14

Supervisory Control and High-level Petri nets

Chiheb Ameur ABID, Sajeh ZAIRI and Belhassen ZOUARI
LIP2 Laboratory - University of Tunis
Tunisia

1. Introduction

The Supervisory Control Theory (SCT) (Ramadge & Wonham, 1989) was developed to provide a formal methodology for the automatic synthesis of controllers for Discrete Event Systems (DES). In this theory, a system, called a plant, is assumed to have uncontrollable behaviours which may violate some desired specifications. Hence, these behaviours have to be controlled by means of a feedback controller, called a controller (or a supervisor), so that the system fulfils the specifications. Primarily, the SCT was studied in the context of automaton based models. More recently, a special interest is given to the Petri net models for the studying of the control problem (Ghaffari et al., 2003; Giua & DiCesare, 1994; Sreenivas & Sreenivas, 1997), since they represent a good trade-off between modelling power and analysis capabilities. For details about the supervisory control problem methods based on Petri nets, one can refer to (Holloway et al., 1997; Su et al., 2005). In addition, high level nets, especially Coloured Petri nets (CP-nets) (Jensen & Rozenberg, 1991), provide a great improvement over the ordinary Petri nets. Notably, the high expressiveness of CP-nets allows to obtain compact models even for large systems, while keeping the same formal analysis capabilities. However, not many works have addressed the supervisory control problem by considering a CP-net as a plant model. In this context, we can cite the method developed in (Makungu et al., 1999). This method addresses the forbidden state problem for a class of CP-nets where the process to be controlled is separated from the control logic.

In this chapter, we review our previous works (Abid & Zouari, 2008; Zouari & Ghedira, 2004; Zouari & Zairi, 2005) for the supervisory control problem of DES modelled by CP-nets. The control specifications herein considered are expressed in terms of forbidden states, i.e. states which have to be avoided by the controlled model. In a first approach, we propose to derive a controller for a plant CP-net model by using the theory of regions. According to the control specifications, the desired behaviours are extracted from the rechability graph associated with the plant model. Then, the theory of regions is used in order to design the controller. Thanks to the expressiveness of CP-nets, as a main advantage, the obtained controller is reduced to one single place. Secondly, we propose to optimise the first approach in order to deal efficiently with symmetric systems. Indeed, the reachability graph of a symmetric system can be represented by an optimised version, called symbolic reachability graph (Chiola et al., 1991; 1997), which is quite smaller. Thereby, the use of symbolic graphs allows to alleviate one important drawback of the latter approach which is the well-known problem of the state space explosion. Moreover and consequently, the use of a smaller graph allows to reduce the complexity of the synthesis process. Finally, we propose an approach which considers as plant model a CP-net that is assumed to be structured on a set of generic processes sharing a set of resources. In addition to the avoiding of the use of the theory of regions, this approach generates a controller as an active process, modelled by a CP-net, and having the advantage to be implemented directly on existing tools such as CPN-Tools (Jensen et al., 2007).

The remainder of the paper is organised as follow. The second section introduces the notation and definitions used in CP-nets. Section 3 provides the basic concepts of the theory of regions. Section 4 deals with the synthesis of a CP-net controller for the forbidden state problem by applying the theory of regions. Section 5 optimises the latter approach for symmetric systems by applying the theory of regions on the basis of symbolic reachability graphs. Section 6 gives how to design a generic CP-net controller for certain systems without using the theory of regions. Finally, section 7 summarizes the main conclusions and perspectives of this chapter.

2. Well-Formed Coloured Petri nets (WF-nets)

High level nets (Jensen & Rozenberg, 1991) represent a natural extension of ordinary Petri net formalism. They enhance both readability and expressivity of Petri nets. As a main advantage, high level nets allow the generation of compact models even for large systems. This extension is mainly done by the introduction of 'colour' structures to identify tokens. Coloured (or in general High-level) Petri nets are particularly well-adapted for the modelling of parametric systems which behaviours depend on the basic structure of the model rather than on the cardinalities of the colour sets. The CP-net model used in this chapter is the Well-Formed coloured Petri nets (WF-nets) model (Jensen & Rozenberg, 1991). WF-nets are equivalent in expressiveness to CP-nets, but are syntactically restricted by enforcing a particular structuring on colour classes and functions.

In this section, we briefly present the different notions related to CP-nets, according to the syntax defined in the WF-net model.

WF-nets have the same modelling power as CP-nets, although syntactically different. Before presenting a formal definition of the model, let us present the related basic notions.

A *multiset* is a set in which given elements may appear several times. Given a set A. $Bag(A)$ denotes the set of finite multisets on A. A multiset $''a''$ can be represented as a sum: $a = \sum_{x \in A} a(x).x$ in which $a(x)$ gives the number of occurrences of the element x in the multiset $''a''$.

Object classes are finite non-empty sets of objects or basic colours. A class may be viewed as a set made up of elements of the same type. We can distinguish particular type of classes, called ordered classes, for which an order relation is defined on its elements.

Colour domains may be defined as a Cartesian product of object classes and is associated with either a transition or a place. When associated with a transition, it defines the set of all its firing instances (coloured firings). When associated with a place, it defines the set of all its possible markings.

Colour functions are a pondered sum of tuples of basic colour functions. Colour functions are associated with the labels of WF-net arcs. These functions allow to specify the number of coloured tokens to be consumed and to be produced when firing a given transition. There are three basic colour functions:

- the *identity* function, used for the choice of any object in a class, specified by a variable X,
- the successor function, used to specify the circular successor of an object in an ordered class, noted $\oplus X$,

- the diffusion function, used to specify all the objects of a class C_i, noted All_{C_i}.

A *guard* is a Boolean function defined on a colour domain and which role is to restrict it to a subdomain. When a guard is associated with the colour domain of a transition, it limits its possible firings. But a guard can also be associated with a colour function labelling an arc in order to indicate whether this arc is valid with respect to the guard value.
Let $[g]$ be a guard and f be a colour function. The guarded function $[g].f$ is defined by:
$\forall c \in C(t), [g]f(c) = (\text{if } g(c) \text{ then } f(c) \text{ else } 0)$.
We can note that a guard can lead to a cancelation between a place and a transition.

Definition 1. *A coloured Petri net is a 6-tuple*
$N =< P, T, C_l, C, W^-.W^+, \Phi, M_0 >$ *where:*
P is a finite set of places,
T is a set of transitions verifying $P \cap T = \emptyset, P \cup T \neq \emptyset$,
$C_l = \{C_1, C_2, ..., C_k\}$ is a set of object classes such that $\forall i, j \in \{1, ..., k\}, i \neq j, C_i \cap C_j = \emptyset$,
C is the colour function, defined from $P \cup T$ into a set of colour domains. An element c of $C(s)$ is a tuple $< c_1, ..., c_k >$ and is called a colour *of s,*
W^-, W^+ are the input and output functions (also called the incidence functions) defined on $P \times T$, such that $W^-(p, t)$ and $W^+(p, t)$ are guarded colour functions representing linear applications mapping $Bag(C(t))$ onto $Bag(C(p))$, for all $(p, t) \in P \times T$,
Φ is a function which associates a guard with any transition. By default Φ is true *for any transition t,*
M_0 the initial marking is a function defined on P, such that $M_0(p) \in Bag(C(p))$, for all $p \in P$.
$W = W^+ - W^-$ indicates the incidence matrix, and $W(p, .)$ is a line vector of such a matrix.

For reasons of clarity, we assume in this paper that the object classes are not ordered.
The dynamic behaviour of a coloured Petri net is determined by the following firing rule:
A transition t is enabled for a colour c and a marking M, denoted by $M[t, c\rangle$, iff $\forall p \in P, M(p) \geq W^-(p, t)(c)$.
The marking M' obtained after the firing of (t, c) is computed as:

$$\forall p \in P, M'(p) = M(p) - W^-(p, t)(c) + W^+(p, t)(c)$$

The notation $M[t, c\rangle M'$ is used to indicate this reachability relation. Using the firing rule, it is possible to construct a reachability graph $R(N)$, whose nodes are the markings reachable from the initial marking, and whose arcs represent the reachability relation. Such an arc is labeled by the transition name and the associated colour involved in the reachability relation between two given nodes.

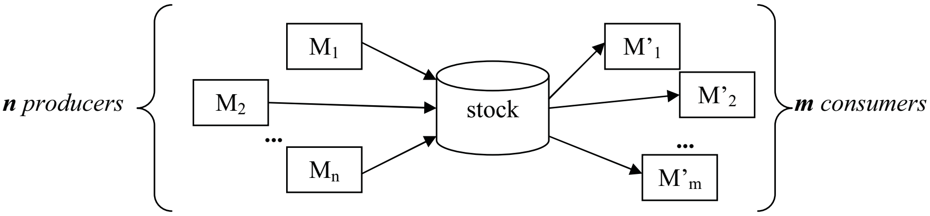


Fig. 1. Problem of producer-consumer

Throughout this chapter, we consider the well-known producer-consumer problem. As it is illustrated in Fig. 1, there are two kinds of machines, namely producers and consumers, sharing a stock. We have n producers and m consumers. A producer can produce an object and transfers it in the stock, while a consumer operates by using an object which have been already produced and transferred in the stock by a producer. The WF-net modelling this problem is illustrated in Fig. 2. For sake of simplicity, we have reduced the behaviour of a consumer to one state and one action. The consumption of an object, deposited in the stock, is traduced by the execution of transition $t3$. When a producer produces an object, it transfers it in the stock by executing transition $t2$. When place $p3$ contains no tokens, it indicates that the stock is full (producers can not transfer a new object in the stock). When place $p4$ contains no tokens, then the stock is empty. C_1, C_2 and C_3 denote the object classes of this net. $C1$ represents the producers, $C2$ allows to indicate the state of the stock and C_3 represents the consumers. The set of places of this net is $P = \{p1, p2, p3, p4\}$. The colour domains of places are: $C(p1) = C(p2) = C_1$, $C(p3) = C(p4) = C_2$ and $C(p5) = C_3$. The set of transitions is $T = \{t1, t2, t3\}$. The colour domains of transitions are: $C(t1) = C_1$,$C(t2) = C_1 \times C_2$ and $C(t3) = C_2 \times C_3$.

