Modeling and Control of Discrete Event Systems Using Finite State Machines with Variables and Their Applications in Power Grids

Junhui Zhao[a], Yi-Liang Chen[b], Zhong Chen[a,c], Feng Lin[a,d*], Caisheng Wang[a,e], and Hongwei Zhang[f]

[a] *Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA*
[b] *Northrop Grumman Aerospace System, Bethpage, NY 11714, USA*
[c] *College of Electrical and Information Engineering, Changsha University of Science and Technology, Changsha, Hunan, China*
[d] *School of Electronics and Information Engineering, Tongji University, Shanghai, China*
[e] *Division of Engineering Technology, Wayne State University, Detroit, MI 48202, USA*
[f] *Department of Computer Science, Wayne State University, Detroit, MI 48202, USA*

[*] Corresponding author: Feng Lin, Address: 5050 Anthony Wayne Dr., Detroit, MI 48201 USA; Phone: +1(313)577-3428; Fax: +1(313)577-1101; E-mail: flin@ece.eng.wayne.edu.

Modeling and Control of Discrete Event Systems Using Finite State Machines with Variables and Their Applications in Power Grids

Control theories for discrete event systems modeled as finite state machines have been well developed to address various fundamental control issues. However, finite state machine model has long suffered from the problem of state explosion that renders it unsuitable for some practical applications. In an attempt to mitigate the state explosion problem, we propose an efficient representation that appends finite sets of variables to finite state machines in modeling discrete event systems. We also present the control synthesis techniques for such finite state machines with variables (FSMwV). We first present our notion and means of control under this representation. We next present our algorithms for both offline and online synthesis of safety control policies. We then apply these results to the control of electric power grids.

Keywords: control synthesis; discrete event systems; finite state machines; PHEV; supervisory control; smart grids

1 Introduction

Modeling and control of discrete event systems (DES) have been studied by control engineers and scientists for more than 25 years. During this period, many modeling approaches have been proposed, including most notably automata or finite state machines [1,2], Petri nets [3,4] and their variations such as vector DES [5,6] and event graphs [7], queuing systems [2] and generalized semi-Markov processes [8].

Among these models, finite state machines are the most straightforward for control. In fact, the supervisory control theory [1,2,9,10] based on finite state machines has been well developed as it addresses the fundamental issues in control of DES. As a result, we now have a good understanding of problems such as controllability, observability, coobservability, normality, decentralization, nondeterminism, etc. We believe that an important reason we have gone this far in a relatively short time period is that we adapted a simple model of finite state machines. Because of this, we can focus our attention on and see the essence of the control problem.

However, finite state machine model has long suffered from the problem of state explosion that renders it unsuitable for some practical applications. For example, to model a buffer of *n* capacity using a finite state machine would require at least *n* states. On the other hand, by using an integer variable to describe the content of the buffer, the number of states required can be drastically reduced. Furthermore, in the case that the capacity of the buffer changes, we can simply modify the range of the variable without remodeling the system.

Meanwhile, the traditional supervisory control techniques focus on (passively) maintaining system safety and liveness by the means of disabling some controllable events. It has neglected the possibility of actively enforcing certain events that is widely practiced in the control of real world DES applications. Event enforcement can be quite useful in both “driving” the system toward the given objective (e.g., marked states) and actively maintaining system safety.

To mitigate the problem of state explosion, we propose to employ both finite state machines and sets of variables in modeling discrete event systems. We call our representation Finite State Machines with Variables (FSMwV)[*]. We show that our FSMwV can represent a broader class of discrete event systems with far smaller numbers of discrete states. The definition of our FSMwV is similar to the Extended Finite State Machines (EFSM) described in [11]. However, the EFSM mechanism was developed for the design verification of circuits but not for the modeling of general discrete event systems. Hence, variables in EFSM are mainly for describing the contents of the circuit inputs/outputs rather than for describing system resources and possible time/resource constraints that FSMwV is designed for. Furthermore, neither concepts of system composition nor control synthesis were developed under the EFSM scheme.

Recently, a method using EFSM to implement the supervisory map as an embedded control was developed [12,13]. The method was extended to decentralized control in [14]. EFSM was also used to verify supervisory control properties in [15]. In [16], the authors proposed to transform a set of extended automata into a set of ordinary automata with equivalent behaviour, but no control synthesis methods were discussed. [17] developed the supervisory control for concurrent systems with EFSM modeled subsystems. In [18], a symbolic transition system model was used, which defines the concept of controllability by applying it to the guards of symbolic transitions, instead of to the events. Neither

[*] Formerly, it is called Finite State Machines with Parameters (FSMwP) [19].

[18] nor [17] investigated the synthesis of optimal (least restrictive) controllers. They also did not consider enforceable events.

In this paper, our focus is on control synthesis using FSMwV. We first extend the scope of the traditional DES control to include both event disablement and event enforcement. We then propose an offline safety control synthesis procedure that takes the advantage of both event disablement and enforcement in order to prevent the controlled system from venturing into the prohibited state space. To address the practical concern of real world implementations, we further present a set of safety control synthesis procedures, based on the limited and/or variable lookahead policies [20,21], that generate the control policies online under the FSMwV modeling framework.

The theoretical results on modeling and control of DES using FSMwV are applied to the safety control of electric power distribution network. DES theories have been explored for applications in power systems [22,23,24,25]. Supervisory control using DES was applied and reported in [22] for line restoration. Hybrid automaton and Petri Nets was used to model power systems for handling inverse problems such as parameter uncertainty and parameter estimation [24]. DES was used in [25] to describe cascading events such as blackouts in power systems. A number of potential power system control problems were discussed in [23]. However, most of the results obtained so far in the area are still preliminary. The relevance and applications of DES to power systems remain not so clear [23]. We model a distribution network by an FSMwV in this paper. We consider both conventional uncontrollable loads and controllable loads (such as plug-in hybrid electric vehicles) by using appropriate variables to avoid possible state explosion. A supervisor is then designed to ensure the network is fully utilized and never overloaded.

The rest of the paper is organized as follows: We present the FSMwV model and its system composition mechanism in Section 2. Some preliminary work on FSMwV was presented in [19]. In Section 3, we describe our notion and means of control and present an offline safety control synthesis algorithm. In Section 4, we present an online synthesis algorithm (and its variations) for safety control policies. In Section 5, we apply the results to the safety control of power distribution network. We conclude the paper in Section 6.

2 Finite State Machines with Variables

In this section, we present the modeling mechanism of finite state machines with variables. First, let us recall that a finite state machine (FSM) is described by a 5-tuple [2]

$$\text{FSM} = (\Sigma, Q, \delta, q_0, Q_m),$$

where Σ is the (finite) event set, Q the (finite) state set, δ: $\Sigma \times Q \rightarrow Q$ the transition function, the q_0 initial state, and Q_m the marked (or final) states.

To introduce variables into an FSM, let $p \in P$ be a vector of variables, where P is some vector space. P can be either finite or infinite. More often, P is over the set of natural numbers. We also introduce guards $g \in G$ that are predicates on the variables p. The transition function δ can be defined as a function from $\Sigma \times Q \times G \times P$ to $Q \times P$ as illustrated in Figure 1. The transition shown is to be interpreted as follows: If at state q, the guard g is true and the event σ occurs, then the next state is q' and the values of variables will be updated to $f(p)$. We denote such a transition by $(q, g \wedge \sigma / p := f(p), q') \in \delta$.

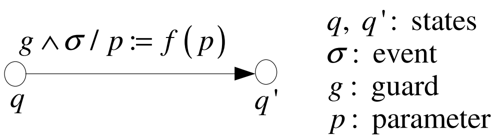


Figure 1. A transition in FSMwV.

If g is absent in the transition, and then the transition takes place when σ occurs. Such a transition is called event transition. If σ is absent, then the transition takes place when g becomes true. Such a transition is called dynamic transition. If p:=$f(p)$ is absent, then no variable is updated during the transition. In summary, a finite state machine with variables can be viewed as a 7-tuple

$$\text{FSMwV} = (\Sigma, Q, \delta, P, G, (q_0, p_0), Q_m),$$

where p_o is the initial value of variables at the initial state q_o.

Without much difficulty, we can regard finite state machines with variables as a special type of Hybrid Machines (HMs) introduced in [26]. In particular, an FSMwV has no continuous dynamics (i.e. $\dot{p}=0$ at any state). The only way to change values of variables is by updating (or re-initialization).

Similar to FSMs and HMs, we can define parallel composition of several FSMwVs running in parallel to form a composite finite state machine with variables (CFSMwV)

$$\text{CFSMwV} = \text{FSMwV}_1 \parallel \text{FSMwV}_2 \parallel \cdots \parallel \text{FSMwV}_n.$$

To define a CFSMwV, we assume that any variable can only be updated by at most one FSMwV. Variables that are not updated by any of the FSMwVs are updated by the unmodeled environment. In general, a variable updated by one FSMwV can be used in another FSMwV. That is, a guard in one FSMwV may depend on a variable updated by another FSMwV.

