Proceedings of the 37th IEEE
Conference on Decision & Control
Tampa, Florida USA • December 1998



# Programmable Timed Petri Nets in the Analysis and Design of Hybrid Control Systems[1]

X.D. Koutsoukos, P.J. Antsaklis. K.X. He, and M.D. Lemmon
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
e-mail:xkoutsou,antsakli,xhe,lemmon@maddog.ee.nd.edu

**Abstract**

In this paper, a class of timed Petri nets, named programmable timed Petri nets is used to model and study switched hybrid systems. Supervisory control of a hybrid system in which the continuous state is transfered to a region of the state space in a way that respects safety specifications on the plant's discrete and continuous dynamics is examined. The approach is illustrated using a power system example.

## 1 Introduction

In hybrid systems the behavior of interest is governed by interacting continuous and discrete dynamic processes. There are several reasons for using hybrid models to represent dynamic behavior of interest. Reducing complexity was and still is an important reason for dealing with hybrid systems. For example, in order to avoid dealing directly with a set of nonlinear equations one may choose to work with sets of simpler equations (e.g. linear), and switch among these simpler models. This paper considers systems that arise when a high-level discrete (-event) supervisor is used to coordinate the actions of various subsystems so that overall system safety is not compromised. These supervised systems can be viewed as a *hybrid* mixture of systems with continuous dynamics (continuous variables) supervised by a switching law generated by a (discrete-event) supervisor described by discrete dynamics (discrete variables).

In order to deal with highly concurrent processes, it is necessary to use discrete-event system models which are better suited to model system concurrency. One such model is the ordinary Petri net [7]. Petri nets can be viewed as a generalization of finite automata. Petri nets provide an excellent tool for easily capturing the inherent concurrency of a complex system as well as providing the means of modeling conflict within the system. In general, a Petri net representation for a concurrent process will be more compact (fewer vertices) than its associated automaton representation and with the use of partial order semantics [5] it is now possible to search the Petri net's state space in an efficient manner. Furthermore, recent results in the supervisory control of discrete-event systems using ordinary Petri nets [6] have made it possible to design supervisors in an efficient and transparent manner.

In Section 2, a class of timed Petri nets named *programmable timed Petri nets* [3] is presented. The main characteristic of the proposed modeling formalism is the introduction of a clock structure which consists of generalized local timers that evolve according to continuous-time vector dynamical equations. They can be seen as an extension of the approach taken in [1] and provides a simple, but powerful way to annotate the Petri net graph with generalized timing constraints expressed by propositional logic formulae. In Section 3, programmable timed Petri nets are used to model switched dynamical systems. Section 4 discusses a Petri net approach to hybrid control. In the last section, the PTPN modeling of hybrid dynamical systems is illustrated with a power system example.

## 2 Programmable Timed Petri Nets

This section introduces a hybrid system model in which timed Petri nets [8] generate the switching logic of the system. In timed Petri nets the firing of a transition occurs over a time interval $[\tau_0, \tau_f]$. The length of this interval is called the transition's *holding time.* A transition $t$ which starts to fire at time $\tau_0$ is said to be *committed.* The duration of the firing interval (holding time) can be characterized in a variety of ways. These time intervals can be controlled by introducing "local" timers which cause transitions to fire when specified conditions *programmed* by the system designer are satisfied. Essentially, this approach characterizes the holding times by logical propositions defined over the times generated by a set of *local clocks.* Petri nets whose

[1]The partial financial support of the National Science Foundation (ECS95-31485) and the Army Research Office (DAAG55-98-1-0199) is gratefully acknowledged.

0-7803-4394-8/98 $10.00 © 1998 IEEE

holding times are defined in this way will be referred to as *programmable timed Petri Nets* (PTPNs).

