DISTRIBUTED OBSERVABILITY OF REGULAR LANGUAGES IS UNDECIDABLE

by

Stavros Tripakis

Memorandum No. UCB/ERL M01/8

5 February 2001

DISTRIBUTED OBSERVABILITY OF REGULAR LANGUAGES IS UNDECIDABLE

by

Stavros Tripakis

Memorandum No. UCB/ERL M01/8

5 February 2001

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Distributed Observability of Regular Languages is Undecidable

Stavros Tripakis

February 5, 2001

1 Introduction

We consider the problem of (distributed) observability. Informally, the problem consists in checking whether a given system behaves correctly or not based on the observed behavior of the system, which might be partial. The system is formalized as a language L of strings over an alphabet Σ. The correct behaviors are formalized as a language $K \subseteq L$. Partial observation is formalized by considering an alphabet $\Sigma_o \subseteq \Sigma$ of observable events and "erasing" all events in $\Sigma - \Sigma_o$ from the observed behaviors. For example, if $\rho = abcabc$ is a behavior of the system, and $\Sigma_o = \{a, b\} \subseteq \Sigma = \{a, b, c\}$, then the observed behavior $\rho/_{\Sigma_o}$ is $abab$.

In the centralized version of the problem, there is one observer which observes all events in Σ_o. Checking observability consists in showing that there do not exist two distinct behaviors in L, such that one belongs to the set of correct behaviors K, the other does not, yet both yield the same observed behavior. We show that checking observability with respect to one observer is decidable.

In the distributed version of the problem, there are $k \geq 2$ observers, i.e., $\Sigma_o = \Sigma_o^1 \cup \cdots \cup \Sigma_o^k$, and observer i observes Σ_o^i (these subalphabets need not be disjoint). Checking observability consists in showing that there do not exist two distinct behaviors in L, such that one belongs to the set of correct behaviors K, the other does not, yet both yield the same observed behavior with respect to each of the observers. We show that checking observability with respect to two observers is undecidable.

Related work

Observability has been considered by many researchers in the discrete-event systems community, e.g., see [7, 2, 10, 12, 4, 1, 14]. Most of the above papers are looking at the problem from the supervisory controller synthesis point of view, since some notion of observability is usually a pre-requisite, and provide necessary and sufficient conditions for controllers to exist. These conditions involve the language of correct behaviors being observable. To our knowledge, however, no decidability results about checking observability have been presented before.

A number of observability definitions have appeared in the literature. We briefly discuss these definitions, for the centralized (one observer) and decentralized (many observers) versions of the problem, respectively.

Centralized observability definitions In [2] the authors introduce the notion of a (M, L)-*recognizable* language: this is essentially what we define as observable language, with the difference that, in their case, M is a *mask* and not a projection, as in our case, i.e., their notion of observed behaviors is more general. In the case where M is a projection, their definition of recognizable languages becomes equivalent to our definition of observable languages.

In [6] the authors introduce two different notions: observable languages and *normal* languages. Their definition of normal languages is equivalent to our definition of observable languages, whereas their definition of observable languages is less strict. Since they are interested in the supervisory control problem with partial observations, it turns out that this problem may have a solution even in the case where the language is not normal, but only observable (w.r.t. their definition). The reason is that the controller is allowed to disable some events even if it cannot observe them.

In summary, let us denote by $\mathsf{rec}_{[2]}$, $\mathsf{nor}_{[6]}$, $\mathsf{obs}_{[6]}$, $\mathsf{obs}_{[11]}$, the classes of recognizable languages according to [2], normal languages according to [6], observable languages according to [6] and observable languages according to [11]. Then:

$$\mathsf{rec}_{[2]} = \mathsf{nor}_{[6]} = \mathsf{obs}_{[11]} \subset \mathsf{obs}_{[6]}$$

where the last inclusion is strict.