To simplify the following definition of parallel composition, we assume that, without loss of generality, all transitions in an FSMwV have been decomposed into event transitions and dynamic transitions, as this can always be done. Hence,

$$\begin{aligned}\text{CFSMwV} &= \text{FSMwV}_1 \| \ldots \| \text{FSMwV}_n \\ &= \left(\Sigma_1, Q_1, \delta_1, P_1, G_1, (q_{o1}, p_{o1}), Q_{m1}\right) \| \ldots \| \left(\Sigma_n, Q_n, \delta_n, P_n, G_n, (q_{on}, p_{on}), Q_{mn}\right) \\ &= \big(\Sigma_1 \cup \ldots \cup \Sigma_n, Q_1 \times \ldots \times Q_n, \delta_1 \times \ldots \times \delta_n, P_1 \cup \ldots \cup P_n, G_1 \cup \ldots \cup G_n, \\ &\quad (q_{o1}, \ldots, q_{on}, p_{o1}, \ldots, p_{on}), Q_{m1} \times \ldots \times Q_{mn})\big) \\ &= (\Sigma, Q, \delta, P, G, (q_o, p_o), Q_m),\end{aligned}$$

where the transition function $\delta = \delta_1 \times \ldots \times \delta_n$ is defined as illustrated in Figures 2 and 3 for n=2. In the figures, l_i can be either an event ($l_i = \sigma_i$) or a guard ($l_i = g_i$). If $l_1 \neq l_2$, then the situation is illustrated in Figure 2. That is, if the transition l_1 occurs at state (q_1, q_2), then the next state is (q_1', q_2). Variable p_1 is updated to $f_1(p_1)$ while p_2 is not updated. On the other hand, if $l_1 = l_2 = l$, then the situation is illustrated in Figure 3. That is, if the transition l occurs at state (q_1, q_2), then the next state is (q_1', q_2'). Variables p_1 and p_2 are updated to $f_1(p_1)$ and $f_2(p_2)$ respectively.

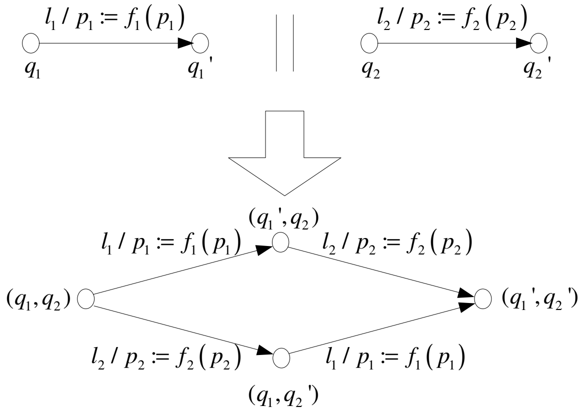


Figure 2. Parallel composition: $l_1 \neq l_2$.

We note that this definition is an extension to that of FSM [2]. Using this parallel composition, we can build large systems from simple components. This procedure can be automated.

To describe the behaviour of an FSMwV, (Σ, Q, δ, P, G, (q_0, p_0), Q_m), we define a run of an FSMwV as a sequence

$$r = (q_o, p_o) \xrightarrow{l_1} (q_1, p_1) \xrightarrow{l_2} (q_2, p_2) \xrightarrow{l_3} (q_3, p_3) \ldots,$$

where l_i is (the label of) the ith transition and (p_i, q_i) is the state and variable values after the ith transition. We denote the set of all possible runs of FSMwV as

$$R(\text{FSMwV}) = \{r\text{: } r \text{ is a run of FSMwV}\}.$$

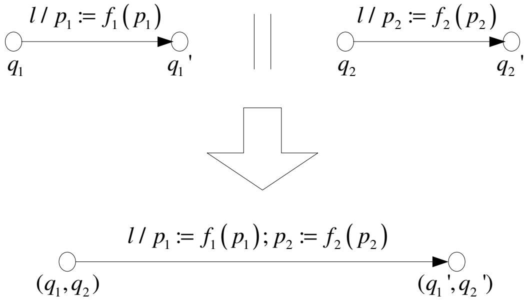


Figure 3. Parallel composition: $l_1 = l_2 = l$.

A trace of a run is the sequence of event transitions in the run

$$s = \sigma_1\sigma_2\sigma_3 \ldots .$$

That is, s is obtained from r by deleting the state information and dynamic transitions.

If an FSMwV is deterministic (which we assume throughout this paper), then a run is uniquely determined by its trace. That is, we can reconstruct a run by looking at its trace and the FSMwV. The set of all traces of an FSMwV is a language denoted by

$$L(\text{FSMwV}) = \{s\text{: } s \text{ is a trace of FSMwV}\}.$$

This language is called the language generated by FSMwV. The language marked by FSMwV is defined as

$$L_m(\text{FSMwV}) = \{s \in L(\text{FSMwV})\text{: the run of } s \text{ ends in a marked state } q \in Q_m\}.$$

Since CFSMwV and FSMwV have the same structure, runs, traces, and languages for CFSMwV are defined similarly.

We often use a legal specification $E \subseteq R(\text{CFSMwV})$ to specify the legal behaviour of the system modeled by a CFSMwV: a run r is legal if and only if it belongs to E. We call this type of specifications dynamic specifications. On the other hand, if the legal behaviour is specified in terms of legal and illegal states, that is, a run r is legal if and only if it does not visit any illegal state, then the specification is called a static specification. It is well known in supervisory control [27] that a dynamic specification can always be translated into a static specification (perhaps at the cost of having more states). Therefore, we will use static specifications in safety controller synthesis.

3 Safety Controller

In this section, we study how to design a safety controller, that is, a controller that guarantees the system will never enter some illegal states. We assume that the system to be controlled is modeled by a CFSMwV:

$$\text{CFSMwV} = (\Sigma, Q, \delta, P, G, (q_0, p_0), Q_m),$$

and the safety requirement is given by a set of illegal states $Q_b \subseteq Q$. Note that the specifications in terms of illegal states are very general and cover a large class of practical situations. For example, we can translate the specification "the variable p shall always be less than a constant c" into an illegal state specification as shown in Figure 4.

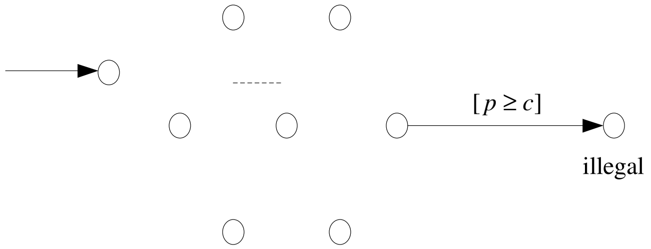


Figure 4. An illegal state specification for $p \geq c$.

The control objective is to make sure that the system never visits any illegal state in Q_b. We assume that there are two control mechanisms that can be used to achieve the control objective.

(1) Disablement: Events in $\Sigma_c \subseteq \Sigma$ can be disabled by a controller. Events $\sigma \in \Sigma_c$ are called controllable events.

(2) Enforcement: Events in $\Sigma_f \subseteq \Sigma$ can be enforced by a controller. Events $\sigma \in \Sigma_f$ are called enforceable events.

We assume that an uncontrollable event cannot be enforced, that is, $(\Sigma - \Sigma_c) \cap \Sigma_f = \phi$, where ϕ denotes the empty set. We also assume that the system is deterministic. That is, any transition in CFSMwV can only lead to one state.

The behavior of the uncontrolled system is described by the set of runs generated by CFSMwV, *R*(CFSMwV). The legal behaviour of the system is described by a subset of runs in *R*(CFSMwV) that does not visit illegal states:

$$E = \{r \in R(\text{CFSMwV}): r \text{ does not visit any illegal states in } Q_b\}.$$

In order to simplify the analysis and synthesis of controllers, we will treat all transitions, including event transitions and dynamic transitions, in a unified manner. To this end, we introduce an artificial uncontrollable event σ_u and extend the event set Σ to include σ_u. To simplify the notation, we will still use Σ to denote the expended event set in the rest of the paper. With σ_u, a dynamic transition $(q, g / p := f(p), q')$ is equivalent to $(q, g \wedge \sigma_u / p := f(p), q')$ for the purpose of controller analysis and synthesis and an event transition $(q, \sigma / p := f(p), q')$ can be viewed as $(q, g \wedge \sigma / p := f(p), q')$ if we let $g = T$.

To investigate the control in a generalized framework, we use generalized control patterns [28] as follows:

$$\Gamma = \{\gamma \subseteq \Sigma : \Sigma - \Sigma_c \subseteq \gamma \vee \gamma \subseteq \Sigma_f\}.$$

This set of control pattern allows two types of control: (1) Disabling some controllable events (that is, those in $\Sigma - \gamma$, if the first disjunction is satisfied); and (2) Enforcing some enforceable events (that is, those in γ, if the second disjunction is satisfied). This is a generalization from pure disablement of standard supervisory control.

Proposition 1: The set of control patterns Γ is closed under union, that is, for all control patterns γ_1, γ_2

$$\gamma_1 \in \Gamma \wedge \gamma_2 \in \Gamma \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma.$$

Proof: Assume that $\gamma_1, \gamma_2 \in \Gamma$, that is $\Sigma - \Sigma_c \subseteq \gamma_1 \vee \gamma_1 \subseteq \Sigma_f$ and $\Sigma - \Sigma_c \subseteq \gamma_2 \vee \gamma_2 \subseteq \Sigma_f$. Consider four possible cases.