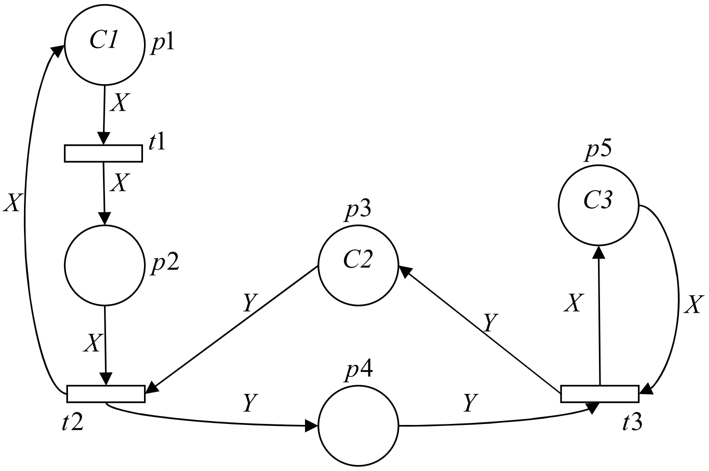


Fig. 2. Well-formed net of producer-consumer problem

3. Theory of regions

The aim of the theory of regions (Badouel et al., 1995) is to decide wether a given automaton is isomorphic to the reachability graph of a net, then constructing it. Ghaffari et al. (Ghaffari et al., 2003) are the first who proposed an adaptation of this theory for the synthesis controller problem using Petri nets. Their proposed method allows to add control places to an initial Petri net in order to avoid reaching undesired states. Considering a plant model modelled by a Petri net, this method starts by constructing its associated reachability graph. After that, one has to provide markings which must be avoided by the system. These markings correspond to *forbidden states*. This step enables to identify the *dangerous markings* as the predecessors of markings that allow reaching forbidden states by *uncontrollable events*. An uncontrollable event corresponds to a transition beyond any control procedure. Forbidden and *blocking* markings are removed in order to obtain a strongly connected modified reachability graph that respects liveness property and implements the maximally permissive legal behaviour. A marking is said blocking if it does not allow to reach a *final state*. Such a state represents a proper termination of some task and corresponds to a stable state of the system. From the obtained graph, the parameters of a pure control place, to be connected to a plant model, are computed by resolving a linear system of equations. According to the theory of regions, three classes of equations are defined. The work of Ghaffari et al. is based on the following theorem which defines how to derive a controller for an ordinary Petri:

Theorem 1. *Let $N =< P, T, W, M_0 >$ be a bounded Petri net such that P is a set of places, T is a set of transitions, and M_0 is its initial marking. Let R be the reachability graph of N. Let R_c be the desired legal behaviour of N (R_c is a subgraph of R). The supervisory control problem can be optimally solved by adding a set of control places P_c to N iff there exists a solution $(M_0(p_c), W(p_c, .)), \forall p_c \in P_c$ satisfying the following equations:*

1. *The* reachability equation *for every marking in R_c:*

$$M(p_c) = M_0(p_c) + W(p_c, .)\overrightarrow{\Gamma}_M \geq 0 \tag{1}$$

where Γ_M is a non oriented path of G from M_0 to M and $\overrightarrow{\Gamma}_M$ is its associated vector, called the vector counting of Γ_M. $\overrightarrow{\Gamma}_M$ is indexed by transitions of T. Each line $\overrightarrow{\Gamma}_M[t]$ represents the algebraic sum of occurrences number of t in Γ.

2. *The* cycle equation *of R_c*

$$W(p_c, .)\overrightarrow{\gamma} = 0, \forall \gamma \in S_c \tag{2}$$

where S_c is the set of cycles of G, and $\overrightarrow{\gamma}$ is the vector counting defined similarly as $\overrightarrow{\Gamma}_M$.

3. *For each pair (M, t) such that t does not fire from M, it exists at least one control place p_c which satisfies the equation of* state separation inequation*:*

$$M_0(p_c) + W(p_c, .)\overrightarrow{\Gamma}_M + W(p_c, t) < 0 \tag{3}$$

Equation of type (1), called reachability conditions, indicates that every reachable marking within the legal behaviour must remain reachable under control. Similarly, the cycle equations (2) indicate that the cycles must remain reachable under control. Finally, an equation of type (3), called an event separation condition, specifies for a pair (M, t) that the control must prevent the transition t from firing in marking M.

4. Synthesis of controllers for CP-nets

In this section, we present a controller synthesis approach for a DES modelled by a CP-net, where the control specifications are expressed in terms of forbidden markings. According to the provided control specifications, we determine the *admissible behaviours* from the reachability graph of the plant model, which are represented by an appropriate graph, called the *admissibility graph*. An admissible behaviour represents a behaviour of the controlled system under both safety specification and non-blocking requirement. Thanks to the expressiveness of CP-nets, the controller to be determined is reduced to one single CP-net place. Its parameters are obtained by applying the theory of regions on the basis of the computed admissibility graph.

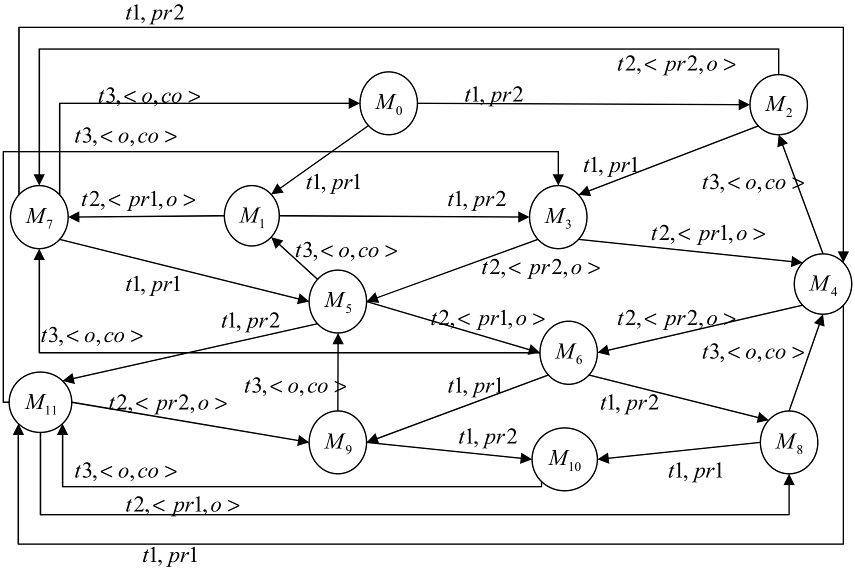


Fig. 3. Reachability graph of the producer-consumer problem

4.1 Computation of the admissibility graph

Let us assume that we have the reachability graph of a plant CP-net model. The control constraints considered herein are specified through a finite set of undesired markings. These markings and blocking ones are qualified as *forbidden markings*. A forbidden marking must not be reached by the controlled model. Thus, the key idea for the determination of the admissibility graph is to remove forbidden markings from the initial reachability graph of the plant model, and also to identify markings which lead inevitably to forbidden ones.

The SCT classifies the transitions into two categories. First category consists of the *controllable transitions* which may be disabled when it is necessary. In contrast, the transitions belonging to the second category, called *uncontrollable transitions*, are beyond any control procedure. Hence, we assume that the transition set T of the plant model is partitioned into two disjoint subsets: the set T_c of controllable transitions, and the set T_u of uncontrollable ones.

The role of a controller is to restrict the behaviour of the plant model by disabling some controllable transitions in order to avoid reaching forbidden states. The disabling of a controllable transition is performed in a *dangerous marking* from which the firing of the transition leads to a forbidden marking. So that, we have to identify the set Ω of *state-transitions* to be disabled. Every element of Ω is a couple $(M,(t,c))$ where M is a dangerous marking, and t is a controllable transition such that the firing of t with colour c from M yields to a forbidden marking.

As we have previously mentioned, the admissibility graph is computed from the reachability graph of the plant model by removing forbidden nodes. In addition, nodes becoming unreachable from the initial marking and the non coreachable markings must be removed from the admissibility graph. The identification of dangerous and forbidden nodes is performed according to the following rules:

- a marking is qualified as dangerous if it has at least one output arc where its destination is a forbidden marking,
- a marking is qualified as forbidden when it has no output arcs and it is not a final marking, or it is a dangerous marking and it has at least one output arc labelled by an uncontrollable transition,
- every forbidden marking must be removed with its input and output arcs.

The computation of the admissibility graph R_c and the set Ω is given by Algorithm 1. It is worth noting that it is an enhanced version of the algorithm proposed in (Zouari & Ghedira, 2004). The algorithm considers the reachability graph R of the plant model, the set FM of specified forbidden markings, the set MS of final markings and the set T_u of uncontrollable transitions. In each iteration of the main loop, we identify forbidden markings. These markings and their input/output arcs are then removed from the graph. After that, we qualify as forbidden the markings which are not reachable from the initial marking. Further, non coreachable nodes are qualified as forbidden. The loop terminates when all forbidden markings are processed.

```
input  : $R$ a reachability graph
         $FM$ is the set of initially specified forbidden markings
         $MS$ is the set of final markings
         $T_u$ is the set of uncontrollable transitions
output: $R_c$ the admissibility graph ; $\Omega$ the set of state-transitions

$DM \leftarrow \emptyset; TE \leftarrow \emptyset; \Omega \leftarrow \emptyset; R_c \leftarrow R$
repeat
    Take a non coloured element $f$ from $FM$
    Colour $f$ in $FM$
    for every input arc $(x, (t, c), f)$ of $f$ do
        if $t \in T_u$ then
            $FM \leftarrow FM \cup \{x\}$
        else
            $DM \leftarrow DM \cup \{x\}$ ; $TE \leftarrow TE \cup \{(x, (t, c))\}$
    Remove $f$, the input and output arcs of $f$ from $R_c$
    for every node $M$ of $R_c$ do
        if $M$ is not reachable from $M_0$ or $M$ is not coreachable or ($M$ has no output arcs and
        $M \notin MS$) then $FM \leftarrow FM \cup \{M\}$
    if $M_0 \in FM$ then exit //there is no solution
until all elements of $FM$ are coloured ;
$DM = DM \setminus FM$
for every element $y$ of $DM$ do
    for any element $(x, (t, c))$ of $TE$ do
        if $y == x$ then $\Omega \leftarrow \Omega \cup \{(x, (t, c))\}$
```

Algorithm 1: Computing the admissibility graph

Let us apply this algorithm to our problem of producer-consumer such that $C1 = \{pr1, pr2\}$, $C2 = \{o, o\}$ and $C3 = \{co\}$. The reachability graph of this problem is given by Fig. 3. Let $M = (p1, p2, p3, p4, p5)$ be the structure of the marking vector. Assuming that $M8 = (pr1, pr2, 0, o, co)$ and $M9 = (pr2, pr1, 0, o, co)$ are the specified forbidden markings. Applying Algorithm 1, we obtain as results the admissibility graph described in Fig. 4 and the set

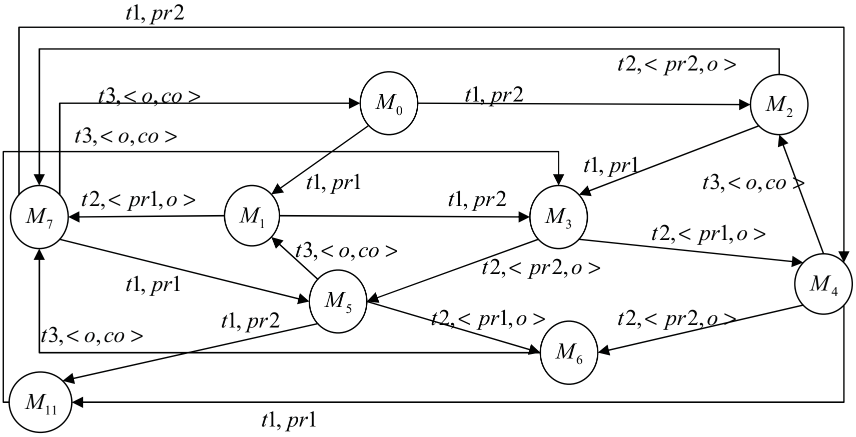


Fig. 4. Admissibility graph of the producer-consumer problem

of state-transitions $\Omega = \{(M6, (t1, pr1)), (M6, (t1, pr2)), (M11, (t2, < pr1, o >)), (M11, (t2, < pr2, o >))\}$.

