2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

A Formula for the Supremal Controllable and Opaque Sublanguage in Discrete Event Systems

Shigemasa Takai† and Yusuke Oka‡

†Graduate School of Science and Technology, Kyoto Institute of Technology
Matsugasaki, Sakyo-ku, Kyoto, 606-8585 Japan
‡Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501 Japan
Email: takai@kit.ac.jp

Abstract—In this paper, we study a property of opacity in the language-based framework of discrete event systems. The problem of synthesizing a supervisor that enforces opacity in a maximally permissive way is addressed. A maximally permissive opacity-enforcing supervisor is realized by an automaton that generates the supremal closed controllable and opaque sublanguage of the generated language of the system. This motivates the study on computability of the supremal sublanguage. We present a formula for computing the supremal sublanguage under some assumption on uncontrollable events. Whenever the languages under consideration are regular, the supremal sublanguage is effectively computed using the presented formula.

1. Introduction

The theory of discrete event systems (DESs) provides powerful mathematical tools for dealing with issues of computer security. The notion of intransitive noninterference (INI) is used to solve several problems in multilevel security systems. In [2], the notion of observability [5], [8] for supervisory control of DESs [9] was extended to capture INI, and a method for verifying the extended version of observability was developed. The notion of opacity characterizes a property that the secrete behavior of the system remains opaque to an external agent [4]. Verification of opacity with respect to the secrete behavior described by state-based predicates was studied in the context of Petri nets [4]. In [10], a notion of K-step opacity was defined with respect to the secrete state set, and a verification method using a certain observer automaton was presented.

The prior works mentioned above focused on verification of security properties. A control theoretic approach is useful to enforce required properties of the system. Recently, opacity-enforcing supervisory control has been studied under the assumption that all events are controllable [1]. The task of a supervisor is to restrict the system behavior so that the opacity property is satisfied. The secrete behavior is assumed to be described by a language, and the language-based notion of opacity is defined. Intuitively, the notion of opacity means that an external agent observing only the occurrences of observable events is unsure whether the executed string is in the secrete language. The class of (prefix-)closed opaque languages has a desirable property that it is closed under union [1]. Thus, there always exists the supremal closed opaque sublanguage of a given language. Sufficient conditions under which the supremal closed opaque sublanguage is regular were presented [1].

In this paper, we follow the framework of [1], and study opacity-enforcing supervisory control in the presence of uncontrollable events. In contrast to [1], we allow the existence of uncontrollable events but only consider the case that there is a single agent to which the secret behavior has to remain opaque. Due to the existence of uncontrollable events, the controllability property [9] has to be taken into account in addition to opacity. A maximally permissive opacity-enforcing supervisor is realized by an automaton that generates the supremal closed controllable and opaque sublanguage of the generated language of the system. We present a formula for computing the supremal closed controllable and opaque sublanguage under some assumption on uncontrollable events. Whenever the languages under consideration are regular, the supremal sublanguage is effectively computed using the presented formula. Note that recently computability of the supremal sublanguage has been proved independently in [6] under different assumptions.

2. Preliminaries

We consider a DES modeled by an automaton $G = (Q, \Sigma, f, q_0, Q_m)$, where Q is the set of states, Σ is the finite set of events, a partial function $f : Q \times \Sigma \to Q$ is the transition function, $q_0 \in Q$ is the initial state, and $Q_m \subseteq Q$ is the set of marked states. If Q is the finite set, G is said to be a finite automaton. Let Σ^* be the set of all finite strings of elements of Σ,

including the empty string ε. The function f can be generalized to $f : Q \times \Sigma^* \to Q$ in the natural way. We use the notation $f(q, s)!$ for any $q \in Q$ and any $s \in \Sigma^*$ to denote that $f(q, s)$ is defined. The generated and marked languages of G, denoted by $L(G)$ and $L_m(G)$, respectively, are defined by $L(G) = \{s \in \Sigma^* \mid f(q_0, s)!\}$ and $L_m(G) = \{s \in \Sigma^* \mid f(q_0, s) \in Q_m\}$.

Let $L \subseteq \Sigma^*$ be a language. We denote the set of all prefixes of strings in L by $\overline{L}$, that is,

$$\overline{L} = \{s \in \Sigma^* \mid \exists t \in \Sigma^*;\ st \in L\}. \tag{1}$$

L is said to be closed if $L = \overline{L}$. We also denote $\overline{\{t\}}$ by $\overline{t}$ for each $t \in \Sigma^*$. For any languages $L_1, L_2 \subseteq \Sigma^*$, their concatenation L_1L_2 is defined as

$$L_1L_2 = \{st \in \Sigma^* \mid s \in L_1,\ t \in L_2\}. \tag{2}$$

Also the quotient operation for $L_1, L_2 \subseteq \Sigma^*$ is defined as

$$L_1/L_2 = \{s \in \Sigma^* \mid \exists t \in L_2;\ st \in L_1\}. \tag{3}$$

The event set Σ is assumed to be partitioned into the observable event set Σ_o and the unobservable event set Σ_{uo} [8]. The natural projection map $P : \Sigma^* \to \Sigma_o^*$ is inductively defined as follows:

- $P(\varepsilon) = \varepsilon$,
- $(\forall s \in \Sigma^*,\ \sigma \in \Sigma)$

$$P(s\sigma) = \begin{cases} P(s)\sigma, & \text{if } \sigma \in \Sigma_o \\ P(s), & \text{if } \sigma \in \Sigma_{uo}. \end{cases} \tag{4}$$

The string $P(s)$ is obtained by erasing all unobservable events from s. If a string $s \in L(G)$ occurs in G, then the string observed by an external agent is $P(s)$. So if $P(s) = P(s')$ for $s, s' \in L(G)$, the agent cannot distinguish between them. For a language $L \subseteq \Sigma^*$, $P(L) \subseteq \Sigma_o^*$ is defined as

$$P(L) = \{P(s) \in \Sigma_o^* \mid s \in L\}. \tag{5}$$

Also, for a language $L' \subseteq \Sigma_o^*$, $P^{-1}(L') \subseteq \Sigma^*$ is defined as

$$P^{-1}(L') = \{s \in \Sigma^* \mid P(s) \in L'\}. \tag{6}$$

3. Opacity-Enforcing Supervisory Control

As in [1], we assume that the secrete behavior of the system G is described by a nonempty language $S \subseteq L(G)$. The notion of opacity means that an external agent observing the behavior of the system through the projection map P is unsure whether the executed string is in the secrete behavior S. Opacity is formally defined as follows.

Definition 1 [1] A closed language $L \subseteq L(G)$ is said to be *opaque* (with respect to $S \subseteq L(G)$) if

$$\forall s \in L \cap S,\ \exists s' \in L - S;\ P(s) = P(s').$$

The notion of opacity requires that for any secrete string $s \in L \cap S$, there exist an indistinguishable non-secrete string $s' \in L - S$. In this paper, we consider a problem of synthesizing an opacity-enforcing supervisor. Let Σ_c and Σ_{uc} be the sets of controllable and uncontrollable events, respectively [9]. A supervisor γ is formally defined by $\gamma : L(G) \to 2^{\Sigma_c}$. An event in $\gamma(s)$ is disabled by γ following the execution of a string $s \in L(G)$. Let $L(G/\gamma)$ be the language generated under the control action of γ [9]. A closed language $L \subseteq L(G)$ is said to be controllable [9] if $L\Sigma_{uc} \cap L(G) \subseteq L$.

Since controllability and opacity are preserved under union [1], [11], there always exists the supremal closed controllable and opaque sublanguage of $L(G)$. We denote this supremal language by $\sup CO(L(G))$. Our purpose is to synthesize a supervisor $\gamma : L(G) \to 2^{\Sigma_c}$ such that $L(G/\gamma) = \sup CO(L(G))$. Such a supervisor enforces opacity in a maximally permissive way.

A maximally permissive opacity-enforcing supervisor can be realized by an automaton that generates $\sup CO(L(G))$. To compute $\sup CO(L(G))$, we consider the following iterative computation:

- $L_0 := L(G)$,
- $(\forall i \geq 0)\ L_{i+1} := \sup C(\sup O(L_i))$,

where $\sup C(L)$ (respectively, $\sup O(L)$) is the supremal closed controllable (respectively, opaque) sublanguage of $L \subseteq L(G)$. If there exists $i \geq 0$ such that $L_{i+1} = L_i$, then $L_i = \sup CO(L(G))$. To our knowledge, the finite convergence of the above iterative computation has not been established. In the next section, we show that the one-step convergence is guaranteed under certain assumption on uncontrollable events.

4. Formula for the Supremal Closed Controllable and Opaque Sublanguage

The following formula for the supremal closed controllable sublanguage $\sup C(L)$ of a closed language $L \subseteq L(G)$ was presented in [3]:

$$\sup C(L) = L - \{(L(G) - L)/\Sigma_{uc}^*\}\Sigma^*. \tag{7}$$

Also, in [1], the following formula for the supremal closed opaque sublanguage $\sup O(L)$ of L was suggested:

$$\sup O(L) = L - \{\Sigma^* - P^{-1}P(L - S)\}\Sigma^*. \tag{8}$$

Remark 1 Assume that $G = (Q, \Sigma, f, q_0, Q_m)$ is a finite automaton, and a closed language $L \subseteq L(G)$ is generated by a finite automaton $G_L = (Q_L, \Sigma, f_L, q_{L0}, Q_{Lm})$. It was shown in [7] that the supremal closed controllable sublanguage $\sup C(L)$ can be computed in $O(|Q| \cdot |Q_L|)$. If the secret behavior

$S \subseteq L(G)$ is also marked by a finite automaton, then the supremal closed opaque sublanguage $\sup O(L)$ can be computed using the formula (8). The complexity of computing $\sup O(L)$ is exponential since we have to construct a *deterministic* automaton for the language $P^{-1}P(L-S)$.

