system, and to avoid such leakage, one must iterate the controller construction. Hence the exact problem is to find a supervisor  $C$  that enforces the opacity of  $S$  w.r.t.  $L(G/C)$  and the projection from  $\Sigma^*$  to  $\Sigma_a^*$ , and this is an intrinsically circular problem because the control objective cannot be expressed without an explicit reference to the controller (this is not true when the control objective is a safety property).

The non-blocking property of the supervisor (another circular problem) was ignored in the works cited above. We shall not address this issue in the present paper, where we still consider finite transition systems, regular secrets and natural projections.

The intrinsic circularity of the supervisory control problem for opacity makes it impossible to give a general solution to this problem by direct application of Ramadge and Wonham's methods.<sup>2</sup> These methods were designed for enforcing integrity

<sup>2</sup>By Ramadge and Wonham's methods, we mean in general finite iterative methods for computing the greatest fixpoints of contractive operators on regular languages, enforcing separately or jointly behavioral properties such as safety, non-blockingness and the like by successive restrictions until these properties finally hold.

properties on plants. Nevertheless, the Ramadge and Wonham's theory can be used directly for enforcing opacity properties when (1)  $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma_a$  or (2)  $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_o$  where  $\Sigma_c$  respectively,  $\Sigma_o$  denote the subsets of actions that can be controlled respectively, observed by the supervisor  $C$  [24]. In both cases, the language of the optimal supervisor  $C$  may be computed by a first fixpoint iteration, yielding the supremal sublanguage  $L'$  of  $L(G)$  with respect to which  $S$  is opaque, followed by a second fixpoint iteration, yielding the supremal controllable and normal sublanguage of  $L'$ . Conditions less restrictive than (2) have been elaborated in [23] to the same effect.

In the remaining situations, a non-trivial adaptation of Ramadge and Wonham's methods is necessary. Some of the difficulties encountered are described in [24] where a specific control synthesis algorithm is defined for the case  $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_o \subseteq \Sigma$ . The purpose of this paper is to propose a new algorithm producing a most permissive and regular supervisory control for opacity in the more general case where both  $\Sigma_c \subseteq \Sigma_o$  and  $\Sigma_a \subseteq \Sigma_o$  but  $\Sigma_c$  and  $\Sigma_a$  are not necessarily comparable.

The remaining sections are organized as follows. Section II fixes some notations, introduces the notion of opacity and presents the opacity control problem. Section III provides a reduction of the opacity control problem to the same problem under full observation. Section IV presents informally the constructions needed to synthesize the controller. Section V is the core of the paper and presents the detailed construction and the formal justification of a solution to the opacity control problem. Section VI concludes the paper.

## II. PRELIMINARIES

### A. Notations and Definitions

Henceforth,  $\Sigma$  is a finite alphabet of actions (denoted  $\sigma, \sigma_i$  and the like),  $G = (Q, \Sigma, \delta, q_0)$  is a deterministic transition system labeled in  $\Sigma$ , with a finite set of states  $Q$ , an initial state  $q_0 \in Q$ , and a partial transition map  $\delta : Q \times \Sigma \rightarrow Q$ . A (partial or incomplete) run  $\rho$  of  $G$  is a finite non-empty sequence  $q_0\sigma_1q_1\sigma_2 \dots q_{n-1}\sigma_n$  alternatively consisting of states  $q_i \in Q$  and actions  $\sigma_i \in \Sigma$  such that  $\delta(q_{i-1}, \sigma_i)$  is defined for  $i = 1 \dots n$  and it is equal to  $q_i$  for  $i = 1 \dots n - 1$ .

The trace  $tr(\rho)$  of the run  $\rho$  is the word  $\sigma_1\sigma_2 \dots \sigma_n \in \Sigma^*$ . The trace of the empty run  $q_0$  is the empty word  $\varepsilon$ . The language of  $G$  is the set of all traces of runs of  $G$  and it is denoted  $L(G)$ . As  $G$  is a deterministic labeled transition system (or LTS), runs and traces of runs are in bijective correspondence. The words in  $L(G)$  may therefore, by an abuse of terminology, be called traces of  $G$ .

Opacity control aims at preventing an inquisitive user, called the adversary, from obtaining or inferring confidential information on the execution of  $G$  from a partial observation of this run. To model the partial observation of runs by the adversary, we let  $\Sigma_a \subseteq \Sigma$  denote the subset of actions of  $G$  that the adversary can observe, and we use the *natural projection*  $\pi_a : \Sigma^* \rightarrow \Sigma_a^*$  defined inductively on words with  $\pi_a(\varepsilon) = \varepsilon$ ,  $\pi_a(\sigma w) = \sigma \pi_a(w)$  if  $\sigma \in \Sigma_a$ , and  $\pi_a(\sigma w) = \pi_a(w)$  otherwise. For any words  $w, w' \in \Sigma^*$ ,  $w \sim_a w'$  denotes the fact that  $\pi_a(w) = \pi_a(w')$ ,

and  $[w]_a = \pi_a^{-1} \circ \pi_a(w)$  denotes the class of  $w$  w.r.t. the induced equivalence. Thus,  $L(G) \cap [w]_a$  is the set of traces of  $G$  that the adversary cannot distinguish from  $w$ .  $Det_a(G)$  denotes the deterministic LTS which is computed from  $G$  by first replacing all transitions labeled in  $\Sigma \setminus \Sigma_a$  with  $\varepsilon$ -transitions and next applying the subset construction on the relabeled version of  $G$ . The construction is recalled hereafter.<sup>3</sup>

**Definition 1:** Let  $G = (Q, \Sigma, \delta, q_0)$ . Then  $Det_a(G) = (\mathcal{X}, \Sigma_a, \Delta, X_0)$  where  $X_0 = \delta(q_0, [\varepsilon]_a)$  and  $\mathcal{X} \subseteq 2^Q$  is the inductive closure of the set  $\{X_0\}$  under the partial transition map  $\Delta(X, \sigma) = \delta(X, [\sigma]_a)$ .

In order to match the system  $G$  with the dynamic estimation of its current state by the adversary, we shall consider the parallel composition  $G \times Det_a(G)$ . This composition operation is recalled below.

**Definition 2:** Let  $G_i = (Q_i, \Sigma_i, \delta_i, q_{i,0})$   $i = 1, 2$  be two LTSs. Their *parallel composition*  $G_1 \times G_2$  is the deterministic transition system  $(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta, (q_{1,0}, q_{2,0}))$  as follows:

$$\delta((q_1, q_2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) & \text{if } \sigma \in \Sigma_1 \cap \Sigma_2, \delta_i(q_i, \sigma) \text{ defined} \\ (\delta_1(q_1, \sigma), q_2) & \text{if } \sigma \in \Sigma_1 \setminus \Sigma_2, \delta_1(q_1, \sigma) \text{ defined} \\ (q_1, \delta_2(q_2, \sigma)) & \text{if } \sigma \in \Sigma_2 \setminus \Sigma_1, \delta_2(q_2, \sigma) \text{ defined} \\ \text{undefined,} & \text{otherwise.} \end{cases}$$

Note that a controlled system  $G/C$  might also be written  $G \times C$  since  $G/C$  is indeed the parallel composition of the plant  $G$  and the controller  $C$ .

### B. The Definition of Opacity

Consider an LTS  $G$  over  $\Sigma$  and a subalphabet  $\Sigma_a \subseteq \Sigma$ . The alphabet  $\Sigma_a$  is the set of actions supplied to the user for interacting synchronously with  $G$ , i.e. for observing the runs of  $G$ . One wants to hide from the user some confidential property of runs, e.g. that a run has never visited some state, or that some action  $\sigma$  has always been immediately followed in the run by some other action  $\sigma'$ , or that some high-level action occurred in the run (according to the terminology for non-interference used in [15]). Such a confidential property of runs may be represented abstractly as a non-empty and regular subset  $S \subseteq L(G)$ , which we call a *secret*. As the user is possibly inquisitive, we call him the *adversary*.

If, for some trace  $w$  of  $G$ ,  $\pi_a(w)$  belongs to  $\pi_a(S)$  but does not belong to  $\pi_a(L(G) \setminus S)$ , then the adversary knows from the observation  $\pi_a(w)$  of the trace  $w$  that this trace belongs to  $S$ , i.e. the trace  $w$  *discloses the secret to the adversary*. In the converse case, no confidential information is leaked and the secret  $S$  is said to be *opaque* w.r.t.  $L(G)$  and  $\Sigma_a$ . Let us state a more precise definition of opacity.

**Definition 3:** For  $S', L' \subseteq \Sigma^*$ , let  $Discloser(S', L')$  be the set of words  $w \in L'$  such that  $[w]_a \cap L' \subseteq S'$ . Then  $S'$  is *opaque* w.r.t.  $L'$  and  $\Sigma_a$  if  $Discloser(S', L') = \emptyset$ .

**Remark 1:** Equivalently,  $S'$  is *opaque* w.r.t.  $L'$  and  $\Sigma_a$  iff  $(\forall s \in L')(\exists s' \in L' \setminus S')s' \sim_a s$ .

<sup>3</sup>We use freely the inductive extension  $\delta : Q \times \Sigma^* \rightarrow Q$  defined with  $\delta(q, \varepsilon) = q$  and  $\delta(q, s\sigma) = \delta(\delta(q, s), \sigma)$  for  $\sigma \in \Sigma$ . We use as well the additive extensions of  $\delta$  to sets of states and/or to sets of words.

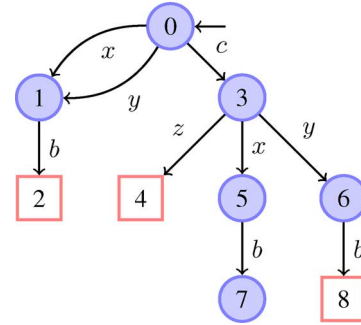


Fig. 1. Case of non-opacity.

**Remark 2:** If  $\mathcal{L}$  is any family of languages and  $S$  is opaque w.r.t.  $L$  and  $\Sigma_a$  for any  $L \in \mathcal{L}$ , then  $S$  is opaque w.r.t. arbitrary unions of languages in  $\mathcal{L}$  and  $\Sigma_a$ .

**Example 1:** Let  $G = (Q, \Sigma, \delta, q_0, Q_S)$  be the automaton depicted in Fig. 1, thus  $Q = \{0, \dots, 8\}$ ,  $\Sigma = \{b, c, x, y, z\}$ ,  $\delta$  is defined by the labeled edges,  $q_0 = 0$ , and  $S = \{xb, yb, cz, cyb\}$  (the traces in  $S$  lead to the squared configurations in Fig. 1). Let  $\Sigma_a = \{b, x, y, z\}$ . Then  $S$  is not opaque w.r.t.  $G$  and  $\Sigma_a$ , since e.g. for the observation  $z$ , the only possible trace is  $c.z \in S$ . Similarly, for the observation  $y.b$ , the only possible traces are  $y.b$  and  $c.y.b$  which both belong to the secret  $S$ . Finally, the trace  $x.b$  belongs to  $S$  but it does not disclose the secret, since  $x.b \sim_a c.x.b$  and  $c.x.b$  does not belong to  $S$ .

For another illustration of the concept of opacity, imagine that  $G$  models a hardware/software system with a bug that cannot be fixed because it is in the hardware and  $S$  is the set of traces of all runs in which the bug occurs, i.e. the secret  $S$  is the run-time extension of the bug. If  $S$  is opaque w.r.t.  $L(G)$ , then a user who can only observe the actions in  $\Sigma_a$  will never identify any consequence of the bug. In the converse case, the best one can do is to hide the bug by wrapping  $G$  in a software interface, restricting the behavior of  $G$  and ensuring in this way that  $S$  and hence the bug cannot be disclosed any more. This can always be done in view of the following proposition.