(1) $\Sigma - \Sigma_c \subseteq \gamma_1 \wedge \Sigma - \Sigma_c \subseteq \gamma_2 \Rightarrow \Sigma - \Sigma_c \subseteq \gamma_1 \cup \gamma_2 \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma$.

(2) $\Sigma - \Sigma_c \subseteq \gamma_1 \wedge \gamma_2 \subseteq \Sigma_f \Rightarrow \Sigma - \Sigma_c \subseteq \gamma_1 \cup \gamma_2 \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma$.

(3) $\gamma_1 \subseteq \Sigma_f \wedge \Sigma - \Sigma_c \subseteq \gamma_2 \Rightarrow \Sigma - \Sigma_c \subseteq \gamma_1 \cup \gamma_2 \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma$.

(4) $\gamma_1 \subseteq \Sigma_f \wedge \gamma_2 \subseteq \Sigma_f \Rightarrow \gamma_1 \cup \gamma_2 \subseteq \Sigma_f \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma$.

Therefore, Γ is closed under union.

Q.E.D

The controller is defined as a mapping from the set of runs *R*(CFSMwV) to the set of control pattern Γ:

$$\psi: R(\text{CFSMwV}) \rightarrow \Gamma.$$

The behavior of the controlled system, denoted by R(CFSMwV, ψ), is given as follows:

(1) $\varepsilon \in R$(CFSMwV, ψ), where ε denotes the empty trace (empty run);

(2) Then inductively,

$$(\forall r = (q_o, p_o) \xrightarrow{l_1} (q_1, p_1) ... \xrightarrow{l_n} (q_n, p_n) \in R(\text{CFSMwV}, \psi))(\forall l_{n+1} = g \wedge \sigma)$$

$$r \xrightarrow{l_{n+1}} (q_{n+1}, p_{n+1}) \in R(\text{CFSMwV}, \psi)$$

$$\Leftrightarrow r \xrightarrow{l_{n+1}} (q_{n+1}, p_{n+1}) \in R(\text{CFSMwV}) \wedge \sigma \in \psi(r).$$

In other words, a transition $(q_n, p_n) \xrightarrow{l_{n+1}} (q_{n+1}, p_{n+1})$ is possible in the closed-loop systems if and only if it is possible in the open-loop system (hence the guard is true) and the event is enabled or enforced. Our goal is to synthesize a controller such that R(CFSMwV, ψ) = E if possible. To find a necessary and sufficient condition for the existence of a controller, controllability is generalized as follows.

Definition 1: A set of possible runs $K \subseteq R(\text{CFSMwV})$ is *controllable* with respect to R(CFSMwV) and Γ if

$$(\forall r \in \overline{K})(\exists \gamma \in \Gamma)(\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r)) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma,$$

where $\overline{K}$ denotes the prefix-closure of K, $\Sigma_{R(\text{CFSMwV})}(r) = \{\sigma \in \Sigma : r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV})\}$, and $\Sigma_K(r) = \{\sigma \in \Sigma : r \xrightarrow{g \wedge \sigma} (q, p) \in \overline{K}\}$.

The following theorem says that controllability is a necessary and sufficient condition for the existence of a controller.

Theorem 1: Given a system CFSMwV and a specification $K \subseteq R$(CFSMwV), a controller ψ exists such that R(CFSMwV, ψ) = K if and only if K is closed and controllable.

Proof: (ONLY IF) Let ψ be a controller such that R(CFSMwV, ψ) = K. Clearly K is closed. We show that K is controllable as follows:

$$\begin{aligned}
& K = R(\text{CFSMwV}, \psi) \\
& \Rightarrow (\forall r \in K)\Sigma_K(r) = \Sigma_{R(\text{CFSMwV}, \psi)}(r) \\
& \Rightarrow (\forall r \in K)\Sigma_K(r) = \Sigma_{R(\text{CFSMwV})}(r) \cap \psi(r) \\
& \Rightarrow (\forall r \in K)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) = \Sigma_{R(\text{CFSMwV})}(r) - \Sigma_{R(\text{CFSMwV})}(r) \cap \psi(r) \\
& \Rightarrow (\forall r \in K)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) = \Sigma_{R(\text{CFSMw V})}(r) - \psi(r) \\
& \Rightarrow (\forall r \in K)(\exists \gamma = \psi(r) \in \Gamma)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma.
\end{aligned}$$

Therefore, K is controllable.
(IF) Since K is closed and controllable,

$$(\forall r \in K)(\exists \gamma \in \Gamma)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma.$$

Let us define the controller ψ: R(CFSMwV)→Γ as follows: For $r \in K$, let $\psi(r)$ be the largest γsatisfies the above equation. By Proposition 1, the largest γexists. For $r \in R$(CFSMwV)–K, let $\psi(r)=\Sigma-\Sigma_c$. We can prove $r \in R$(CFSMwV, ψ) ⇔ $r \in K$ by induction on the length $|r|$ of r as follows:

Base: Since K is closed, $\varepsilon \in K$. By the definition of R(CFSMwV, ψ), $\varepsilon \in R$(CFSMwV, ψ). Therefore,

$$\varepsilon \in R(\text{CFSMwV}, \psi) \Leftrightarrow \varepsilon \in K.$$

Induction Hypothesis (IH): Assume that for all r such that the length $|r| \leq d$, and d is a positive integer.

$$r \in R(\text{CFSMwV}, \psi) \Leftrightarrow r \in K.$$

Induction Step: We need to prove that for all $r \xrightarrow{g \wedge \sigma} (q, p)$ such that $| r \xrightarrow{g \wedge \sigma} (q, p) |= d + 1$,

$$r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV}, \psi) \Leftrightarrow r \xrightarrow{g \wedge \sigma} (q, p) \in K \ .$$

Indeed,

$$\begin{aligned}
& r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV}, \psi) \\
\Leftrightarrow\ & r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV}) \wedge \sigma \in \psi(r) \wedge r \in R(\text{CFSMwV}, \psi) \\
\Leftrightarrow\ & r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV}) \wedge \sigma \in \psi(r) \wedge r \in K \qquad\qquad \text{By IH} \\
\Leftrightarrow\ & \sigma \in \Sigma_{R(\text{CFSMwV})}(r) \wedge \sigma \in \psi(r) \wedge r \in K \\
\Leftrightarrow\ & \sigma \in \Sigma_{R(\text{CFSMwV})}(r) \wedge \sigma \notin \Sigma_{R(\text{CFSMwV})}(r) - \psi(r) \wedge r \in K \\
\Leftrightarrow\ & \sigma \in \Sigma_{R(\text{CFSMwV})}(r) \wedge \sigma \notin \Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) \wedge r \in K \\
\Leftrightarrow\ & \sigma \in \Sigma_{R(\text{CFSMwV})}(r) \wedge \sigma \in \Sigma_K(r) \wedge r \in K \\
\Leftrightarrow\ & r \xrightarrow{g \wedge \sigma} (q, p) \in R(\text{CFSMwV}) \wedge r \xrightarrow{g \wedge \sigma} (q, p) \in K \wedge r \in K \\
\Leftrightarrow\ & r \xrightarrow{g \wedge \sigma} (q, p) \in K.
\end{aligned}$$

This proves the theorem.

Q.E.D

If the specification *E* is not controllable, we will find the largest subset of *E* that is controllable. In fact, we can show that the supremal controllable subset of *E* always exists. To this end, let us define the set of all controllable subset of *E* as

$$C(E)=\{K \subseteq E\text{: } K \text{ is closed and controllable with respect to } R(\text{CFSMwV}) \text{ and } \Gamma\}.$$

Then we have the following theorem.

Theorem 2: If K_1, $K_2 \in C(E)$, then $K_1 \cup K_2 \in C(E)$. Therefore, the supremal controllable subset of *E*, denoted by sup$C(E)$, exists.

Proof: Let K_1, $K_2 \in C(E)$ and $K = K_1 \cup K_2$. Obviously *K* is closed. Since both K_1 and K_2 are controllable, we have

$$\begin{aligned}
&(\forall r \in K_1)(\exists \gamma_1 \in \Gamma)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_{K_1}(r) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma_1 \,, \\
&(\forall r \in K_2)(\exists \gamma_2 \in \Gamma)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_{K_2}(r) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma_2 \,.
\end{aligned}$$

To prove *K* is controllable, we need to show

$$(\forall r \in K)(\exists \gamma \in \Gamma)\Sigma_{R(\text{CFSMwV})}(r) - \Sigma_K(r) = \Sigma_{R(\text{CFSMwV})}(r) - \gamma \,.$$

Since $K = K_1 \cup K_2$, there are three possible cases.