Let $\mathcal{N} = (P, T, I, O)$ be an ordinary Petri net [7]. We introduce a set, $\mathcal{X}$, of $N$ *local clocks* where the $i$th clock $\mathcal{X}_i$ is denoted by the triple $(\dot{x}_i, x_{i0}, \tau_{i0})$. $x_{i0} \in \Re^n$ is a real vector representing the clock's offset. $\tau_{i0}$ is an initial time (measured with respect to the global clock) indicating when the local clock was started. $\dot{x}_i : \Re^n \rightarrow \Re^n$ is a Lipschitz continuous automorphism over $\Re^n$ characterizing the local clock's rate. Assume that the clock rate $\dot{x}_i$ is denoted by the automorphism $f$. The *local time* generated by the $i$th clock will be denoted as $x_i$ which is a continuous differentiable function over $\Re^n$ that is the solution to the initial value problem,

$$\frac{dx_i}{d\tau} = f(x_i), \;\; x_i(\tau_{i0}) = x_{i0} \qquad (1)$$

for $\tau > \tau_{i0}$. We therefore see that the local timers are vector dynamical equations. The local time of the $i$th timer at global time $\tau$ is denoted as $x_i(\tau)$ and the timer's rate is denoted as $\dot{x}_i(\tau)$. We say that the *state* of the $i$th timer is the ordered pair $z_i(\tau) = (x_i(\tau), \dot{x}_i(\tau))$. The interval $[\tau_0, \tau_f]$ over which a transition $t$ will be firing is going to be characterized by formulae in a propositional logic whose atomic formulas are equations over the local times or clock rates of $\mathcal{X}$.

**Definition 1** An *atomic formula*, $p$, takes one of the following forms: (i) a *time constraint* of the form $h(x_i) = 0$ or $h(x_i) < 0$ where $h : \Re^n \rightarrow \Re$ is a real valued function, (ii) a *rate constraint* of the form $\dot{x}_i = f$ which means that the $i$th clock's rate $\dot{x}_i$ is equal to the vector field $f : \Re^n \rightarrow \Re^n$, and (iii) a *reset equation* of the form $x_i(\tau) = \bar{x}_0$ which says that the $i$th clock's local time at global time $\tau$ is set to the vector $\bar{x}_0$.

**Definition 2** A *well-formed formula* (WFF) is define as any expression generated by a finite number of applications of the following rules: (i) any atomic formula is a WFF, (ii) if $p$ and $q$ are WFFs, then $p \wedge q$ is a WFF, (iii) if $p$ is a WFF, then $\tilde{p}$ is a WFF.

The set of all WFFs formed in this manner will be denoted as $\mathcal{P}$. Consider an ordinary Petri net, $\mathcal{N} = (P, T, I, O)$ and a set of logical timers, $\mathcal{X}$. A *programmable timed Petri net* (PTPN) is denoted by the ordered tuple $(\mathcal{N}, \mathcal{X}, \ell_P, \ell_T, \ell_I, \ell_O)$ where the functions $\ell_P : P \rightarrow \mathcal{P}$, $\ell_T : T \rightarrow \mathcal{P}$, $\ell_I : I \rightarrow \mathcal{P}$, and $\ell_O : O \rightarrow \mathcal{P}$ label the places, transition, input arcs, and output arcs (respectively) of the Petri net $\mathcal{N}$ with WFFs in $\mathcal{P}$. The *syntax* for WFFs is defined with respect to the underlying Petri net structure of the form $\mathcal{N} = (P, T, I, O)$ and the set of local clocks $\mathcal{X}$. The local clock state $z$ at time $\tau$ is said to *satisfy* a formula $p \in \mathcal{P}$ if $p$ is "true" for the given clock state, $z(\tau)$. The truth of the WFFs is understood in the usual sense.

## 3 PTPN Modeling of Switched Systems

A switched system is a continuous-time system whose structure changes in a discontinuous manner as the system state evolves into switching sets. More formally, such systems are often represented by the equations

$$\begin{aligned} \dot{x} &= f_{i(\tau)}(x(\tau), w(\tau)) & (2)\\ i(\tau) &= q(x(\tau), i(\tau^-)) & (3) \end{aligned}$$

where $x : \Re \rightarrow \Re^n$ and $i : \Re \rightarrow \mathbb{Z}^+$ denote the continuous and discrete states of the system, respectively. The signal $w : \Re \rightarrow \Re^m$ is an exogenous disturbance. The continuous dynamics are controlled by a finite collection of $N$ control strategies $\mathcal{D} = \{f_1, f_2, \cdots f_N\}$ where $f_i : \Re^n \times \Re^m \rightarrow \Re^n, i = 1, \ldots, N$ are locally Lipschitz continuous functions. The discrete state of the system is controlled by a *successor* function $q : \Re^n \times \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ which determines the next possible discrete state $i(\tau)$ at time $\tau$ given the current continuous state and the "previous" discrete state $i(\tau^-)$, where $i(\tau^-)$ denotes the left hand limit of $i$ at time $\tau$.