Decentralized observability definitions The authors of [2] extend their definition of recognizable languages to the decentralized case as well. Decentralized recognizability is equivalent to the definition of *weak decomposability* of [10]. The authors of [10] introduce also another two notions of observability, namely, *strong decomposability* and *co-observability*. These different notions relate differently to the necessary and sufficient conditions for the existence of decentralized controllers given in the above paper. For more details, we refer the reader to [2, 10].

Here, we summarize the relations of the above definitions with ours:

$$\mathsf{s}-\mathsf{dec}_{[10]} \subset \mathsf{co}-\mathsf{obs}_{[10]} \subset \mathsf{w}-\mathsf{dec}_{[10]} = \mathsf{rec}_{[2]} \subset \mathsf{obs}_{[11]}$$

where all inclusions are strict.

Other work [7] study the problem of decentralized control with respect to "local" specifications, e.g., two supervisors S_1 and S_2 are synthesized independently with respect to local specifications ϕ_1 and ϕ_2, and their combined control on the plant results in the behavior $\phi_1 \wedge \phi_2$. This may be called "modular" controller synthesis and essentially has to do with breaking the problem into smaller ones.

[4] study the same problem for a more general class of controllers where the composition with the plant is not necessarily synchronous. [14] consider other ways of combining control actions with the plant, including "fusion by union" of events.

[12] and [1] consider the problem of decentralized control where the controllers are more powerful, in the sense that they can exchange information during the execution of the plant.

[13] study the controller synthesis problem in the centralized case. [5] examine the complexity of this problem under incomplete information (partial observation).

On a slightly different setting, [8] studies the decidability and complexity of the problems of distributing a centralized program on a decentralized processor architecture, or synthesizing a decentralized program from scratch.

2 The Observability Problem

Preliminaries. Let Σ be a finite alphabet. Σ^* denotes the set of all finite strings over Σ. We denote by ϵ the empty string. Σ^+ is the set of all finite strings over Σ except the empty string, $\Sigma^* \setminus \{\epsilon\}$. Given two strings ρ_1 and ρ_2, $\rho_1\rho_2$ is the concatenation of ρ_1 and ρ_2.

For $\Sigma_1 \subseteq \Sigma$, we define the *projection* of a string $\rho \in \Sigma^*$ to Σ_1, denoted $\rho/_{\Sigma_1}$, as the string $\rho_1 \in \Sigma_1^*$, where ρ_1 is obtained from ρ by erasing all letters not in Σ_1. For example, if $\Sigma = \{a, b, c\}$ and $\Sigma_1 = \{a, c\}$, then $abbcbacb/_{\Sigma_1} = acac$.

Observability definition. Let $K, L \subseteq \Sigma^*$ be two regular languages over Σ, such that $K \subseteq L$.

Given alphabets $\Sigma_i \subseteq \Sigma$, $i = 1, ..., k$, we say that K is *observable with respect to* L *and* $\Sigma_1, ..., \Sigma_k$ if for all $\rho_1, \rho_2 \in L$,

$$(\forall i = 1, ..., k, \rho_1/_{\Sigma_i} = \rho_2/_{\Sigma_i}) \Rightarrow (\rho_1 \in K \Leftrightarrow \rho_2 \in K).$$

So K is observable iff there are no two different strings ρ_1, ρ_2 in L, such that $\rho_1 \in K$, $\rho_2 \notin K$, but ρ_1 and ρ_2 yield the same projections to all Σ_i.

The intuition behind the above definition is as follows. Consider a system which generates behaviors in L, where K are the correct behaviors, and $L - K$ the erroneous behaviors. Each behavior of the system is observed by all k observers. At the end of the execution of the system, the observers get together and decide whether the behavior was correct or not. They can only do that if all behaviors that yield the same observations (to all observers) are either all correct or all erroneous.

3 Decidability for One Observer

Theorem 3.1 *Given regular languages* $K \subseteq L \subseteq \Sigma^*$, *and* $\Sigma_1 \subseteq \Sigma$, *there is an algorithm to decide whether or not* K *is observable with respect to* L *and* Σ_1.

