Supervisory Control of Extended Finite Automata Using Transition Projection

Mohammad Reza Shoaei, Lei Feng, Bengt Lennartson

Abstract—A limitation of the Ramadge and Wonham framework for the supervisory control theory is the explicit state representation using finite automata, often resulting in complex and unintelligible models. Extended finite automata (EFAs), i.e., deterministic finite automata extended with variables, provide compact state representation and then make the control logic transparent through logic expressions of the variables. A challenge with this new control framework is to exploit the rich state abstraction of subsystems and the global model of the entire system is unnecessary. Sufficient conditions are presented to guarantee the decentralized supervisors result in maximally permissive and nonblocking control to the entire system.

I. INTRODUCTION

Supervisory control theory (SCT), established by Ramadge and Wonham [1], [2], is a formal framework for the modeling and control of discrete-event systems (DES). Problems that SCT can address include dynamic allocation of resources, the prevention of system blocking, etc. and, within these constraints, maximally permissive system behavior. Although SCT can systematically synthesize supervisory controllers that are able to prevent a DES from executing undesirable behavior, industrial acceptance is scarce. A number of issues that hinder industrial use have been identified by various researchers such as [3], [4]. Two main issues are the lack of a compact representation of large models and computational complexity.

In the former case, Extended Finite Automata (also called Symbolic Transition Systems), which are ordinary automata augmented with discrete variables, guard expressions and action functions is introduced in [5] and [6]. Extended Finite Automata (EFAs) have been used in several research works and successfully applied to a range of examples such as [7], [8]. Beside a number of methods for synthesizing EFAs [9], [10], [11], the EFA framework in [6] has been implemented in Supremica [12], a verification and supervisory control tool.

Even though EFAs simplify the modeling experience by providing a compact modeling and representation, SCT analysis is performed on their underlying automata models and therefore, the fundamental obstruction to the development of SCT, i.e., the computational complexity of synthesizing maximally permissive and nonblocking supervisors, still remains. Indeed, the nonblocking supervisory control problem for DES is NP-hard [13]. It is well known that the exponential complexity of supervisor design arises from synchronizing subsystems into a global system. Researchers are, therefore, seeking effective control methods for various subclasses of DES that enjoy special structures. Such structures will admit modularity [14], [15], [16], [17] and model abstraction [18], [19], [20] to circumvent computing global dynamic models.

The most effective model abstraction operator in SCT is the causal reporter map having the observer property [14]. While [21] treats hierarchical control using general causal reporter maps, Feng and Wonham [22], [23], construct model abstractions only with natural observers, i.e., natural projections [24], [25], [26] with the observer property. In this method, if two components share only a small number of common events, their abstractions tend to be small, and either verifying the nonconflicting property (if it holds) or designing a coordinator to achieve it may require only modest effort. Natural projection is a language-theoretic operation, which needs the language of a system to be known or can be obtained by its generators, for instance, automata. Unfortunately, this cannot be applied for DES modeled by EFAs. In particular, it makes no sense to speak of the language of individual extended automata, i.e., the language of the components can both be larger than or smaller than the language of the synchronized system. Hence, one cannot enjoy the compositional computation of natural projections.

In this paper, we introduce the transition projection which is an extension of natural projection that can be applied directly on the transition systems of the EFAs rather than their underlying finite automata. Sufficient conditions are presented for maximally permissive nonblocking and controllable controllers with partial observation in EFAs, by preserving the information needed for reliable representation of the nonblocking and controllability properties.

This paper is organized as follows: Section II briefly describes Extended Finite Automata modeling formalism used to model our problems. In Section III, we introduce a model abstraction using transition projection, that is the projection on transition systems, followed by Sections IV and V in which transition projection properties are explained. A practical example has been modeled and abstracted in Section VI and we conclude our work in Section VII.
II. PRELIMINARIES

A. Languages and Automata

The behavior of DES [25], [26] is described in terms of event sequences and regular languages [2]. A regular language is a subset of strings that can be recognized by a finite automaton (FA) \( G = (Q, \Sigma, \rightarrow, q^0, Q^m) \). \( Q \) is the finite state set. \( \Sigma = \Sigma_c \cup \Sigma_u \) is a non-empty finite event set called alphabet. \( \rightarrow \subseteq Q \times \Sigma \times Q \) is the state transition relation mapping elements of \( Q \times \Sigma \times Q \) into singletons of \( Q \). The element \( q^0 \in Q \) is the initial state and \( Q^m \subseteq Q \) is the set of marked states. The transition relation in \( G \) is written in infix notation \( p \xrightarrow{\varepsilon} q \). Let \( \Sigma^* \) be the set of all finite strings over \( \Sigma \), including the empty string \( \varepsilon \). Then, these notations can be extended to strings in \( \Sigma^* \) in the natural way by letting \( p \xrightarrow{\varepsilon} p \) for all \( p \in Q \) and \( p \xrightarrow{\sigma} q \) if \( p \rightarrow r \) and \( r \xrightarrow{\sigma} q \) for \( s \in \Sigma^*, \sigma \in \Sigma, r \in Q \). Let \( p \xrightarrow{\sigma} \) denote the existence of at least one state \( q \) such that \( p \xrightarrow{\varepsilon} q \), and \( p \xrightarrow{\sigma} q \) the existence of a string \( s \in \Sigma^* \) such that \( p \xrightarrow{\sigma} q \). Automaton \( G \) is deterministic if \( p \xrightarrow{\sigma} q \) and \( p \xrightarrow{\sigma} q \) always implies \( q = q \).

An important property of an automaton is nonblocking. The automaton \( G \) is nonblocking if any state reachable from the initial state \( q_0 \) can also reach a marked state via some string, i.e., \((\forall q \in Q) q_0 \xrightarrow{\sigma} q \Rightarrow q \xrightarrow{\sigma} p \) for some \( p \in Q^m \).