Let $\mathcal{D} = \{f_1, \ldots, f_N\}$ be a set of $N$ Lipschitz continuous vector fields and let $\mathcal{G} = \{h_1, \ldots, h_M\}$ be a set of smooth hypersurfaces in $\Re^n$. The functions in $\mathcal{G}$ are sometimes referred to as the *guards* of the system. Consider a network $\mathcal{N} = (P, T, I, O)$ and a set of timers $\mathcal{X}$ where the $i$th timer has rate $\dot{x}_i$, initial time $x_{i0}$, and reset time $\tau_{i0}$. We label the places, transitions, and arcs of the Petri net $\mathcal{N}$ with WFFs defined over the timer states, $z_i$. In particular, these labels are defined as follows. (i) Let $J(p)$ be a subset of $\{1, \ldots, N\}$ associated with place $p \in P$ representing those clocks associated with place $p$. $\ell_P(p)$ is a WFF of the form,

$$\ell_P(p) = \bigwedge_{i \in J(p)} ((\dot{x}_i = f_j) \wedge (\tau_{i0} = \tau)) \qquad (4)$$

This formula is interpreted as follows. When place $p$ is marked, then the timer states, $z_i$, for all $i \in J(p)$ are reset to satisfy $\ell_P(p)$. In particular, this means that the initial time, $\tau_{i0}$, and the clock rate, $\dot{x}_i$, are reset to the values specified in the equation. The label $\ell_P(p)$ is therefore used to represent switching of the system's vector field when events occur (i.e. transitions fire). (ii) $\ell_T(t)$ is chosen to be a tautology. (iii) Let $J(p, t)$ be a subset of $\{1, \ldots, M\}$ denote a set of hypersurfaces in $\mathcal{G}$ associated with the input arc, $(p, t)$. $\ell_I((p, t))$ is chosen to be a WFF whose truth commits the transition $t$ to firing provided this transition is already enabled. In particular, we confine our attention to WFFs of the form

$$\ell_I((p, t)) = \bigwedge_{i \in J(p,t)} (h_i(x(\tau)) < 0) \qquad (5)$$

This condition allows $t$ to be committed to firing when the continuous state (at time $\tau$) satisfies the listed set of inequalities with respect to the hypersurfaces in $\mathcal{G}$.

We refer to $\ell_I((p,t))$ as the input guard equation. (iv) $\ell_O((t,p))$ is chosen as a WFF whose truth completes the firing of the transition, $t$, assuming that transition $t$ is enabled and committed. These conditions also take the same form as the input guard equation (5) labeling the network's input arcs.

Use the guidelines mentioned above, we can construct PTPN for switched systems characterized by a generalization of equations (2) and (3). The generalization we consider treats the discrete state $i$ in equation (3) as a vector in $\{0,1\}^N$ rather than a nonnegative integer in $\mathbb{Z}^+$. Let $i \in \{0,1\}^N$ be represented by the vector $i = [i_1, i_2, \cdots, i_N]$ where $i_j \in \{0,1\}$ for all $j = 1, \ldots, N$ is the $j$th element of the vector $i$. We can therefore generalize equations (2) and (3) as follows. Let the mapping $f$ in equation (2) be written as $f = [f_1, f_2, \cdots, f_n]$ where $f_j : \Re^n \times \Re^m \times \{0,1\}^N \to \Re$ is a scalar function representing the rate of change for the $j$th continuous state. Also let the mapping $q$ in equation (3) be written as $q = [q_1, q_2, \ldots, q_n]$ where $q_j : \Re^n \times \{0,1\}^N \to \{0,1\}$ is a scalar function representing change of $j$th discrete state. We assume $f$ and $q$ are both partial functions of the discrete vector $i \in \{0,1\}^N$ which means that $f$ and $q$ may not exist for all $i$. We represent the switched system by the equations

$$\dot{x}_j(\tau) = f_j(x(\tau), w(\tau), i(\tau)), \; j = 1, \ldots, n \quad (6)$$