In order to guarantee the one-step convergence of the iterative computation for the supremal closed controllable and opaque sublanguage $\sup CO(L(G))$, we impose the following condition C) on uncontrollable events:

C) $\forall \sigma \in \Sigma_{uc} \cap \Sigma_o$, $\forall s, s' \in L(G)$; $P(s) = P(s') \wedge s\sigma \in L(G) \Rightarrow s'\sigma \in L(G)$.

The above condition requires that for any two indistinguishable strings, if an uncontrollable and observable event is feasible after one string, then it is also feasible after the other string.

When $G = (Q, \Sigma, f, q_0, Q_m)$ is a finite automaton, the condition C) can be verified in the following way. We construct the testing automaton $T = (Z, \Sigma^T, g, z_0, Z)$ as follows:

- $Z = Q \times Q$ and $z_0 = (q_0, q_0)$.
- $\Sigma^T = (\Sigma \cup \{\varepsilon\}) \times (\Sigma \cup \{\varepsilon\}) - \{(\varepsilon, \varepsilon)\}$.
- $g : Z \times \Sigma^T \to Z$ is defined as follows: For each $z = (q_1, q_2) \in Z$ and $\sigma^T = (\sigma_1, \sigma_2) \in \Sigma^T$, $g(z, \sigma^T)!$ if and only if
  - if $\sigma_i \neq \varepsilon$ then $f(q_i, \sigma_i)!$ $(i = 1, 2)$,
  - $P(\sigma_1) = P(\sigma_2)$.

  If $g(z, \sigma^T)!$, then $g(z, \sigma^T) = (q_1', q_2')$, where

$$q_i' = \begin{cases} f(q_i, \sigma_i), & \text{if } \sigma_i \neq \varepsilon \\ q_i, & \text{otherwise.} \end{cases} \tag{9}$$

The following proposition shows that the problem of verifying the condition C) is transformed to a reachability problem in the testing automaton T.

Proposition 1 Assume that $G = (Q, \Sigma, f, q_0, Q_m)$ is a finite automaton. The condition C) does not hold if and only if there exist $\sigma \in \Sigma_{uc} \cap \Sigma_o$ and a reachable state $(q_1, q_2) \in Z$ of the testing automaton T such that $f(q_1, \sigma)$ is defined and $f(q_2, \sigma)$ is not.

In the following theorem, the one-step convergence of the iterative computation for $\sup CO(L(G))$ is established under the condition C).

Theorem 1 Assume that the condition C) holds. Then,

$$\sup CO(L(G)) = \sup C(\sup O(L(G))). \tag{10}$$

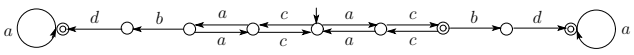

Figure 1: Automaton G for Example 1.

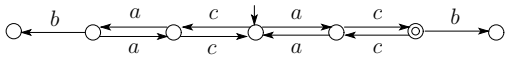

Figure 2: Automaton which generates $\sup O(L(G))$.

The following example illustrates the computation of $\sup CO(L(G))$ under the condition C).

Example 1 We consider an automaton G shown in Fig. 1, where the initial state is identified by the cycle with an arrow $\downarrow$, and a marked state is identified by a double circle. Let $\Sigma_o = \{a, b, d\}$, $\Sigma_{uo} = \{c\}$, $\Sigma_c = \{a, b, c\}$, and $\Sigma_{uc} = \{d\}$. Then this automaton satisfies the condition C).

We suppose that the secrete behavior is described by the marked language $L_m(G)$, that is, $S = L_m(G)$. We can easily verify that the generated language $L(G)$ is not opaque. For example, let us consider a secrete string $acbd \in S$. Then any string $s \in L(G)$ with $P(s) = P(acbd) = abd$ belongs to S.

We first compute the supremal closed opaque sublanguage $\sup O(L(G))$. An automaton that generates $\sup O(L(G))$ is shown in Fig. 2. We next compute the supremal closed controllable sublanguage $\sup C(\sup O(L(G)))$ of $\sup O(L(G))$. The sublanguage $\sup C(\sup O(L(G)))$ is generated by an automaton shown in Fig. 3. By Theorem 1, $\sup C(\sup O(L(G)))$ is the supremal closed controllable and opaque sublanguage of $L(G)$, and a maximally permissive opacity-enforcing supervisor is realized by the automaton of Fig. 3.