(1) $r \in K_1$ and $r \in K_2$: In this case, let $\gamma = \gamma_1 \cup \gamma_2$. By Proposition 1, $\gamma_1 \in \Gamma \wedge \gamma_2 \in \Gamma \Rightarrow \gamma_1 \cup \gamma_2 \in \Gamma$. Also $\Sigma_K(r)$=$\Sigma_{K_1 \cup K_2}(r)$=$\Sigma_{K_1}(r) \cup \Sigma_{K_2}(r)$. Therefore,

$$\begin{aligned}
&\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_K(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_1\cup K_2}(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-(\Sigma_{K_1}(r)\cup\Sigma_{K_2}(r))\\
&=(\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_1}(r))\cap(\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_2}(r))\\
&=(\Sigma_{R(\text{CFSMwV})}(r)-\gamma_1)\cap(\Sigma_{R(\text{CFSMwV})}(r)-\gamma_2)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-(\gamma_1\cup\gamma_2)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\gamma.
\end{aligned}$$

(2) $r \in K_1$ and $r \notin K_2$: In this case, let $\gamma = \gamma_1 \in \Gamma$. Then,

$$\begin{aligned}
&\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_K(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_1\cup K_2}(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-(\Sigma_{K_1}(r)\cup\Sigma_{K_2}(r))\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_1}(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\gamma_1\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\gamma.
\end{aligned}$$

(3) $r \notin K_1$ and $r \in K_2$: In this case, let $\gamma = \gamma_2 \in \Gamma$. Then,

$$\begin{aligned}
&\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_K(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_1\cup K_2}(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-(\Sigma_{K_1}(r)\cup\Sigma_{K_2}(r))\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_{K_2}(r)\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\gamma_2\\
&=\Sigma_{R(\text{CFSMwV})}(r)-\gamma.
\end{aligned}$$

So, in any case,

$$(\forall r\in K)(\exists\gamma\in\Gamma)\Sigma_{R(\text{CFSMwV})}(r)-\Sigma_K(r)=\Sigma_{R(\text{CFSMwV})}(r)-\gamma\,.$$

Q.E.D

By this result, we can find the least restrictive safety controller that ensures the closed-loop system will never visit illegal states. Our strategy to synthesize the least restrictive safety controller is as follows: Initially, the system can be in any legal state of the system. However, the system may move to an illegal state via some transitions. So it is important to study transitions on the boundary (from a legal state to an illegal state). If a transition is associated with a controllable event (i.e., transition $(q, g \wedge \sigma / p := f(p), q')$ with $\sigma \in \Sigma_c$) , then the transition can be disabled and we do not need to worry about it. On the other hand, if a transition is associated with an uncontrollable event, then we must prevent it from occurring by either making sure that its guard is not true or pre-empting the transition with an enforceable event if possible. This implies that we must strengthen (or tighten) the conditions under which the system can stay in legal states. We call these conditions safety conditions. We use I_q to denote the safety condition for state q. The key to synthesizing the least restrictive safety controller is to update I_q iteratively so that after the procedure converges, the transitions on the boundary are either disabled or pre-empted. To do this formally, let us denote the number of iterations by k. Initially, we let safety condition $I_q(0)=T$ for all legal states $q \notin Q_b$ and $I_q(0) = F$ for all illegal states $q \in Q_b$. For a legal state $q \notin Q_b$, its safety condition $I_q(k)$ is updated as:

$$I_q(k+1) = I_q(k) \wedge (\neg(\bigvee_{(q, g\wedge\sigma / p:=f(p), q')\in\delta\wedge\sigma\notin\Sigma_c} (g \wedge \neg I_{q'}(k)|_{p:=f(p)}))$$

$$\vee (\bigvee_{(q, g\wedge\sigma / p:=f(p), q')\in\delta\wedge\sigma\in\Sigma_f} (g \wedge I_{q'}(k)|_{p:=f(p)}))).$$

This formula implies that the new safety condition will be true only if the old safety condition $I_q(k)$ is true and either there are no uncontrollable transitions leading to illegal states, $\neg(\bigvee_{(q, g\wedge\sigma / p:=f(p), q')\in\delta\wedge\sigma\notin\Sigma_c} (g \wedge \neg I_{q'}(k)|_{p:=f(p)}))$, or there are some enforceable transitions leading to legal states, $(\bigvee_{(q, g\wedge\sigma / p:=f(p), q')\in\delta\wedge\sigma\in\Sigma_f} (g \wedge I_{q'}(k)|_{p:=f(p)}))$.

Since Q is finite by definition, whether the above iteration will converge (terminate) or not depends on the set P. If P is finite, then the iteration is guaranteed to converge. If P is infinite, then the iteration may or may not converge. In the example below, we show that in some cases even if P is infinite, the iteration still converges.

When the iteration converges, we have $I_q(k+1) = I_q(k)$. Denote $I^*_q = I_q(k+1) = I_q(k)$. We can obtain the controller ψ:R(CFSMwV)→Γ as follows: Let $r \in R$(CFSMwV) be a run ending at (q, p). Then

$$\psi(r) = \begin{cases} \{\sigma\in\Sigma : (q, g\wedge\sigma / p := f(p), q')\in\delta\wedge\neg(g\wedge\neg I^*_{q'}|_{p:=f(p)})\}\cup(\Sigma-\Sigma_c) \\ \qquad\qquad \text{if } \neg(\bigvee_{(q, g\wedge\sigma / p:=f(p), q')\in\delta\wedge\sigma\notin\Sigma_c} (g\wedge\neg I^*_{q'}|_{p:=f(p)})) \\ \{\sigma\in\Sigma_f : (q, g\wedge\sigma / p := f(p), q')\in\delta\wedge(g\wedge I^*_{q'}|_{p:=f(p)})\} \\ \qquad\qquad \text{otherwise.} \end{cases}$$

Clearly $\psi(r) \in \Gamma$ and under this control, the closed-loop system will satisfies safety condition I^*_q for all legal state $q \notin Q_b$. We show that ψ: R(CFSMwV)→Γ is indeed the controller we want.

Theorem 3: After the iteration converges, the controller ψ: R(CFSMwV)→Γ designed above generates the supremal controllable subset sup$C(E)$. In other words,

$$R(\text{CFSMwV}, \psi) = \sup C(E).$$

Proof: We need to prove (1) R(CFSMwV, ψ) is controllable; (2) R(CFSMwV, ψ) $\subseteq E$; and (3) for all other subset $K \subseteq R$(CFSMwV) such that K is controllable and $K \subseteq E$, $K \subseteq L$(CFSMwV, γ).

(1) R(CFSMwV, ψ) is controllable:
R(CFSMwV, ψ) is generated by a controller. By Theorem 1, it is controllable.

(2) R(CFSMwV, ψ) $\subseteq E$:
During the iteration, all safety conditions are strengthened, that is, $I^*_q \Rightarrow I_q$ for all legal state $q \notin Q_b$. Therefore, R(CFSMwV, ψ) $\subseteq E$.

(3) For all other subset $K \subseteq R$(CFSMwV) such that K is controllable and $K \subseteq E$, $K \subseteq L$(CFSMwV,γ):
During the iteration, a safety condition is strengthened only if not doing so will result in violation of specification E. Hence, no other controller can generate a larger subset than L(CFSMwV,γ) without violating E. Since K is controllable, by Theorem 1, it can be generated by a controller. Therefore, K is controllable and $K \subseteq E$ imply $K \subseteq L$(CFSMwV,γ).

Q.E.D

Note that we assume that in the controlled system, the transitions enforced by the controller will occur before the occurrence of any uncontrollable transition. This assumption is reasonable because we do not consider time in the FSMwV model. If time is of importance, then we shall use hybrid machine model of [29] rather than FSMwV model. Let us now illustrate the above results by an example.

Example 1: Consider the system described by the CFSMwV in Figure 5. The CFSMwV has three events α, β, η and one variable $p \in P$, where P is the set of natural numbers. The illegal state is $Q_b = \{6\}$ (shaded in the figure). The controllable events are $\Sigma_c = \{\beta, \eta\}$. The enforceable event is $\Sigma_f = \{\eta\}$. Our goal is to synthesize a safety controller to ensure that the system will never enter the illegal state.

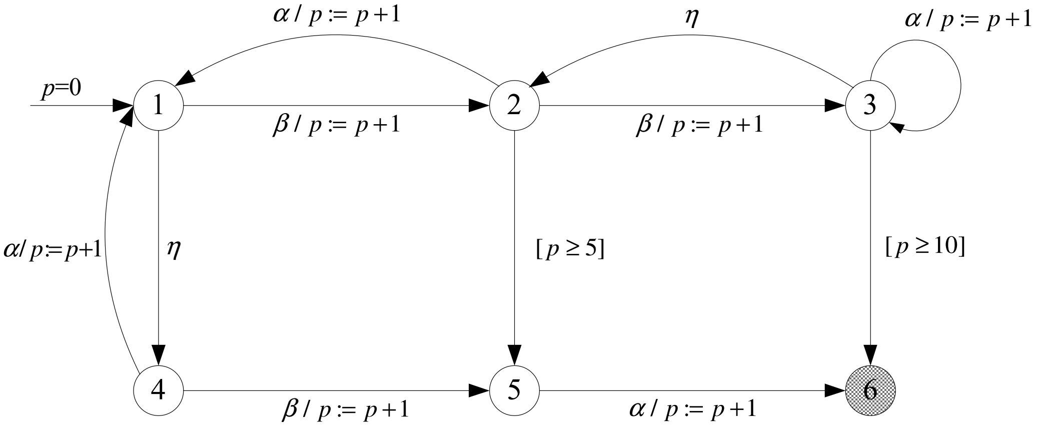


Figure 5. CFSMwV for Example 1.