$$i_k(\tau) = q_k(x(\tau), i(\tau^-)), \; k = 1, \ldots, N \quad (7)$$

We say the model is *well-posed* if for all $i, i' \in \{0,1\}^N$ such that $f_j(x,w,i)$ and $f_j(x,w,i')$ exist, then $f_j(x,w,i) = f_j(x,w,i')$ whenever the $l$th components, $i_l$ and $i'_l$, are marked (i.e. $i_l = i'_l = 1$). This condition ensures that the marking of the $l$th component of the discrete state $i$ has a unique set of differential equations associated with it.

We can now associate a Petri net $\mathcal{N} = (P, T, I, O)$ with the switched system characterized by equation (6) and (7), by letting the set of places be $P = \{1, 2, \ldots, N\}$ and the set of transitions $T = \{(i,j) \in \{0,1\}^N \times \{0,1\}^N | q(x,i) = j \text{ exists}\}$. The input and output arcs are obtained by examining the transitions in $T$. The set of input arcs are characterized by the equation $I = \{(p,t) \in P \times T | t = (i,j), i_p = 1\}$ and the set of output arcs by $O = \{(t,p) \in T \times P | t = (i,j), j_p = 1\}$.

## 4 Supervision of Hybrid Systems

This section describes how switching policies that guarantee the safe operation of hybrid systems can be incorporated in the PTPN model. In particular, the supervisory control of a hybrid system in which the continuous state is transfered to a region of the state space in a way that respects safety specifications on the plant's discrete and continuous dynamics is examined.

The discrete specifications represent logical constraints on the switching policy (for example mutual exclusion constraints) and are expressed as linear predicates on the marking of the Petri net. A DES control method, namely supervisory control of Petri nets based on place invariants [10, 6] is applied to satisfy these discrete specifications. In view of the continuous dynamics, an algorithm based on the notion of a common flow region is used to determine the exact mode switching between the subsystems and to characterize the length of time each subsystem will be active. This is accomplished by determining the set of hypersurfaces $\{h_i\}$, $i \in J(p,t)$ which are used to label the input and output arcs (see equation (5)).

We introduce now some additional notation that will be useful later in the section. The firing times of transition $t$ are described by $\sigma^t(n)$, $n \in \mathbb{Z}^+$, where $\sigma^t(k) \in \Re$ represents the duration of the $k^{th}$ firing of transition $t$. During the time interval $\sigma^t(k)$ the tokens of the input places of transition $t$ do not change. These tokens are put into the output places of $t$ upon the completion of firing of the transition, according to the enabling condition of the untimed Petri net. We assume that $0 < \Delta \leq \sigma^t(n) < \infty$, for some $\Delta \in \Re$, for all firings $n$ and transitions $t$. Next, a firing event is defined as the pair $(t, \tau)$ which denotes that the transition $t$ starts firing at time $\tau$. Consider the sequence of firing events

$$s = (t_{i_0}, \tau_0), (t_{i_1}, \tau_1), \ldots, \; i_j \in \{1, \ldots, N\}, \; j = 0, 1, \ldots$$

where $j$ denotes the ordering of the transitions that fire. For example, $s = (t_1, \tau_0), (t_3, \tau_1), \ldots$ denotes that $t_1$ fires at $\tau_0$, next $t_3$ fires at $\tau_1$ and so on. The firing time intervals are defined by the equation $\sigma^{t_i}(k) = \tau_{k+1} - \tau_k$. At the $k^{th}$ firing of the network, the transition $t_i$ starts firing (at time $\tau_k$) for $\sigma^{t_i}(k)$ time units (until $\tau_{k+1}$).