**Proposition 1 ([22]):** Given a system  $G$  and a set of traces  $S$ , there always exists a supremal prefix-closed sublanguage  $L'$  of  $L(G)$  such that  $S$  is opaque w.r.t.  $L'$  and  $\Sigma_a$ , namely the language  $L' = L(G) \setminus ((L(G) \setminus \pi_a^{-1} \circ \pi_a(L(G) \setminus S)).\Sigma^*)$ .

The proof of this proposition follows from the definition of opacity and Remark 2.

### C. Opacity Control Problem

Given a system  $G$  and a secret  $S$ , our goal is to enforce the opacity of  $S$  on  $G$  by supervisory control. The search space for possible controls over  $G$  is determined by two subsets of  $\Sigma$ , a subset  $\Sigma_c$  of *controllable* actions and a subset  $\Sigma_o$  of *observable* actions. It is assumed that after a run of  $G$  with trace  $w$ , the information available for controlling the next action of  $G$  is the observed trace  $\pi_o(w)$ , where  $\pi_o$  is the natural projection from  $\Sigma^*$  to  $\Sigma_o^*$ .

A *control* is a map  $f : \Sigma_o^* \rightarrow 2^\Sigma$  from observed traces to subsets of  $\Sigma$  such that only the supersets of  $\Sigma_{uc} = \Sigma \setminus \Sigma_c$  can appear in the range of  $f$  (this restriction reflects that a controller

can never disable any uncontrollable action). Applying the control  $f$  to  $G$  means disabling after  $w$  all actions which do not belong to  $f(\pi_o(w))$ . We let  $L(G/f)$  denote the induced restriction of the language generated by  $G$  under the control  $f$ .

*Remark 3:* If  $C$  is a (possibly infinite) LTS such that  $L(C) = L(G_{\Sigma^*}/f)$ , where  $G_{\Sigma^*}$  is any LTS such that  $L(G_{\Sigma^*}) = \Sigma^*$ , then  $L(G/f) = L(G \times C)$ . The LTS  $C$  is called a *controller*, and the parallel composition  $G \times C$  is often written  $G/C$  to stress this interpretation. Indeed,  $f$  and  $C$  determine each other up to the constraint  $L(G/f) = L(G \times C)$ . So, one can work indifferently with control maps  $f$  or with controllers  $C$ . •

The problem we want to solve may be described roughly as follows.

*Problem 1:* Given a finite deterministic transition system  $G$  labeled over  $\Sigma$ , a regular subset  $S \subseteq \Sigma^*$  (the secret), and three subalphabets  $\Sigma_a$ ,  $\Sigma_o$  and  $\Sigma_c$  of  $\Sigma$ , compute a most permissive control  $f$  such that  $S$  is opaque w.r.t.  $L(G/f)$  and  $\Sigma_a$ .

A classical Ramadge and Wonham's theorem states that a prefix-closed sublanguage  $K$  of  $L(G)$  may be obtained as the induced restriction of the language generated by  $G$  under some control  $f$  (in formulas,  $K = L(G/f)$ ) if and only if  $K$  is controllable and observable according to the definitions below (for a complete presentation of supervisory control, the reader is referred e.g. to [25]).

*Definition 4:* A prefix-closed language  $K \subseteq L(G)$  is controllable w.r.t.  $L(G)$  and  $\Sigma_c$  if  $K \cdot \Sigma_c \cap L(G) \subseteq K$ . •

*Definition 5:* A prefix-closed language  $K \subseteq L(G)$  is observable w.r.t.  $L(G)$ ,  $\Sigma_o$  and  $\Sigma_c$  if, for any  $s, s' \in K$  with identical observations  $\pi_o(s) = \pi_o(s')$  and for any controllable action  $\sigma \in \Sigma_c$ ,  $(s\sigma \in L \wedge s'\sigma \in K) \Rightarrow s\sigma \in K$ . •

Another classical theorem states that if every controllable action is observable, i.e. in the case  $\Sigma_c \subseteq \Sigma_o$ , then a prefix-closed language  $K$  is observable if and only if it is *normal* according to the definition below.

*Definition 6:* A prefix-closed language  $K \subseteq L(G)$  is normal w.r.t.  $L(G)$  and  $\Sigma_o$  if  $\pi_o^{-1} \circ \pi_o(K) \cap L(G) \subseteq K$ . •

Prefix-closedness, controllability, and normality are preserved under arbitrary unions of languages. Using this fact, one may show that under the assumption  $\Sigma_c \subseteq \Sigma_o$ , any language  $K$  has a supremal prefix-closed, controllable and observable sublanguage. Using Remark 2, it should moreover be clear that a formal solution to Problem 1 is obtained by defining  $L(G/f)$  as the union  $K^\dagger$  of all languages  $K \subseteq L(G)$  such that  $K$  is prefix-closed, controllable and observable, and  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . Unfortunately, this does not indicate that  $f$  or  $L(G/f)$  can be effectively computed. It should moreover be noted that, if the conditions stated above are fulfilled only for  $K = \emptyset$  (the empty language that does not even contain the empty word), then there exists no solution  $f$  to Problem 1 (because  $\varepsilon \in L(G/f)$  for any  $f$ ). The opacity of  $S$  w.r.t.  $L(G) \cap \Sigma_{uc}^*$  and  $\Sigma_a$  is a sufficient condition to the existence of solutions to Problem 1.<sup>4</sup> A list of alternative conditions under which effective algorithms have been proposed for computing the most permissive opacity control is stated in the following proposition.

<sup>4</sup>This condition is not necessary. For instance, let  $L(G) = \varepsilon + (x+c)(a+\varepsilon)$  and  $S = xa$  with  $\Sigma_a = \{a\}$ ,  $\Sigma_c = \{c\}$ , and  $\Sigma_o = \{x, c\}$ , then  $S$  is not opaque w.r.t.  $L(G) \cap \Sigma_{uc}^*$  but it is opaque w.r.t.  $L(G)$ !

*Proposition 2:* Given a finite transition system  $G$  and a regular secret  $S$ , assume that  $S$  is opaque w.r.t.  $L(G) \cap \Sigma_{uc}^*$  and  $\Sigma_a$ . In each of the following four situations, one can effectively compute the most permissive control  $f$  enforcing the opacity of  $S$  w.r.t.  $L(G/f)$  and  $\Sigma_a$ :

- 1)  $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma_a \subseteq \Sigma$  [24];
- 2)  $\Sigma_o = \Sigma$  and  $(\forall s, s' \in L(G))(\forall \sigma \in \Sigma_{uc} \cap \Sigma_a) s \sim_a s' \wedge s\sigma \in L(G) \Rightarrow s'\sigma \in L(G)$  [23];
- 3)  $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_o \subseteq \Sigma$  [24];
- 4)  $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_o \subseteq \Sigma$  [24]. ■

In cases (1), (2) and (3),  $L(G/f)$  is obtained by computing first the supremal sublanguage  $SupOp(L(G))$  of  $L(G)$  with respect to which  $S$  is opaque (given by Proposition 1), and next the supremal controllable and normal sublanguage  $SupCoNo(SupOp(L(G)))$  of the former. In case (4), on account of the assumption that the adversary knows exactly the control map, it would be necessary to iterate further the two operations  $SupOp$  and  $SupCoNo$  in alternation, but it has been shown in [24] that such an iteration may not stabilize, and a non-trivial adaptation of Ramadge and Wonham's methods was in fact necessary to compute  $L(G/f)$  with an algorithm that always terminates.

In the rest of the paper, we assume that  $\Sigma_c \subseteq \Sigma_o$  as in all cases listed in Proposition 2. We also suppose that  $\Sigma_a \subseteq \Sigma_o$ , meaning that a controller has at least as precise information as the adversary on the actual run of  $G$ . But we *do not suppose* that  $\Sigma_c$  and  $\Sigma_a$  are comparable, thus jointly extending conditions (2), (3) and (4) of Proposition 2. The exact problem which we want to solve may finally be stated as follows:

*Problem 2 (Opacity Control Problem):* Given a finite deterministic transition system  $G$  labeled over  $\Sigma$ , a regular subset  $S \subseteq \Sigma^*$  (the secret), a subalphabet  $\Sigma_o \subseteq \Sigma$  (the actions which a controller can observe), and two arbitrary subalphabets  $\Sigma_c \subseteq \Sigma_o$  (the actions which a controller can disable) and  $\Sigma_a \subseteq \Sigma_o$  (the actions visible to the adversary), let  $\mathcal{K}$  denote the set of non-empty prefix-closed sublanguages  $K$  of  $L(G)$  such that  $K$  is controllable w.r.t.  $L(G)$  and  $\Sigma_c$ ,  $K$  is normal w.r.t.  $L(G)$  and  $\Sigma_o$ , and  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . The problem is twofold.

- i) Show that it is decidable whether  $\mathcal{K}$  is empty.
- ii) Show that for non-empty  $\mathcal{K}$ , the union  $\bigcup \mathcal{K}$  of all languages in  $\mathcal{K}$  is a regular language  $K^\dagger$ , and construct from  $G$  and  $S$  a finite state machine generating  $K^\dagger$ .

At this stage, we would like to indicate briefly potential applications. In the security literature, opacity is usually considered in the context of infinite systems, including for instance unbounded Petri nets. As opacity is not decidable in general [11], the best one can do is often model-checking systems for sufficient opacity conditions. A complementary technique would be to enforce opacity by supervisory control on finite abstractions of infinite systems and then to lift control to the original systems for subsequent model-checking. A field of direct application of regular opacity control is the guidance of semi-autonomous agents traveling through finite networks, with the objective of preventing current positions from being known to adversaries that receive partial information from sensors. The technical development presented in Section V will be illustrated with a reduced example of this kind.

### III. A REDUCTION TO OPACITY CONTROL UNDER FULL OBSERVATION

We show in this section that solving the Opacity Control Problem under the assumption  $\Sigma = \Sigma_o$  (full observation) induces a general solution of the Opacity Control Problem. The parameter  $\Sigma_o$  of the Opacity Control Problem will therefore be eliminated from the subsequent sections where the problem is afforded a solution.

Define  $\mathcal{F}(\Sigma, L(G), S, \Sigma_c, \Sigma_o, \Sigma_a) = \mathcal{K}$  (the set specified in the statement of the Opacity Control Problem). Let  $\mathcal{K}' = \mathcal{F}(\Sigma_o, \pi_o(L(G)), S', \Sigma_c, \Sigma_o, \Sigma_a)$  where  $S' = \pi_o(S) \setminus \pi_o(L(G) \setminus S)$ . Note that  $S' \subseteq \pi_o(S)$ , and  $S'$  is a regular language.

*Proposition 3:* Let  $K \in \mathcal{K}$ , then  $\pi_o(K) \in \mathcal{K}'$ .

*Proof:*  $K \subseteq L(G) \Rightarrow \pi_o(K) \subseteq \pi_o(L(G))$ , and the non-emptiness and prefix-closeness of  $K$  entail similar properties for  $\pi_o(K)$ .

We show that  $\pi_o(K)$  is controllable w.r.t.  $\pi_o(L(G))$  and  $\Sigma_c$ . Let  $w' \in \pi_o(K)$  and  $\sigma \in \Sigma_o \setminus \Sigma_c$  such that  $w'\sigma \in \pi_o(L(G))$ . As  $L(G)$  is prefix-closed,  $w' = \pi_o(w)$  for some  $w \in L(G)$  such that  $w\sigma \in L(G)$ . Let  $w' = \pi_o(v)$  for some  $v \in K$ . Then  $\pi_o(v) = \pi_o(w)$ . As  $K$  is normal w.r.t.  $L(G)$  and  $\Sigma_o$ ,  $w \in L(G) \wedge v \in K \Rightarrow w \in K$ . As  $w\sigma \in L(G)$  and  $\sigma \notin \Sigma_c$ ,  $w\sigma \in K$  by controllability of  $K$ . Therefore,  $w'\sigma \in \pi_o(K)$  as required.