Note that, by definition, the symbol \( \varepsilon \) does not belong to either of \( \Sigma, \Sigma_c, \) or \( \Sigma_u \). If it is to be included, the event sets \( \Sigma_\varepsilon = \Sigma \cup \{ \varepsilon \}, \Sigma_{\varepsilon,c} = \Sigma_c \cup \{ \varepsilon \}, \text{ and } \Sigma_{\varepsilon,u} = \Sigma_u \cup \{ \varepsilon \} \) are used instead. Given two event sets \( \Sigma \) and \( \Sigma_0 \subseteq \Sigma \), the natural projection is the function \( P : \Sigma^* \rightarrow \Sigma_0^* \) such that \( P(\varepsilon) = \varepsilon, P(\sigma) = \varepsilon, \sigma \in \Sigma - \Sigma_0 \), and \( P(\sigma) = \varepsilon, \sigma \in \Sigma_0 \).

The effect of \( P \) on a string \( s \in \Sigma^* \) is just to erase the events in \( s \) that do not belong to \( \Sigma_0 \), but keep the events in \( \Sigma_0 \) unchanged. The inverse image of the natural projection \( P \) is a function \( P^{-1} : Pwr(\Sigma_0^*) \rightarrow Pwr(\Sigma^*) \) where \( Pwr \) is the power set.

B. Extended Finite Automata

An extended automaton can be extended with a set of variables to an Extended Finite Automaton (EFA) whose transitions are augmented with conditions and actions on these variables to enjoy a compact and symbolic description of DES.

Let \( V = \{ v_1, \ldots, v_n \} \) be the set of \( n \) typed variables over the finite domain \( D = D_1 \times \cdots \times D_n \). Let \( E(V) \) denote the set of \( (\sigma) \) evaluations \( \eta \) that assigns values to variables. \( G \) is the set of Boolean conditions over \( V \) in which each condition \( g \), also called guard, is a propositional logic formula whose propositional symbols are of the form \( \overline{v} \in D \), where \( \overline{v} = (v_1, \ldots, v_n) \) is an \( n \)-tuple of pairwise distinct variables in \( V \), and \( D \) is a subset of the domain \( D \). For the sake of simplicity, we write the propositional symbols such as \( \overline{\sigma} = \overline{\sigma} \leq 2 \) instead of \( \{ (x, y) \in \{ (m, n) \in D_n \times D_n | m - n \leq 2 \} \} \). A satisfaction relation \( \models \) for a guard \( g \) can be defined as a set of pairs \( (\eta, g) \) indicating the evaluations \( \eta \) for which the guard \( g \) is satisfied. It is written \( \eta \models g \) instead of \( (\eta, g) \in \models \).

Given two guards \( g \) and \( h \), we say that \( g \) is a subguard of \( h \), denoted \( g \leq h \), if \( g \land h = g \). We say \( g \) and \( h \) have the same satisfaction, denoted \( g = h \), if for all \( \eta \in E(V), \eta \models g \iff \eta \models h \).

Let \( A \) be the set of actions over the set of variables in which \( a \in A \) is a function \( a : D \rightarrow D \), updating the current variables value to a new value. The symbol \( \xi \) is used to indicate that variables are not updated, namely, takes their current values.

**Definition 1 (Extended Finite Automaton)**

An extended finite automaton over the set of variables \( V \) is a tuple \( (L, D, \Sigma, T, \ell^0, d^0, L^m, D^m) \) where:

- \( L \) is a finite set of discrete locations,
- \( D = D_1 \times \cdots \times D_n \) is the finite domain of variables,
- \( \Sigma \) is a nonempty finite set of events (alphabets),
- \( T \subseteq L \times \Sigma \times G \times A \times L \) is the transition relation,
- \( \ell^0 \in L \) is the initial location,
- \( d^0 \in D \) is the tuple of initial values,
- \( L^m \subseteq L \) is the set of marked (desired) locations,
- \( D^m \subseteq D \) is the set of marked values of the variables.

The notation \( \ell \xrightarrow{\sigma} g/a \ell \) is used as shorthand for \( (\ell, \sigma, g, a, \ell) \in T \). If the guard of the conditional transition \( \ell \xrightarrow{\sigma} g/a \ell \) is a tautology then we simply write \( \ell \xrightarrow{\sigma} \ell \). For two EFAs \( E_1 \) and \( E_2 \), with the same set of events, domain, initial location, and initial variables value we say \( E_1 \) is a sub-EFA of \( E_2 \), written \( E_1 \subseteq E_2 \), if \( L_1 \subseteq L_2, T_1 \subseteq T_2, D^{m}_1 \subseteq D^{m}_2 \), and \( L^{m}_1 \subseteq L^{m}_2 \).

The semantics of an EFA is described in terms of a DFA.

**Definition 2 (EFA Semantics)**

Let \( E = (L, D, \Sigma, T, \ell^0, d^0, L^m, D^m) \) be an EFA over the set of variables \( V \). The DFA \( G(E) \) of \( E \) is the tuple \( (Q_E, \Sigma_E, \rightarrow_E, q^0_E, Q^m_E) \) where \( Q_E = L \times D, \Sigma_E = \Sigma, q^0_E = (\ell^0, d^0), Q^m_E = L^m \times D^m, \rightarrow_E \subseteq Q \times \Sigma \times Q \) is defined by the following rule:

\[ \ell \xrightarrow{\sigma} g/a \ell \land \eta \models g \]

\[ (\ell, \eta) \xrightarrow{\sigma} (\ell, a(\eta)) \]
definition of EFSC, it is assumed that the variables are shared by all EFAs with the same initial values.