In the nonlinear control literature, switching has been used to expand the domain of attraction of operation points in control systems [4]. In the hybrid systems case, we assume that for each control strategy there exists a unique equilibrium point for the resulting continuous subsystem $f_i$, $i = 1, \ldots, N$. Each equilibrium has a domain of attraction associated with it. The idea is to switch at discrete time instants from one mode (subsystem $f_i$) to another in a way that the system gradually progresses from one equilibrium to another towards the final equilibrium. This can be formalized using an *invariant based approach* for hybrid systems proposed in [9]. A *common flow region* for a given target region, is defined as a set of states which can be driven to the target region with the same control policy. A common flow region can be determined by a set of hypersurfaces. The basic property of the regions defined by these hypersurfaces is that their boundaries satisfy certain conditions that preclude the state trajectories from crossing them. Sufficient conditions for a set of hypersurfaces to form a common flow region are given in [9]. These hypersurfaces can be either *invariant* under the vector field of the given control policy (see [9]) or *cap boundaries* for the given vector field. For a hypersurface $h_c$ to form a cap boundary for a given vector field $f$ and common flow region $B$, the following condition must be satisfied

$$\nabla_\xi h_c(\xi) \cdot f(\xi) < 0,\ \forall \xi \in B \cap Null(h_c) \tag{8}$$

In this paper, a Lyapunov approach is followed to efficiently compute hypersurfaces that form common flow regions for each control policy. Each common flow region is identified as a subset of an invariant manifold defined by a Lyapunov functional and is associated with a control policy. Consider the hypersurface $h_c(x)$ that forms a cap boundary for the common flow region $B$. Assume that there exists an appropriate Lyapunov function $V(x)$ for the vector field $f$. Then $\Omega_c = \{x \in \Re^n | \ V(x) \leq c\}$ is bounded and the hypersurface $h_c(x) = V(x) - c$ is a cap boundary candidate. The constant parameter $c$ can be selected appropriately so that $h_c(x)$ bounds the common flow region $B$. Based on these results, appropriate cap boundaries can be determined efficiently using Lyapunov theory. Furthermore, the design based on Lyapunov functions will exhibit desirable robustness properties. The next proposition gives sufficient conditions for the state to progress from one equilibrium point to another.

**Proposition 1** *Let $f_{i_1}, f_{i_2} \in \mathcal{D}$ satisfy the following assumptions. Each $f_i$ admits an isolated equilibrium point $\bar{x}_i$, and $\bar{x}_i$ is asymptotically stable w.r.t. $f_i$. For each $f_i$ there exists an appropriate Lyapunov function $V_i : \Re^n \to \Re$ and $\Omega_i = \{x \in \Re^n | \ V_i(x) \leq c_i\}$ such that*

$$\begin{aligned} V(x) > 0,\ \forall x \in \Omega_i \\ V(x) \to \infty \ as \ ||x|| \to \infty \\ \dot{V}(x) < 0,\ \forall x \in \Omega_i \end{aligned} \tag{9}$$

*In addition, assume that $\Omega_{i_1} \cap \Omega_{i_2} \neq \emptyset$ and $\bar{x}_{i_1} \in R' = int(\Omega_{i_1} \cap \Omega_{i_2})$, then for every $x_0 \in \Omega_{i_1}$ there exists a switching sequence $s(x_0, \tau_0) = (i_1, \sigma^{t_{i_1}}(k_0)), (i_2, \sigma^{t_{i_2}}(k_1))$ which drives the state to a region $R$ of the equilibrium point $\bar{x}_{i_2}$.*

The following corollary gives sufficient conditions for a switching sequence generated by the controlled Petri net to drive the continuous state $x_0$ to a target region of the state space. It is assumed that the initial conditions belong to the region of attraction $\Omega_{i_0}$ of the first control policy and that the state progresses towards $\bar{x}_{i_m} \in \Omega_{i_m}$ by allowing switchings to occur on the intersection $\Omega_{i_j} \cap \Omega_{i_{j+1}}$ of consecutive invariant manifolds. In the case when all the pairs of subsystems satisfy Proposition 1, the set $\Omega_{i_j} \cap \Omega_{i_{j+1}}$ will be nonempty and the proof is straightforward.

