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 \( \Sigma \). The correct behaviors are formalized as a language \( K \subseteq L \). Partial observation is formalized by considering an alphabet \( \Sigma_\circ \subseteq \Sigma \) of observable events and "erasing" all events in \( \Sigma - \Sigma_\circ \) from the observed behaviors. For example, if \( \rho = abcabc \) is a behavior of the system, and \( \Sigma_\circ = \{a, b\} \subseteq \Sigma = \{a, b, c\} \), then the observed behavior \( \rho/\Sigma_\circ \) is \( abab \).

In the centralized version of the problem, there is one observer which observes all events in \( \Sigma_\circ \). 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., \( \Sigma_\circ = \Sigma_\circ^1 \cup \cdots \cup \Sigma_\circ^k \), and observer \( i \) observes \( \Sigma_\circ^i \) (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. These definitions, for the centralized (one observer) and decentralized 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 \(\text{rec}_2\), \(\text{nor}_6\), \(\text{obs}_{11}\), 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:

\[
\text{rec}_2 = \text{nor}_6 = \text{obs}_{11} \subset \text{obs}_6
\]

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:

\[
\text{s-dec}_{10} \subset \text{co-obs}_{10} \subset \text{w-dec}_{10} = \text{rec}_2 \subset \text{obs}_{11}
\]

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 \(\phi_1\) and \(\phi_2\), and their combined control on the plant results in the behavior \(\phi_1 \land \phi_2\). 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.


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

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

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

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

$$(\forall i = 1, \ldots, k, \rho_1/\Sigma_i = \rho_2/\Sigma_i) \Rightarrow (\rho_1 \in K \iff \rho_2 \in K).$$

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

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 $\Sigma_1$. 
Proof sketch: Let $A_K$ and $A_{L-K}$ be automata that recognize $K$ and $L - K$ respectively. Let $A = A_K \times E A_{L-K}$ 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 $\Sigma_1$ 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 little 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 $\Sigma$, 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 $A, B \subseteq T$, $A = \{w_1, w_2, ..., w_n\}$ 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 $w_i w_i \cdots w_i = u_i u_i \cdots u_i$. 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 $\Sigma$ such that $K$ is observable with respect to $L$ and $\Sigma_1, \Sigma_2 \subseteq \Sigma$ 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 $w_i w_i \cdots w_i = u_i u_i \cdots u_i$. Then, let $\rho_1 = w_i a_i w_i a_i \cdots w_i a_i$ and $\rho_2 = u_i a_i u_i a_i \cdots u_i a_i$. Both $\rho_1, \rho_2 \in L$, but only $\rho_1 \in K$. However, $\rho_1 / \Sigma_1 = w_i w_i \cdots w_i = u_i u_i \cdots u_i = \rho_2 / \Sigma_1$. And $\rho_1 / \Sigma_2 = a_i a_i \cdots a_i = \rho_2 / \Sigma_2$. Therefore, $K$ is not observable.

4
In the other direction, assume that $K$ is not observable, that is, there exist $\rho_1, \rho_2 \in L$, $\rho_1 \neq \rho_2$, $\rho_1 \notin K$, $\rho_2 \notin K$, such that $\rho_1/\Sigma_1 = \rho_2/\Sigma_1$ and $\rho_1/\Sigma_2 = \rho_2/\Sigma_2$. By definition of $L$ and $K$, $\rho_1$ must be of the form $w_1a_1w_2a_2 \cdots w_ik_a_i$ and $\rho_2$ must be of the form $u_1a_1u_2a_2 \cdots u_k a_i$. Since $\rho_1/\Sigma_1 = \rho_2/\Sigma_1$, we have $w_1w_2 \cdots w_k = u_1u_2 \cdots u_k$, which means that the answer to the above PCP is "yes".

![Figure 1: The system automaton, generating $L$.](image)

5 Discussion

The definitions and undecidability for two observers can be extended to $\omega$-languages.

References