The projected language  $\pi_o(K)$  is certainly normal w.r.t.  $\pi_o(L(G))$  and  $\Sigma_o$ , because  $\pi_o(K) \subseteq \pi_o(L(G)) \subseteq \Sigma_o^*$ .

We show finally that  $S'$  is opaque w.r.t.  $\pi_o(K)$  and  $\Sigma_a$ . Let  $w' \in S' \cap \pi_o(K)$ , then  $w' = \pi_o(w)$  for some  $w \in K \subseteq L(G)$  and by definition of  $S'$ ,  $w \in S$ . As  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ ,  $\pi_a(w) = \pi_a(v)$  for some  $v \in K \setminus S$ . Let  $v' = \pi_o(v)$ , then  $\pi_a(w') = \pi_a(w) = \pi_a(v) = \pi_a(v')$ ,  $v' \in \pi_o(K)$ , and  $v' \notin S'$  because  $v \in \pi_o^{-1}(v')$ ,  $v \in K \subseteq L(G)$ , and  $v \notin S$ . ■

*Proposition 4:* Let  $K' \in \mathcal{K}'$ , then  $\pi_o^{-1}(K') \cap L(G) \in \mathcal{K}$ .

*Proof:* Let  $K = \pi_o^{-1}(K') \cap L(G)$ , then  $K \subseteq L(G)$ ,  $K$  is prefix-closed because  $K'$  and  $L(G)$  are prefix-closed, and  $K \neq \emptyset$  because  $K'$  is a non-empty subset of  $\pi_o(L(G))$ .

We show that  $K$  is controllable w.r.t.  $L(G)$  and  $\Sigma_c$ . Let  $w\sigma \in L(G)$  with  $w \in K$  and  $\sigma \in \Sigma \setminus \Sigma_c$ . If  $\sigma \notin \Sigma_o$ , then  $w\sigma \in K$  because  $\pi_o(w\sigma) = \pi_o(w) \in K'$ . Suppose now that  $\sigma \in \Sigma_o$ . Then  $w' = \pi_o(w)$  belongs to  $K'$ ,  $\sigma \in \Sigma_o \setminus \Sigma_c$ , and  $w'\sigma \in \pi_o(L(G))$ . As  $K'$  is controllable w.r.t.  $\pi_o(L(G))$  and  $\Sigma_c$ ,  $w'\sigma \in K'$ . Therefore,  $w\sigma \in \pi_o^{-1}(K')$ , and since  $w\sigma \in L(G)$ ,  $w\sigma \in K$  as required.

It follows directly from the definition  $K = \pi_o^{-1}(K') \cap L(G)$  that  $K$  is normal w.r.t.  $L(G)$  and  $\Sigma_o$ .

We show finally that  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . Let  $w \in S \cap K$  and let  $w' = \pi_o(w)$ , then  $w' \in K'$ , hence  $\pi_o^{-1}(w') \cap L(G) \subseteq K$ . If  $\pi_o^{-1}(w') \cap L(G)$  is not included in  $S$ , then  $\pi_o(v) = \pi_o(w)$  and thus  $\pi_a(v) = \pi_a(w)$  for some  $v \in \pi_o^{-1}(w') \cap (K \setminus S)$ . If  $\pi_o^{-1}(w') \cap L(G)$  is included in  $S$ , then  $w' \in S'$  by definition of this set. As  $S'$  is opaque w.r.t.  $K'$  and  $\Sigma_a$ ,  $\pi_a(w') = \pi_a(v')$  for some  $v' \in K' \setminus S'$ . By definition of  $S'$ ,  $v' = \pi_o(v)$  for some  $v \in L(G) \setminus S$ . Now  $v \in \pi_o^{-1}(v')$  and  $v' \in K'$ , hence  $\pi_o^{-1}(v') \cap L(G) \subseteq K$  by definition of  $K$ . Therefore,  $v \in K \setminus S$ . Finally,  $\pi_a(v) = \pi_a(w)$  because  $\pi_a(w) = \pi_a(w')$  and  $\pi_a(v) = \pi_a(v')$ . ■

*Proposition 5:* The Opacity Control Problem with the parameters  $(\Sigma, L(G), S, \Sigma_c, \Sigma_o, \Sigma_a)$  is equivalent to the same problem with the parameters  $(\Sigma_o, \pi_o(L(G)), S', \Sigma_c, \Sigma_o, \Sigma_a)$ .

*Proof:* In view of Propositions 3 and 4,  $\mathcal{K} \neq \emptyset$  if and only if  $\mathcal{K}' \neq \emptyset$ . Note that both operations  $\pi_o(\cdot) : \mathcal{K} \rightarrow \mathcal{K}'$  and  $\pi_o^{-1}(\cdot) \cap L(G) : \mathcal{K}' \rightarrow \mathcal{K}$  are monotone. Moreover, the following relations hold for all  $K \in \mathcal{K}$  and  $K' \in \mathcal{K}'$  (establishing thus a Galois connection between  $\mathcal{K}$  and  $\mathcal{K}'$ ):

- $\pi_o(K) \subseteq K' \Rightarrow K \subseteq \pi_o^{-1}(K') \cap L(G)$ ;
- $K \subseteq \pi_o^{-1}(K') \cap L(G) \Rightarrow \pi_o(K) \subseteq K'$ ;
- $K \subseteq \pi_o^{-1} \circ \pi_o(K) \cap L(G)$ ;
- $K' = \pi_o(\pi_o^{-1}(K') \cap L(G))$ .

One deduces the following. If  $\cup \mathcal{K} = K^\dagger \in \mathcal{K}$  then for any  $K' \in \mathcal{K}'$ ,  $\pi_o^{-1}(K') \cap L(G) \subseteq K^\dagger \subseteq \pi_o^{-1} \circ \pi_o(K^\dagger) \cap L(G)$ , hence  $\pi_o(\pi_o^{-1}(K') \cap L(G)) = K' \subseteq \pi_o(K^\dagger)$ . If  $\cup \mathcal{K}' = K'^\dagger \in \mathcal{K}'$  then for any  $K \in \mathcal{K}$ ,  $\pi_o(K) \subseteq (K'^\dagger)$ , hence  $K \subseteq \pi_o^{-1}(K'^\dagger) \cap L(G)$ . Thus,  $\cup \mathcal{K} \in \mathcal{K}$  if and only if  $\cup \mathcal{K}' \in \mathcal{K}'$ . As both operators  $\pi_o(\cdot)$  and  $\pi_o^{-1}(\cdot) \cap L(G)$  preserve regular languages, the proof of the proposition follows. ■

Based on Proposition 5, whenever  $\Sigma_a \subseteq \Sigma_o \subseteq \Sigma$ , one can reformulate the opacity control problem in terms of the abstract system induced by the observation map  $\pi_o$  and a new secret  $S'$  derived from  $S$ , solve the problem in this abstract setting, and lift up the solution  $K'^\dagger$  to the original setting as  $K^\dagger = \pi_o^{-1}(K'^\dagger) \cap L(G)$ .

### IV. AN INFORMAL PRESENTATION OF THE CONSTRUCTIONS

In this section, we sketch the intuitions under the methods that will be employed for solving Problem 2. Henceforth, based on Proposition 5, we assume w.l.o.g. that  $\Sigma = \Sigma_o$ . Moreover, we assume that the transition system  $G$  recognizes the secret  $S$ , i.e.  $G = (Q, \Sigma, \delta, q_0, Q_S)$  such that for any  $s \in \Sigma^*$ ,  $s \in L(G)$  iff  $\delta(q_0, s)$  is defined and  $s \in S$  iff  $\delta(q_0, s) \in Q_S$ . This second condition, even though it does not hold for arbitrary  $G$  and  $S$ , always holds for the parallel composition of  $G$  and a complete deterministic automaton recognizing  $S$ .

Let  $\mathcal{K}$  denote the set of all *non-empty* prefix-closed and controllable languages  $K \subseteq L(G)$  such that  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . It was observed in Section II that, if  $\mathcal{K}$  differs from the empty set, then  $K^\dagger = \cup \mathcal{K}$  is in  $\mathcal{K}$ . We want to construct a finite automaton  $A^\dagger$  that generates  $K^\dagger$ , showing that this language is regular and providing as a by-product the most permissive control  $f^\dagger$  enforcing the opacity of  $S$  w.r.t.  $L(G)/f^\dagger$  and  $\Sigma_a$ . In case when  $K^\dagger = \emptyset$ , no control  $f$  on  $G$  can enforce the opacity of  $S$ . In order to avoid this special case, we assume henceforth that  $q_0 \notin Q_S$  and  $\delta(q_0, \sigma)$  is defined only for  $\sigma \in (\Sigma_c \cap \Sigma_a)$ . This property may always be enforced on  $G$  by adding if necessary a dummy transition from a dummy initial state to the actual initial state of  $G$ . In this way, the condition  $K^\dagger \neq \emptyset$  is replaced equivalently with the condition  $K^\dagger \neq \{\varepsilon\}$ . The latter condition may of course be decided upon from any automaton  $A^\dagger$  generating  $K^\dagger$ .

As  $S$  is the set of traces of  $G$  recognized by the accepting states in  $Q_S$ , a trace  $s \in \Sigma^*$  discloses the secret  $S$  to the adversary if  $\delta(q_0, [s]_a) \subseteq Q_S$ . Therefore,  $K^\dagger = L(G)$  if  $\delta(q_0, [s]_a) \not\subseteq Q_S$  for every  $s \in S$ . The usual method for testing this is to construct an LTS  $Det_a(G) = (\mathcal{X}, \Sigma_a, \Delta, X_0)$  as indicated in Definition 1, and to check that no reachable state

$X \in \mathcal{X}$  is a subset of  $Q_S$ . In case when  $X \subseteq Q_S$  for some  $X \in \mathcal{X}$ , a first and necessary step towards computing  $K^\dagger$  is to compute the LTS  $A = G \times Det_a(G)$ , thus matching the system  $G$  with the dynamic estimation of its current state by the adversary. This LTS, equipped with the set of accepting states  $Q \times 2^{Q_S}$ , recognizes exactly the set of traces which disclose the secret  $S$ .

Let us explain more precisely the contribution brought by the LTS  $A$  to the construction of a control map  $f$  enforcing the opacity of  $S$ . Define  $\Sigma_{ua} = \Sigma \setminus \Sigma_a$ . The initial state of  $A$  is the pair  $(q_0, E_0)$  defined with  $E_0 = \delta(q_0, \Sigma_{ua}^*)$ , and the reachability set of  $A$  is the inductive closure of the set  $\{(q_0, E_0)\}$  under the partial transition map  $\delta((q, E), \sigma) = (\delta(q, \sigma), \delta(E, \Sigma_{ua}^* \sigma \Sigma_{ua}^*))$  (where  $E \subseteq Q$ ). When a run with trace  $s$  is performed in  $G$ , the matching run of  $A$  leads to a pair  $(q, E)$  where  $q$  is the state reached by  $G$  and  $E = \delta(q_0, [s]_a)$  is the most accurate estimate of  $q$  that can be computed by an adversary knowing  $G$  and  $\pi_a(s)$ . Therefore, the adversary can disclose the secret if and only if  $A$  has at least one *loosing path*, that is a path leading to a *loosing configuration*  $(q, E)$  such that  $E \subseteq Q_S$ . Here and in the sequel, the term *configuration* is used as a synonym of state for all automata with sets of states included in  $Q \times 2^Q$ , while the term *state* is reserved to the elements of  $Q$ .