The results of the iteration process to calculate I_q at different states is given in Table 1 and shown in Figure 6.

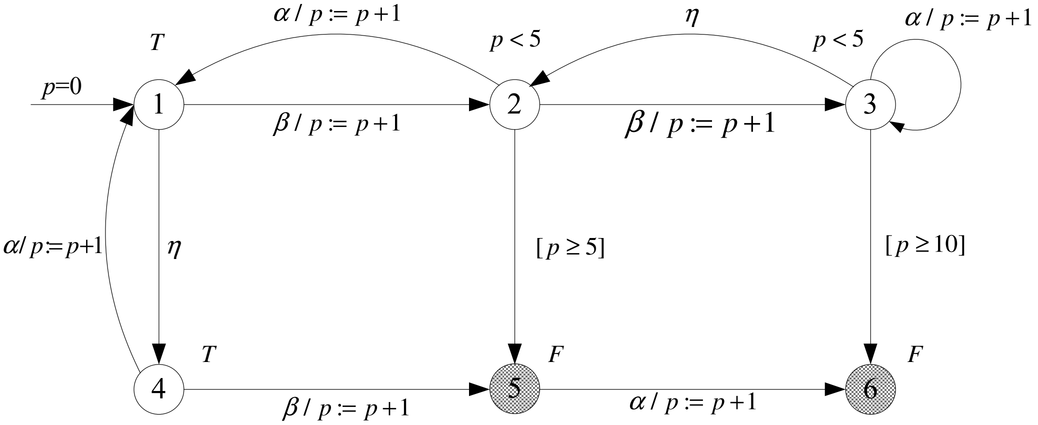


Figure 6. Resulting CFSMwV after the iteration converges.

The control is given as: at state 1 and 2, the controller will disable β if $p \geq 4$; at state 3, the controller will enforce η if $p \geq 4$; and at state 4, the controller always disables β.

Note that the controlled system can loop between states 1 and 4 infinitely many times. Hence, the value of p can increase unboundedly. This example shows that even if P is infinite (the set of natural numbers), the iteration still converges.

Table 1. Calculation of I_q at six states

State / k	1	2	3	4	5	6
0	T	T	T	T	T	F
1	T	T	T	T	T $\wedge$ {$\neg$(T$\wedge\neg$F)} = F	F
2	T	T$\wedge${$\neg$[(T$\wedge\neg$T)$\vee$($p \geq 5\wedge\neg$F)]} = p<5	T$\wedge${$\neg$[(T$\wedge\neg$T)$\vee$($p \geq 10\wedge\neg$F)]$\vee$(T$\wedge$T]} = T	T	F	F
3	T	p<5$\wedge${$\neg$[(T$\wedge\neg$T)$\vee$($p \geq 5\wedge\neg$F)]} = p<5 **I_2*= I_2(2) = I_2(3), stop!**	T$\wedge${$\neg$[(T$\wedge\neg$T)$\vee$($p \geq 10\wedge\neg$F)]$\vee$(T$\wedge$p<5)} = p<10	T	F	F

…	…	…	…	…	…	…
8	T	$p < 5$	$p<5$	T	F	F
9	T	$p < 5$	$p<5$ **I_3*= I_3(9)= I_3 (8), stop!**	T	F	F

4 Online Safety Controller

As it has been demonstrated in standard supervisory control theory, online synthesis of safety controllers has advantages in various applications. If the system to be controlled is large and complex, then offline control synthesis may not be feasible, because it tries to compute the control actions for all possible states and values of variables. Therefore, for large and complex systems, online synthesis is a good alternative because online synthesis only tries to compute the control action for the current state and the current values of variables. Furthermore, online synthesis can be used even if the system to be controlled is time-varying, while offline synthesis cannot be used for time-varying systems. In this section, we will discuss online synthesis of safety controllers using FSMwV model.

To design a safety controller online, we can use either limited lookahead policies or variable lookahead policies [20,21]. In both cases, a forward looking tree representing all possible future behaviour from the current state is constructed. Since the variable values at the current state are known (under our assumption of full observation), all guards can be evaluated. If a guard is true, the transition will be included in the tree; otherwise, the transition (and its continuation) will be discarded in the tree.

After the tree is constructed, the online control synthesis is similar to that of offline. It is actually simpler because of the following two reasons: (1) there are no loops in the tree structure; and (2) guards of all transitions have been evaluated as either true or false. Transitions with false guards are discarded. As before, dynamic transitions with true guards can be treated as same as uncontrollable event transitions by introducing a fictitious new uncontrollable event σ_u .

Since the offline synthesis algorithm has been discussed in the previous section, the key to controller synthesis is to construct the forward looking tree. This is the focus of this section.

During the tree construction, after evaluating guards, transitions of various types are replaced as follows:

(1) $q \xrightarrow{T\wedge\sigma} q'$ replace by $q \xrightarrow{\sigma} q'$
(2) $q \xrightarrow{F\wedge\sigma} q'$ discarded
(3) $q \xrightarrow{T} q'$ replace by $q \xrightarrow{\sigma_u} q'$
(4) $q \xrightarrow{F} q'$ discarded

For limited lookahead policies, the tree construction ends after N steps. The legality of the states at the boundary is determined by the attitude used. If the conservative attitude is used, then all the states at the boundary are considered illegal. This guarantees that the resulting control policy is safe. However, conservative attitude may result in a smaller (that is, more restrictive) control policy or even an empty control policy, which means that the controller will have an error. On the other hand, if the optimistic attitude is used, then all the states at the boundary, except those belonging to Q_b, are considered legal. This attitude will result in a more flexible control policy. However, it may also lead to an unrecoverable error, as it may be too late for an optimistic controller to prevent some illegal behaviour when it sees illegal states.

For variable lookahead policies, the tree construction will continue until some termination conditions are satisfied. We use the following three termination conditions:

(1) A branch terminates at state q if q is an illegal state;
(2) A branch terminates at state q if there is no forcible transition leaving q to a legal state but there is an uncontrollable transition leaving q to an illegal state. In this case, state q is illegal;
(3) A branch terminates at state q if all the transitions leaving q are controllable. In this case, state q is legal regardless of the legality of the following states.

If the tree construction for variable lookahead policies terminates, that is, if every branch ends with one of the three termination conditions satisfied, then the variable lookahead policy obtained is guaranteed to be safe and least restrictive. In other words, it will achieve the same performance as the controller synthesized offline.

Unlike limited lookahead policies, the tree construction for variable lookahead policies may not terminate. In such cases, we can combine limited lookahead policies with variable lookahead policies. In other words, we construct the tree as in variable lookahead policies until it reaches the N-step boundary. We then use either conservative or optimistic attitude for the boundary state as in limited lookahead policies.

Example 2: Let us now demonstrate the online synthesis of the safety controller for the same system as in Example 1. We consider the initial state with *p=0*. The tree with *N=3* is constructed as shown in Figure 7. The shaded state is illegal.

If the conservative attitude is used, then all states at the bottom layer are considered illegal. By applying the synthesis algorithm, the illegal states are "propagated" upward as shown in Figure 8. The control action at the root (i.e. at the initial state) is to enable β, η and enforce nothing.

If the optimistic attitude is used, then all states at the bottom layer, except the left most one, are considered legal. The synthesis algorithm finds bad states as shown in Figure 9. The resulting control action at the initial state is the same as for the conservative attitude.

If variable lookahead policy is used, then some branches will terminate early as shown in Figure 10. This control synthesis results in the same control action at the initial state.

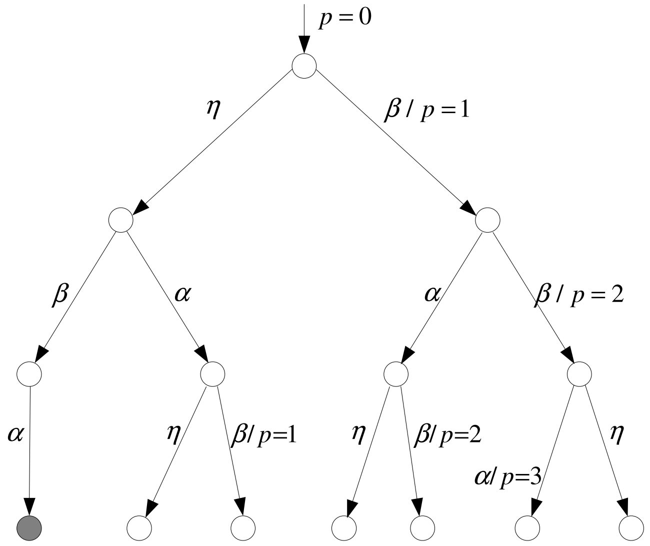


Figure 7. Online expansion of the CFSMwV in Figure 5, where $N = 3$.