Proof sketch: Let A_K and A_{L-K} be automata that recognize K and $L-K$ respectively. Let $A = A_K \times_{\Sigma_1} A_{L-K}$ be a product automaton of A_K and A_{L-K} which is defined synchronizing all transitions of A_K and A_{L-K} labeled with the same letter in Σ_1 and interleaving asynchronously all transitions labelled otherwise. A state (s, s') of A is defined to be accepting iff s is an accepting state of A_K and s' is an accepting state of A_{L-K}. Then, K is observable iff A has an accepting behavior.

4 Undecidability for Two Observers

(Stavros: I'm going to change this section slightly, to give a littly bit simpler proof that is also going to be used for diagnosability.)

Theorem 4.1 *The problem of deciding, given two regular languages $K \subseteq L$ over Σ, whether K is observable with respect to L and $\Sigma_1, \Sigma_2 \subseteq \Sigma$, is undecidable.*

Proof: We reduce Post's Correspondence Problem (PCP) to the observability problem. PCP is known to be undecidable [3].

First we recall PCP. We are given a finite alphabet T and two sets of strings $A, B \subset T$, $A = \{w_1, w_2, ..., w_n\}$ and $B = \{u_1, u_2, ..., u_n\}$. We are asked: do there exist indices $i_1, ..., i_k \in [1..n]$, $k \geq 1$, such that $w_{i_1} w_{i_2} \cdots w_{i_k} = u_{i_1} u_{i_2} \cdots u_{i_k}$. We cannot give an algorithmic answer to this question.

We now translate the above instance of PCP to an observability problem. Let $a_1, a_2, ..., a_n$ be new letters, not in T. Let $\Sigma_1 = T$, $\Sigma_2 = \{a_1, ..., a_n\}$ and $\Sigma = \Sigma_1 \cup \Sigma_2$. We will construct languages K and L over Σ such that K is observable with respect to L and Σ_1, Σ_2 iff the answer to the above PCP is "no".

The automaton recognizing L is shown in figure 1. From the initial state, it can move non-deterministically and with no input to two accepting states. From each of these two states, there is a sequence of transitions (a loop) that brings the automaton back to the same state. For example, if the automaton is in the upper accepting state, it can read the string $w_1 a_1$ and get back to the same accepting state, or read $w_2 a_2$ and get back to this state, and so on. If the automaton is in the lower accepting state, it can read $u_1 a_1$ and get back to this state, and so on.

The specification K is given by the regular expression:

$$K = (w_1 a_1 + \cdots + w_n a_n)^*.$$

That is, the behaviors ending in the upper accepting state are considered correct and the behaviors ending in the lower accepting state are wrong.

Assume first that the answer to the above PCP is "yes", that is, there exist indices $i_1, ..., i_k \in [1..n]$, $k \geq 1$, such that $w_{i_1} w_{i_2} \cdots w_{i_k} = u_{i_1} u_{i_2} \cdots u_{i_k}$. Then, let $\rho_1 = w_{i_1} a_{i_1} w_{i_2} a_{i_2} \cdots w_{i_k} a_{i_k}$ and $\rho_2 = u_{i_1} a_{i_1} u_{i_2} a_{i_2} \cdots u_{i_k} a_{i_k}$. Both $\rho_1, \rho_2 \in L$, but only $\rho_1 \in K$. However, $\rho_1/_{\Sigma_1} = w_{i_1} w_{i_2} \cdots w_{i_k} = u_{i_1} u_{i_2} \cdots u_{i_k} = \rho_2/_{\Sigma_1}$. And $\rho_1/_{\Sigma_2} = a_{i_1} a_{i_2} \cdots a_{i_k} = \rho_2/_{\Sigma_2}$. Therefore, K is not observable.