Suppose the adversary can win. One must impose on  $G$  some control  $f$  such that, for any  $s$  in  $L(G)$  and for any action  $\sigma_c \in \Sigma_c$ , if  $s \in L(G/f)$  and  $\delta((q_0, E_0), s\sigma_c u)$  is a loosing configuration of  $A$  for some  $u \in \Sigma_{uc}^*$ , then  $\sigma_c \notin f(s)$ . Traces  $s \in L(G)$  are fully observable, hence the corresponding configurations  $\delta((q_0, E_0), s)$  of  $A$  may be used to determine the values of  $f(s)$ . It suffices indeed to track controllable actions  $\sigma_c$  backwards from loosing configurations on acyclic paths of  $A$ , and to define  $f$  such that the last controllable transition is disabled on each loosing path. Then, no loosing configuration of  $A$  can be reached in  $A/f$ , and by construction,  $L(G/f) = L(A/f)$  is a superset of  $K^\dagger$ .

However,  $L(A/f)$  may be larger than  $K^\dagger$ , because the configuration  $\delta((q_0, E_0), s) = (q, E)$  reached in  $A/f$  by a trace  $s \in L(G/f)$  does not always reflect the most accurate estimate of  $q = \delta(q_0, s)$  for an adversary knowing  $G$ ,  $f$  and  $\pi_a(s)$ . As  $L(G/f)$  is strictly smaller than  $L(G)$ ,  $E$  may indeed be strictly larger than  $\delta(q_0, [s]_a \cap L(G/f))$ , and if this smaller set is included in  $Q_S$ , then the adversary can still disclose the secret  $S$  in the controlled system  $G/f$  since he knows exactly the control map  $f$ .

At this stage, it would be helpful to compute an automaton  $A'$  generating  $L(G/f)$  such that each trace  $s \in L(G/f)$  leads from the initial state of  $A'$  to the configuration  $(q, E)$  given by  $q = \delta(q_0, s)$  and  $E = \delta(q_0, [s]_a \cap L(G/f))$ .<sup>5</sup> Unfortunately, such an automaton does not exist in general. One reason among others is that two traces  $s, s' \in L(G/f)$  inducing different control values  $f(s)$  and  $f(s')$  might nevertheless lead to the same configuration  $(q, E)$ , thus preventing  $A'$  from generating  $L(G/f)$ .

*Example 2:* To illustrate this point, consider again the system  $G$  and the secret  $S$  defined in Example 1. Assume now that

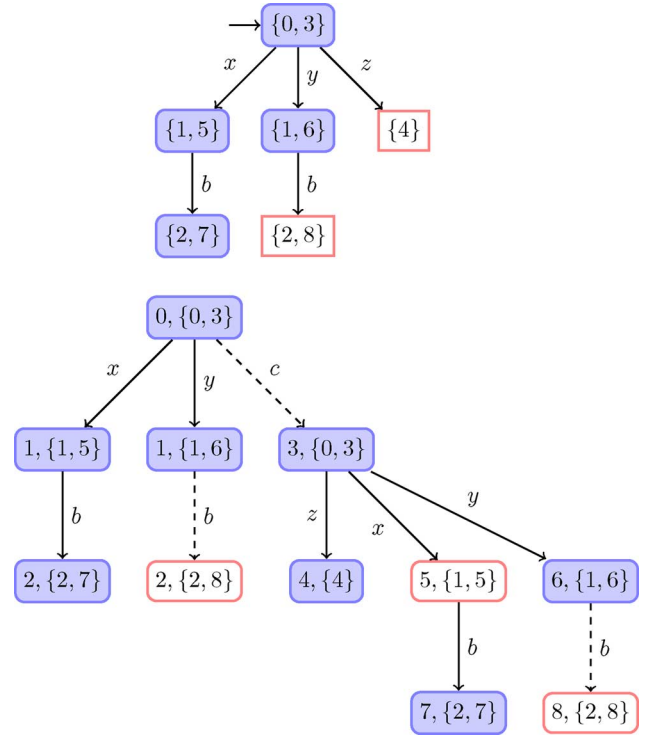


Fig. 2.  $Det_a(G)$  and  $A = G \times Det_a(G)$ .

$\Sigma_a = \{b, x, y, z\}$  and  $\Sigma_c = \{b, c\}$ . The LTSs  $Det_a(G)$  and  $A = G \times Det_a(G)$  are shown in Fig. 2.

The loosing configurations that disclose  $S$  are squared. The control  $f$  must disable the last controllable transition on each loosing path as indicated in dashed lines, hence  $f(\varepsilon) = \{x, y\}$ ,  $f(x) = \{b\}$ , and  $f(y) = \emptyset$ . In particular,  $x, y \in L(G/f)$  and  $f(x) \neq f(y)$ . However, an adversary who knows that  $G$  is controlled by  $f$  computes the same state estimate  $\{1\}$  for  $\delta(0, x)$  and  $\delta(0, y)$ . Now, if one lets accordingly  $\delta((0, \{0\}), x) = (1, \{1\}) = \delta((0, \{0\}), y)$  in  $A'$ , then one reaches a confused situation: on the one hand,  $\delta((1, \{1\}), b)$  should be defined because  $b \in f(x)$ , and on the other hand,  $\delta((1, \{1\}), b)$  should be undefined because  $b \notin f(y)$ !  $\diamond$

Furthermore, even if the construction of  $f$  from  $A$ , in a first stage, and of  $A'$  from  $f$  and  $A$ , in a second stage, were possible, the computation of  $f^\dagger$  would be very slow, as we explain below. Given a trace  $s \in L(G/f)$  with  $\delta((q_0, E_0), s) = (q, E)$ , suppose that  $\delta(q_0, [s]_a \cap L(G/f)) = E'$  such that  $E' \subset E$  and  $(q, E')$  is *not* a configuration of  $A$ . Suppose that from every state  $q' \in E'$ , exists in  $G$  exactly one transition  $\delta(q', \sigma) = q_S$  where  $q_S \in Q_S$  and all the transitions from states  $q' \in E'$  are labeled with the same action  $\sigma \in (\Sigma_a \cap \Sigma_c)$ . Although it was already evident from  $G$  (as opposed to  $G/f$ ) that, if an adversary ever gets the estimate  $E'$  of  $q = \delta(q, s)$ , then the action  $\sigma$  must be control-disabled after  $s$ , this fact is ignored in the definition of the control  $f$  since  $(q, E')$  is not a configuration of  $A$ .

*Example 3:* Back to example 2, one may see that if, after control  $f$  is imposed on  $G$ , the configuration  $(1, \{1\})$  may be reached in  $G/f$ , then the secret will be disclosed by the trace  $x.b$  in the controlled system  $G/f$ . It is thus ultimately necessary to disable  $b$  at configuration  $(1, \{1\})$  (which solves at the same time the confusion encountered in Example 2).  $\diamond$

<sup>5</sup>This was the construction used in [24] to deal with case (4) in Proposition 1.

In order to cope with these shortcomings, we will replace automata defined over subsets of  $Q \times 2^Q$  with partial maps  $d : Q \times 2^Q \times \Sigma \rightarrow Q$  such that either  $d(q, E, \sigma) = \delta(q, \sigma)$  or it is undefined. Each partial map  $d$  generates a corresponding automaton with the initial state  $(q_0, \{q_0\})$ . The reachability set of the generated automaton is the inductive closure of the set  $\{(q_0, \{q_0\})\}$  under the partial transition map  $\delta_d : (Q \times 2^Q) \times \Sigma \rightarrow (Q \times 2^Q)$  defined from  $d$  as follows. Let  $d : Q \times 2^Q \times \Sigma_{ua}^* \rightarrow Q$  be defined inductively with  $d(q, E, \sigma s) = d(d(q, E, \sigma), E, s)$  for  $\sigma \in \Sigma_{ua}$ . Let  $d$  denote also the additive extension of the latter map to sets of states and to languages. We define  $\delta_d((q, E), \sigma) = (q', E')$  with  $q' = d(q, E, \sigma)$ ,  $E' = E$  if  $\sigma \notin \Sigma_a$ , and  $E' = d(E, E, \Sigma_{ua}^* \sigma)$  otherwise.

Consider e.g. the automaton  $A = G \times \text{Det}_a(G)$ . We will replace this automaton with the partial transition map  $d_0$  defined with  $d_0(q, E, \sigma) = \delta(q, \sigma)$ . The map  $d_0$  generates from  $(q_0, \{q_0\})$  an automaton  $A_0$  isomorphic to  $A$ . The isomorphism maps each state  $(\delta(q_0, s), \delta(q_0, [s]_a))$  of  $A$  to a similar state  $(\delta(q_0, s), E)$  where the *condensed state estimate*  $E$  is equal to  $\{q_0\}$  if  $[s]_a = [\varepsilon]_a$  and otherwise to  $\delta(q_0, [s]_a \cap \Sigma_a^* \Sigma_a)$ . Note that the map  $d_0$  may also serve to generate sequences of transitions  $(q_i, E_i) \xrightarrow{\sigma_i} (q_{i+1}, E_{i+1})$  from initial configurations  $(q, E)$  that cannot be reached in  $A$  but may play a role *later on* during the iterative computation of the optimal control.

Continuing to mimic the approach in which a first control function  $f$  was derived from  $A$ , we will replace the computation of  $f$  by the computation of a partial map  $d_1 : Q \times 2^Q \times \Sigma \rightarrow Q$ , such that  $d_1(q, E, \sigma)$  will be either equal to  $d_0(q, E, \sigma)$  or undefined. The latter case will occur whenever  $\sigma$  is uncontrollable and the configuration  $(q', E') = \delta_{d_0}((q, E), \sigma)$  discloses the secret, i.e. when all states reached in  $G$  by firing sequences in  $\Sigma_{ua}^*$  from states in  $E'$  are in  $Q_S$ . The automaton generated by the map  $d_1$  from  $\{(q_0, \{q_0\})\}$  can now play the role that we had assigned to our ghost automaton  $A'$ .

Proceeding similarly from  $d_1$ , we will construct an inductive sequence of partial maps  $d_{i+1} = \phi(d_i)$  such that  $d_{i+1}(q, E, \sigma) = d_i(q, E, \sigma)$  or it is undefined. These maps induce a corresponding sequence of finite automata  $A_i$ , generated from  $\{(q_0, \{q_0\})\}$ , with decreasing languages  $L(A_i) \supseteq L(A_{i+1})$ . As there exist finitely many partial maps from  $Q \times 2^Q \times \Sigma$  to  $Q$ , the decreasing sequence  $(d_i)_{i \in \mathbb{N}}$  stabilizes, i.e.  $d_n = \phi(d_n)$  for some  $n$ . Let  $d^\dagger = d_n$  and  $A^\dagger = A_n$  for the least such  $n$ , then we will prove that  $L(A^\dagger) = K^\dagger$ , thus showing that the most permissive control  $f^\dagger$  enforcing the opacity of  $S$  is regular.<sup>6</sup>

We would like to complete this informal presentation of the results and constructions stated in Section V by explaining a little more the intuitions under state estimates  $E$  in pairs  $(q, E) \in Q \times 2^Q$ . In such pairs,  $E$  is best envisaged as the set of states that an adversary feels  $G$  may have reached *immediately after the last action*  $\sigma \in \Sigma_a$  he has observed in some trace  $w\sigma$  of  $G$  with partially observed prefix  $w$ . Suppose the adversary moreover knows that the control imposed on  $G$  after his last observed action in  $\Sigma_a$  agrees with  $d_i$ , i.e. that  $G$  behaves according to the sequences of transitions defined by  $\delta_{d_i}$  from

<sup>6</sup>We recall that  $K^\dagger$  is always non-empty owing to the assumptions made in the beginning of this section.

the configuration  $(q, E)$ . Then, the estimate  $E$  provides him enough information to determine all states that might have been reached in  $G$  under the same control by executing arbitrary traces  $w'\sigma s$  such that  $w \sim_a w'$  and  $s \in \Sigma_{ua}^*$ . The reason why we have chosen to work with condensed state estimates is that this facilitates greatly the construction of the partial transition map  $\delta_d$  from the control map  $d$ . For any  $s \in L(G/f^\dagger)$ , the state estimate  $E$  in  $\delta_{d^\dagger}((q_0, \{q_0\}), s) = (q, E)$  is indeed the set of states that an adversary believes  $G$  may have reached under the control  $f^\dagger$  after the last action  $\sigma \in \Sigma_a$  in the trace  $s$ .

