12th IFAC/IEEE Workshop on Discrete Event Systems
Cachan, France. May 14-16, 2014

Generalised Search for the Observer Property in Discrete Event Systems [1]

H. J. Bravo [*] **P. N. Pena** [**] **A. E. C. da Cunha** [***]
R. Malik [****] **J. E. R. Cury** [†]

[*] *Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais, Brazil (e-mail: hugobravoc@ufmg.br)*
[**] *Departamento de Engenharia Eletrônica, Universidade Federal de Minas Gerais, Brazil (e-mail:ppena@ufmg.br)*
[***] *Seção de Engenharia Elétrica, Instituto Militar de Engenharia, Brazil (email: carrilho@ime.eb.br)*
[****] *Department of Computer Science, The University of Waikato, New Zealand, (e-mail: robi@waikato.ac.nz)*
[†] *Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina, Brazil (e-mail: cury@das.ufsc.br)*

Abstract: This paper proposes a procedure to compute abstractions with the observer property (OP) for discrete event systems. The procedure is a generalisation of an algorithm proposed before by the authors, which is based on a quadratic algorithm to test whether a given projection has the observer property. The new version proposed in this paper supports systems that have cycles of non-relevant events, thus removing a restriction of the previous version. Nevertheless, it retains its cubic complexity, which means that the method is asymptotically faster than other methods proposed in the literature to solve the same problem.

Keywords: Discrete Event Systems, Natural Projections, Observer Property.

1. INTRODUCTION

The *Observer Property (OP)* is fundamental for the consistency of abstractions of Discrete Event Systems in Supervisory Control Theory (Ramadge and Wonham, 1989), in particular for hierarchical control (Wong and Wonham, 1996; Hill and Tilbury, 2006; da Cunha and Cury, 2007; Feng and Wonham, 2008; Schmidt et al., 2008; Schmidt and Breindl, 2011) and for the verification of the absence of conflicts in modular control (Hill and Tilbury, 2006; Flordal and Malik, 2009; Pena et al., 2009).

The problem of searching for abstractions obtained by natural projections with OP consists of the refinement of a given system and its natural projection that do not satisfy the OP until the OP is satisfied for the refined system and the projection. The refinement can be performed either by selecting and turning a non-relevant event or particular transition into relevant. Some approaches that deal with the search of the OP for projections are (Schmidt and Moor, 2006; Feng and Wonham, 2010; Pena et al., 2010). On the other hand, in (Wong and Wonham, 2004), a similar problem is treated for causal reporter maps (Zhong and Wonham, 1990).

Particularly, the approach (Pena et al., 2010) proposes a search algorithm based on the OP-verifier (Pena et al., 2008), which determines whether or not an abstraction has the Observer Property. The OP-verifier has a restriction to be applied only to systems with no cycles of non-relevant events, and the OP-search algorithm (Pena et al., 2010) has the same restriction.

Recently, Pena et al. (2014) presented a generalisation of the OP-verifier algorithm that overcomes the limitation on the cycles of non-relevant events. As shown in this paper, the verifier (Pena et al., 2014) does not exhibit the properties of the verifier (Pena et al., 2008) that were exploited in the OP-search (Pena et al., 2010). Therefore, in this paper, we propose a modification of the verifier (Pena et al., 2014), a structure named *hybrid verifier*, which has favourable properties that make it possible to implement a new OP-search algorithm.

This paper is organised as follows. Section 2 introduces preliminary concepts necessary to the development of the paper. Section 3 describes the OP-verifier (Pena et al., 2014) and some related properties. Section 4 introduces the hybrid verifier and discusses its properties. Section 5 introduces a new version of the OP-search that exploits the properties of the hybrid verifier, presents an example and complexity analysis. The results are summarised in Section 6.

2. PRELIMINARIES

In this section, some basic concepts of the Supervisory Control Theory (Ramadge and Wonham, 1989) are revisited. The reader is referred to Cassandras and Lafortune (2008) for a detailed introduction to the theory.

[1] This work has been supported by the Brazilian agencies CAPES, FAPEMIG and CNPq. The fifth author is supported in part by CNPq grant 300953/93-3.