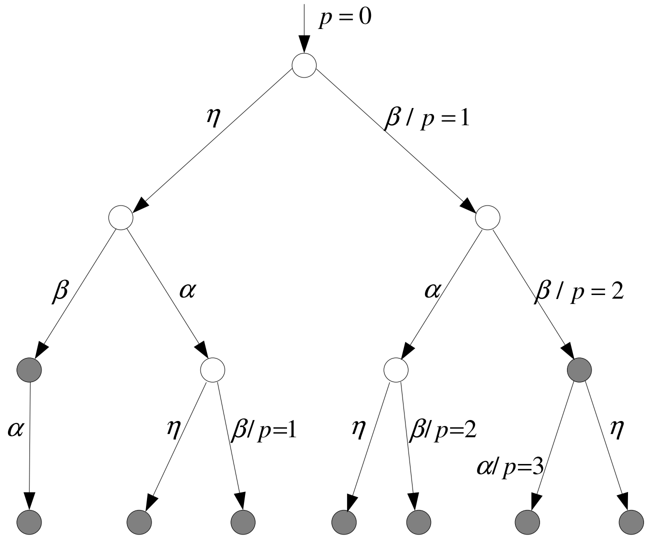


Figure 8. Online control synthesis in Example 2: with conservative attitude.

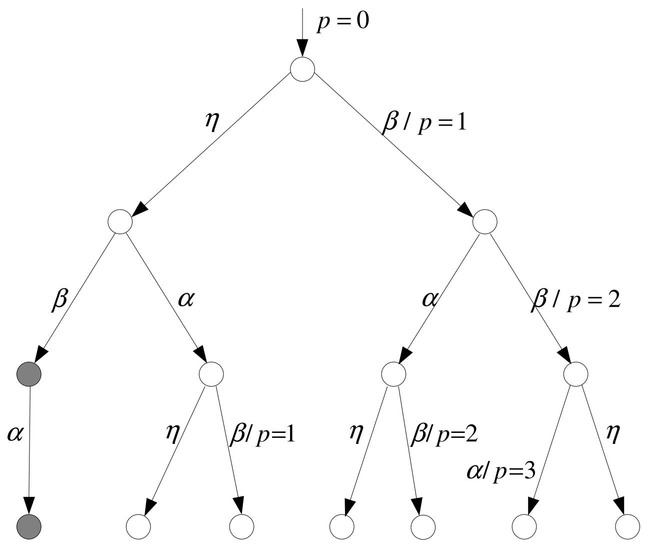


Figure 9. Online control synthesis in Example 2: with optimistic attitude.

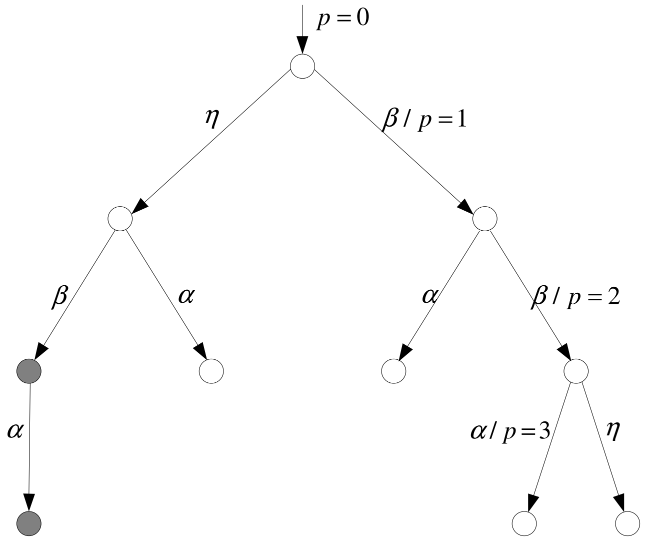


Figure 10. Online control synthesis in Example 2: with variable lookahead.

5 Applications to Power Grids

In this section, we apply the results obtained in the previous section to power grids that needs to accommodate more and more use of PHEVs. This is because transportation electrification is viewed as one of the most viable ways to reduce CO_2 emissions and the gasoline dependency. It is projected that the cumulative sales of electrical vehicles (EVs) and PHEVs will reach 16 million by 2030 [30]. The increasing number of PHEVs will post new challenges to the existing power grid, as they will become a large load to the grid [31]. Clearly uncontrolled charging of a large number of PHEVs will be a big burden to the grid. Adding this load to the conventional residential and industry loads may cause the power grid to be overloaded and hence negatively impacts the grid. Solving this problem by building new generation stations is neither economic nor environmental friendly. On the other hand, when to charge a PHEV is not often time critical. So the charging of PHEVs can be controlled optimally under the constraints of generation and transmission capacity of the existing power grid [32]. In the rest of the paper, we will use FSMwV to model a small distribution network and use supervisory control to control the charging of PHEVs.

5.1 *Distribution Networks*

A distribution network connects the output terminals of a distribution substation to the input terminals of customer loads. Let us consider a typical distribution network shown in Figure 11. We assume that there are N nodes (or buses) in the distribution network. We consider radial distribution networks in this paper. Interconnected distributed networks can also be considered, but not discussed in this paper. For each node i, all the conventional local loads are lumped together and denoted as $p_{i,i}$. For instance, all the local load at *Node 2* is denoted as $p_{2,2}$. All the power lines including transformers connected to *Node i* should not be overloaded. For example, for the local loads connected to *Node 2*, $p_{1,2}$, $p_{2,3}$ and $p_{2,2}$ all should be within their corresponding limits $p_{1,2,m}$, $p_{2,3,m}$ and $p_{2,2,m}$. We call $p_{1,2}$ the incoming power to *Node 2*, at the same time, $p_{2,3}$ and $p_{2,4}$ are called the outgoing powers. At each node, there is a power meter to measure the power of each line connected to the node.

The power loss of the distribution network is neglected. We assume that if the power of a power line is 10% over its limit, the circuit breaker (CB) will trip to protect the line and other devices. This constraint can be readily changed to any actual protection setting in a distribution network. For the purpose of simplification, only PHEVs are considered as controllable loads. The control target is to avoid the over loading type of tripping while satisfying all the load demands as much as possible. Therefore the only safety criterion considered now is the power limit of each node in the distribution network. Since the incoming powers and the outgoing powers are the summation of the local loads, the illegal condition can also be considered as the overload of every local load power line.

A PHEV load is assume to be $n_i \times m$, where n_i is the number of PHEVs being charged at the node i and m is the power consumed by each PHEV at the unit of kilowatts (kW). Three scenarios were proposed in [33] to charge the PHEVs and one of them, $m=6kW$, is used in this paper. All local loads are calculated as conventional loads plus the PHEV load, that is, $p_{i,i}+6n_i$. For instance, the local loads at *Node 2* is $p_{2,2}+6n_2$. The control must ensure that all local loads connected to all nodes do not exceed their limits. For example, for the local loads connected to *Node 2*, $p_{2,2}+6n_2$ must be within its corresponding limit $p_{2,2,m}$.

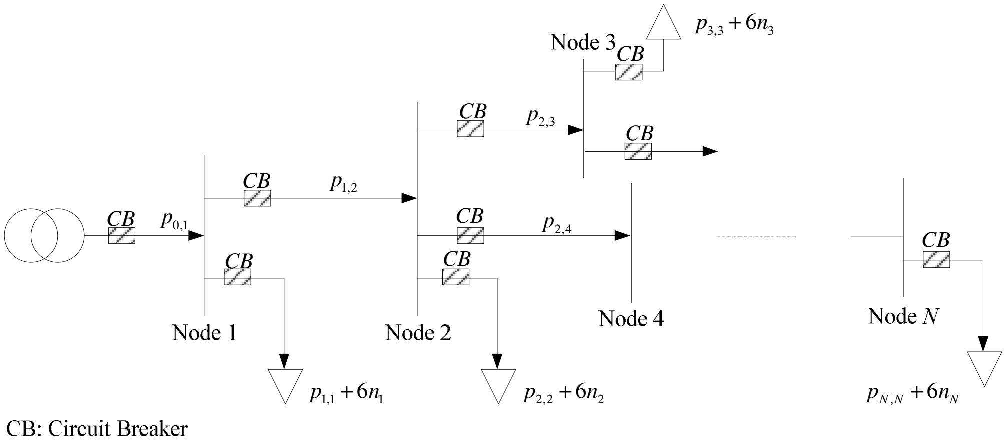
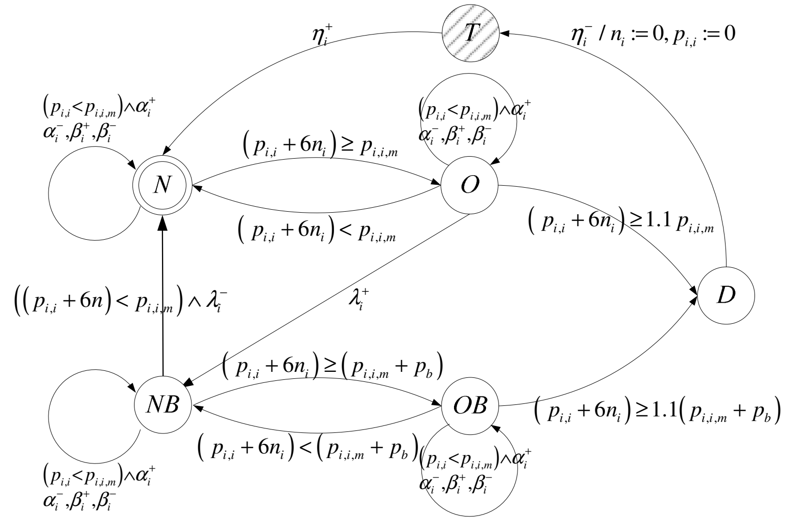


Figure 11. A distribution network with N nodes.