Finally, we would like to give intuitions about the structure of the automaton  $A^\dagger$ . The set of reachable states of this automaton may be partitioned into disjoint subsets  $\mathcal{Q}(E)$  such that  $(q', E') \in \mathcal{Q}(E)$  if and only if  $E = E'$ . For each  $E$ , the full restriction of  $A^\dagger$  induced on  $\mathcal{Q}(E)$  is isomorphic to a subautomaton of  $G$ , where the isomorphism maps  $(q, E) \xrightarrow{\sigma} (q', E)$  to  $q \xrightarrow{\sigma} q'$ . The automaton  $A^\dagger$  may therefore be seen as a mode automaton, with one mode per estimate  $E$ . In each mode  $E$ , the automaton  $A^\dagger$  exerts *state based* control on  $G$  until, at some state  $(q, E)$ , it enables some action  $\sigma \in \Sigma_a$  which the adversary is aware of. Then  $A^\dagger$  jumps to a new mode  $E'$ , reflecting the new condensed estimate gained by the adversary, and it enters  $\mathcal{Q}(E')$  at  $(q', E')$  such that  $q \xrightarrow{\sigma} q'$  in  $G$ . The new mode is  $E' = d^\dagger(E, E, \Sigma_{ua}^* \sigma)$ . Note that the adversary's view of  $G/f^\dagger$  is isomorphic to the quotient of  $A^\dagger$  obtained by removing first all transitions with labels in  $\Sigma_{ua}$  and then collapsing  $\mathcal{Q}(E)$  to a single state for each mode  $E$ .

## V. THE TECHNICAL DEVELOPMENT

Let  $G = (Q, \Sigma, \delta, q_0, Q_S)$  with  $q_0 \notin Q_S$  and  $\delta(q_0, \sigma)$  undefined for  $\sigma \in \Sigma \setminus (\Sigma_a \cap \Sigma_c)$ ,  $L(G) = \{s \in \Sigma^* \mid \delta(q_0, s) \text{ defined}\}$  and  $S = \{s \in \Sigma^* \mid \delta(q_0, s) \in Q_S\}$ . Throughout the section,  $d : Q \times 2^Q \times \Sigma \rightarrow Q$  denotes a partial map such that  $d(q, E, \sigma)$  is either equal to  $\delta(q, \sigma)$  or undefined. Let us recall the following.

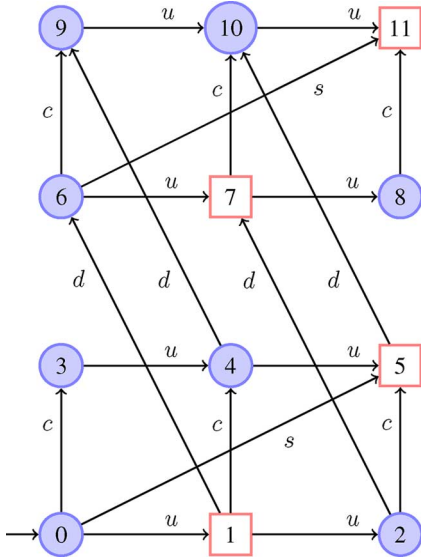
*Definition 7:* Given  $d : Q \times 2^Q \times \Sigma \rightarrow Q$ , define inductively  $d(q, E, \varepsilon) = q$  and  $d(q, E, \sigma s) = d(d(q, E, \sigma), E, s)$  for  $\sigma \in \Sigma_{ua}$  and  $s \neq \varepsilon$ . Let  $\delta_d : (Q \times 2^Q) \times \Sigma \rightarrow (Q \times 2^Q)$  be the partial transition map on  $Q \times 2^Q$  such that  $\delta_d((q, E), \sigma) = (q', E')$  is given by  $q' = d(q, E, \sigma)$ ,  $E' = E$  if  $\sigma \notin \Sigma_a$ , and  $E' = d(E, E, \Sigma_{ua}^* \sigma)$  otherwise. For  $s \in \Sigma^*$ , define inductively  $\delta_d((q, E), \varepsilon) = (q, E)$  and  $\delta_d((q, E), \sigma s) = \delta_d(\delta_d((q, E), s), \sigma)$ . Let  $\mathcal{A}(d)$  denote the automaton which is generated by the partial transition map  $\delta_d$  from the initial state  $(q_0, \{q_0\})$ . Thus  $\mathcal{A}(d) = (\Theta_d, \Sigma, \delta_d, (q_0, \{q_0\}))$  where  $\Theta_d$  is the closure of the set  $\{(q_0, \{q_0\})\}$  under  $\delta_d$ . Finally let  $\mathcal{L}(d) = L(\mathcal{A}(d))$ . •

*Remark 4:*  $\mathcal{L}(d) \subseteq L(G)$  by definition of  $\delta_d$ . •

*Remark 5:* If  $d' \subseteq d$  in the sense that  $d(q, E, \sigma)$  is defined and equal to  $d'(q, E, \sigma)$  whenever the latter is defined, then for any  $s \in \Sigma^*$ ,  $\delta_{d'}((q, E), s) = (q', E')$  entails that  $\delta_d((q, E), s) = (q', E'')$  for some  $E'' \supseteq E'$ . Therefore,  $d' \subseteq d \Rightarrow \mathcal{L}(d') \subseteq \mathcal{L}(d)$ . •

*Example 4:* Let  $G$  be the transition system depicted in Fig. 3, with  $Q = \{0, \dots, 11\}$ ,  $q_0 = 0$ ,  $Q_S = \{1, 5, 7, 11\}$ , and  $\Sigma = \{u, d, c, s\}$ .

Let  $\delta : Q \times \Sigma \rightarrow Q$  be the partial map defined according to the arcs of the picture.  $G$  may be seen as a representation

Fig. 3.  $G$ .

of all sequences of possible moves of an agent in a three storey building with a south wing and a north wing, both equipped with lifts and both connected by a corridor at each floor. Moreover, there is a staircase that leads from the first floor in the south wing to the third floor in the north wing. The agent starts from the first floor in the south wing. He can walk up the stairs ( $s$ ) or walk through the corridors ( $c$ ) from south to north without any control. The lifts can be used several times one floor upwards ( $u$ ) and at most once one floor downwards ( $d$ ) altogether. The moves of the lifts are controllable. Thus  $\Sigma_c = \{u, d\}$  and  $\Sigma_{uc} = \{s, c\}$ . The secret is that the agent is either at the second floor in the south wing or at the third floor in the north wing. The adversary may gather the exact subsequence of moves in  $\Sigma_a = \{s, c, u\}$  from sensors, but he cannot observe the upwards moves of the lifts. Thus  $\Sigma_a = \{u, c, s\}$  and  $\Sigma_{ua} = \{d\}$ .

Let  $d_0 : Q \times 2^Q \times \Sigma \rightarrow Q$  be the map defined with  $d_0(q, E, \sigma) = \delta(q, \sigma)$ . The automaton  $\mathcal{A}(d_0)$  defined from  $G$  is depicted in Fig. 4. Each configuration  $(q, E)$  of the automaton is represented with  $q$  in the first line and with the list of elements of  $E$  in the second line. For instance

$$\begin{aligned} \delta_{d_0}((1, \{1\}), c) &= (d_0(1, \{1\}, c), d_0(\{1\}, \{1\}, d^*c)) \\ &= (4, \{4, 9\}). \end{aligned}$$

In fact,

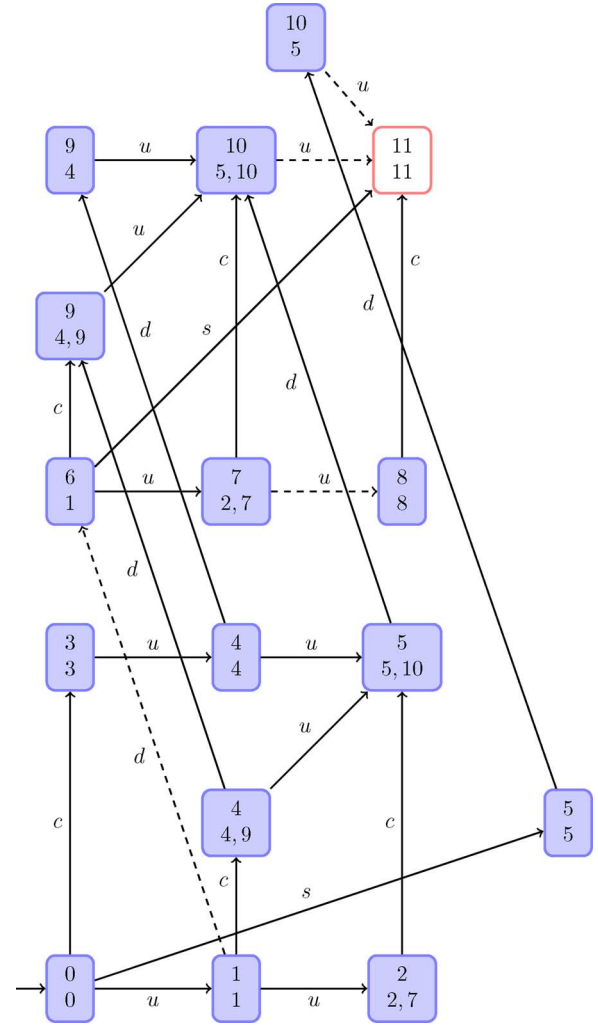
- $d_0(1, \{1\}, c) = \delta(1, c) = 4$ ;
- $d_0(1, \{1\}, d) = \delta(1, d) = 6$ ;
- $d_0(1, \{1\}, dc) = d_0(d_0(1, \{1\}, d), \{1\}, c) = \delta(6, c) = 9$ .

Similarly

$$\begin{aligned} \delta_{d_0}((4, \{4, 9\}), u) &= (d_0(4, \{4, 9\}, u), d_0(\{4, 9\}, \{4, 9\}, d^*u)) \\ &= (5, \{5, 10\}) \end{aligned}$$

because

$$\begin{aligned} d_0(4, \{4, 9\}, d^*u) &= \{d_0(4, \{4, 9\}, u), d_0(d_0(4, \{4, 9\}, d), \{4, 9\}, u)\} \\ &= \{\delta(4, u), \delta(\delta(4, d), u)\} = \{5, 10\}. \end{aligned}$$

Fig. 4.  $\mathcal{A}_{d_0}$ .

In contrast

$$\begin{aligned} \delta_{d_0}((4, \{4\}), d) &= (d_0(4, \{4\}, d), \{4\}) \\ &= (\delta(4, d), \{4\}) = (9, \{4\}) \end{aligned}$$

because  $d \in \Sigma_{ua}$ .  $\diamond$

All other examples presented in the section are continuations of Example 4.