4.2 Synthesis of the controller

The controller synthesis consists in solving numerous systems similar to those of type (1), (2) and (3) formulated from the admissibility graph. The solutions of obtained systems allow to determine the necessary parameters of the controller which is expressed in term of CP-nets. Thanks to the expressiveness power of CP-nets, the controller is reduced to one single place. Therefore, the necessary parameters allowing to build the controller (a CP-net place) p_c and achieving its connection to the plant model are its colour domain C_{pc}, its initial marking $M_0(p_c)$ and its incidence vector $W(p_c, .)$.
First, we propose to reformulate the equations/inequations of type (1), (2) and (3) in order to deal with CP-nets instead of ordinary Petri nets. Finally, we give the algorithm allowing the synthesis of a controller for a plant CP-net model.

Let us consider the admissibility graph R_c of a plant CP-net model, and the set Ω of state-transitions, and p_c be the controller CP-net place to determine. We denote its colour domain by C_{pc}. Each object of C_{pc} is related to one or several elements of Ω. As it is stated by the theory of regions, the controller place p_c must satisfy the reachability condition. This condition guarantees that every marking in the admissible behaviours remains reachable under control. Let M be a marking of the admissibility graph R_c. The reachability condition related to M is:

$$\forall v \in C_{pc}, M(p_c)(v) = M_0(p_c)(v) + W(p_c, .)(v)\overrightarrow{\Gamma}_M \geq 0 \qquad (4)$$

where $\overrightarrow{\Gamma}_M$ is the vector counting of any non oriented path Γ_M in R_c from M_0 to M.
For instance, we consider the marking M_7 of the admissible graph in Fig. 4. For every $v \in C_{pc}$, the related reachability condition to M_7 is: $M_7(p_c)(v) =$

$M_0(p_c)(v) + W(p_c, t1)(pr1)(v) + W(p_c, t2)(< pr1, o >)(v)$.

Cycle equations ensure that the cycles of R_c must remain reachable under control. Hence, the place p_c has to satisfy the following equation:

$$\forall v \in C_{pc}, \forall \gamma \in S_c, W(p_c, .)(v)\overrightarrow{\gamma} = 0 \tag{5}$$

where S_c denotes the set of cycles of R_c and $\overrightarrow{\gamma}$ is the vector counting of the cycle γ.
In the admissibility graph of Fig. 4, the cycle equation related to the oriented cycle (M_0, M_1, M_7) is expressed by the equation $\forall v \in C_{pc}, W(p_c, t1)(pr1)(v) + W(p_c, t2)(< pr1, o >)(v) + W(p_c, t3)(< o, co >)(v) = 0$.

Finally, a event separation equation associated with an element $(M, (t, c))$ of Ω allows the controller p_c to prevent the firing of t in M with colour c:

$$\forall v \in C_{pc}, M_0(p_c)(v) + W(p_c, .)(v)\overrightarrow{\Gamma}_M + W(p_c, t)(c)(v) < 0 \tag{6}$$

As an example, the disabling of the event separation event $(M6, t1, pr1)$ is ensured through the inequation $\forall v \in C_{cp}, M_0(p_c)(v) + W(p_c, t1)(pr1)(v) + W(p_c, t1)(pr2)(v) + W(p_c, t2)(< pr2, o >)(v) + W(p_c, t2)(< pr1, o >)(v) + W(p_c, t1)(pr1)(v) < 0$.

Our proposed Algorithm 2 builds, in an incremental manner, the controller components. In each iteration of the algorithm, it solves one system where a new object of C_{pc} is introduced as an unknown factor, and a new element of Ω is considered in (6).
Then, we have to *fold the partial solution* $W(p_c, t_i)(c)(v_j)$ in order to determine $W(p_c, .)(v_j)$. Indeed, the colour domain $C(t_i)$ is partitioned into k_i sets E_i^s $(s = 1..k_i)$ such that: $c', c'' \in E_i^s$, iff $W(p_c, t_i)(c')(v_j) = W(p_c, t_i)(c'')(v_j) = \alpha_i^s$. Hence, for every transition t_i, the colour functions associated with the same set E_i^s are grouped. Thus, the folding is achieved as follows:

$$W(p_c, t_i)(v_j) = \sum_{s=1}^{k_i}[\bigvee_{c' \in E_i^s}[\bigwedge_{X \in var(t_i)}(X = X(c'))].\alpha_i^s] \tag{7}$$

```
input  : $R_c$ is the admissibility graph
         $\Omega$ the set of event separation instances
output: $M_0(p_c)$ and $W(p_c, .)$
$j \leftarrow 0$
Compute the basis cycles of $R_c$
Generate the reachability conditions (4)
Compute the independent cycle equations (5)
repeat
    $j \leftarrow j + 1$
    Let $v_j$ be new object Let $(M, (t, c)) \in \Omega$
    $\Omega = \Omega \setminus \{(M, (t, c))\}$
    Generate the event separation condition (6) for $(M, (t, c))$
    Solve the system made up of (4), (5) and (6) after replacing $v$ by $v_j$.
    if there is no solution then
        the algorithm terminates as the legal behaviour can not be enforced
    else
        Remove from $\Omega$ the elements having the same solution
        for every transition $t_i \in T$ do Fold the solution
until $\Omega = \emptyset$ ;
```

Algorithm 2: Synthesis of the controller

Applying Algorithm 2 on the basis of admissibility graph of Fig. 4 and the set $\Omega = \{(M6, (t1, pr1)), (M6, (t1, pr2)), (M11, (t2, < pr1, o >)), (M11, (t2, < pr2, o >))\}$ of state-transitions, we obtain the controller place p_c having as colour domain the set $C_{pc} = \{v1\}$. The controlled model is illustrated by Fig. 5. As an example showing the operation of folding, we consider the partial solutions $W(p_c, t1)(v1)(pr1)$ and $W(p_c, t1)(v1)(pr1)$. Indeed, we have $W(p_c, t1)(v1)(pr1) = W(p_c, t1)(v1)(pr2) = 1$. Consequently, the folding of these two partial solutions according to (7) gives $W(p_c, t1)(v1) = 1[X = pr1 \vee X = pr2]$.

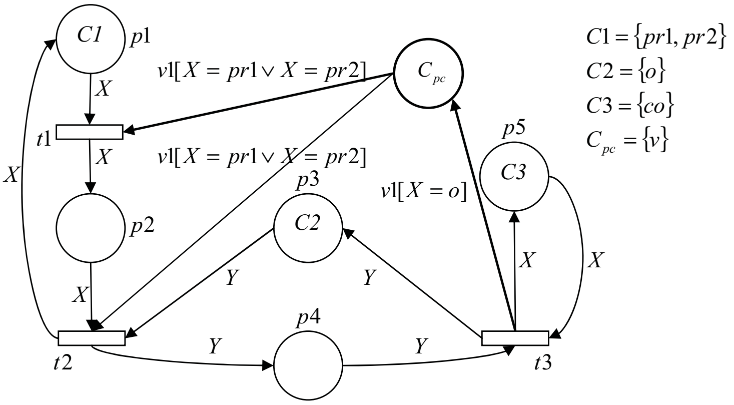


Fig. 5. Controlled model

5. Optimised controller synthesis for symmetric systems

In practice, reachability graphs obtained from CP-nets are often huge, thereby it becomes sometimes impossible to perform the synthesis process. As an attractive solution to alleviate this issue is the use of *Symbolic reachability graphs* (SRGs) instead of ordinary reachability graphs. Indeed, an SRG allows the construction of a reduced representation of the ordinary state space without unfolding of colour sets. Experiments (Daws & Tripakis, 1998) have proven that the size of the symbolic reachability graph is quite small in practice. In addition, the building of an SRG is performed automatically from the structure of a WF-net by exploiting its *behaviour symmetries*.

Following the same steps of the approach described in the previous section, we propose to optimise the controller synthesis for symmetric DES. The optimisation is achieved mainly by applying the theory of regions on the basis of symbolic reachability graphs instead of ordinary reachability graphs. Given a CP-net as a plant model, we build its SRG. Then, a treatment is required in order to produce a unique representation for the arcs of the SRG. From the obtained SRG, we determine the graph modelling the desired behaviours according to the control specifications. Finally, the theory of regions is applied on the basis of the latter graph in order to derive the controller, which is represented by a single place expressed in terms of CP-nets.

5.1 Symbolic reachability graphs

The symbolic reachability graph of WF-net is based on the idea of *symmetry* of objects of the basic colour classes. Intuitively, a behaviour symmetry is the fact to do *not distinguish* the identities of colours that potentially have the same evolving.

For instance, if we consider the CP-net modelling the dining philosophers problem (Chiola et al., 1997), it is not necessary to distinguish the identities of philosophers. Indeed, for *any* philosopher, the associated structural behaviour may be expressed in terms of synchronisation with its right and left neighbours. In this case, philosophers have symmetrical behaviours. In many other classical problems, behaviour symmetries may be obtained from colours representing processes (clients, servers,...) or resources.