**Definition 3 (EFSC).**

Let \( E = (L_k, D, \Sigma_k, T_k, \ell_k^0, d^0, L_k^n, D_k^n) \), \( k = 1, 2 \), be two EFAs over the set of shared variables \( V \). The Extended Full Synchronous Composition of \( E_1 \) and \( E_2 \) is the tuple

\[
E_1 \parallel E_2 = (L, D, \Sigma, T, \ell_0, d_0, L^n, D^n),
\]

where \( L = L_1 \times L_2 \), \( \Sigma = \Sigma_1 \cup \Sigma_2 \), \( \ell_0 = \langle \ell_1^0, \ell_2^0 \rangle \), \( L^n = L_1^n \times L_2^n \), and \( T \) is defined by the following rules:

\[
\begin{align*}
\ell_1 \sigma_1 \mathrel{\triangleleft} \ell_1, g_1/a_1, \ell_2 \sigma_2 \mathrel{\triangleleft} \ell_2, g_2/a_2 & \quad \Rightarrow \quad \langle \ell_1, \ell_2 \rangle \sigma_1 \mathrel{\triangleleft} g_1/a_1 \langle \ell_1, \ell_2 \rangle \sigma_2 \mathrel{\triangleleft} g_2/a_2,
\end{align*}
\]

\[
\begin{align*}
\ell_2 \sigma_2 \mathrel{\triangleleft} \ell_2, g_2/a_2, \ell_1 \sigma_1 \mathrel{\triangleleft} \ell_1, g_1/a_1 & \quad \Rightarrow \quad \langle \ell_1, \ell_2 \rangle \sigma_1 \mathrel{\triangleleft} g_1/a_1 \langle \ell_1, \ell_2 \rangle \sigma_2 \mathrel{\triangleleft} g_2/a_2,
\end{align*}
\]

such that \( g = g_1 \land g_2 \) and

\[
a = \begin{cases} 
 a_1 & \text{if } a_1 = a_2 \\
 a_1 & \text{if } a_2 = \xi \\
 a_2 & \text{if } a_1 = \xi \\
 \eta & \text{otherwise}
\end{cases} \quad \text{and} \quad \eta \neq \eta.
\]

Note that, if the action functions of \( E_1 \) and \( E_2 \) try to update a shared variable to different values, the variable is, by default, not updated.

We introduce the notion of local events for a system consists of more than one EFA component which will be used later. For an event \( \sigma \), let \( \text{Act}(\sigma) \subseteq A \) and \( \text{Con}(\sigma) \subseteq G \) be the sets of actions and guards, respectively, retrieved from all transitions labeled with \( \sigma \).

**Definition 4 (Local Event).**

An event \( \sigma \in \Sigma_i \) is local to \( E_i, i \in I \) where \( I \) is an index set if for all \( j \in I \) we have (i) \( \sigma \in \Sigma_i - \bigcup \Sigma_j (j \neq i) \), (ii) \( (\forall g \in \text{Con}(\sigma)) g \) is a tautology, (iii) \( (\forall a \in \text{Act}(\sigma); \forall \eta \in E(V); \forall g \in \bigcup G) \eta \equiv g \Rightarrow a(\eta) \equiv g \).

In above, condition (i) guarantees that the event \( \sigma \) only appears in \( E_i \), (ii) ensures that guards on any transition labeled by \( \sigma \) evaluates to true; hence \( \sigma \) can cause the transition to occur at any time, and (iii) guarantees that the actions on all transitions labeled by \( \sigma \) has no effect on any guard in the system. We say an event is shared with other EFAs when it is not local. Also, any transition labeled with a local event is called a local transition.

Moreover, we use the notion of executions (also called runs) to describe a possible behavior of the transition system.

**Definition 5 (Execution Fragment).**

An execution fragment \( \varrho \) in \( E \) is a series of finite transitions in \( T, \varrho = \ell_0 \sigma_1 \ell_1 \sigma_2 \ell_2 \ldots \sigma_{i-1} \ell_{i-1} \sigma_i \ell_i, (0 \leq i < n) \), where \( n \geq 0 \) and the variables evaluation \( \eta_{i+1} = a(\eta_i) \). The integer \( n \) is the length of the \( \varrho \) and \( \varrho = \ell_0 \) for some \( \ell_0 \in L \) is a legal execution fragment of length \( n = 0 \).

The first and last location of \( \varrho \) is denoted by \( \text{first}(\varrho) \) and \( \text{last}(\varrho) \), respectively. We call an execution fragment \( \varrho \) initial if \( \text{first}(\varrho) = \ell_0 \), marked if \( \text{last}(\varrho) \in L^n \), and local if all of its transitions are local. For two execution fragments \( \varrho, \dot{\varrho} \), we say \( \varrho \) is a precedence of \( \dot{\varrho} \), written \( \varrho \sqsubseteq \dot{\varrho} \), if \( \text{last}(\dot{\varrho}) = \text{first}(\varrho) \) and we say \( \varrho = \dot{\varrho} \) if they have the same alternating sequence of transitions up to renaming of locations.

**C. Supervisory Control of EFA**

SCT is a formal framework for the modeling and control of DES consisting of a plant and a specification. A supervisor for a control problem modeled by EFAs can be symbolically computed using the algorithm presented in [11]. The algorithm iteratively strengthens the guards on conditional transitions to avoid forbidden or blocking states.

Given a DES control problem, we assume that the plant is modeled by an EFA \( P \) and the specification by an EFA \( Sp \).

The specification can be represented, without loss of generality, by a set of forbidden locations which can be obtained by a refined plant model \( R \) with the same behavior as \( P \) such that the executions not allowed in \( Sp \) end up in certain forbidden locations in \( R \). See [11] for more elaboration on refinement. From now on, we assume that the plant model is given as the refined EFA \( R \) and the specification is given as the set of forbidden locations \( L_f \subseteq L_R \).

Let us denote the set of safe locations by \( L_s = L - L_f \) and recall the set of reachable states \( Q_R \) in \( G(H) \). A state \( q = (\ell, \eta) \in Q_R \) is a forbidden state iff \( \ell \in L_f \), otherwise, \( q \) is a safe state. In the sequel, \( R_s \) denotes the EFA obtained from \( R \) by assigning false to the guard \( g \) of every transition \( \ell \mathrel{\triangleleft} g/a \ell \) for which \( \ell \in L_f \), i.e., \( \ell \) is a forbidden location. \( R_s \) is constructed such that \( R_s \subseteq R \) and is called the safe sub-EFA of \( R \).