Throughout the section, we let  $\theta = (\tilde{q}, \tilde{E}) \in Q \times 2^Q$  with  $\tilde{q} \in d(\tilde{E}, \tilde{E}, \Sigma_{ua}^*)$ . The following three lemmas show that if a sequence of transitions  $(q_i, E_i) \xrightarrow{\sigma_i} (q_{i+1}, E_{i+1})$ ,  $i = 1 \dots n$ , is generated from  $(q_1, E_1) = (\tilde{q}, \tilde{E})$  using the transition map  $\delta_d$ , then for each  $i$ ,  $d(E_{i+1}, E_{i+1}, \Sigma_{ua}^*)$  is the best estimate of the state  $q_i = \delta(\tilde{q}, \sigma_1 \dots \sigma_i)$  that the adversary can obtain from  $\pi_a(\sigma_1 \dots \sigma_i)$  and the knowledge that  $\tilde{q} \in d(\tilde{E}, \tilde{E}, \Sigma_{ua}^*)$ .

**Lemma 1:** Let  $\delta_d(\theta, s) = (q, E)$  and  $\delta_d(\theta, s') = (q', E')$ , then  $\pi_a(s) = \pi_a(s') \Rightarrow E = E'$ .

**Proof:** We use an induction on the length of  $\pi_a(s)$ . If this length is zero, i.e.  $s, s' \in \Sigma_{ua}^*$ , then  $E = \tilde{E}$  and  $E' = \tilde{E}$  by Definition 7, hence  $E = E'$ . Assume now that the lemma holds when  $\pi_a(s)$  has length  $n$ , and consider  $s, s' \in \Sigma^*$  with  $\pi_a(s) = \pi_a(s') \in \Sigma_a^{n+1}$ . Let  $s = s_1 \sigma_a s_2$  and  $s' = s'_1 \sigma_a s'_2$  such



that  $\pi_a(s_1) = \pi_a(s'_1) \in \Sigma_a^n$  and  $\sigma_a \in \Sigma_a$ , hence  $s_2, s'_2 \in \Sigma_{ua}^*$ . By the induction hypothesis, if we let  $\delta_d(\theta, s_1) = (q_1, E_1)$  and  $\delta_d(\theta, s'_1) = (q'_1, E'_1)$ , then  $E_1 = E'_1$ . By Definition 7, if we let  $\delta_d(\theta, s_1\sigma_a) = (q_2, E_2)$  and  $\delta_d(\theta, s'_1\sigma_a) = (q'_2, E'_2)$  then  $E_2 = d(E_1, E_1, \Sigma_{ua}^*\sigma_a)$  and  $E'_2 = d(E'_1, E'_1, \Sigma_{ua}^*\sigma_a)$ , hence  $E_2 = E'_2$ . Now  $(q, E) = \delta_d((q_2, E_2), s_2)$  and  $(q', E') = \delta_d((q'_2, E'_2), s'_2)$ . As  $s_2, s'_2 \in \Sigma_{ua}^*$ , by Definition 7,  $E = E_2$  and  $E' = E'_2$ , hence  $E = E'$ . ■

*Example 5:* To illustrate the previous lemma, we can see that  $\delta_{d_0}((4, \{4\}), u) = (5, \{5, 10\})$  and

$$\begin{aligned} \delta_{d_0}((4, \{4\}), du) &= \delta_{d_0}(\delta_{d_0}((4, \{4\}), d), u) \\ &= \delta_{d_0}((9, \{4\}), u) = (10, \{5, 10\}) \end{aligned}$$

have the same condensed state estimate. ◇

*Lemma 2:*  $\delta_d(\theta, s) = (q, E) \Rightarrow q \in d(E, E, \Sigma_{ua}^*)$ .

*Proof:* If  $s = \varepsilon$ , then  $\delta_d(\theta, s) = \theta$  and the property to show coincides with the assumed property of  $\theta$ . If  $s = s'\sigma$  with  $\sigma \in \Sigma$ , let  $\delta_d(\theta, s') = (q', E')$ . One may assume by induction on words that  $q' \in d(E', E', \Sigma_{ua}^*)$ , thus  $q' = d(q'', E', s'')$  for some  $q'' \in E'$  and  $s'' \in \Sigma_{ua}^*$ . Then  $q = d(q', E', \sigma) = d(d(q'', E', s''), E', \sigma) = d(q'', E', s''\sigma) \in d(E', E', \Sigma_{ua}^*\sigma)$ . One proceeds by cases. Suppose that  $\sigma \notin \Sigma_a$ , then  $\delta_d((q', E'), \sigma) = (q, E)$  entails  $E = E'$  and the desired result follows from  $q \in d(E', E', \Sigma_{ua}^*)$ . Suppose that  $\sigma \in \Sigma_a$ , then  $\delta_d((q', E'), \sigma) = (q, E)$  entails  $E = d(E', E', \Sigma_{ua}^*\sigma)$  and therefore  $q \in E$ . The desired result follows from  $E \subseteq d(E, E, \Sigma_{ua}^*)$ . ■

*Example 6:* Consider  $\delta_{d_0}((4, \{4\}), d) = (9, \{4\})$ , then  $d_0(4, \{4\}, d) = \delta(4, d) = 9$ , and  $9 \in d_0(\{4\}, \{4\}, d)$ . ◇

*Lemma 3:* Let  $\tilde{q} = q_0$  and  $\tilde{E} = \{q_0\}$ , then  $\delta_d(\theta, s) = (q, E)$  and  $q' \in d(E, E, \Sigma_{ua}^*)$  jointly entail  $\delta_d(\theta, s') = (q', E')$  for some  $s'$  such that  $\pi_a(s) = \pi_a(s')$ .

*Proof:* We use an induction on the length of  $\pi_a(s)$ . Let  $\pi_a(s)$  have length zero. By Definition 7,  $E = \tilde{E}$ , hence  $q' = d(q_0, \{q_0\}, s')$  for some  $s' \in \Sigma_{ua}^*$ , and  $\pi_a(s) = \pi_a(s')$ . By Definition 7,  $\delta_d(\theta, s') = (d(q_0, \{q_0\}, s'), \{q_0\}) = (q', E)$  as desired. Assume now that the proposition holds when  $\pi_a(s)$  has length  $n$ , and consider  $s \in \Sigma^*$  with  $\pi_a(s) \in \Sigma_a^{n+1}$ . Let  $s = s_1\sigma_a s_2$  such that  $\pi_a(s_1) \in \Sigma_a^n$  and  $\sigma_a \in \Sigma_a$ , hence  $s_2 \in \Sigma_{ua}^*$ . Let  $\delta_d(\theta, s_1) = (q_1, E_1)$ . By Definition 7,  $E = d(E_1, E_1, \Sigma_{ua}^*\sigma_a)$ . As  $q' \in d(E, E, \Sigma_{ua}^*)$ ,  $q' = d(q'_2, E, s'_2)$  for some  $q'_2 \in E$  and  $s'_2 \in \Sigma_{ua}^*$ . As  $E = d(E_1, E_1, \Sigma_{ua}^*\sigma_a)$ ,  $q'_2 = d(q'_1, E_1, s''_2\sigma_a)$  for some  $q'_1 \in E_1$  and  $s''_2 \in \Sigma_{ua}^*$ . As  $\pi_a(s_1)$  has length  $n$ , the induction hypothesis applies to  $\delta_d(\theta, s_1) = (q_1, E_1)$  and to  $q'_1 \in E_1 \subseteq d(E_1, E_1, \Sigma_{ua}^*)$ . Therefore,  $\delta_d(\theta, s'_1) = (q'_1, E_1)$  for some  $s'_1$  such that  $\pi_a(s_1) = \pi_a(s'_1)$ . Now  $\delta_d(\theta, s'_1 s''_2 \sigma_a) = (q'_2, d(E_1, E_1, \Sigma_{ua}^*\sigma_a)) = (q'_2, E)$ , hence  $\delta_d(\theta, s'_1 s''_2 \sigma_a s'_2) = (q', E)$ , establishing the lemma. ■

*Example 7:* We have  $\delta_{d_0}((4, \{4\}), u) = (5, \{5, 10\})$ ,  $10 \in d_0(\{5, 10\}, \{5, 10\}, \varepsilon)$ , and  $(10, \{5, 10\}) = \delta_{d_0}((4, \{4\}), ud)$ . ◇

We will now investigate which words in  $\mathcal{L}(d)$  actually disclose the secret  $S$  to the adversary, and how one can remedy these security failures. First, let us introduce a definition.

*Definition 8:* Given a partial map  $d : Q \times 2^Q \times \Sigma \rightarrow Q$ , let  $\mathcal{LE}(d) = \{E \subseteq Q \mid E \neq \emptyset \wedge d(E, E, \Sigma_{ua}^*) \subseteq Q_S\}$  be the

associated set of *loosing estimates*, and for any  $\theta = (\tilde{q}, \tilde{E})$  in  $Q \times 2^Q$  such that  $\tilde{q} \in d(\tilde{E}, \tilde{E}, \Sigma_{ua}^*)$ , let

$$\mathcal{LT}(d, \theta) = \{s \in \Sigma^* \mid \delta_d(\theta, s) \in Q \times \mathcal{LE}(d)\}$$

be the set of *loosing traces* w.r.t. state  $\tilde{q}$  of  $G$ , adversary's state estimate  $\tilde{E}$  and control  $d$ . •

*Example 8:* Given that  $d_0(E, E, \Sigma_{ua}^*) = E \cup \delta(E, d)$  for all  $E \subseteq Q$ ,  $\mathcal{LE}(d_0)$  contains exactly one losing estimate, namely the singleton set  $\{11\}$ . Thus for instance,  $ds, duuc, dcuu$  belong to  $\mathcal{LT}(d_0, (1, \{1\}))$ . Note that  $\{5\}$  is not a losing estimate, because  $d_0(5, \{5\}, d) = 10 \notin Q_S$ . ◇

The subset of words in  $\mathcal{L}(d)$  that disclose the secret may now be recognized by the automaton  $\mathcal{A}(d)$  (see Definition 7) equipped with the set of accepting states  $Q \times \mathcal{LE}(d)$ , as stated in the following proposition.

*Proposition 6:* For any  $s \in \mathcal{L}(d)$ ,  $[s]_a \cap \mathcal{L}(d) \subseteq S$  iff  $\delta_d((q_0, \{q_0\}), s) \in Q \times \mathcal{LE}(d)$ .

*Proof:* Let  $s \in \mathcal{L}(d)$  such that  $[s]_a \cap \mathcal{L}(d) \subseteq S$ . Let  $\delta_d((q_0, \{q_0\}), s) = (q, E)$ , and let  $q' \in d(E, E, \Sigma_{ua}^*)$ . By Lemma 3,  $(q', E) = \delta_d((q_0, \{q_0\}), s')$  for some  $s' \in [s]_a$ . As  $s' \in [s]_a$  and  $s' \in \mathcal{L}(d)$ , we have  $s' \in [s]_a \cap \mathcal{L}(d) \subseteq S$ . As  $\delta(q_0, s') = q'$ , it follows that  $q' \in Q_S$ . Therefore,  $E \in \mathcal{LE}(d)$ . To show the converse implication, let  $\delta_d((q_0, \{q_0\}), s) = (q, E) \in Q \times \mathcal{LE}(d)$ , hence  $d(E, E, \Sigma_{ua}^*) \subseteq Q_S$ . By Lemma 1, for any  $s' \in [s]_a \cap \mathcal{L}(d)$ , if we let  $q' = \delta(q_0, s')$  then  $\delta_d((q_0, \{q_0\}), s') = (q', E)$ . By Lemma 2,  $q' \in d(E, E, \Sigma_{ua}^*) \subseteq Q_S$ . Therefore,  $[s]_a \cap \mathcal{L}(d) \subseteq S$ . ■

*Proposition 7:* If  $\theta = \delta_d((q_0, \{q_0\}), \tilde{s})$ , then

$$\mathcal{LT}(d, \theta) = \{s \in \Sigma^* \mid \tilde{s}s \in \mathcal{L}(d) \wedge [\tilde{s}s]_a \cap \mathcal{L}(d) \subseteq S\}.$$