A symbolic marking (SM), a node of a SRG, is a marking, the colours of which are gathered into equivalence classes, forgetting the identity of colours but keeping the cardinality of each represented equivalence class. The SMs are constructed using symmetries that are computed without building the ordinary reachability graph.

In well-formed nets, due the restricted operators defined on object classes, it has been proved that symmetrical colours in a given marking cause the same behaviour. The colours which have structurally similar behaviour, i.e. that can be exchanged at any point in the evolution of the system with no impact on the sequences of firable transitions, are grouped into *static subclasses*, which are not modified during the construction.

Let us consider that any C_i class of objects is partitioned into n_i static subclasses:

$$\forall i \in \{1, ..., n\}, C_i = \bigcup_{q=1}^{n_i} D_{i,q}$$

For instance, in our considered problem of producer-consumer, all producers behave symmetrically, thus C_1 corresponds to one static subclass denoted by $D_{1,1}$. Similarly, C_2 corresponds to one static class denoted by $D_{2,1}$, and C_3 corresponds to a static subclass denoted by $D_{3,1}$.

In contrast, a *dynamic subclass* is a subset of a static subclass. It groups colours having the same distribution throughout the places of the WF-net. A dynamic subclass is characterised by its cardinality and by the static subclass to which it belongs. Although the number and cardinality of these dynamic subclasses evolve during the SRG construction, dynamic subclasses always constitute a partition of static subclasses (the producers that are working and those that are waiting for instance). Thus dynamic subclasses concisely represent the permutations that are permitted on an SM without modifying future sequences of firable transitions.
Now, we give the formal definition of an SM. Intuitively, an SM is expressed by a product of dynamic subclasses.

Definition 2. *Let $I = \{1, .., n\}$ be the set of class indexes. A symbolic marking $\mathcal{M}$ is a 4-tuple $R =< m, card, d, marq >$ satisfying:*

- *$m : I \longrightarrow N$, such that $m(i)$ (denoted also m_i) is the number of dynamic subclasses of C_i in $\mathcal{M}$. The set of dynamic subclasses of C_i in $\mathcal{M}$ is $\widehat{C}_i = \{Z_i^j | 0 < j \leq m_i\}$,*
- *$card : (\bigcup_{i \in I} \widehat{C}_i) \longrightarrow \mathbb{N}$,*
- *$d : (\bigcup_{i \in I} \widehat{C}_i) \longrightarrow \mathbb{N}$ such that:*
    1. *$d(Z_i^j)$ is the index q of a static subclass $D_{i,q}$,*
    2. *$\forall i \in I, \forall j, k$ s.t. $0 < i \leq n \wedge 0 < j < k \leq m_i, d(Z_i^j) \leq d(Z_i^k)$*
- *$\forall p \in P, marq(p) : \otimes_{i \in I} (\widehat{C}_i)^{e_i} \longrightarrow \mathbb{N}$ where e_i represents the number of occurrences of C_i in $C(p)$, and $\otimes$ denotes the Cartesian product.*

Moreover, R must satisfy: $\forall M \in \mathcal{M}, \forall i \in I, \exists \psi_i : C_i \longrightarrow \widehat{C}_i$ such that:

1. *$|\psi_i^{-1}(Z_i^j)| = card(Z_i^j)$*
2. *$\forall i \in I, \exists D_{i,q}$ such that $\psi_i^{-1}(Z_i^j) \subseteq D_{i,q}$ and $q = d(Z_i^j)$*
3. *$\forall p \in P, \forall c \in C(p), M(p, \otimes_{i \in I} \otimes_{j=1}^{e_i} c_i^j) = M(p)(\otimes_{i \in I} \otimes_{j=1}^{e_i} \psi_i(c_i^j))$*

In order to illustrate an example of an SM, we consider the SRG of the producer-consumer given in Fig. 6. Let us consider the initial SM $\mathcal{M}_0 = (Z_1^1, 0, Z_2^1, 0, Z_3^1)$ of this SRG. This SM is expressed by the dynamic subclass Z_1^1 in place $p1$, the dynamic subclass Z_2^1 in place $p3$ and Z_3^1 in place p_5. Since the cardinality of Z_1^1 is $|Z_1^1| = 2$, and $d(Z_1^1) = 1$ (Z_1^1 is a subset of $D_{1,1}$), then Z_1^1 represents all the elements of $D_{1,1}$, namely the two tokens $pr1$ and $pr2$. Following the same reasoning for Z_2^1, we conclude that it represents all the elements of the second static subclass $D_{2,1}$. Also, Z_3^1 represents the elements of $D_{3,1}$.
In order to build directly a new SM from a current one, the classical notion of a transition instance is replaced by the notion of *symbolic instance*. It corresponds to a splitting of the dynamic subclasses of the current SM in order to isolate quantities of colours that can be used for the *symbolic firing*. Indeed, in a symbolic firing, instance dynamic subclasses are assigned to the transition parameters instead of objects. When an instance dynamic subclass is assigned to a parameter, it means that any object in the subclass can be assigned to the parameter.
Let $I = \{1, ..., n\}$ be the set of class indexes. Let t be a transition, the colour domain of which is $C(t) = \bigotimes_{i=1}^{n} \bigotimes_{j=1}^{e_i} C_i$. Let $\mathcal{M}$ be a symbolic marking and $\mathcal{R}$ a symbolic representation of $\mathcal{M}$.
We say that $(\bigotimes_{i=1}^{n} \bigotimes_{j=1}^{e_i} Z_i^{\lambda_i(j), \mu_i(j)})$ is a *symbolic instance* for t wrt. $\mathcal{R}$, if and only if:
$\lambda = \{\lambda_i : \{1, ..., e_i\} \rightarrow \mathbb{N}^*\}$ and $\mu = \{\mu_i : \{1, ..., e_i\} \rightarrow \mathbb{N}^*\}$ such that $\forall i \in I, \forall x \in \{1, ..., e_i\}$,

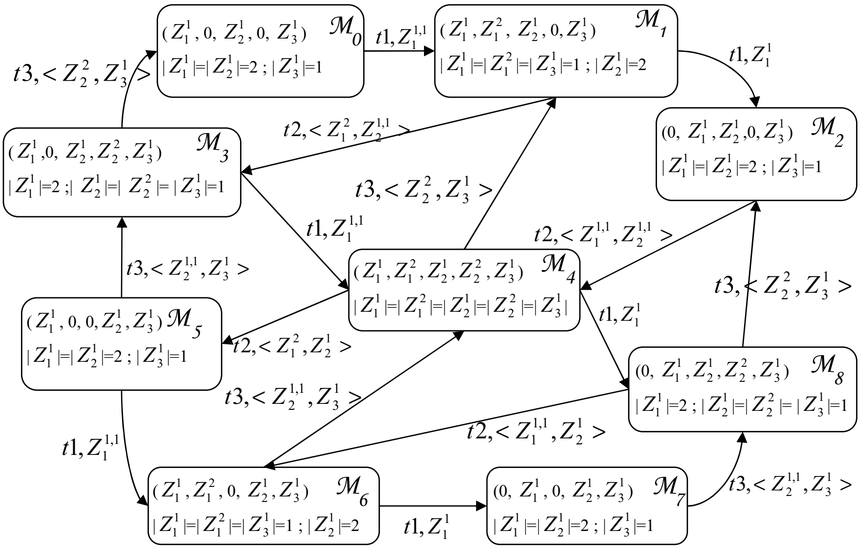

$(Z_1^1, 0, Z_2^1, 0, Z_3^1)$ $\mathcal{M}_0$
$|Z_1^1|=|Z_2^1|=2 ; |Z_3^1|=1$
$t1, Z_1^{1,1}$
$(Z_1^1, Z_1^2, Z_2^1, 0, Z_3^1)$ $\mathcal{M}_1$
$|Z_1^1|=|Z_1^2|=|Z_3^1|=1 ; |Z_2^1|=2$
$t1, Z_1^1$
$(0, Z_1^1, Z_2^1, 0, Z_3^1)$ $\mathcal{M}_2$
$|Z_1^1|=|Z_2^1|=2 ; |Z_3^1|=1$
$t3, < Z_2^2, Z_3^1 >$
$(Z_1^1, 0, Z_2^1, Z_2^2, Z_3^1)$ $\mathcal{M}_3$
$|Z_1^1|=2 ;| Z_2^1|=| Z_2^2|=|Z_3^1|=1$
$t2, < Z_1^2, Z_2^{1,1} >$
$t3, < Z_2^2, Z_3^1 >$
$t1, Z_1^{1,1}$
$t2, < Z_1^{1,1}, Z_2^{1,1} >$
$t3, < Z_2^{1,1}, Z_3^1 >$
$(Z_1^1, Z_1^2, Z_2^1, Z_2^2, Z_3^1)$ $\mathcal{M}_4$
$|Z_1^1|=|Z_1^2|=|Z_2^1|=|Z_2^2|=|Z_3^1|$
$t3, < Z_2^2, Z_3^1 >$
$(Z_1^1, 0, 0, Z_2^1, Z_3^1)$ $\mathcal{M}_5$
$|Z_1^1|=|Z_2^1|=2 ; |Z_3^1|=1$
$t1, Z_1^1$
$t2, < Z_1^2, Z_2^1 >$
$(0, Z_1^1, Z_2^1, Z_2^2, Z_3^1)$ $\mathcal{M}_8$
$|Z_1^1|=2 ; |Z_2^1|=|Z_2^2|=|Z_3^1|=1$
$t3, < Z_2^{1,1}, Z_3^1 >$
$t2, < Z_1^{1,1}, Z_2^1 >$
$t1, Z_1^{1,1}$
$(Z_1^1, Z_1^2, 0, Z_2^1, Z_3^1)$ $\mathcal{M}_6$
$|Z_1^1|=|Z_1^2|=|Z_3^1|=1 ; |Z_2^1|=2$
$t1, Z_1^1$
$(0, Z_1^1, 0, Z_2^1, Z_3^1)$ $\mathcal{M}_7$
$|Z_1^1|=|Z_2^1|=2 ; |Z_3^1|=1$
$t3, < Z_2^{1,1}, Z_3^1 >$

Fig. 6. Symbolic reachability graph of the producer-consumer problem

- $\lambda_i(x) \leq \mathcal{R}.m(i)$,
- $\mu(i) \leq \mathcal{R}.card(Z_i^{\lambda_i(x)})$
- $\forall k \in \mathbb{N}^*$ s.t. $k < \mu_i(x), \exists x' < x$ s.t. $\lambda_i(x') = \lambda_i(x) \wedge \mu_i(x') = k$

If $e_i = 0$, we do not define λ_i and μ_i.