**Definition 6 (Nonblocking, Safety, Controllability).**

[11] Let \( R \) be an EFA, \( L_f \) its set of forbidden locations, and \( R_s \) its safe subautomaton. A reachable state \( q \in Q_R \) is: (a) nonblocking if there exists a state \( p \in Q_R \) such that \( q \mathrel{\Rightarrow} p \) for some string \( s \in \Sigma^* \); (b) safe if \( q \in Q_R \), and (c) \((R, L_f, \Sigma_u)\)-controllable (or simply controllable when clear from context) if \( q \) is safe and \( \forall \sigma \in \Sigma_R(q) \cap \Sigma_u \) where \( \Sigma_R(q) \) denote the set of active events, we have \( Q_R(q, \sigma) \subseteq Q_R \). The EFA \( R \) is, respectively, nonblocking, safe, and controllable if every reachable state of \( R \) is, respectively, nonblocking, safe, and controllable.

A supervisor \( S \) for \( R \) can be seen as a function \( S : T \to G \) which maps each transition to a supervision guard such that \( S(\ell \mathrel{\triangleleft} g/a \ell) \leq g \) if \( \sigma \in \Sigma_s \) and \( S(\ell \mathrel{\triangleleft} g/a \ell) = g \) if \( \sigma \in \Sigma_u \). Let \( R^S \) denote the sub-EFA obtained from \( R \) by replacing its guards by those provided by \( S \). Then, \( S \) is said to be nonblocking if \( R^S \) is nonblocking and safe if \( R^S \) is safe. In case \( R^S \) is blocking or uncontrollable, a search will be performed to find a safe and nonblocking supervisor \( S \) such that \( R^S \subseteq R_s \). Let \( S(R, L_f) \) denote the set of nonblocking and safe supervisor candidates of \( R \), then \( S^T := \sup(S(R, L_f)) \), is the most permissive nonblocking and safe supervisor compare to any other supervisor in \( S(R, L_f) \).
when the latter is nonempty. The $R^S$ is called the supremal controllable and nonblocking sub-EFA of $R_s$.

$R^S$ is calculated by the Supervisory Synthesis for EFA (SSEFA) using a fixed-point iteration method. Given a refined EFA $R$ and a set $L_f \subset L$ of forbidden location, SSEFA($R, L_f$) computes stronger, maximally permissive, guards for the transitions of $R$ in $N$ steps such that the obtained EFA is nonblocking, safe, and controllable [11].

III. EFA Projection

Traditionally, brute-force computation is used for verification and coordination [25], [18]. This we wish to avoid since the nonblocking supervisory control problem in SCT [18] is NP-hard [13]. Abstraction introduces hierarchy into system structure, as it reports only the events shared with other subsystems and conceals the rest. The fewer the reported events, the greater state reduction will be achieved. In order to use a model abstraction using natural projection on EFAs, we introduce the transition projection which is an extension of natural projection to abstract systems modeled by EFAs.

We present sufficient conditions for optimal nonblocking controller with partial observation in EFA by preserving the information needed for reliable representation of the nonblocking and controllability properties.

For an EFA $E$ with the set of events $\Sigma$, the transition projection, written with a slight abuse of notation $\bar{P}$, for the conditional transition relation $T$ and the set $\Sigma_\ell \subseteq \Sigma$ is a function $\bar{P} : T \times \Sigma_\ell \rightarrow T$ defined as follows: for every transition $\ell \xrightarrow{\sigma, g, a, \ell} \ell' \in T$,

$$\bar{P}(\ell \xrightarrow{\sigma, g, a, \ell}, \ell) = \ell \xrightarrow{\sigma, g, a, \ell'}, \sigma \neq \gamma$$

$$\bar{P}(\ell \xrightarrow{\sigma, g, a, \ell}, \gamma) = \ell \xrightarrow{\sigma, g, a, \ell'}$$

The transition projection $\bar{P}$ replaces the label of transitions labeled by events in $\Sigma_\ell$ with symbol $\varepsilon$. In effect, an EFA is allowed to make a transition spontaneously, without receiving an input event. Extending $T$ to its power set $Pwr(T)$, we get $\bar{P} : Pwr(T) \times \Sigma_\ell \rightarrow Pwr(T)$ such that for any $\tau \in \Sigma_\ell$, $N \subseteq T \cap \Sigma_\ell : \bar{P}(N, \tau) = \{ \ell \xrightarrow{\sigma, g, a, \ell} | (\ell, \sigma, g, a, \ell) \in T \}$. If we extend $\Sigma_\ell$ to its power set $Pwr(\Sigma_\ell)$, $\bar{P}$ becomes $\bar{P} : Pwr(T) \times Pwr(\Sigma_\ell) \rightarrow Pwr(T)$ such that for $A \subseteq \Sigma_\ell$, $N \subseteq T : \bar{P}(N, A) = \bigcup \{ \bar{P}(N, \tau) | \tau \in A \}$. If the action of $\bar{P}$ on $T$ is understood then $\bar{P}(T, \Sigma_\ell)$ may be written $\bar{P}_\Sigma T$ and if $\bar{P}$ is defined $\bar{P}T$.