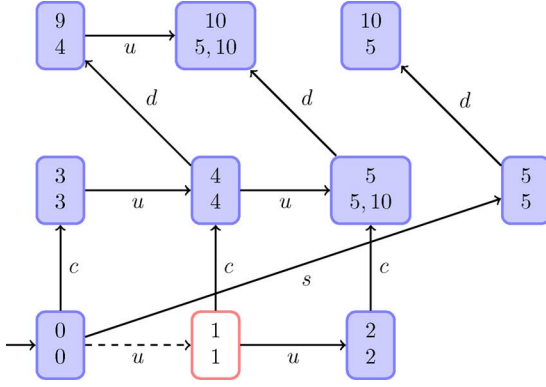
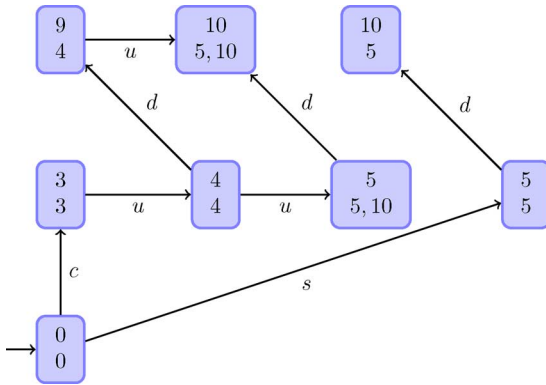
*Proof:* Let  $s \in \mathcal{LT}(d, \theta)$ , then by definition,  $\delta_d((q_0, \{q_0\}), \tilde{s}s) = (q, E)$  with  $E \in \mathcal{LE}(d)$ . By Proposition 6,  $[\tilde{s}s]_a \cap \mathcal{L}(d) \subseteq S$ . To prove the converse inclusion relation, consider  $s \in \Sigma^*$  such that  $\tilde{s}s \in \mathcal{L}(d)$  and  $[\tilde{s}s]_a \cap \mathcal{L}(d) \subseteq S$ . Let  $\delta_d((q_0, \{q_0\}), \tilde{s}s) = (q, E)$ . By Proposition 6,  $\delta_d((q_0, \{q_0\}), \tilde{s}s) \in Q \times \mathcal{LE}(d)$ , hence  $E \in \mathcal{LE}(d)$ . Now  $\delta_d((q_0, \{q_0\}), \tilde{s}s) = \delta_d(\theta, s)$ , hence  $s \in \mathcal{LT}(d, \theta)$ . ■

Proposition 7 tells us that, if  $\theta = (\tilde{q}, \tilde{E}) = \delta_d((q_0, \{q_0\}), \tilde{s})$  for some trace  $\tilde{s} \in L(G)$  and some partial map  $d : Q \times 2^Q \times \Sigma \rightarrow Q$ , then for any  $s \in \mathcal{LT}(d, \theta)$ , if an adversary gets the state estimate  $\tilde{E}$  immediately after the trace  $\tilde{s}$  has been executed in  $G \times \mathcal{A}(d)$ , then he can infer from the projection  $\pi_a(s)$  of the subsequent trace  $s$  executed in  $G \times \mathcal{A}(d)$  that  $\tilde{s}s$  is in the secret set  $S$ . More generally, even though the configuration  $(d, \theta)$  may not be reachable in  $\mathcal{A}(d)$ , if  $s \in \mathcal{LT}(d, \theta)$  and  $\theta = \delta_{d'}((q_0, \{q_0\}), \tilde{s})$  for some  $d' \subseteq d$  and  $\tilde{s} \in \Sigma^*$ , then  $s \in \mathcal{LT}(d', \theta)$ . The reasons why things should be so have been explained and illustrated in Section IV.

Based on Proposition 7, we immediately have the following.

*Corollary 1:* If  $\mathcal{LT}(d, \theta) = \emptyset$  for every configuration  $\theta = \delta_d((q_0, \{q_0\}), \tilde{s})$  reached in  $\mathcal{A}(d)$ , then  $\text{Discloser}(S, \mathcal{L}(d)) = \emptyset$ , i.e.  $S$  is opaque w.r.t.  $\mathcal{L}(d)$  and  $\Sigma_a$ .

We have now in hands all elements needed to compute  $d^\dagger : Q \times 2^Q \times \Sigma \rightarrow Q$  such that  $\text{Discloser}(S, \mathcal{L}(d^\dagger)) = \emptyset$  and  $\mathcal{L}(d^\dagger) = K^\dagger$  is the largest controllable sublanguage of  $L(G)$  with this property.

Fig. 5.  $\mathcal{A}_{d_1}$ .Fig. 6.  $\mathcal{A}_{d_2}$ .

**Definition 9:** Given  $d : Q \times 2^Q \times \Sigma \rightarrow Q$ , let  $\phi(d) \subseteq d$  be the partial map such that

- $\phi(d)(q, E, \sigma)$  is undefined if  $\sigma \in \Sigma_c$  and
  - either  $q \notin d(E, E, \Sigma_{ua}^*)$
  - or  $\mathcal{LT}(d, \theta) \cap \Sigma_{uc}^* \neq \emptyset$  for  $\theta = \delta_d((q, E), \sigma)$ ,
- $\phi(d)(q, E, \sigma) = d(q, E, \sigma)$  otherwise.

**Example 9:** Let  $d_1 = \phi(d_0)$ . We restrict our attention to pairs  $(q, E)$  such that  $E \subseteq E'$  for some reachable configuration  $(q, E')$  of  $\mathcal{A}(d_0)$ , which is enough for computing  $\mathcal{A}_{d_1}$ . Then  $d_1(q, E, u)$  is undefined for  $q = 7$  and  $\emptyset \neq E \subseteq \{2, 7\}$  or for  $q = 10$  and any  $\emptyset \neq E \subseteq \{5, 10\}$ . Similarly,  $d_1(q, E, d)$  is undefined for  $q = 1$  and  $E = \{1\}$ . This is so because  $ds \in \mathcal{LT}(d_0, (1, \{1\}))$  and  $s$  is not controllable. The resulting automaton  $\mathcal{A}_{d_1}$  is depicted in Fig. 5.  $\diamond$

It is important to note that the emptiness of the set  $\mathcal{LT}(d, \theta) \cap \Sigma_{uc}^*$  may be checked on the finite automaton generated by the partial transition map  $\delta_d$  from the initial state  $\theta = (\tilde{q}, \tilde{E})$ .

**Definition 10:** Let  $d^\dagger = d_n$  for the least  $n$  such that  $d_{n+1} = d_n$  where  $d_{i+1} = \phi(d_i)$  and  $d_0 : Q \times 2^Q \times \Sigma \rightarrow Q$  is the map defined with  $d_0(q, E, \sigma) = \delta(q, \sigma)$ .

**Example 10:** Given that  $d_1(E, E, \Sigma_{ua}^*) = E \cup \{d_1(q, E, d) | q \in E\}$ ,  $\mathcal{LE}(d_1)$  contains exactly one losing estimate, namely the singleton set  $\{1\}$ . Thus,  $u \in \mathcal{LT}(d_1, (0, \{0\}))$  and  $d_2(0, \{0\}, u)$  is undefined. The automaton  $\mathcal{A}_{d_2}$  is depicted in Fig. 6. The reader may verify that  $\phi(d_2) = d_2$ .  $\diamond$

The partial map  $d^\dagger$  is well defined since  $\phi(d) \subseteq d$  for all  $d$  and there exist finitely many partial maps  $d : Q \times 2^Q \times \Sigma \rightarrow Q$ . Note that  $\mathcal{L}(d_0) = L(G)$ , thus  $\mathcal{L}(d_0)$  is a controllable sublanguage of  $L(G)$ . It may then be shown by an induction on  $i \geq 0$  that  $\mathcal{L}(d_i)$  is a controllable sublanguage of  $L(G)$  for all  $i$ . Indeed, by Definition 9, if  $s \in \mathcal{L}(d_i) \setminus \mathcal{L}(d_{i+1})$  and  $s'$  is the longest prefix of  $s$  in  $\mathcal{L}(d_{i+1})$ , then  $s = s'\sigma_c s''$  for some action  $\sigma_c \in \Sigma_c$ .

The following propositions show that the language  $K^\dagger = \mathcal{L}(d^\dagger)$  is the largest controllable sublanguage of  $L(G)$  such that  $S$  is opaque w.r.t.  $K^\dagger$  and  $\Sigma_a$ .

**Proposition 8:**  $S$  is opaque w.r.t.  $K^\dagger = \mathcal{L}(d^\dagger)$  and  $\Sigma_a$ .

**Proof:** Let  $d_n = \phi(d_n)$ , hence  $K^\dagger = \mathcal{L}(d_n)$ . Assume for contradiction that  $S$  is not opaque w.r.t.  $\mathcal{L}(d_n)$  and  $\Sigma_a$ . Then there exists  $s \in \mathcal{L}(d_n)$  such that  $[s]_a \cap \mathcal{L}(d_n) \subseteq S$ . By Proposition 6,  $s \in \mathcal{LT}(d_n, (q_0, \{q_0\}))$ . We claim that  $s \in \Sigma_{uc}^*$ . In order to establish this property, assume for contradiction that  $s = s_1 \sigma s_2$  with  $\sigma \in \Sigma_c$  and  $s_2 \in \Sigma_{uc}^*$ . Let  $\delta_{d_n}((q_0, \{q_0\}), s_1) = (q_1, E_1)$  and  $\delta_{d_n}((q_1, E_1), \sigma) = (q_2, E_2)$ , then by Proposition 7,  $s_2 \in \mathcal{LT}(d_n, (q_2, E_2))$ . As  $s_2 \in \Sigma_{uc}^*$ , by Definition 9,  $\phi(d_n)(q_1, E_1, \sigma)$  is undefined. As  $d_n(q_1, E_1, \sigma) = q_2$ , it is impossible that  $d_n = \phi(d_n)$ . It follows from this contradiction that  $s \in \Sigma_{uc}^*$ . Recalling that  $\mathcal{L}(d_n) \subseteq L(G)$ , we observe now that necessarily  $s = \varepsilon$ , because  $s \in \Sigma_{uc}^*$  and  $\delta(q_0, \sigma)$  is undefined for all  $\sigma \in \Sigma_{uc}$ . Now  $[\varepsilon]_a \cap \mathcal{L}(d_n) \not\subseteq S$  since it has been assumed that  $q_0 \notin Q_S$ , hence it is impossible that  $[\varepsilon]_a \cap \mathcal{L}(d_n) \subseteq S$ . It follows from this second contradiction that  $S$  is opaque w.r.t.  $\mathcal{L}(d_n)$  and  $\Sigma_a$ .  $\blacksquare$

**Proposition 9:** Let  $K$  be any prefix-closed and controllable sublanguage of  $L(G)$  such that  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . Then  $K \subseteq \mathcal{L}(d_i)$  for all  $i$ .