5.2 *Model*

We use FSMwV to model the distribution network. The model of the local load FSMwV$_{i,i}$ is shown in Figure 12. The states set $Q_{i,i}$ contains six states representing load level: the marked state N is for $0\leq(p_{i,i}+6n_i)<p_{i,i,m}$; O is for $p_{i,i,m}\leq(p_{i,i}+6n_i)<1.1p_{i,i,m}$; sometimes the backup power source is used to handle emergent situation, and *NB* is used to denote the state that the backup power source is used and $p_{i,i,m}\leq(p_{i,i}+6n_i)<(p_{i,i,m}+p_b)$, where p_b denotes the capacity of the backup power source; *OB* is for $(p_{i,i,m}+p_b)\leq(p_{i,i}+6n_i)<1.1(p_{i,i,m}+p_b)$; D denotes for the dangerous state and at D the circuit breaker will be tripped to protect the power line thereby moving the system to the illegal state T. Six dynamic transitions are defined correspondingly as: $N\rightarrow O$ when $(p_{i,i}+6n_i)\geq p_{i,i,m}$; $O\rightarrow N$ when $(p_{i,i}+6n_i)<p_{i,i,m}$; $NB\rightarrow OB$ when $(p_{i,i}+6n_i)\geq(p_{i,i,m}+p_b)$; $OB\rightarrow NB$ when $(p_{i,i}+6n_i)<(p_{i,i,m}+p_b)$; $O\rightarrow D$ when $(p_{i,i}+6n_i)\geq 1.1p_{i,i,m}$; and $OB\rightarrow D$ when $(p_{i,i}+6n_i)>1.1(p_{i,i,m}+p_b)$.

Eight events in $\Sigma_{i,i}$ are defined as follows: α_i^+ is for "increase the conventional load"; α_i^- is for "decrease the conventional load"; β_i^+ is for "add one PHEV"; β_i^- is for "remove one PHEV"; λ_i^+ is for "add the backup power source"; λ_i^- is for "remove the backup power source"; η_i^- is for "trip the circuit switch" and η_i^+ is for "restore the power line". Two variables, the conventional loads $p_{i,i}$ and number of PHEVs being charged n_i, will be updated with the occurrence of corresponding events as: α_i^+ with $p_{i,i} := p_{i,i} + 1kW$; α_i^- with $p_{i,i} := p_{i,i} - 1kW$; β_i^+ with $n_i:=n_i+1$; β_i^- with $n_i:=n_i-1$; η_i^- with $n_i:=0$ and $p_{i,i} := 0$. We assume that charging PHEV can be controlled (disabled). Therefore, the controllable event set is $\Sigma_c = \{\beta_i^+\}$. We assume that the events in $\Sigma_f = \{\lambda_i^+, \lambda_i^-, \eta_i^+\}$ are enforceable. As for the event β_i^-, we will consider two scenarios, one is uncontrollable and unenforceable (cannot unplug a PHEV) and the other is enforceable (can unplug a PHEV). We will discuss these two scenarios separately and compare their effects in the control.



T
η_i^+
$\eta_i^- / n_i := 0, p_{i,i} := 0$
$(p_{i,i}<p_{i,i,m})\wedge\alpha_i^+$
$\alpha_i^-, \beta_i^+, \beta_i^-$
N
$(p_{i,i}+6n_i) \geq p_{i,i,m}$
$(p_{i,i}<p_{i,i,m})\wedge\alpha_i^+$
$\alpha_i^-, \beta_i^+, \beta_i^-$
O
$(p_{i,i}+6n_i) < p_{i,i,m}$
$(p_{i,i}+6n_i) \geq 1.1\, p_{i,i,m}$
$((p_{i,i}+6n) < p_{i,i,m}) \wedge \lambda_i^-$
λ_i^+
D
$(p_{i,i}+6n_i) \geq (p_{i,i,m}+p_b)$
NB
OB
$(p_{i,i}+6n_i) \geq 1.1(p_{i,i,m}+p_b)$
$(p_{i,i}+6n_i) < (p_{i,i,m}+p_b)$
$(p_{i,i}<p_{i,i,m})\wedge\alpha_i^+$
$\alpha_i^-, \beta_i^+, \beta_i^-$
$(p_{i,i}<p_{i,i,m})\wedge\alpha_i^+$
$\alpha_i^-, \beta_i^+, \beta_i^-$

Figure 12. FSMwV model for local load at node *i*.

Three assumptions for the FSMwV model of local loads are made as follows: (1) The occurrence of α_i^+ has a guard $p_{i,i}<p_{i,i,m}$ since the conventional local loads normally cannot exceed the limit; (2) The occurrence of λ_i^- has a guard $(p_{i,i}+6n_i)<p_{i,i,m}$, because we cannot remove the backup power source if the system will be overloaded; and (3) Initial limitation of the state *OB* is set as: $(p_{i,i}+6n_i)<1.1(p_{i,i,m}+p_b)$.

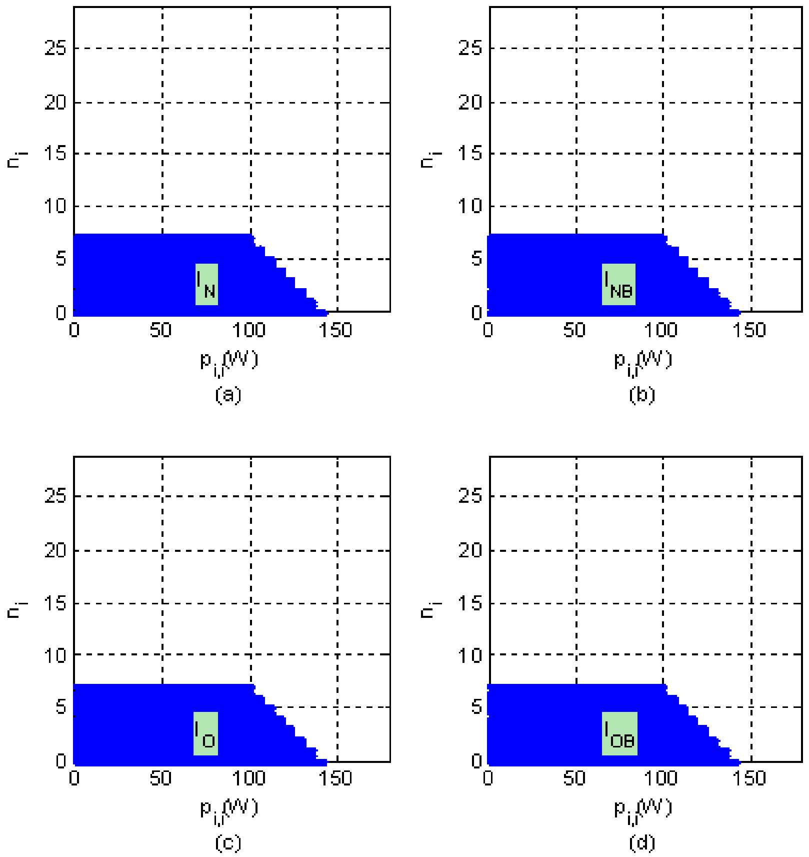
With the help of variables $p_{i,i}$ and n_i, the dynamic of local load at node *i* is clearly represented by the FSMwV model without having a large number of states. This FSMwV model clearly shows the relationship among the system status, the penetration of PHEVs and the amount of conventional loads in a more efficient way.

5.3 *Offline Safety Control*

To synthesize a safety controller, two scenarios for event β_i^-, uncontrollable and enforceable, are considered. We use the method described in Section 3 to calculate safety conditions I_q iteratively. We assign the variables as: $p_{i,i,m}$ = 100 *kW*, p_b = 30 *kW*. Since the iterations are rather involved and time consuming, we write a computer program to do the calculations.

When the event β_i^- is considered as uncontrollable and unenforceable, the safety regions representing safety conditions I_q of states *N, NB, O* and *OB* are shown in Figure 13 after 101 iterations. We do not show safety conditions I_T and I_D, because they are simple: I_T is always "False" and I_D is "False" after the first iteration since the transition from state D to state T is uncontrollable.

From Figure 13, we can see that the safety regions of states *N, NB, O* and *OB* are all very small. Intuitively, this is because if the controller cannot unplug PHEVs, then it must be very conservative when it allows PHEVs to charge. The maximal number PHEVs can be charged is only 7. This is the case even if the conventional loads are very low. This means the capacity of the distribution network (and the generation capacity) is not fully utilized. This control is not suitable for the increasing use of PHEVs.



(a)

(b)

(c)

(d)

Figure 13. Safety regions when β_i^- is uncontrollable
(a) State *N*, (b) State *NB*, (c) State *O* and (d) State *OB*.

When the event β_i^- is considered as enforceable, the safety regions representing safety conditions I_q of states *N, NB, O* and *OB* are shown in Figure 14 after 33 iterations. The I_D is also "False" after the first iteration.

It is clear from Figure 14 that the safety regions of states *N, NB, O* and *OB* are much bigger than the uncontrollable scenario. This is because if PHEVs can be unplugged by the controller, then the control of charging of PHEVs becomes more flexible. The control strategy is based on two premises: to guarantee the safety of the system (to avoid entering the illegal states) and to give preference to uncontrollable conventional loads. This control not only ensures the safety of the distribution network, but also takes full advantage of its capacity. It allows as many PHEV to be charged as possible.