A symbolic instance for a transition is a product of dynamic subclasses. A dynamic subclass may occur several times: if some μ_i values are equal, with respect to the same dynamic subclasses, then the same object is referred. In practice, one can note that λ_i and μ_i are specified only if necessary.

As an example for a symbolic firing, we consider the SRG of fig 6. In SM $\mathcal{M}_0$, two producers are in place $p1$, namely $pr1$ and $pr2$. Since X may be bound to any producer, then two coloured firings are possibles: the firing of transition $t1$ with colour $pr1$ or with colour $pr2$. All elements within a dynamic subclass Z_i^j are fully equivalent, then there is only one way to bind a variable to any Z_i^j, whatever its cardinality. Hence, a single symbolic binding is possible from the SM $\mathcal{M}_0$, X is bound to (a value in) Z_1^1. These possible firings are represented by one symbolic firing inducing an arc labelled with $(t1, Z_1^{1,1})$ such that $Z_1^{1,1}$ represents any object of Z_1^1. The new induced SM by the firing of $t1$ in $\mathcal{M}_0$ is $\mathcal{M}_1 = p1(Z_1^1) + p2(Z_1^2) + p3(Z_2^1) + p5(Z_3^1)$ such that $|Z_1^1| = |Z_1^2| = |Z_3^1| = 1$ and $|Z_2^1| = 2$. Z_1^1 represents an object of $D_{1,1}$, while Z_1^2 represents another object of the same static subclass. Z_2^1 represents the two objects of $D_{2,1}$.

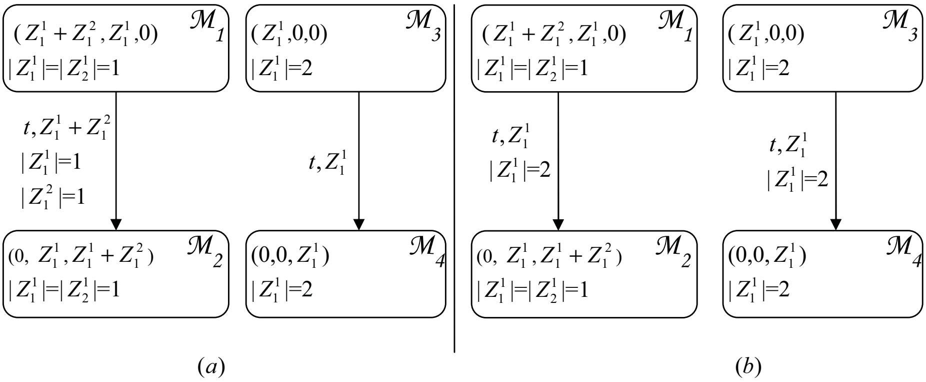

$(Z_1^1+Z_1^2, Z_1^1, 0)$ $|Z_1^1|=|Z_2^1|=1$ $\mathcal{M}_1$
$t, Z_1^1+Z_1^2$ $|Z_1^1|=1$ $|Z_1^2|=1$
$(0, Z_1^1, Z_1^1+Z_1^2)$ $|Z_1^1|=|Z_2^1|=1$ $\mathcal{M}_2$
$(Z_1^1, 0, 0)$ $|Z_1^1|=2$ $\mathcal{M}_3$
t, Z_1^1
$(0, 0, Z_1^1)$ $|Z_1^1|=2$ $\mathcal{M}_4$
(a)
$(Z_1^1+Z_1^2, Z_1^1, 0)$ $|Z_1^1|=|Z_2^1|=1$ $\mathcal{M}_1$
t, Z_1^1 $|Z_1^1|=2$
$(0, Z_1^1, Z_1^1+Z_1^2)$ $|Z_1^1|=|Z_2^1|=1$ $\mathcal{M}_2$
$(Z_1^1, 0, 0)$ $|Z_1^1|=2$ $\mathcal{M}_3$
t, Z_1^1 $|Z_1^1|=2$
$(0, 0, Z_1^1)$ $|Z_1^1|=2$ $\mathcal{M}_4$
(b)

Fig. 7. Canonisation of SRG arcs

5.2 Canonisation of arcs

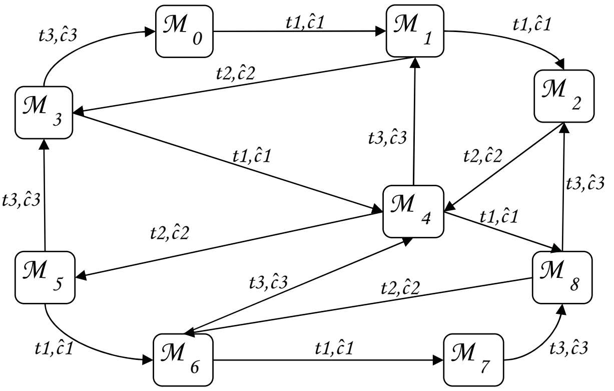
The parameters of the controller are determined by resolving linear system equations formulated from the admissibility graph. Most terms of the formulated equations are determined from the arcs of the admissibility graph. When dealing with SRGs, the main issue is that the same symbolic firing instances can be encoded differently in the graph.

A symbolic firing of a transition t in an SM $\mathcal{M}$ indicates the removed objects from the input places of t in the corresponding arc of the SRG. The removed objects are represented by a product of dynamic subclasses. These dynamic subclasses correspond to partial instances of dynamic subclasses of $\mathcal{M}$. Therefore, the same objects may not have the same representation in arcs. As a consequence, the comparison of two symbolic firings related to the same transition becomes more complex, since it requires the computation of represented objects by the dynamic subclasses labelling the arcs, and then one has to check whether the sets of objects are the same, or not. In order to simplify this operation of comparison, we propose to define a canonical representation of dynamic subclasses labelling arcs induced by symbolic firings. By this way, we guarantee an unique representation for the labels of arcs. This unique representation allows to compare two arcs, induced by the symbolic firing of the same transition, by an equality test between the dynamic subclasses labelling the arcs. Such comparison is performed according to the following three criterions:

- they have the same labels,
- they belong to the same static subclass,
- they have the same cardinalities.

Similarly to the canonisation of SMs, we exploit two properties on dynamic subclasses, called minimality and ordering in order to canonise the representations of arcs.

Given an arc labelled with $(t, \widehat{c})$. $\widehat{c}$ is a tuple indexed by the input places of t, where each component is a product of dynamic subclasses specifying the objects moved from the input places of t. The same algorithms used for the canonisation of SMs can be applied for arcs. Here, we will not present in detail these algorithms, since they can be found in (Chiola et al., 1991). We just note that this computation is organized in two steps:



$$\hat{c}1 = Z_1^1; \left|Z_1^1\right| = 1; d(Z_1^1) = 1$$

$$\begin{cases} \hat{c}2 =< Z_1^1, Z_2^1 >; \\ \left|Z_1^1\right| = \left|Z_2^1\right| = 1; \\ d(Z_1^1) = d(Z_2^1) = 1 \end{cases}$$

$$\begin{cases} \hat{c}3 =< Z_2^1, Z_3^1 >; \\ \left|Z_2^1\right| = \left|Z_3^1\right| = 1; \\ d(Z_2^1) = d(Z_3^1) = 1 \end{cases}$$

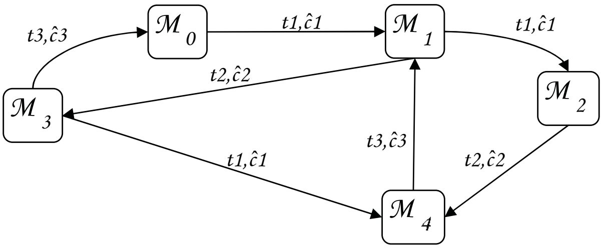
Fig. 8. SRG of consumer-producer problem with canonised arcs

- the computation of a minimal representation. In this step, we compute the smallest number of the dynamic subclasses. Thus, two dynamic subclasses having the same distribution in $\widehat{c}$ are grouped into one dynamic subclass.
- the search of an ordered representation. It consists of readjusting the dynamic subclasses indexes according to an ordering criterion. The ordering criterion has to be defined in a such way that it will set a unique indexing scheme for dynamic subclasses, i.e. it has to be a total order. Here, we choose the lexicographic order on the input places associated with the considered transition.

Figure 7*(a)* gives a subgraph of an SRG illustrating the issue related to representation of arcs. Here, the WF-net of the SRG consists of three places $p1$, $p2$ and $p3$. Its object class contains two objects. Although the two symbolic firings of t in SMs $\mathcal{M}_1$ and $\mathcal{M}_3$ are identical, they are represented differently. Indeed, a symbolic firing of transition t is performed by removing two objects from its input place $p1$ and transferring them into place $p3$. In marking $\mathcal{M}_1$ the two objects in place $p1$ are represented by the sum $Z_1^1 + Z_1^2$. Consequently, the firing of t in $\mathcal{M}_1$ induces an arc labelled with $(t, Z_1^1 + Z_1^2)$. In marking $\mathcal{M}_3$ the two objects in place $p1$ are represented by one dynamic subclass Z_1^1 such that $|Z_1^1| = 2$. Firing of t in $\mathcal{M}_3$ induces an arc labelled with (t, Z_1^1). Performing the canonisation of arcs, we obtain the same label for the two arcs as it is illustrated by Fig. 7*(b)*. Now let us consider the SRG of Fig. 6. Canonising the arcs of this graph gives the graph shown in Fig. 8. The latter graph will be used in order to determine the controlled behaviour respecting the control specification.

5.3 Synthesis of controller based on SRGs

Given an SRG with canonised arcs of a plant model, the determination of a controller is ensured through three steps. First, one has to provide the control specifications. According to these specifications, we build, in a second step, a symbolic graph implementing the desired behaviour from the SRG. Finally, in third step, we apply the theory of regions on the basis of the latter graph in order to determine the controller which is represented by one CP-net place.



$$\hat{c}1 = Z_1^1; \left|Z_1^1\right| = 1; d(Z_1^1) = 1$$

$$\begin{cases} \hat{c}2 =< Z_1^1, Z_2^1 >; \\ \left|Z_1^1\right| = \left|Z_2^1\right| = 1; \\ d(Z_1^1) = d(Z_2^1) = 1 \end{cases}$$

$$\begin{cases} \hat{c}3 =< Z_2^1, Z_3^1 >; \\ \left|Z_2^1\right| = \left|Z_3^1\right| = 1; \\ d(Z_2^1) = d(Z_3^1) = 1 \end{cases}$$

Fig. 9. Admissibility symbolic graph

Step 1 - Control specifications

In a forbidden state problem, the control specifications are defined by providing a set of undesirable (forbidden) states. The provided states will be used to extract the desired behaviours from the SRG of the plant model. Therefore, the undesired states have to be expressed in terms of SMs corresponding to nodes of the considered SRG. Such specifications fit well with the nature of symmetric systems made up of several components behaving similarly.