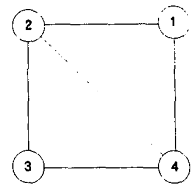


**Figure 1:** The example power system

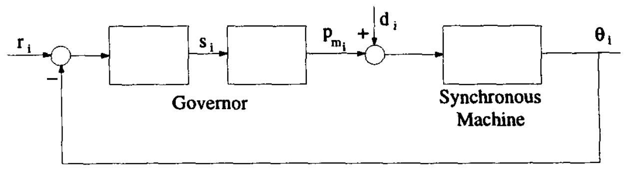


**Figure 2:** Simplified generator block diagram

**Corollary 1** *Suppose there exists a switching sequence accepted by the controlled Petri net such that every pair $(f_{i_j}, f_{i_{j+1}})$ satisfies Proposition 1 . Given a target region $R$ such that $\bar{x}_{i_m} \in int(R)$, there exists switching policy to drive the continuous state from any initial condition $x_0 \in \Omega_{i_0}$ to the region $R$ in finite time. The firing time intervals $\sigma^t(n)$ will be chosen so that the switchings occur while $x \in int(\Omega_{i_j} \cap \Omega_{i_{j+1}})$.*

The switching policy is implemented by assigning WFFs to the input and output arcs of the controlled Petri net that have the form of equation (5) and represent the regions where switchings are allowed to occur.

## 5 Analysis and Design of Power Systems using PTPN

In this section, a power system example is used to illustrate the use of the PTPN in modeling multiagent systems. The supervisor control methodology presented in Section 4 is used to drive the setpoint of the system to a desired region of the state space.

We consider the 4 node power system shown in Fig. 1. Each node in the figure represents a generator and the arcs denote the transmission lines between generators. A simplified block diagram for each generator is shown in Fig. 2. A generator consists of a synchronous machine and a governor that controls the mechanical power input to the rotor. The continuous state of the $i$th generator is characterized by its rotor angle, $\theta_i$, the rotor angle's rate of change $\dot{\theta}_i$, the variation in the mechanical power $p_{m_i}$, and the change $s_i$ in the valve displacement that determines the input to the turbine. Without loss of generality, we assume that node 4 is a reference node, so that $\theta_4 = \dot{\theta}_4 = p_{m_4} = s_4 = 0$. The continuous state of the system $x \in \Re^{12}$ is

$$x = [\theta_1, \dot{\theta}_1, p_{m_1}, s_1, \theta_2, \dot{\theta}_2, p_{m_2}, s_2, \theta_3, \dot{\theta}_3, p_{m_3}, s_3]^T$$

The differential equations for the $i$th generator are

$$\begin{aligned} \frac{d^2\theta_i}{d\tau^2} &= -D_i\frac{d\theta_i}{d\tau} + \frac{1}{M_i}(p_{m_i} + d_i) \\ \frac{dp_{m_i}}{d\tau} &= \frac{1}{T_T}(s_i - p_{m_i}) \\ \frac{ds_i}{d\tau} &= \frac{1}{T_G}(-s_i - \frac{1}{R}\frac{d\theta_i}{d\tau} + r_i) \end{aligned} \quad (10)$$

where $D_i, M_i, T_T, T_G, R$ are constants determined by the physical characteristics of the generator. In the above equations, $d_i$ represents the variation in the load $d_i = v_i + \delta p_i$ which consists of two terms. The term $v_i$ represents a significant change in the load that the system should response to and $\delta p_i$ represents the variation of the real power due to the generator coupling. From the power flow equation we obtain that

$$\delta p_i = \sum_j b_{ij} \cos(\theta_i - \theta_j)\theta_i \quad (11)$$

where $b_{ij}$ is a constant based on the transmission line parameters. The input $r_i$ represents the increment in the speed changer position which determines increases or decreases in the power demand. From the preceding equations, we obtain the following linearized system.

$$\begin{aligned} \dot{x} &= \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} x + \begin{bmatrix} B_1 & 0 & 0 \\ 0 & B_2 & 0 \\ 0 & 0 & B_3 \end{bmatrix} u \\ z &= Cx = [\theta_1, \theta_2, \theta_3]^T \end{aligned}$$

where

$$A_{ii} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{b_{ii}}{M_i} & -D_i & \frac{1}{M_i} & 0 \\ 0 & 0 & -\frac{1}{T_T} & \frac{1}{T_T} \\ 0 & -\frac{1}{RT_G} & 0 & -\frac{1}{T_G} \end{bmatrix},$$

$$A_{ij} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{b_{ij}}{M_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$B_i = \begin{bmatrix} 0 & 0 \\ \frac{1}{M_i} & 0 \\ 0 & 0 \\ 0 & \frac{1}{T_G} \end{bmatrix}, \text{ and } u = \begin{bmatrix} v_1 & 0 & 0 \\ r_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & r_2 & 0 \\ 0 & 0 & v_3 \\ 0 & 0 & r_3 \end{bmatrix}.$$