Given any EFA, Algorithm 1, denoted by $\bar{P}$, computes the projected EFA. The intuition of the algorithm is the following. Let $\Sigma_\ell(\ell)$ be the set of $\varepsilon$-closure of a location $\ell$ in $E$. $\Sigma_\ell(\ell)$ is constructed recursively by finding every location that can be reached from $\ell$ along any path whose transitions are all labeled $\varepsilon$. Formally, (1) $\ell \in \Sigma_\ell(\ell)$, (2) $(\forall \ell' \in \Sigma_\ell(\ell)) \ell \xrightarrow{\varepsilon, g, a, \ell'} \ell' \Rightarrow \ell' \in \Sigma_\ell(\ell)$. The location set of $E$ will be denoted by $\ell$, with element $\ell$ that label $\varepsilon$-closure subsets of $E$. The transition system of $E$ is constructed as follows. Define the initial location subset $\ell^0 := \Sigma_\ell(\ell^0)$. Choose $\ell_1 \in \Sigma - \Sigma_\ell$ and define $\ell_1 := \bigcup_{\ell \in \ell^0} \{ \ell \xrightarrow{\sigma, g, a, \ell} | (\ell, \sigma, g, a, \ell) \in T \}$. Define $\ell_2$ similarly, from $\ell^0$ and $\ell_2 \in \Sigma - \Sigma_\ell - \{ \ell_1 \}$, and repeat until $\Sigma - \Sigma_\ell$ is exhausted. The subset obtained at any step is discarded if it is empty or if it appeared previously. This process yields a list of (final) distinct nonempty subsets $\ell^0, \ell^1, \ldots, \ell^k$ and one-step ‘subset’ transitions of form $(\ell^0, \sigma, g, a, \ell^1), \sigma \in \Sigma - \Sigma_\ell$, $i \in \{0, 1, \ldots, k\}$. The procedure is repeated with each of the subsets $\ell^k, \ell^k, \ldots, \ell^k$ and each $\sigma \in \Sigma - \Sigma_\ell$, until no new subset transitions are obtained. The result is the projected EFA $E' = (\ell, D, \Sigma, \bar{T}, \bar{\ell}^0, \bar{d}^0, \bar{L}^m, D^m)$, where $\bar{L}$ is the final list $(\bar{\ell}^0, \bar{\ell}^1, \ldots, \bar{\ell}^k)$, $(\bar{\ell}^0, \bar{\ell}^1, \ldots, \bar{\ell}^k)$, and $(\ell, \sigma, g, a, \ell') \in T$ iff $(\ell, \sigma, g, a, \ell') \in T$ for some $\ell \in \ell$, $\ell' \in \ell'$, $\sigma \in \Sigma - \Sigma_\ell$.

We assume a DES to consist of a group of simple plant EFAs subject to a conjunction of modular control specifications. Consider a system consisting of two EFA components, $E_1$ and $E_2$. To obtain a reduction of the system, we could first compute the systems global behavior $E_1 \parallel E_2$ and then its transition projection. However, when, however, the local events of the two components are all defined the result is obtained more economically from reductions of the components.

**Proposition 1.**
Let $E_k = (L_k, D, \Sigma_k, T_k, \ell^0_k, d^0_k, L_k^m, D_k^m), k = 1, 2$, be two EFAs. Consider $T$ as the set of transition relation for $E_1 \parallel E_2$ and $\Sigma_k \subseteq \Sigma_k \cup \Sigma_2$. Define $\bar{P} : T \times \Sigma_k \rightarrow T$ and $\bar{Q}_i : T_i \times (\Sigma_k \cap \Sigma_i) \rightarrow T_i (i = 1, 2)$, if $\Sigma_k$ is the set of local events then $\bar{P}(E_1 \parallel E_2, \Sigma_k) \in \bar{Q}_1(E_1, \Sigma_1 \cap \Sigma_k) \parallel \bar{Q}_2(E_2, \Sigma_2 \cap \Sigma_k).

**Proof:** See [28].

The extension to an arbitrary number of synchronized factors is straightforward and is left out.
IV. OBSERVER OF EFA

Consider a DES described by EFA $E$. Given a set of local events, we can define the transition projection $\hat{P} : T \times \Sigma_\ell \rightarrow T$ and then the projected EFA $\tilde{P}(E, \Sigma_\ell)$. The resulting projected EFA is not guaranteed to be coreachable, or nonblocking for $E$. Crucial to successful model abstraction using transition projection is that the projected system contains necessary and sufficient information needed for reliable representation of the nonblocking property. Therefore, one must carefully select the local events of a DES.

A “good” selection of local events for any transition projection is whenever a projected EFA reaches a marked location via some projected execution fragments, the original system must be able to reach a marked location by those execution fragments as follows.

**Definition 7** $(E$-observer$)$.

Assume a nonblocking EFA $E$ and let $\Sigma_\ell \subseteq \Sigma$ be the subset of local events. The transition projection $\hat{P} : T \times \Sigma_\ell \rightarrow T$ is an $E$-observer, if for all initial execution fragments $\varrho_1$ and $\varrho_2$ and for all marked execution fragment $\varrho_3$ in $E$ such that $\varrho_3 \subseteq \varrho_1$ and $\hat{P}\varrho_3 = \hat{P}\varrho_4$, there exists a marked execution fragment $\varrho_4$ in $E$ such that $\varrho_4 \subseteq \varrho_2$ and $\hat{P}\varrho_4 = \hat{P}\varrho_2$.

Note that if $\Sigma_\ell$ is equal to $\Sigma$ or $\emptyset$ for an EFA $E$ then $\hat{P}$ is automatically an $E$-observer. It can be shown, by similar reasoning on reachable states as in [29], that the model abstractions computed by transition projection with observer property are guaranteed to have location sizes no larger than the original model.

For a system consists of more than one plant component it would be more economical to check the observer property component-wise without computing the synchronous product first. Proposition 2 presents a sufficient condition for this simplification to be valid.

**Proposition 2.** Let $E_k = (L_k, D, \Sigma_k, T_k, \ell_0, \delta_0, L_0^k, D_0^k), k = 1, 2$, be two nonblocking EFAs. Consider $T$ as the set of transition relation for $E_1 \| E_2$. Define the transition projections $\hat{P} : T \times \Sigma_\ell \rightarrow T$ and $\hat{Q}_i : T_1 \times (\Sigma_1 \cap \Sigma_2) \rightarrow T_i$ ($i = 1, 2$) where $\Sigma_\ell \subseteq \Sigma := \Sigma_1 \cup \Sigma_2$. If $\Sigma_\ell$ is the set of local events and for both $i = 1, 2$, $Q_i$ is an $E_i$-observer, then $\hat{P}$ is an $E_1 \| E_2$-observer.

**Proof:** See [28].

As we establish a “reliable interface” for EFAs by introducing $E$-observer, the interaction between two complex system may be examined through their projections rather than their global behavior. If $\hat{P}$ has the observer property, we can check if two EFAs $E_1$ and $E_2$ are synchronously nonconflicting by checking whether their projections are synchronously nonconflicting and we may save significant computational effort, in accordance with the following.