We consider the SRG of Fig. 6. As control specifications, we assume that one aims to restrict the stock capacity to one object. Such specifications can be symbolically expressed by providing the undesirable SMs $\mathcal{M}_5$, $\mathcal{M}_6$ and $\mathcal{M}_7$.

Step 2 - Computing the desired behaviours

Here, the desired behaviours are implemented by a subgraph of the SRG, called the *admissibility symbolic graph*. Algorithm 1 can be easily adapted to SRGs in order to compute the admissibility symbolic graph from an SRG and the set Ω_{SM} of SM-transitions. In a SRG, the set Ω_{SM} is used instead of the set Ω of state-transitions. Each element of Ω_{SM} is couple $(\mathcal{M}, (t, \widehat{c}))$ where $\mathcal{M}$ is a *dangereous SM* in which the symbolic firing of transition $(t, \widehat{c})$ must be forbidden, i.e. t must be prevented from firing in M with every colour $c \in \widehat{c}$. The main change to be made for Algorithm 1 is to deal with SMs (resp. symbolic instances) instead of ordinary markings (resp. coloured firings).

For example, let us consider the SRG of Fig. 6 and the control specifications corresponding to the restriction of the stock capacity to one object. Computing the admissibility symbolic graph, we obtain the graph illustrated by Fig. 9 and the set of SM-transitions $\Omega_{SM} = \{(\mathcal{M}_4, (t1, \widehat{c}1)), (\mathcal{M}_8, (t2, \widehat{c}2))\}$.

Step 3 - Construction of the controller

In this final step, we construct the controller by applying the theory of regions on the basis of the admissibility symbolic graph. Using CP-nets allows to represent the obtained controller by one CP-net place p_c which will be connected to the plant model. For this, we have to determined its colour domain C_{pc}, its initial marking $M_0(p_c)$ and its incidence vector $W_c(p_c, .)$. Using a symbolic graph requires the reformulation of the three classes of equations/inequations (4), (5) and (6) according to the structure of the symbolic graphs. Indeed, in an ordinary reachability graph, these equations are deduced from the basic relation between two reachable markings.

Let $v \in C_{pc}$. Let M,M' be two reachable markings. Let t be a transition enabled in M for $c \in C(t)$, such that $M[t,c\rangle M'$, then:
$\forall v \in C(p_c)$,

$$M'_c(p_c)(v) = M_c(p_c)(v) + W_c(p_c,t)(c)(v) \tag{8}$$

where $M'_c(p_c)(v)$ is the number of v in place p_c at the marking M', $M_c(p_c)(v)$ is the number of v in place p_c in M and $W_c(p_c,t)(c)(v)$ is the incidence value associated with place p_c and transition t for the colour $(c)(v)$.
Further, we have $W_c(p_c,t)(c)(v)$=$W_c(p_c,t)(c')(v), \forall c' \in C(t); \widehat{c'} = \widehat{c}$.
Thus, there exists two symbolic markings $\mathcal{M}$ and $\mathcal{M}'$ such that $M \in \mathcal{M}$, $M' \in \mathcal{M}'$ and $\mathcal{M}[[t,\widehat{c}\rangle\rangle\mathcal{M}'$. Thus, the equation (8) can be written as follows:

$$M'_c(p_c)(v) = M_c(p_c)(v) + W_c(p_c,t)(\widehat{c})(v) \tag{9}$$

Therefore, the three classes of equations depicted by the theory of regions can be reformulated for an SRG as follows:

- The reachability conditions (4) indicate that every reachable SM must remain reachable under control. $\forall \mathcal{M} \in R_c$,

$$M(p_c)(v) = M_0(p_c)(v) + W(p_c,.)(v)\overrightarrow{\Gamma}_{\mathcal{M}} \geq 0 \tag{10}$$

  where M is an ordinary marking represented by $\mathcal{M}$, $M(p_c)(v)$ is the occurrence of a given object v in the marking of p_c, $W(p_c,.)(v)$ is the incidence vector relatively to the object v, $\Gamma_{\mathcal{M}}$ is any non oriented path in R_c from $\mathcal{M}_0$ to $\mathcal{M}$, and $\overrightarrow{\Gamma}_{\mathcal{M}}$ is its associated vector. $\overrightarrow{\Gamma}_{\mathcal{M}}$ is indexed by transitions of T and is called the *vector counting* of $\Gamma_{\mathcal{M}}$. Each line of $\overrightarrow{\Gamma}_{\mathcal{M}}$ represents the sum of occurrence number of $(t_i,\widehat{c})$ such that $c \in C(t_i)$ and $(t_i,\widehat{c})$ labels an arc in the considered path. Formally:

$$\overrightarrow{\Gamma}_{\mathcal{M}} = \begin{pmatrix} \Gamma^1_{\mathcal{M}} \\ : \\ \Gamma^m_{\mathcal{M}} \end{pmatrix}$$

  where $\Gamma^i_{\mathcal{M}} = \sum_{(t_i,\widehat{c})\in\Gamma} \alpha_i(\widehat{c}).\widehat{c} \ \forall i \in \{1,...,m\}$ where $\alpha_i(\widehat{c})$ is the occurrence number of $(t_i,\widehat{c})$ in the path $\Gamma_{\mathcal{M}}$ and which is determined from R_c.

- the cycle equations (5) indicate that the cycles of the admissibility graph R_c must remain reachable under control. In other words, the place p_c should satisfy (for every object v of C_{pc}):

$$W(p_c,.)\overrightarrow{\gamma} = 0, \forall \gamma \in S_c \tag{11}$$

  where S_c is the set of cycles of the admissibility graph R_c. $\overrightarrow{\gamma}$ is a vector counting of a cycle γ defined similarly as $\Gamma_{\mathcal{M}}$. It is worth to note that according to well known results of graph theory, the cycle equations can be reduced to independent cycle equations of basis cycles (Schrijver, 1986).

- an event separation condition (6) indicates that the control must prevent from firing a state transition of Ω. In other words, the place p_c and an occurrence object v of C_{pc}, must solve at least one event separation instance $(\mathcal{M},(t,\widehat{c}))$ from Ω:

$$M_0(p_c)(v) + W(p_c,.)(v)\overrightarrow{\Gamma}_{\mathcal{M}} + W(p_c,t)(v)(\widehat{c}) < 0 \tag{12}$$

In order to derive a controller, Algorithm 2 can be applied such that we replace respectively the admissibility graph and set of state-transitions by the admissibility symbolic grah and the SM-transitions. In addition, here the folding is performed differently, since each partial solution, obtained for a new object v_j in an iteration, consists of $M_{pc}(v_j)$ and $\forall t_i \in T, \forall \widehat{c} \in \widehat{C}(t_i); W(p_c, t_i)(\widehat{c})(v_j)$. The aim is to determine for every transition t_i the incidence vector $W(p_c, t_i)(v_j)$. Hence, we propose to achieve the folding of a partial solution through the following two steps for every transition t_i:

- We determine for every colour c represented by $\widehat{c} \in \widehat{C}(t_i)$ the colour function $W(p_c, t_i)(c)(v_j)$ which is equal to $W(p_c, t_i)(\widehat{c})(v_j)$,
- We partition the colour domain $C(t_i)$, of a transition t_i, into k_i sets E_i^s, $(s = 1..k_i)$ such that $c, c' \in E_i^s$, iff $W(p_c, t_i)(c)(vj) = W(p_c, t_i)(c')(v_j) = \alpha_i^s$. Then, the incidence function for the transition t_i is

$$W(p_c, t_i)(v_j) \leftarrow \sum_{s=1}^{k_i} \left[\bigvee_{c' \in E_i^s} \left[\bigwedge_{X \in var(t_i)} (X = X(c')) \right] \right] \alpha_i^s$$

where $var(t_i)$ is the set of variables appearing on arcs that have t_i as source or destination.

Let us consider the admissibility graph shown in Fig. 9 and the set $\Omega_{SM} = \{(\mathcal{M}_4, (t1, \widehat{c}1)), (\mathcal{M}_8, (t2, \widehat{c}2))\}$. The obtained controller consists of one place p_c as it is shown in Fig. 10. One can easily check that the symbolic graph of the controlled model is exactly the admissibility symbolic graph of the plant model.

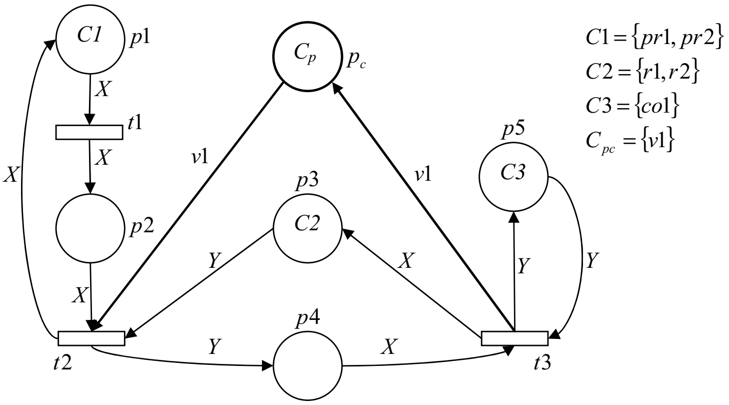


Fig. 10. Controlled model

6. Active controller

Considering as a plant model a CP-net that is assumed to be structured on a set of generic processes sharing a set of resources, i.e. a resource allocation system, we present the "Active Controller" approach allowing to simplify the design of a generic CP-net controller for a such system. In previous sections, we have used the theory of regions in order to derive a controller for a DES modelled by a CP-net. The use of the theory of regions requires the resolution of numerous systems allowing the determination of the necessary parameters of the controller. The ``Active Controller´´ approach was introduced to avoid solving such systems. The key idea of this approach is that the controller must be able to handle enough information to detect reaching a dangerous state, and from which, it removes appropriate authorisations in order to disable the firing of some transitions.

The major difference, from previous approaches, lies in the synthesis of the controller. In the ``Active Controller´´ approach, the controller is characterised by a CP-net subnet (with a fixed structure (4 places, 2 transitions)) representing its behaviour. The variable part (basic colour sets, initial marking and some arc expressions connecting the controller CP-net to the original CP-net) of this controller CP-net depends on the specificity of the studied system and it is generated from the control specification. More precisely, it is defined based on the set Ω of state-transitions. So, the same model can be used in several applications. Thus, the controller synthesis may be considered as parametrable.