The initial setpoint is chosen so that $\theta_i = 0, i = 1,2,3$. *It is* assumed that a load change occurs at $\tau = 0$ and the increment of the speed changer $r_i$ is determined according to a prespecified rule to generate allocation levels. Both $v_i(t)$ and $r_i(t)$ are described by known

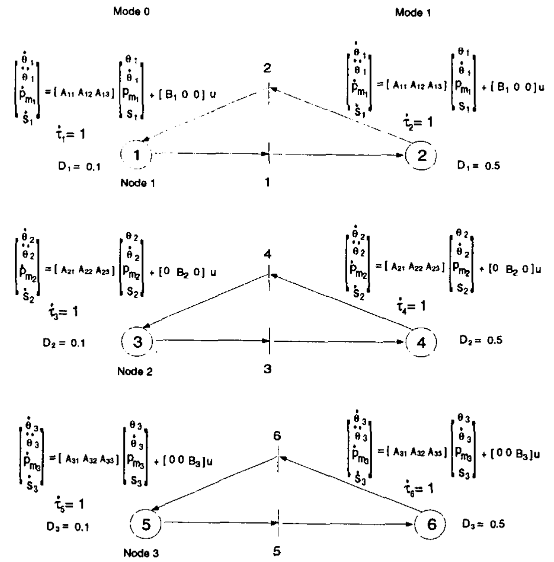


**Figure 3:** Original Petri net model of the power system

step inputs. The objectives for the allocation of generation is to maintain zero frequency error and to be in accordance with dispatching principles. The allocation of generation results in a new setpoint. The control objective is to drive the new setpoint of the system in a region where $|\theta_i| < 0.3$. A switching policy is used to help achieve this goal. It is assumed that each generator has two winding ratios to choose from, $D_{i0}$ and $D_{i1}$, for $i = 1, 2$ and 3. It is assumed that $D_{i0} = 0.1$ and $D_{i1} = 0.5$. The $i$th generator (node) is in discrete state 0 if the first winding ratio is used (i.e. $D_i = D_{i0}$) and is in state (mode) 1 otherwise. There are two conditions which the generators need to respond to. First, the generator must respond to a large setpoint, therefore the supervisor strategy forces the $i$th generator to switch from mode 0 to 1 when $|\theta_i| > 0.25$. Second, if a load change is detected and node is in mode 1, then the generator will switch to mode 0 to be able to track the load faster (smaller damping). The switch from mode 1 to mode 0 will be constrained to reset the operating mode after 5 seconds.

The strategy outlined above can be applied to each generator in a decoupled manner. We can therefore construct a network, $\mathcal{N}$, to represent the logical states of the system. We generalize the discrete state $i$ to a vector $i = [i_1, i_2, \cdots, i_6]$ where the $k$th component represent the marking of the $k$th place. We let the set of places $P = \{1, 2, \ldots, 6\}$ represent three generators in two different modes in the following way. Let place $2i-1$ represent generator $i$ in mode 0 and place $2i$ represent generator $i$ in mode 1, $i = 1,2,3$. It is easy to show that the preceding construction satisfies the "well-posed" condition. We can therefore associate $\left[\dot{\theta}_i, \ddot{\theta}_i, \dot{p}_{m_i}, \dot{s}_i\right]$, $(D_i = D_{i0})$ with place $2i-1$ and