In this framework, the behaviour of a Discrete Event System is described by strings of events taken from a finite alphabet Σ. The set of all finite strings of events in Σ, including the empty string ε, is denoted by Σ^*. A subset $L \subseteq \Sigma^*$ is called a *language*. The *concatenation* of strings $s, u \in \Sigma^*$ is written as su. A string $s \in \Sigma^*$ is called a *prefix* of $t \in \Sigma^*$, written $s \leq t$, if there exists $u \in \Sigma^*$ such that $su = t$. The *prefix-closure* $\overline{L}$ of a language $L \subseteq \Sigma^*$ is the set of all prefixes of strings in L, i.e., $\overline{L} = \{\, s \in \Sigma^* \mid s \leq t \text{ for some } t \in L \,\}$.

In this paper, finite-state automata are used to represent languages. A (nondeterministic) finite-state automaton is a tuple $G = (\Sigma, Q, \rightarrow, Q^\circ)$, where Σ is the finite set of events, Q is the set of states, $\rightarrow \subseteq Q \times \Sigma \times Q$ is the transition relation, and $Q^\circ \subseteq Q$ is the set of initial states. The automaton G is said to be *deterministic* if $|Q^\circ| \leq 1$ and $x \xrightarrow{\sigma} y_1$ and $x \xrightarrow{\sigma} y_2$ imply that $y_1 = y_2$.

The transition relation is extended to strings over Σ^* by making $x \xrightarrow{\varepsilon} x$ for every $x \in Q$ and $x \xrightarrow{s\sigma} z$ if $x \xrightarrow{s} y$ and $y \xrightarrow{\sigma} z$ for some $y \in Q$. Also, $x \xrightarrow{s}$ means $x \xrightarrow{s} y$ for some $y \in Q$ and $x \rightarrow y$ means $x \xrightarrow{s} y$ for some $s \in \Sigma^*$. The notation $x \nrightarrow y$ represents that there is no $s \in \Sigma^*$ such that $x \xrightarrow{s} y$, and $x \nrightarrow$ if there is no $y \in Q$ such that $x \xrightarrow{s} y$. The notation can also be used in sets of states or automata such as: $X \xrightarrow{s} Y$ for $X, Y \subseteq Q$ means that $x \xrightarrow{s} y$ for some $x \in X$ and $y \in Y$, and $G \xrightarrow{s}$ means $Q^\circ \xrightarrow{s}$.

The *generated language* of G is defined as $\mathcal{L}(G) = \{\, s \in \Sigma^* \mid Q^\circ \xrightarrow{s} \,\}$. To express the marking of strings, the alphabet Σ is assumed to contain the special event $\omega \in \Sigma$, which may only appear on selfloops, i.e., $x \xrightarrow{\omega} y$ implies $x = y$. The marked states of G are indicated by selfloops labelled with the event ω, and the *marked language* of G is defined as $\mathcal{L}_m(G) = \{\, s \in (\Sigma \setminus \{\omega\})^* \mid s\omega \in \mathcal{L}(G) \,\}$. A state $x \in Q$ is *accessible* if $G \rightarrow x$, and *co-accessible* if $x \xrightarrow{tw}$ for some $t \in \Sigma^*$. An automaton G is said to be *accessible* if all states are accessible and *nonblocking* if all accessible states are co-accessible.

Let $G = (\Sigma, Q, \rightarrow, Q^\circ)$ be an automaton and $\sim \subseteq Q \times Q$ be an equivalence relation on Q. The *quotient automaton* of G is

$$G/\!\sim \;= (\Sigma, Q/\!\sim, \rightarrow/\!\sim, \widetilde{Q}^\circ) \;, \tag{1}$$

such that $\rightarrow/\!\sim \;= \{\, ([x], \sigma, [y]) \mid x' \xrightarrow{\sigma} y' \text{ for some } x' \in [x] \text{ and } y' \in [y] \,\}$ and $\widetilde{Q}^\circ = \{\, [x^\circ] \mid x^\circ \in Q^\circ \,\}$. Here, $[x] = \{\, x \in Q \mid x \sim x' \,\}$ denotes an equivalence class of $x \in Q$ and $Q/\!\sim \;= \{\, [x] \mid x \in Q \,\}$ is the set that includes all equivalence classes.