The active controller acts as priority process having two states. It is either in a ``Monitor´´ state in which it observes the evolution of the original network, or in an ``Alert´´ state from which it inhibits the firing of some transitions. The controller permanently maintain, in a dedicated place, the current marking of the original system. The controller enters the ``Alarm´´ state, if it detects, based on an appropriate marking of an additional place, that the original system has reached a dangerous state. When entering the ``Alarm´´ state, the controller removes specific marking from an appropriate additional place to disable the firing of the forbidden transitions associated with this global state. The controller lets the original system to evolve towards an admissible state. Once the overall current state is changed, the controller leaves the ``Alarm´´ state.

The functioning of the controller is based on the higher priority associated with its transitions. Such a priority enables it to preempt the transitions of the original system.

The ``Active Controller´´ approach is based on two steps:

- the admissibility computation: This computation is done as explained in section 4.1.
- the construction of the controller: this step allows the generation of the controller CP-net and its connection to the plant CP-net model.

Fig. 11 describes the inputs and the outputs of each step. The following section introduces the controller construction method. The determination of the admissible behaviour has already been presented in section 4.1.

6.1 Controller construction method

At the beginning of this step, we assume that we have already generated the set Ω of state-transitions by Algorithm 1. Based on this information, we generate the controller, modelled by a CP-net, and we define its connection to the plant CP-net model. The synthesis of a such controller may be automatised. It is worth noting that the generated CP-net is autonomous and does not require external devices to ensure the control.

At this level, we handle two kinds of specifications:

- the control specification. More precisely:
  - the set of dangerous markings (denoted DM),

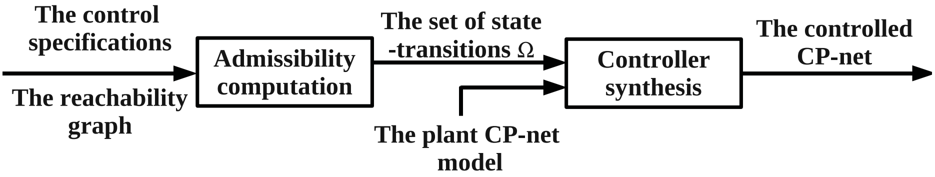


Fig. 11. ˋActive Controller˝ steps

  - the set of forbidden state-transitions associated with each dangerous marking d ∈ DM (denoted FT(d)). FT(d) may be viewed as an application from DM to the set of subsets of T.

- the uncontrolled system specification represented by a CP-net N such that: $N =< P, T, C_l, C, W^-, W^+, \Phi, M_0 >$.

The output of this step is a new CP-net such that its functioning automatically satisfies the control specifications. As previously explained, the key idea of the ˋActive Controller˝ is that the controller must handle enough information to detect the reaching of a dangerous state. Further, in a dangerous marking, we remove appropriate authorisations by disabling the firing of some coloured transitions, called *forbidden coloured transitions*. This method relies on the following points:

- The information related to dangerous states and their associated forbidden transitions are defined by the initial marking of a specific place. This marking is computed on the basis of the set Ω.
- The current state (marking) of the plant model is handled in a special added place. This information is modelled by a composed token (tuple of colours) in accordance with the CP-nets semantics.
- An authorisation is associated with each coloured forbidden transition to enable its firing. All authorisations are managed in a third added place.
- The generated controller CP-net has two supplementary transitions. The first one is fired when a dangerous state is detected, while the second is fired when the dangerous state is quitted. These two controller transitions must be immediately fired when enabled to remove or replace the appropriate authorisations. Thus, they must have the highest priority over all the other transitions.
- The necessary additional CP-net components (colour functions, synchronisation arcs, markings, etc.) must be also defined to ensure the desired management of the controller.

In the following, we formally detail the generation of the controlled CP-net model.
The model representing the system under control is a CP-net N^* obtained from N so that $N^* =< P^*, T^*, C_l{}^*, C^*, W^{*-}, W^{*+}, \Phi^*, M^*{}_0 >$ where:
$P^* = P \cup \{$CM, DaM, AT, AS$\}$, with:

CM representing the Current Marking,
DaM representing the Dangerous Markings,
AT representing the Authorisations for forbidden Transitions, and
AS representing the Alert State of the controller.

$T^* = T \cup \{\text{A-In, A-Out}\}$, with:

A-In representing entering the alert state,
A-Out representing quitting the alert state,
A-In and A-Out have the highest priority.

$C_l{}^* = C_l \cup \{C_{num}, C_{FT}\}$, with:

$C_{num} = \{0, 1, 2, , MaxInt\}$ is a class representing a set of finite positive integers. Its elements will model the occurrence of some given tokens. We assume MaxInt large enough to be greater than the bound of the maximum occurrences of any token in a reachable marking. As, we deal with bounded CP-nets, this property holds,

C_{FT} is a class representing all coloured forbidden transitions. The different element of this class will be defined based on the application FT.

Each of the added controller place is characterised by a specific colour and by an initial marking. Those parameters are generated as follows:

Place CM has a complex colour domain that is a Cartesian product of C_{num} performed on the basis of the number of process classes, the number of places per process and the resource class. The role of CM is to handle information about the current state of the controlled system.
The token marking of CM is a long tuple made up of counters where each one holds the information about the occurrence of tokens in a given place (according to the lexical order) among process places and the occurrence of tokens in the resource place. The colour domain of CM will be defined as follows:

$$C^*(\text{CM}) = \bigotimes_{i=1}^{|C_R|} C_{num} \bigotimes_{j=1}^{|C_P|} \bigotimes_{k=1}^{|NbP(P_j)|} C_{num}$$

Where:
* C_R represents the resource class,
* C_P represents the different types of processes,
* $NbP(P_j)$ defines the number of places associated with the process type P_j. CM is always mono-marked and its initial marking M_0^*(CM) is performed on the basis of the initial marking of the CP-net associated with the studied plant. M_0^*(CM) may be algorithmically determined.

The colour domain of the place DaM is:

$$C^*(\text{DaM}) = C^*(\text{CM}) \times C_{FT}$$

The initial marking of DaM is not updated, since this place is only read accessed. The number of tokens in DaM is equal to $\sum\limits_{d \in DM} |FT(d)|$

The colour domain and the initial marking of the place AT are: C^*(AT)= C_{FT} , and M_0^*(AT) = C_{FT}. Indeed, initially, all forbidden transitions are authorised.

The colour domain of the place AS is: C^*(AS)= C^*(CM). Initially this place is empty M_0^*(AS) = $\varnothing$.

Finally, the colour functions of the arcs connecting the controller places to a subset of T and to the transitions A-In and A-Out must be defined.

As the role of CM is to hold the current marking of N, it is connected to every transition of T using an input arc (reading marking) and an output arc (updating marking).

$\forall t \in T, W^{*-}$(CM,t)=$< X_{1,1}, ..., X_{k,xk}, Y_1, ..., Y_u > = < X >$

where $X_{i,j}$ is a variable defined on C_{num} computing the number of tokens in the place i of the process type j and Y_u is a variable defined on C_{num} reading the occurrence of colour u in the resource places;

$\forall t \in T, W^{*+}$(CM,t)= $< X'_{1,1}, ..., X'_{k,xk}, Y'_1, ..., Y'_u > = < X' >$

where $X'_{i,j}$ and Y'_u are variables defined on C_{num} and determined as follows :

$X'_{i,j}$=$X_{i,j} - \chi$, with χ= $W^+(p_{ij}, t) - W^-(p_{ij}, t)$,

and

Y'_u= $Y'_u - \xi$, where ξ is computed as follows:

$$\left.\begin{array}{l} W^-(r,t) = \sum\limits_i \alpha_i . r_i \\ W^+(r,t) = \sum\limits_i \alpha'_i . r_i \end{array}\right. \Rightarrow \xi = \alpha'_u - \alpha_u$$

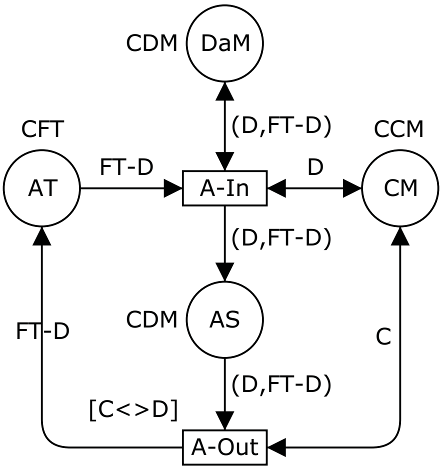


Fig. 12. The ˵Active Controller˝ subnet

The place AT is connected to every forbidden transition by one input/output arc in order to check the presence of the associated firing authorisation:

$\forall t \in C_{FT}$, W^{*+}(AT, t)= W^{*-}(AT, t)= $< Xt >$,

where Xt is defined on C_{FT} and represent the identity of the coloured forbidden transition.

The colour functions of the arcs connecting the controller places to the transition A-In and A-Out are defined as follows:

W^{*-}(DaM, A-In)=W^{*+}(DaM, A-In)=$< D, FT\text{-}D >$;
W^{*-}(CM, A-In)=W^{*+}(CM, A-In)=$< D >$;
W^{*-}(AT, A-In)=$< FT\text{-}D >$; W^{*+}(AS, A-In)=$< D, FT\text{-}D >$;
W^{*+}(AS, A-Out)=$< D, FT\text{-}D >$; W^{*-}(CM, A-Out) =$< C >$;
W^{*+}(AT, A-Out)=$< FT\text{-}D >$;

D and $C \in$ C(CM) (i.e. they are a tuple of variables) , $FT\text{-}D \in C_{FT}$
Transition A-Out is associated with the predicate: $[C \neq D]$

Fig. 12 represents the CP-net modelling the controller behaviour. It is worth to note that the controller CP-net is connected to the plant CP-net model through the places AT and CM as it was previously defined.
In the next section, we apply the generation of the controller to our problem of producer-consumer.

6.2 Example

We consider the previously introduced producer-consumer problem modelled by the WF-net of Fig. 2. We assume that $C1 = \{2pr\}$, $C2 = \{o1, o2\}$ and $C3 = \{co\}$. The reachability graph of this problem is given by Fig. 13.