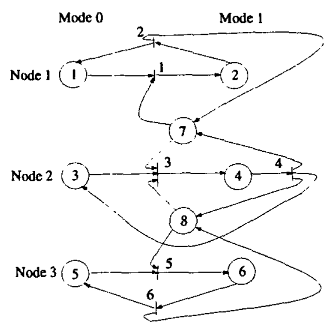


**Figure 4:** Controlled Petri net model of the power system

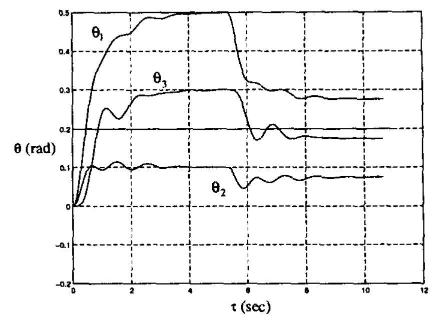


**Figure 5:** Simulation results

$\left[\dot{\theta}_i, \ddot{\theta}_i, \dot{p}_{m_i}, \dot{s}_i\right]$, $(D_i = D_{i1})$ with place $2i, (i = 1,2,3)$ as timers. We also associate each place with a local timer $\tau_i$. Six transitions are then derived to represent the switching policy between the different modes of the generator. The PTPN model of the original system is shown in Fig. 3.

In practice, an additional requirement is due to the fact that the generators are coupled by the transmission lines shown in Fig. 1. It is shown in [2] that if two neighboring generator nodes (i.e. nodes 1 and 2 or nodes 2 and 3) are both in mode 1, the rotor angle of the generators will exhibit large variations in the presence of the disturbance. The additional requirement is implemented via supervisory control [6] that prevents adjacent generators from being in mode 1 at the same time. The supervised system is shown in Fig. 4. The supervisor is implemented by adding the control places 7 and 8 to ensure that adjacent generators enter mode 1 in a mutually exclusive manner.

According to Proposition 1, the switching policy described above will drive the new setpoint to the desired region if we can guarantee that the switchings will occur inside the regions of attraction of the corresponding equilibrium points. Each equilibrium point is determined by the matrix $A$ and the step inputs $v_i$ and $r_i$. The matrix $A$ is Hurwitz for both modes 0 and 1, and therefore, the region of attraction of the equilibrium for each linearized system can be estimated by a Lyapunov functional.

A particular case is used to illustrate the supervisor control methodology. It is assumed that initially, generators 1 and 3 are in mode 1 and generator 2 at mode 0. A known load change occurs in the generators 1 and 3 at $t = 0$. Generators 1 and 3 will switch to mode 0 where they will soon reach the steady state. After 5 seconds the generators will switch to mode 1 to drive the new setpoint to a desired region so that the system is protected from random load disturbances. For this to be possible, we require that the switching will occur while the state lies in the region of attraction of the desired setpoint. For example, the trajectories of $\theta_i, i = 1,2,3$ when $v_1 = r_1 = 0.5, v_2 = r_2 = 0$ and $v_3 = r_3 = 0.3$ are shown in Fig. 5.

## References


[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P.-H. Ho, X. Nicollin, A. Oliveiro, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical and Computer Science*, 138:3–34, 1995.

[2] X. Koutsoukos, K. He, M. Lemmon, and P. Antsaklis. Timed Petri nets in hybrid systems: Stability and supervisory control. *Journal of Discrete Event Dynamic Systems: Theory and Applications*, 8(2):137–173, 1998.

[3] M. Lemmon, K. He, and C. Bett. Modeling hybrid control systems using programmable Petri nets. In *3rd International Conference ADMP'98*, pp. 177–184, Reims, France, March 1998.

[4] N. McClamroch, C. Rui, I. Kolmanovsky, and M. Reyhanoglu. Hybrid closed loop systems: A nonlinear control perspective. In *Proc. of the 36th IEEE Conference on Decision and Control*, pp. 114–119, 1997.

[5] K. McMillan. A technique of a state space search based on unfoldings. *Formal Methods in System Design*, 6(1):45–65, 1995.

[6] J. Moody and P. Antsaklis. *Supervisory Control of Discrete Event Systems using Petri Nets.* Kluwer Academic Publishers, 1998.

[7] W. Reisig. *Petri Nets.* Springer-Verlag, 1985.

[8] J. Sifakis. Use of Petri nets for performance evaluation. In *Measuring, modelling and evaluating computer systems.* North Holand, 1977.

[9] J. Stiver, P. Antsaklis, and M. Lemmon. Interface and controller design for hybrid control systems. In P. Antsaklis *et al.*, eds, *Hybrid Systems II*, Vol. 999 of *LNCS*, pp. 462–492. Springer, 1995.

[10] K. Yamalidou, J. Moody, M. Lemmon, and P. Antsaklis. Feedback control of Petri nets based on place invariants. *Automatica*, 32(1):15–28, 1996.