**Proof:** In order to establish the proposition, we assume that  $K \not\subseteq \mathcal{L}(d_i)$  for some  $i$  and we search for a contradiction. As  $\mathcal{L}(d_i) \supseteq \mathcal{L}(d_{i+1})$  for all  $i$ , we can moreover assume that  $i$  is the least integer such that  $K \not\subseteq \mathcal{L}(d_i)$ . As  $L(G) = \mathcal{L}(d_0)$ , we have  $i \neq 0$ . Let  $s$  be a minimal word w.r.t. the prefix order in  $K \setminus \mathcal{L}(d_i)$ . As  $\mathcal{L}(d_i)$  is prefix-closed,  $s \neq \varepsilon$ . Let  $s = s'\sigma$  with  $\sigma \in \Sigma$ . As  $s$  has no strict prefix in  $K \setminus \mathcal{L}(d_i)$ , necessarily  $s' \in K \cap \mathcal{L}(d_i)$ . Thus,  $s' \in \mathcal{L}(d_i)$ ,  $s'\sigma \notin \mathcal{L}(d_i)$ , and  $s'\sigma \in \mathcal{L}(d_{i-1})$  since  $s'\sigma = s \in K \subseteq \mathcal{L}(d_{i-1})$ . By construction of the map  $d_i = \phi(d_{i-1})$ ,  $\sigma \in \Sigma_c$  and  $\mathcal{LT}(d_{i-1}, \theta) \cap \Sigma_{uc}^* \neq \emptyset$  for  $\theta = \delta_{d_{i-1}}((q_0, \{q_0\}), s'\sigma)$ . By Proposition 7, there exists  $s'' \in \Sigma_{uc}^*$  such that  $s'\sigma s'' \in \mathcal{L}(d_{i-1})$  and  $[s'\sigma s'']_a \cap \mathcal{L}(d_{i-1})$  is included in  $S$ . Now  $s = s'\sigma$  is in  $K$ ,  $s'\sigma s''$  is in  $L(G)$  because  $\mathcal{L}(d_{i-1}) \subseteq L(G)$ , and  $s'' \in \Sigma_{uc}^*$ . As  $K$  is a controllable sublanguage of  $L(G)$ , it follows that  $s'\sigma s''$  is in  $K$ . As  $[s'\sigma s'']_a \cap \mathcal{L}(d_{i-1}) \subseteq S$  and  $K \subseteq \mathcal{L}(d_{i-1})$  by assumption on  $i$ ,  $[s'\sigma s'']_a \cap K$  is included in  $S$ . This contradicts the hypothesis that  $S$  is opaque w.r.t.  $K$  and  $\Sigma_a$ . Therefore, the proposition has been established.  $\blacksquare$

**Theorem 1:**  $K^\dagger = \mathcal{L}(d^\dagger)$  is the largest prefix-closed and controllable sublanguage of  $L(G)$  s.t.  $S$  is opaque w.r.t.  $K^\dagger$  and  $\Sigma_a$ .

**Proof:** This is an immediate consequence of Propositions 8 and 9.  $\blacksquare$

**Theorem 2:**  $\mathcal{L}(d^\dagger)$  is a regular language.

**Proof:** This follows from the fact that  $\mathcal{A}(d^\dagger)$  is an automaton with set of reachable states included in the finite set  $Q \times 2^Q$ .  $\blacksquare$

With Theorems 1 and 2, we have reached the objectives announced in Sections I and IV. Namely, the control  $f^\dagger$  induced by the automaton  $A^\dagger = \mathcal{A}(d^\dagger)$  is the optimal control enforcing the opacity of  $S$  on  $G$ , and this control is regular.

*Example 11:* The optimal control defined by the automaton  $\mathcal{A}_{d_2}$  prevents the agent from using the lift of the south wing, and it also prevents the agent from using the lift of the north wing from the second floor to the third floor at any time after he has used this lift downwards.  $\diamond$

## VI. CONCLUSION

Given a system modeled by a finite transition system  $G$  over  $\Sigma^*$ , a regular secret  $S \subseteq \Sigma^*$  and an adversary observing a subset  $\Sigma_a \subseteq \Sigma$  of the events of  $G$ , we have addressed the problem of computing the supremal controller that enforces the opacity of  $S$  on  $G$  while observing and controlling respective subsets  $\Sigma_o \subseteq \Sigma$  and  $\Sigma_c \subseteq \Sigma_o$  of events of  $G$ . Assuming that  $\Sigma_a \subseteq \Sigma_o$ , we have shown that this supremal controller is regular. The question is open whether the supremal controller is still regular and effectively computable when  $\Sigma_a$  and  $\Sigma_o$  are not comparable. We would like to add that the control synthesis algorithm which has been proposed can be easily adapted to enforce the joint opacity of several regular secrets  $S_1, \dots, S_n$  (recognized by corresponding subsets of states  $Q_{S_1}, \dots, Q_{S_n}$  of  $G$ ). This may serve, e.g., to hide the current position of a token traveling through a graph with  $n$  vertices, or to stipulate that no bit of a secret data byte should be leaked (by requiring the joint opacity of some predicate  $S$  representing this data byte and of the complementary predicate). This covers also the notion of secrecy as defined in [26]. The non-blocking property of supervisors was ignored in this work. However, under full observation (i.e. when  $\Sigma = \Sigma_o$ ), a straightforward adaptation of our work suffices to enforce also the deadlock-freeness of the controlled system  $G/f^\dagger$ . It suffices indeed to state in Definition 9 that  $\phi(d)(q, E, \sigma)$  is undefined whenever  $\sigma \in \Sigma_c$  and some deadlocked configuration can be reached from  $\delta_d((q, E), \sigma)$  by some uncontrollable sequence of events  $s \in \Sigma_{uc}^*$ .<sup>7</sup>

Many applications in which security issues cannot be ignored deal with infinite data types. Such systems or services are naturally modeled with infinite transition systems. In order to avoid that confidential information leaks from such infinite systems, it seems important to investigate combined techniques of opacity control and abstract interpretation. A possible scheme is to enforce opacity by supervisory control on finite abstractions of infinite systems and then to lift control to the original systems for subsequent model-checking.

## REFERENCES

- [1] G. Lowe, "Towards a completeness result for model checking of security protocols," *J. Comput. Security*, vol. 7, no. 2–3, pp. 89–146, 1999.
- [2] B. Blanchet, M. Abadi, and C. Fournet, "Automated verification of selected equivalences for security protocols," in *Proc. 20th IEEE Symp. Logic Comput. Sci. (LICS'05)*, Chicago, IL, Jun. 2005, pp. 331–340.
- [3] N. Hadj-Alouane, S. Lafrance, F. Lin, J. Mullins, and M. Yeddes, "On the verification of intransitive noninterference in multilevel security," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 5, pp. 948–957, Oct. 2005.
- [4] F. B. Schneider, "Enforceable security policies," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 1, pp. 30–50, 2000.
- [5] J. Ligatti, L. Bauer, and D. Walker, "Edit automata: Enforcement mechanisms for run-time security policies," *Int. J. Inf. Sec.*, vol. 4, no. 1–2, pp. 2–16, 2005.
- [6] V. Darmaillacq, J.-C. Fernandez, R. Groz, L. Mounier, and J.-L. Richier, "Test generation for network security rules," in *Proc. TestCom'06*, 2006, vol. 3964, pp. 341–356.
- [7] G. Le Guernic, "Information flow testing—The third path towards confidentiality guarantee," in *Proc. Adv. Comput. Sci. (ASIAN'07)*, 2007, pp. 33–47.
- [8] J. Dubreil, T. Jéron, and H. Marchand, "Monitoring information flow by diagnosis techniques," in *Proc. Eur. Control Conf. (ECC)*, Budapest, Hungary, Aug. 2009, pp. 2854–2859.
- [9] M. Bishop, *Introduction to Computer Security*. Reading, MA: Addison-Wesley Professional, 2004.
- [10] L. Mazaré, "Using unification for opacity properties," in *Proc. 4th IFIP WG1.7 Workshop Issues Theory Security (WITS'04)*, Barcelona, Spain, 2004, pp. 165–176.
- [11] J. W. Bryans, M. Koutny, L. Mazaré, and P. Y. A. Ryan, "Opacity generalised to transition systems," *Int. J. Inform. Security*, vol. 7, no. 6, pp. 421–435, May 2008.
- [12] A. Saboori and C. Hadjicostis, "Notions of security and opacity in discrete event systems," in *Proc. 46th IEEE Conf. Decision Control*, 2007, pp. 5056–5061.
- [13] A. Saboori and C. Hadjicostis, "Verification of initial-state opacity in security applications of DES," in *Proc. 9th Int. Workshop Discrete Event Systems (WODES'08)*, May 2008, p. 6.
- [14] S. Schneider and A. Sidiropoulos, "CSP and anonymity," in *Proc. Comput. Security (ESORICS'96)*, 1996, pp. 198–218.
- [15] R. Focardi and R. Gorrieri, "Classification of security properties: Information flow," in *Proc. Found. Security Anal. Design*, 2000, pp. 331–396.
- [16] M. Sampath, R. Sengupta, S. Lafortune, K. Sinaamohideen, and D. Tenekeztzis, "Failure diagnosis using discrete event models," *IEEE Trans. Control Syst. Technol.*, vol. 4, no. 2, pp. 105–124, Mar. 1996.
- [17] T. Jéron, H. Marchand, S. Pinchinat, and M.-O. Cordier, "Supervision patterns in discrete event systems diagnosis," in *Proc. Workshop Discrete Event Syst. (WODES'06)*, Ann-Arbor, MI, Jul. 2006, pp. 262–268.
- [18] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control Optim.*, vol. 25, no. 1, pp. 206–230, Jan. 1987.
- [19] P. J. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.
- [20] L. Ricker, "A question of access: Decentralized control and communication strategies for security policies," in *Proc. 8th Int. Workshop Discrete Event Syst.*, Jun. 2006, pp. 58–63.
- [21] F. Cassez, J. Mullins, and O. Roux, "Synthesis of non-interferent distributed systems," in *Proc. 4th Int. Conf. Math. Methods, Models Architectures Comput. Network Security (MMM-ACNS'07)*, Jan. 2007, pp. 159–170.
- [22] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau, "Concurrent secrets," *Discrete Event Dyn. Syst.*, vol. 17, pp. 425–446, 2008.
- [23] S. Takai and Y. Oka, "A formula for the supremal controllable and opaque sublanguage arising in supervisory control," *SICE J. Control, Meas., Syst. Integr.*, vol. 1, no. 4, pp. 307–312, Mar. 2008.
- [24] J. Dubreil, P. Darondeau, and H. Marchand, "Opacity enforcing control synthesis," in *Proc. Workshop Discrete Event Syst. (WODES'08)*, Gothenburg, Sweden, Mar. 2008, pp. 28–35.
- [25] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Norwell, MA: Kluwer Academic, 1999.
- [26] R. Alur, P. Černý, and S. Zdancewic, "Preserving secrecy under refinement," in *Proc.: 33rd Int. Colloq. Automata, Language Programming (ICALP'06)*, 2006, pp. 107–118.

<sup>7</sup>It is worth noting that the reduction from partial observation to total observation presented in Section III works for opacity but does not work for deadlock-freeness.



**Jérémy Dubreil** received the M.Eng. degree in telecommunications from Telecom Bretagne, Brest, France, in 2006 and the Ph.D. degree from the University of Rennes 1, Rennes, France, in 2009.

He currently holds a postdoctoral position at the Laboratoire d'Informatique de l'Ecole Polytechnique, Paris, France. His research interests include verification of possibilistic and probabilistic information flow properties and the application of control theory, modal logic and game theory for the synthesis of secure system.



**Hervé Marchand** received the Ph.D. degree in computer science from "Université de Rennes 1," Rennes, France, in 1997.

In 1998, he spent one year at the University of Michigan, Ann Arbor. Since 1998, he holds an INRIA research scientist position (Rennes, France). His research interests include Supervisory Control, automatic test generation and diagnosis of discrete events systems and their applications to security. He is also interested in high-level languages for reactive and real-time systems programming.



**Philippe Darondeau** received the M.S. degree in applied mathematics from Ensimag, Grenoble, France, in 1970.

He worked at Ensimag until 1978, as a Researcher at CNRS, and wrote his Thèse d'Etat on protection in operating systems. He then moved to Irisa, Rennes, France, and worked subsequently in the field of concurrency. His favorite topics in that domain were process equivalences, true concurrency, fairness and computability. He left CNRS for INRIA in 1991, and then oriented his research at Irisa on Petri net

synthesis and, more recently, on DES supervision.