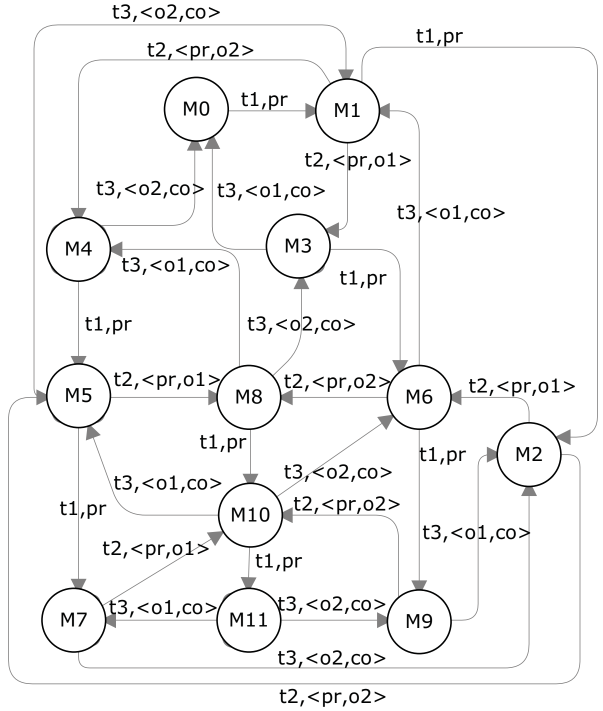


Fig. 13. The reachability graph

Assuming that the stock must contain at most one object, such a specification induces that $M8$, $M10$ and $M11$ are the forbidden markings. Applying Algorithm 1, we obtain the admissibility graph described in Fig. 14 and the set of state-transitions $\Omega = \{(M5,(t2,< pr,o1 >)),(M6,(t2,< pr,o2 >)),$
$(M7,(t2,< pr,o1 >)),(M9,(t2,< pr,o2 >))\}$
Let us illustrate the different variable elements of the Active Controller associated with the considered producer-consumer system.

C^*(CM)= $C_{num} \times C_{num} \times C_{num} \times C_{num} \times C_{num}$
The first and the second elements respectively define the number of producers in places $p1$ and $p2$. The third element computes the number of consumers in place $p5$. The fourth (respectively the fifth element) handles the number of objects of type $o1$ (respectively $o2$) in place $p4$.

C_{FT}={$t2PrO_1$, $t2PrO_2$}
$t2PrO_1$ (respectively $t2PrO_2$) represents the authorisation to fire transition $t2$ with colour $< pr,o_1 >$ (respectively $< pr,o_2 >$).

M_0^*(CM)=<2,0,1,0,0>
M_0^*(DaM)=<<1,1,1,0,1>,$t2PrO_1 >$ + <<1,1,1,1,0>,$t2PrO_2 >$ + <<0,2,1,0,1>,$t2PrO_1 >$

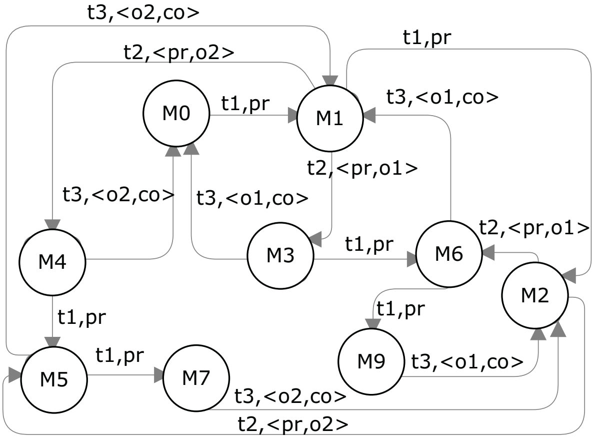


Fig. 14. The admissibility graph

+ <<0,2,1,1,0>,$t2PrO_2$ >
M_0^*(AT)= $t2PrO_1$ + $t2PrO_2$
$\forall t \in T$, W^{*-}(CM,t)=$< X_{1,1}, X_{1,2}, X_{2,1}, Y_1, Y_2 >$

W^{*+}(CM,t1$\{pr\}$)=$< X_{1,1} - 1, X_{1,2} + 1, X_{2,1}, Y_1, Y_2 >$
W^{*+}(CM,t2$\{< pr, o1 >\}$)=$< X_{1,1} + 1, X_{1,2} - 1, X_{2,1}, Y_1 + 1, Y_2 >$
W^{*+}(CM,t2$\{< pr, o2 >\}$)=$< X_{1,1} + 1, X_{1,2} - 1, X_{2,1}, Y_1, Y_2 + 1 >$
W^{*+}(CM,t3$\{< o1, co >\}$)=$< X_{1,1}, X_{1,2}, X_{2,1}, Y_1 - 1, Y_2 >$
W^{*+}(CM,t3$\{< o2, co >\}$)=$< X_{1,1}, X_{1,2}, X_{2,1}, Y_1, Y_2 - 1 >$

7. Conclusion

In this chapter, we have dealt with the control of DES, modelled by CP-nets, for the problem of forbidden states. The use of CP-nets allows compact models even for large and complex systems. The first approach, based on the theory of regions, can be addressed to any kind of DES modelled by CP-nets. Considering a CP-net as plant model, in a first step of this approach, the graph implementing the desired behaviours is determined from the reachability graph of the considered DES according to the control specifications. Then, the theory of regions is applied in order to design the controller. Thanks to the expressiveness of CP-nets, the obtained controller is represented by one single place. In a second approach, we propose to cope with the combinatorial explosion of state space for symmetric systems. Indeed, the state space of a symmetric system can be represented by a condensed version, the symbolic reachability graph, which is quite smaller. Following similar steps as the first approach, the second approach allows to deal efficiently with symmetric systems. Indeed, the theory of region is applied on the basis of a symbolic reachability graph instead of the ordinary one. Finally, third approach avoids the use of the theory of regions which requires the resolution of numerous linear systems in order to determine the controller. Indeed, the generated controller is an active process, modelled by a generic CP-net, that permanently observes the plant model to detect the reaching of dangerous states, and then it removes appropriate authorisations.

8. References

Abid, C. & Zouari, B. (2008). Synthesis of controllers using symbolic reachability graphs, *Proceedings of 9th International Workshop of Discrete Event Systems (WODES'08)*, Goteborg, pp. 314–321.

Badouel, E., Bernardinello, L. & Darondeau, P. (1995). Polynomial algorithms for the synthesis of bounded nets, *Proceedings of the 6th International Joint Conference CAAP/FASE on Theory and Practice of Software Development*, Vol. 915, Lecture Notes In Computer Science, Aarhus, pp. 364–378.

Chiola, G., Dutheillet, C., Franceschinis, G. & Haddad, S. (1991). On well-formed coloured nets and their symbolic reachability graph, *in* K. Jensen & G. Rozenberg (eds), *High-Level Petri Nets – Theory and Application*, Springer, pp. 373–396.

Chiola, G., Dutheillet, C., Franceschinis, G. & Haddad, S. (1997). A symbolic reachability graph for coloured Petri nets, *Theoretical Computer Science* **176**(1-2): 39–65.

Daws, C. & Tripakis, S. (1998). Model checking of real-time reachability properties using abstractions, *TACAS*, pp. 313–329.

Ghaffari, A., Rezg, N. & Xie, X. (2003). Design of live and maximally permissive petri net controller using the theory of regions, *Proceedings of IEEE Transactions on Robotics and Automation*, Vol. 19, Aarhus, pp. 137–142.

Giua, A. & DiCesare, F. (1994). Petri net structural analysis for supervisory control, *IEEE Transactions on Robotics and Automation* **10**(2): 185–195.

Holloway, L. E., Krogh, B. H. & Giua, A. (1997). A survey of petri net methods for controlled discrete eventsystems, *Discrete Event Dynamic Systems* **7**(2): 151–190.

Jensen, K., Kristensen, L. M. & Wells, L. (2007). Coloured petri nets and cpn tools for modelling and validation of concurrent systems, *Int. J. Softw. Tools Technol. Transf.* **9**(3): 213–254.

Jensen, K. & Rozenberg, G. (1991). *High-Level Petri Nets: Theory and Application*, Springer Verlag.

Makungu, M., Barbeau, M. & St-Denis, R. (1999). Synthesis of controllers of process modeled as coloured petri nets, *Journal Discrete Event Dynamic Systems Theory Applications Kluwer Academic Publishers* **Vol. 9**(No. 2): 147–169.

Ramadge, P. & Wonham, W. (1989). The control of discrete event systems, *Proceedings of IEEE*, Special Issue on Discrete Event Dynamic Systems, pp. 81–98.

Schrijver, A. (1986). *Theory of Linear and Integer Programming*, John Wiley and Sons, NY.

Sreenivas, S. & Sreenivas, R. S. (1997). On the existence of supervisory policies that enforce liveness in discrete event dynamic systems modeled by controlled petri nets, *IEEE Transactions on Automatic Control* **42**: 94–5.

Su, H. Y., Wu, W. M. & Chu, J. (2005). Liveness problem of petri nets. supervisory control theory for discrete event systems, *ACTA AUTOMATICA SINICA* **31**(1): 143–150.

Zouari, B. & Ghedira, K. (2004). Synthesis of controllers using coloured petri nets and theory of regions, *Proceedings of IFAC Workshop on Discrete Event Systems (WODES'04)*, Reims, pp. 231–236.

Zouari, B. & Zairi, S. (2005). Synthesis of active controller for resources allocation systems, *Proceedings of Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools (CPN'05)*, pp. 79–98.

Petri Nets Applications
Edited by Pawel Pawlewski

ISBN 978-953-307-047-6
Hard cover, 752 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Petri Nets are graphical and mathematical tool used in many different science domains. Their characteristic features are the intuitive graphical modeling language and advanced formal analysis method. The concurrence of performed actions is the natural phenomenon due to which Petri Nets are perceived as mathematical tool for modeling concurrent systems. The nets whose model was extended with the time model can be applied in modeling real-time systems. Petri Nets were introduced in the doctoral dissertation by K.A. Petri, titled "„Kommunikation mit Automaten" and published in 1962 by University of Bonn. During more than 40 years of development of this theory, many different classes were formed and the scope of applications was extended. Depending on particular needs, the net definition was changed and adjusted to the considered problem. The unusual "flexibility" of this theory makes it possible to introduce all these modifications. Owing to varied currently known net classes, it is relatively easy to find a proper class for the specific application. The present monograph shows the whole spectrum of Petri Nets applications, from classic applications (to which the theory is specially dedicated) like computer science and control systems, through fault diagnosis, manufacturing, power systems, traffic systems, transport and down to Web applications. At the same time, the publication describes the diversity of investigations performed with use of Petri Nets in science centers all over the world.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821