**Theorem 1** (Synchronously Nonconflicting Criterion). Let $E_k = (L_k, D, \Sigma_k, T_k, \ell_0, \delta_0, L_0^k, D_0^k), k = 1, 2$, be two EFAs and let $\Sigma_\ell \subseteq \Sigma := \Sigma_1 \cup \Sigma_2$ be the set of local events. If $\hat{Q}_i : T_1 \times (\Sigma_1 \cap \Sigma_\ell) \rightarrow T_i$ are $E_i$-observer ($i = 1, 2$), then $E_1 \| E_2$ is nonblocking if and only if $\dot{Q}_1(E_1, \Sigma_1 \cap \Sigma_\ell) \| \dot{Q}_2(E_2, \Sigma_2 \cap \Sigma_\ell)$ is nonblocking.

The proof needs the following Lemma. Let denote $E = \emptyset$ when there is no ongoing transition from the initial location of $E$.

**Lemma 1.** In the notation of Proposition 2, define the transition projection $\hat{P}_i : T \times (\Sigma_i - \Sigma_\ell) \rightarrow T$ ($i, j = 1, 2; i \neq j$).

If $\Sigma - \Sigma_\ell \neq \emptyset$ and there exists $\ell_i, \tilde{\ell}_i \in T_i$ such that $\sigma \in \Sigma - \Sigma_\ell$ for some $\ell_i, \tilde{\ell}_i \in L_i$ ($i = 1, 2$) then $E_1 \| E_2 \neq \emptyset \Leftrightarrow \dot{Q}_1(E_1, \Sigma_1 \cap \Sigma_\ell) \| \dot{Q}_2(E_2, \Sigma_2 \cap \Sigma_\ell) \neq \emptyset$.

**Proof:** See [28].

Returning to the proof of Theorem 1, define the transition projections $\hat{P}_i : T \times (\Sigma_i - \Sigma_\ell) \rightarrow T$ ($j \neq i$), $\hat{Q}_i : T \times (\Sigma_i \cap \Sigma_\ell) \rightarrow T$, $\hat{Q}_i := \dot{Q}_i \circ \hat{P}_i (i, j = 1, 2)$ and let $E_1 = \dot{Q}_1(E_1, \Sigma_1 \cap \Sigma_\ell), E_2 = \dot{Q}_2(E_2, \Sigma_2 \cap \Sigma_\ell)$.

**Proof of Theorem 1:** (if) Let $\varrho_2$ be an initial execution fragment in $E_1 \| E_2$. We must show that there exists a marked execution fragment $\varrho_3$ such that $\varrho_3 \subseteq \varrho_2$. Applying $\hat{P}_1$ to $\varrho_2$, we get $\hat{P}_1\varrho_3 \subseteq \varrho_3$. We also know that $\hat{P}(\varrho_3) \in \dot{Q}(E_1 \| E_2, \Sigma_\ell)$. Because of the assumption that $\Sigma_\ell$ is the set of local events and by Proposition 1, $\hat{P}\varrho_3 \subseteq \hat{P}\varrho_2$. Applying $\hat{R}_i$ on both sides, we get $\hat{R}_i \circ \hat{P}_i \varrho_3 = \hat{R}_i \circ \hat{P}\varrho_3$, and as required is marked and $\varrho_3 \subseteq \varrho_2$.

(only if) According to the assumption $E_1 \| E_2$ is nonblocking and therefore, for any initial execution fragment $\varrho_3$ there exists a marked execution fragment $\varrho_3$ such that $\varrho_3 \subseteq \varrho_2$. Applying $\hat{P}$ on both $\varrho_2$ and $\varrho_3$, we get, respectively, $\hat{P}_1\varrho_3 \subseteq \hat{P}\varrho_3$ and $\hat{P}_1\varrho_3 \subseteq \hat{P}\varrho_3$. This implies that $\hat{P}_1\varrho_3 \subseteq \hat{P}\varrho_3$, and as required is marked and $\varrho_3 \subseteq \varrho_2$.

In case two EFAs $E_1$ and $E_2$ are synchronously conflicting, a third EFA $E$, called a coordinator, must be introduced to resolve the conflict. We can now, instead of computing the coordinator directly from the two EFAs themselves, we perform this computation through their abstractions.

**Proposition 3.** Let $E_k = (L_k, D, \Sigma_k, T_k, \ell_0, \delta_0, L_0^k, D_0^k), k = 1, 2$, be two synchronously conflicting EFAs and let $\Sigma_\ell \subseteq \Sigma := \Sigma_1 \cup \Sigma_2$ be the set of local events. If $\hat{Q}_i : T_1 \times (\Sigma_i \cap \Sigma_\ell) \rightarrow T_i$ are
Fig. 1: For the EFA E with $\Sigma_\ell = \{\alpha, \beta\}$, $\Sigma_u = \{\beta, \gamma\}$ the transition projection $\bar{P}$ is not OCC since there exists an execution fragment $\bar{q} = \ell_0 \xrightarrow{\alpha} \ell_1 \xrightarrow{\beta} \ell_2 \xrightarrow{\gamma} \ell_3$ such that $\gamma \notin \Sigma_\ell$, $\{\alpha, \beta\} \in \Sigma_\ell$, $\gamma \in \Sigma_u$ but $\alpha \notin \Sigma_u$.

$E_i$-observer $(i = 1, 2)$ and there exists an EFA E such that $\bar{Q}_1(E_1, \Sigma_1 \cap \Sigma_\ell)$ and $\bar{Q}_2(E_2, \Sigma_2 \cap \Sigma_\ell)$, it can resolve the conflict between $E_1$ and $E_2$.

V. OPTIMAL NONBLOCKING AND CONTrollable SUPERVISOR

An optimal supervisor with full observation usually disables the nearest controllable events preceding or “upstream” to a prohibited uncontrollable event (say, $\sigma$). If, however, some of these controllable events are unobservable, a decentralized supervisor must disable controllable events further back, and so is more restrictive. For this restriction to be relaxed, the local event set must be selected properly enough to contain all the upstream controllable events nearest to $\sigma$. Such a decentralized supervisor will prevent the occurrence of an uncontrollable event while allowing maximal freedom of system behavior. A transition projection with such a local event set is called Output Consistent (OCC).

Definition 8 (OCC).