In the other direction, assume that K is not observable, that is, there exist $\rho_1, \rho_2 \in L$, $\rho_1 \neq \rho_2$, $\rho_1 \in K$, $\rho_2 \notin K$, such that $\rho_1/_{\Sigma_1} = \rho_2/_{\Sigma_1}$ and $\rho_1/_{\Sigma_2} = \rho_2/_{\Sigma_2}$. By definition of L and K, ρ_1 must be of the form $w_{i_1}a_{i_1}w_{i_2}a_{i_2}\cdots w_{i_k}a_{i_k}$ and ρ_2 must be of the form $u_{i_1}a_{i_1}u_{i_2}a_{i_2}\cdots u_{i_k}a_{i_k}$. Since $\rho_1/_{\Sigma_1} = \rho_2/_{\Sigma_1}$, we have $w_{i_1}w_{i_2}\cdots w_{i_k} = u_{i_1}u_{i_2}\cdots u_{i_k}$, which means that the answer to the above PCP is "yes". ■

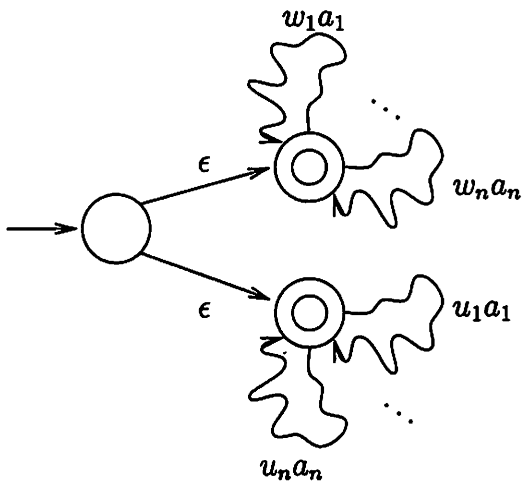

Figure 1: The system automaton, generating L.

5 Discussion

The definitions and undecidability for two observers can be extended to ω-languages.

References

[1] G. Barrett and S. Lafortune, "On the Synthesis of Communicating Controllers with Decentralized Inf ormation Structures for Discrete-Event Systems". IEEE Conference on Decision and Control 1998.

[2] R. Cieslak, C. Desclaux, A.S. Fawaz and P. Varaiya. "Supervisory control of discrete-event processes with partial observations". IEEE Trans. Automatic Control, vol. 33, pp. 249-260, 1988.

[3] M. Garey and D. Johnson. *Computers and Intractability: a guide to the theory of NP-completeness.* Freeman, 1979.

[4] R. Kumar, M.A. Shayman, "Centralized and Decentralized Supervisory Control of Nondeterministic Systems Under Partial Observation", SIAM Journal on Control and Optimization, vol. 35(2), pp. 363-383, 1997.

[5] O. Kupferman and M. Vardi, "Synthesis with Incomplete Information", ICTL 1997.

[6] F. Lin and W.M. Wonham. "On observability of discrete-event systems", Information Sciences, vol. 44, pp. 173-198, 1988.

[7] F. Lin and W.M. Wonham. "Decentralized supervisory control of discrete-event systems", Information Sciences, vol. 44, pp. 199-224, 1988.

[8] A. Pnueli and R. Rosner, "Distributed Reactive Systems are Hard to Synthesize", Proceedings of the 31th IEEE Symposium Foundations of Computer Science (FOCS 1990), pages 746-757, 1990.

[9] P.J.G. Ramadge and W.M. Wonham, "The control of discrete event systems", *Proceedings of the IEEE*, January, 1989.

[10] K. Rudie and W.M. Wonham. "Think globally, act locally: Decentralized supervisory control", IEEE Trans. Automatic Control, vol. 37, pp. 1692-1708, 1992.

[11] S. Tripakis, "Distributed Observability of Regular Languages is Undecidable", This document.

[12] K.C. Wong and J.H. van Schuppen, "Decentralized supervisory control of discrete-event systems with communication", WODES '96, IEE, 1996.

[13] H. Wong-Toi and D. L. Dill. "Synthesizing processes and schedulers from temporal specifications", Proceedings of the 1990 Computer-Aided Verification Workshop, Lecture Notes in Computer Science, Volume 531, Springer-Verlag, 1991.

[14] T. Yoo and S. Lafortune, "New Results on Decentralized Supervisory Control of Discrete-Event Systems", IEEE Conference on Decision and Control 2000.