The following example shows that the formula of Theorem 1 is not true without the condition C).

Example 2 We consider an automaton G shown in Fig. 4. Let $\Sigma_o = \{a, b, d\}$, $\Sigma_{uo} = \{c\}$, $\Sigma_c = \{c\}$, and $\Sigma_{uc} = \{a, b, d\}$. Then this automaton does not satisfy the condition C). For example, for $a \in \Sigma_{uc} \cap \Sigma_o$ and $cad, ad \in L(G)$, we have $P(cad) = P(ad) = ad$, $cada \in L(G)$ and $ada \notin L(G)$.

We suppose that $S = L_m(G)$. We first compute the supremal closed opaque sublanguage $\sup O(L(G))$.

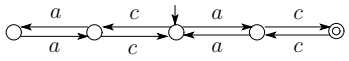

Figure 3: Automaton which generates $\sup C(\sup O(L(G)))$.

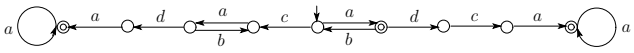

Figure 4: Automaton G for Example 2.

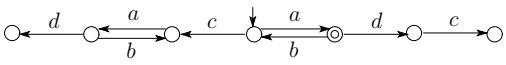

Figure 5: Automaton which generates $\sup O(L(G))$.

An automaton that generates $\sup O(L(G))$ is shown in Fig. 5. We next compute the supremal closed controllable sublanguage $\sup C(\sup O(L(G)))$ of $\sup O(L(G))$. The sublanguage $\sup C(\sup O(L(G)))$ is generated by an automaton shown in Fig. 6. Note that for $a \in \sup C(\sup O(L(G))) \cap S$, there does not exist $s \in \sup C(\sup O(L(G))) - S$ such that $P(a) = P(s)$. Thus, $\sup C(\sup O(L(G)))$ is not opaque, which implies that the formula of Theorem 1 does not hold.

The following theorem shows that the formula presented in Theorem 1 is simplified under a stronger assumption that all uncontrollable events are unobservable.

Theorem 2 Assume that $\Sigma_{uc} \subseteq \Sigma_{uo}$. Then,

$$\sup CO(L(G)) = \sup O(L(G)). \tag{11}$$

5. Conclusion

In this paper, we have studied opacity-enforcing supervisory control of DESs in the language-based framework. The contribution of the paper is a formula for computing the supremal closed controllable and opaque sublanguage, which is derived under some assumption on uncontrollable events. Whenever the languages under consideration are regular, the supremal sublanguage is effectively computed using the presented formula.

Acknowledgment

This work was supported in part by MEXT under Grant-in-Aid for Scientific Research (No. 17360198 and No. 18560433).

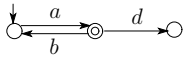

Figure 6: Automaton which generates $\sup C(\sup O(L(G)))$.

References

[1] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau, "Concurrent secrets," *Discrete Event Dynamic Syst.: Theory and Appl.*, vol.17, no.4, pp.425–446, 2007.

[2] N. Ben 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., Part B: Cybern.*, vol.35, no.5, pp.948–958, 2005.

[3] R. D. Brandt, V. K. Garg, R. Kumar, F. Lin, S. I. Marcus, and W. M. Wonham, "Formulas for calculating supremal controllable and normal sublanguages," *Syst. Contr. Lett.*, vol.15, no.2, pp.111–117, 1990.

[4] J. W. Bryans, M. Koutny, and P. Y. A. Ryan, "Modelling opacity using Petri nets," *Electronic Notes in Theoretical Computer Science*, vol.121, pp.101–115, 2005.

[5] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Trans. Automat. Contr.*, vol.33, no.3, pp.249–260, 1988.

[6] J. Dubreil, P. Darondeau, and H. Marchand, "Opacity enforcing control synthesis," *Proc. 9th Int. Workshop Discrete Event Syst.*, pp.28–35, 2008.

[7] R. Kumar, V. Garg, and S. I. Marcus, "On controllability and normality of discrete event dynamical systems," *Syst. Contr. Lett.*, vol.17, no.3, pp.157–168, 1991.

[8] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Inform. Sci.*, vol.44, no.3, pp.173–198, 1988.

[9] P. J. Ramadge and W. M. Wonham: "Supervisory control of a class of discrete event processes," *SIAM J. Contr. Optim.*, vol.25, no.1, pp.206–230, 1987.

[10] A. Saboori and C. Hadjicostis, "Notions of security and opacity in discrete event systems," *Proc. 46th IEEE Conf. Decision and Contr.*, pp.5056–5061, 2007.

[11] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language," *SIAM J. Contr. Optim.*, vol.25, no.3, pp.637–659, 1987.