Let $E = (L, D, \Sigma, T, \ell_0, d^0, L^m, D^m)$ be an EFA and let $\Sigma_\ell, \Sigma_u \subseteq \Sigma$ be the local and uncontrollable event sets. The transition projection $\bar{P} : T \times \Sigma_\ell \to T$ is output control consistent (OCC) for the EFA E, if for every finite execution fragment $q$ of the form

$\bar{q} = q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_{n-1}} q_n \xrightarrow{\sigma_n} q_{n+1}$

which satisfies the conditions that $n \geq 1$, $\sigma_i \in \Sigma - \Sigma_\ell$, $\sigma_j \in \Sigma_\ell$ ($j \in \{n\langle i \rangle \}$) and $\sigma_n \in \Sigma - \Sigma_\ell$, we have the property that $\sigma_n \in \Sigma_u \implies (\forall i \in n) \sigma_j \in \Sigma_u$.

In above definition, when $\sigma_n$ is not local and uncontrollable, its immediately preceding local events must all be uncontrollable, namely, its nearest controllable event must not be unobservable.

Example 1. Consider EFA $E$ in Fig. 1 where $\Sigma = \{\alpha, \beta, \gamma, \delta\}$, $\Sigma_\ell = \{\alpha, \beta\}$, $\Sigma_u = \{\beta, \gamma\}$ are, respectively, the sets of alphabet, local events, and uncontrollable events. Let $P : T \times \Sigma_\ell \to T$ be the transition projection. In this, $\bar{P}$ is not OCC for $E$ since there exists an execution fragment $q = \ell_0 \xrightarrow{\alpha} \ell_1 \xrightarrow{\beta} \ell_2 \xrightarrow{\gamma} \ell_3$ such that $\gamma \notin \Sigma_\ell$, $\alpha, \beta \in \Sigma_\ell$ but $\alpha \notin \Sigma_u$.

We can now state a sufficient condition for Optimal Nonblocking and Controllable Supervisor (ONCS).

Theorem 2 (ONCS).

Let $E$ be a nonblocking EFA along with local and uncontrollable event sets $\Sigma_\ell, \Sigma_u \subseteq \Sigma$, respectively. Define the transition projection $\bar{P} : T \times \Sigma_\ell \to T$ and let EFA $\bar{E} = \bar{P}(E, \Sigma_\ell)$. Suppose the set of forbidden locations is $\bar{L}_f \subset \bar{L}$. If the transition projection $\bar{P}$ is an $E$-observer and OCC for $\bar{E}$, then $supS(E, \bar{L}_f) = supS(\bar{E}, \bar{L}_f)$.

$supS(\bar{E}, \bar{L}_f)$ denotes the optimal nonblocking and controllable supervisor with full observation that can be obtained for $E$. Similarly, $supS(E, \bar{L}_f)$ describes the decentralized supervisor with partial observation on $\Sigma - \Sigma_\ell$. When this supervisor is synchronized with the plant, the final controlled behavior is the system on the right side of the equation.

Proof: It needs to be shown that the reachable states of the fixed points of $supS(E, \bar{L}_f)$ and $supS(\bar{E}, \bar{L}_f)$ is, i.e., $G^N := (SSEFA(E, \bar{L}_f))^N$ and $\bar{G}^N := (SSEFA(\bar{E}, \bar{L}_f))^N$, respectively, are the same. This can be proved by an induction on the step iterator $j$.

(\subseteq) BASE: Let $j = 0$. By definition $G^N \subseteq L \times D = G^0$. INDUCTION: Assuming that the property holds for $j$ it needs to be shown that it also holds for $j+1$. Let $p = (\ell, \eta) \in G^N$. By the inductive assumption it holds that $p \notin G^j$. Assume that $p \notin G^{j+1}$. This implies that either $p$ is $(\alpha)$ uncontrollable or $(\beta)$ blocking state and therefore removed by the synthesis algorithm.

(\alpha) Then there exists $\nu \in \Sigma u$ such that $\nu \xrightarrow{\nu}_G \notin G^j$ for some $\nu = (\ell, \eta) \in L \times D$. Assume $\nu \in \Sigma - \Sigma_\ell$. Then $\nu$ is not projected by $\bar{P}$ so the same transition exists in $E$. Therefore, $\nu \xrightarrow{\nu}_G \notin G^N \supseteq G^N$. But then $p \notin G^N$, which is in contradiction. Now, assume $\nu \notin \Sigma_\ell$. Then for all execution fragments of the form $p \xrightarrow{\nu}_G \cdots \xrightarrow{\nu}_G \cdots \xrightarrow{\nu}_G$, $t \xrightarrow{\nu}_G \nu (k \geq 1)$ in $G^j$ such that $\sigma_1 = v$ and satisfies the conditions $\sigma_1, \ldots, \sigma_k \in \Sigma_\ell(i \in k\langle 1 \rangle)$ and $\sigma_k \notin \Sigma - \Sigma_\ell$. Observe that the subset of states $\{p, \cdots , t\}$ is labeled by $p$ in $E$. If $\sigma_k \notin \Sigma_u$ then by the assumption that $\bar{P}$ is OCC for $E$ we can immediately see $(\forall i \in k)\sigma_i \notin \Sigma_u$. Consequently, $p \xrightarrow{\nu}_G \cdots \xrightarrow{\nu}_G \nu \notin G^{j+1}$ and $p \notin G^N$ which is in contradiction. If $\sigma_k \in \Sigma_u$ then it must be the case that $q \xrightarrow{\nu}_G \notin G^j$. Assume $\sigma_2 \in \Sigma_\ell$, then $q \in G^j$ which is in contradiction. Therefore, $\sigma_2$ must be uncontrollable, i.e., $\sigma_2 \in \Sigma_u$. By similar reasoning we can see that $\sigma_i \in \Sigma_u (\forall i \in k\langle 1 \rangle)$. Since $\sigma_k \in \Sigma_u$ implies that the state $t$ is not removed by the synthesis algorithm hence $\{p, q, \cdots , t\} \in G^j$ which is in contradiction to the assumption that $q \notin G^j$.