In this paper, Σ is partitioned as $\Sigma = \Sigma_r \cup \Sigma_{nr}$, where Σ_r is the set of *relevant* events and Σ_{nr} is the set of *non-relevant* events. For $\Sigma_r \subseteq \Sigma$, the *natural projection* $\theta \colon \Sigma^* \rightarrow \Sigma_r^*$ maps sequences of events in Σ^* to sequences of events in Σ_r^* by erasing events that are not in Σ_r. This concept can be extended to languages by $\theta(L) = \{\, t \in \Sigma_r^* \mid t = \theta(s) \text{ for some } s \in L \,\}$. A property that characterises the natural projection is presented in the following.

Definition 1. (Wong et al., 2000) Let $L \subseteq \Sigma^*$ and $\theta \colon \Sigma^* \rightarrow \Sigma_r^*$ be the natural projection. If for all $s \in \overline{L}$ and all $t \in \Sigma_r^*$ such that $\theta(s)t \in \theta(L)$, there exists $t' \in \Sigma^*$ such that $\theta(st') = \theta(s)t$ and $st' \in L$, then $\theta(L)$ has the *observer property (OP)*.

In words, if a projection satisfies the observer property then the tasks that are performed in the abstracted, i.e., projected, automaton correspond uniquely to marked strings in the original automaton. The observer property is also applied to automata: $\theta(G)$ is called an OP-abstraction if $\theta(\mathcal{L}_m(G))$ has the observer property (Pena et al., 2008).

3. VERIFICATION OF THE OBSERVER PROPERTY

In this section, the verifier introduced in (Pena et al., 2014) is described along with some of its properties and their use in the search of the observer property.

3.1 Constructing the Automaton G_{nr}

In order to deal with systems with cycles of non-relevant events, an automaton G_{nr} is introduced. Let $G = (\Sigma, Q, \rightarrow, Q^\circ)$ be a deterministic nonblocking automaton, and let $\Sigma = \Sigma_r \cup \Sigma_{nr}$. The relation $\xrightarrow{nr} \subseteq Q \times Q$ is defined as follows:

$$x \xrightarrow{nr} y \iff x \xrightarrow{s} y \text{ for some } s \in \Sigma_{nr}^* \;; \tag{2}$$

$$x \xleftrightarrow{nr} y \iff x \xrightarrow{nr} y \text{ and } y \xrightarrow{nr} x \;. \tag{3}$$

If $x \xleftrightarrow{nr} y$, then the states x and y are called *strongly Σ_{nr}-connected states*. If G does not have two distinct Σ_{nr}-connected states, then it is said to be *Σ_{nr}-acyclic*. Any set of strongly Σ_{nr}-connected states is called a *strongly Σ_{nr}-connected component (Σ_{nr}-SCC)* of G.

If each Σ_{nr}-SCC of G is contracted into a single state, the resulting automaton is a *Σ_{nr}-acyclic* automaton, and is called the *strongly Σ_{nr}-connected component automaton G_{nr}*. Formally, it is the quotient automaton

$$G_{nr} = G/\!\xleftrightarrow{nr} = (\Sigma, Q_{nr}, \rightarrow_{nr}, Q_{nr}^\circ) \;. \tag{4}$$

By construction, G_{nr} does not have cycles of non-relevant events (except for selfloops), namely, if $[x] \xleftrightarrow{nr} [y]$ then $[x] = [y]$. Also, for $y \in Q$, $[y]$ is a *terminal component* if, for all $\sigma \in \Sigma_{nr}$ and all $z \in Q$ such that $[y] \xrightarrow{\sigma} [z]$, it is true that $[y] = [z]$.

Example 2. The strongly connected components automaton G_{nr} constructed from G in Fig. 1 for $\Sigma_r = \{\lambda, \omega\}$ is shown in Fig. 2. The states 1, 2, and 3 form the Σ_{nr}-SCC $[1] = \{1, 2, 3\}$ in G_{nr}. Also, $[0] = \{0\}$ and $[4] = \{4\}$.

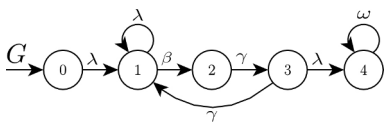

Fig. 1. Automaton G.

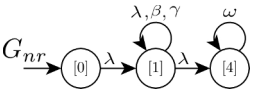

Fig. 2. Automaton G_{nr}.