6 Conclusion

In this paper, we have presented our work on control synthesis under the modeling framework of Finite State Machine with Variables. We have described our extension of the scope of the traditional DES (*i.e.*, supervisory) control to include both event disablement and enforcement for the control of discrete event systems modeled as FSMwV. We have proposed an offline safety control synthesis procedure that takes the advantage of both event disablement and enforcement in order to prevent the controlled system from venturing into illegal states. We have further presented online safety control synthesis procedures based on the limited/variable lookahead policies to address the practical concern of real world implementation. We have also applied the theoretical results to control PHEVs in power distribution networks.

Acknowledgement

This research is supported in part by the National Science Foundation of USA under Grant ECS-0823865, the National Natural Science Foundation of China under Grants 60904019 and 60804042.

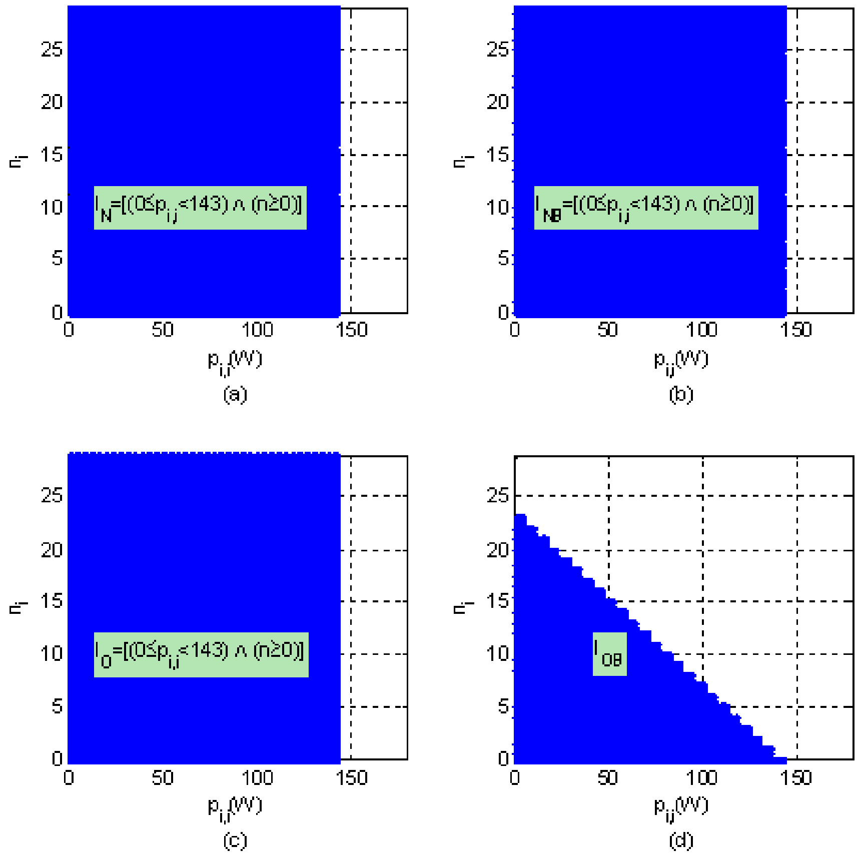


Figure 14. Safety area when β_i^- is enforceable event of
(a) State *N*, (b) State *NB*, (c) State *O* and (d) State *OB*.

References

[1] P. J. Ramadge, W. M. Wonham, Supervisory control of a class of discrete event processes, SIAM J. Control Optim., 25 (1987) 206-230.

[2] C. G. Cassandras, S. Lafortune, Introduction to Discrete Event Systems, Kluwer, 1999.

[3] J. L. Peterson, Petri Net Theory and the Modeling of Systems, Prentice-Hall, Upper Saddle River, NJ, 1987.

[4] L. E. Holloway, B. H. Krogh, A. Giua, A survey of Petri Net methods for controlled discrete event ystems, Discret. Event Dyn. S., 7 (1997) 151-190.

[5] Y. Li, W. M. Wonham, Control of vector discrete event systems I – the base model, IEEE Trans. Automat. Control, 38 (1993) 1214-1227.

[6] Y. Li, W. M. Wonham, Control of vector discrete event systems II– controller synthesis, IEEE Trans. Automat. Control, 39 (1994) 512-531.

[7] G. Cohen, S. Gaubert, J. P. Quadrat, Timed-event graphs with multipliers and homogeneous min-plus systems, IEEE Trans. Automat. Control, 43 (1998) 1296-1302.

[8] P. W. Glynn, A GSMP formalism for discrete event systems, Proc. of IEEE, 77 (1989) 14-23.

[9] F. Lin, W. M. Wonham, On observability of discrete event systems, Inform. Sci., 44 (1988) 173-198.

[10] P. J. Ramadge, W. M. Wonham, The control of discrete event systems, Proc. of the IEEE, 77 (1989) 81-98.

[11] K. T. Cheng, A. S. Krishnakumar, Automatic generation of functional vectors using the extended finite state machine model, ACM T. Des. Automat. El., 1 (1996) 57-79.

[12] Y. Yang, P. Gohari, Embedded supervisory control of discrete-event system, Proc. IEEE Int. Conf. Autom. Sci. Eng., (2005) 410-415.

[13] Y. Yang, A. Mannani, P. Gohari, Implementation of supervisory control using extended finite state machines, Int. J. Syst. Sci., 39 (2008) 1115-1125.

[14] A. Mannani, Y. Yang, P. Gohari, Distributed extended finite state machines: communication and control, Proc. Int. Workshop Discret. Event Syst., (2006) 161-167.

[15] A. Voronoc, K. Akesson, Verification of supervisory control properties of finite automata extended with variables, Technical Report, Chalmers University of Technology (2009).

[16] M. Skoldstam, K. Akesson, M. Fabian, Modelling of discrete event systems using finite automata with variables, Proc. IEEE Conf. Control and Decis., (2007) 3387-3392.

[17] B. Gaudin, P. Deussen, Supervisory control on concurrent discrete event systems with variables, Proc. Am. Control Conf., (2007) 4274-4279.

[18] T. L. Gall, B. Jeannet, H. Marchand, Supervisory control of infinite symbolic systems using abstract interpretation, Proc. IEEE Conf. Decis. Control and Eur. Control Conf., (2005) 30-35.

[19] Y.-L. Chen and F. Lin, “Modeling of discrete event systems using finite state machines with parameters”, Proceedings of the 2000 IEEE International Conference on Control Applications, (2000), 941-946.

[20] S. L. Chung, S. Lafortune, F. Lin, Limited lookahead policies in supervisory control of discrete event systems, IEEE Trans. Automat. Control, 37 (1992) 1921-1935.

[21] N. B. Hadj-Alouane, S. Lafortune, F. Lin, Variable lookahead supervisory control with state information, IEEE Trans. Automat. Control, 39 (1994) 2398-2410.

[22] J. H. Prosserl, J. Selinskyl, H. G. Kwatny, M. Kaml, Supervisory control of electric power transmission networks, IEEE Trans. Power Syst., 10 (1995) 1104-1110.

[23] L. H. Fink, Discrete events in power systems, Discret. Event Dyn. S., 9 (1999) 319-330.

[24] I. A. Hiskens, Power system modeling for inverse problems, IEEE Trans. Circuits Systems I, 51 (2004) 539-551.

[25] H. Zhao, Z. Mi, H. Ren, Modeling and analysis of power system events, Proc. IEEE Power Eng. Soc. Gen. Meet., Montreal, Canada (2006).

[26] H. Heymann, F. Lin, G. Meyer, Synthesis of minimally restrictive legal controllers for a class of hybrid systems, Hybrid Syst. IV, Lecture Notes in Comput. Sci., 1273 (1997) 134-159.

[27] M. Heymann, F. Lin, Discrete event control of nondeterministic systems, IEEE Trans. Automat. Control, 43 (1998) 3-17.

[28] Y. Li, F. Lin, Z. H. Lin, A generalized framework for supervisory control of discrete event systems, Int. J. Intell. Control Syst., 2 (1998) 139-159.

[29] M. Heymann, F. Lin, G. Meyer, Synthesis and viability of minimally interventive legal controllers for hybrid systems, Discret. Event Dyn. S., 8 (1998) 105-135.

[30] M. Duvall, Grid integration of plug-in hybrid electric vehicles, Technical Report, Electric Power Research Institute (2009).

[31] C. Roe, F. Evangelos, J. Meisel, A. P. Meliopoulos, T. Overby, Power system level impacts of PHEVs’, Proc. Int. Conf. Syst. Sci., Hawaii, USA, (2009) 1-10.

[32] K. Clement-Nyns, E. Haesen, J. Driesen, The impact of charging plug-in hybrid electric vehicles on a residential distribution grid, IEEE Trans. Power Syst., 25 (2010) 371-380.

[33] R. Green, L. Wang, M. Alam, The impact of plug-in hybrid electric vehicles on distribution networks: a review and outlook, Renew. Sust. Energ. Rev., 15 (2011) 544-553.