(\beta) Then $p \xrightarrow{\nu}_G \nu \xrightarrow{\nu}_G \nu$ implies $q \notin L^m \times D^m$, $s \in \Sigma_\ell$, and $t \in (\Sigma - \Sigma_\ell)^*$. Since $\bar{P}$ is E-observer then $p \xrightarrow{\nu}_G q \in \bar{G}^N$ also implies $q \notin L^m \times D^m$ thus $p \notin \bar{G}^N$, which is in contradiction to the initial assumption.
Proposition 4. Let $E_1$ and $E_2$ be two nonblocking EFAs along with local and uncontrollable event sets $\Sigma_i, \Sigma_u \subseteq \Sigma := \Sigma_1 \cup \Sigma_2$, respectively, and let $E := E_1 \parallel E_2$. Define the transition projections $\bar{P} : T_E \times \Sigma \rightarrow T_{E_1} \bar{Q}_i : T_i \times (\Sigma_i \cap \Sigma_u) \rightarrow T_i$ $(i = 1, 2)$ and let $EFA \ E = \bar{P}(E, \Sigma_u)$. Suppose the set of forbidden locations is $L_f \subseteq L_E$. If for $i = 1, 2$, $\bar{Q}_i$ is an $E_i$-observer and OCC for $E_i$ then $\sup(S(E, L_f)) = \sup(S(E, L_f))∥E$.

Proof: See [28].

By an argument similar to that for Theorem 2 we can further extend the Proposition 4 for $n$ number of EFAs as follows.

Corollary 1. Let $E_1$ and $E_2$ be two nonblocking EFAs along with local and uncontrollable event sets $\Sigma_i, \Sigma_u \subseteq \Sigma := \Sigma_1 \cup \Sigma_2$, respectively, and let $E := E_1 \parallel E_2$. Define the transition projections $\bar{P} : T_E \times \Sigma \rightarrow T_{E_1} \bar{Q}_i : T_i \times (\Sigma_i \cap \Sigma_u) \rightarrow T_i$ $(i = 1, 2)$ and let $EFA \ E = \bar{P}(E, \Sigma_u)$. Suppose the set of forbidden locations is $L_f \subseteq L_E$. If for $i = 1, 2$, $\bar{Q}_i$ is an $E_i$-observer and OCC, then $\sup(S(E, L_f)) = \sup(S(E, L_f))∥E$.

Proof: The proof is similar to that of Theorem 2 and Proposition 4 by considering $E_2$ as $E_2∥\cdots∥E_n$.

This property was pointed out by [15], [26] and later in more general form by [23], and Corollary 1 extends it to systems modeled by EFAs.

VI. EXAMPLE - MANUFACTURING WORKCELL

Consider a manufacturing workcell borrowed from [11], consisting of three machines M1, M2, and M3, working on parts stored in two buffers B1 and B2 of size 16 and 8, respectively. To increase the practical usage and complexity of the workcell, two inspection units TU1 and TU2 are added to randomly inspect parts from B1 and B2. Parts which are qualified will be returned to the buffers otherwise will be eliminated. Fig. 2 shows the workcell product flow and Fig. 3 illustrates the EFA components of the system in which the events with exclamation mark are the uncontrollable events and the shaded circles are the marked locations. The domain of variables $b1$ and $b2$ are, respectively, $D_1 = \{0, 1, 2, \ldots, 16\}$ and $D_2 = \{0, 1, 2, \ldots, 8\}$ that indicates the number of parts in the two buffers and their maximum capacity. B1 and B2 initially contain no part, i.e., $D^0 = \{0, 0\}$, and all values are marked $D^m = D_1 \times D_2$. The workcell specifications are as follows: SPEC1: buffers B1 and B2 must not overflow, i.e., a machine must not try to put a part in a buffer when it is full, formally, when $b1 = 16$ or $b2 = 8$. SPEC2: buffers B1 and B2 must not underflow, i.e., a machine must not try to take a part from a buffer when it is empty, formally, when $b1 = 0$ or $b2 = 0$. The corresponding EFAs for SPEC1 and SPEC2 are depicted in Fig. 3(f) and Fig. 3(g), respectively.

To apply the model abstraction using transition projection as mentioned earlier, first we find the local events in the system by checking the conditions in Definition 4 for all the events. The first candidates for the set of local events are

Fig. 2: The manufacturing workcell control flow.

Fig. 3: EFA components of the example

Fig. 4: Abstracted EFA models of TU1 and TU2.
<table>
<thead>
<tr>
<th>TABLE I: Optimal nonblocking supervisory synthesis results of the manufacturing workcell example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Models</td>
</tr>
<tr>
<td>Abtracted Models</td>
</tr>
</tbody>
</table>

\[ \Sigma_i = \{ \text{in}, \text{test}_1, \text{test}_2, \text{!o}_1, \text{!o}_2 \} \]. In this, the events “!o_1” and “!o_2” cannot fulfill the observer property and therefore, are eliminated from the list. Also, the event “in” is found to be inconsistent with OCC conditions thus is removed. Finally, the remaining local events \( \Sigma_i = \{ \text{test}_1, \text{test}_2 \} \) are used to project the EFA components. The projection of TU1 and TU2 are depicted in Fig. 4(a) and Fig. 4(b), respectively, while the rest are the same as the original ones. The optimal nonblocking and controllable guards added to the abstracted system are the same as the original system in [11]. Table I shows the result of optimal nonblocking supervisory synthesis for both original and abstracted systems.

VII. CONCLUSION

In this paper we have extended previous work on model abstraction by natural projection with a modified observer property to include the EFA modeling formalism. Transition projection is introduced as an extension of natural projection for EFAs by directly projecting the conditional transitions. We independently compute the projection of the low-level components without regard to their mutual conflict. Subsequently, to reduce computational complexity, we compute the high-level coordinators based only on abstracted models of the low-level components. Effective and consistent model abstraction is accomplished through transition projections with the observer and OCC properties. A manufacturing workcell example demonstrates the computational effectiveness and practical usage of the proposed approach. A special case of this abstraction, including additional structural reduction, has been applied on a large-scale manufacturing workcell [30], where more than 98% of the computational time and space has been saved.

REFERENCES