A Reversibility Enforcement Approach for Petri Nets Using Invariants

Hanife Apaydın Özkan
Anadolu University
Electrical and Electronics Engineering
26555 Eskisehir
Turkey
hapaydin1@anadolu.edu.tr

Aydın Aybar
Anadolu University
Electrical and Electronics Engineering
26555 Eskisehir
Turkey
aaybar@anadolu.edu.tr

Abstract: Petri net model which is one of the most common modelling method of discrete event systems, is considered to enforce reversibility in this work. Reversibility guarantees that the intial state is reachable from any state in the reachability set of given Petri net. An approach, enforcing reversibility, is presented in this work. In this approach, the minimal T-invariants and the firing sequences coressponding to the determined T-invariants are determined. Then, a set of markings, which is a subset of reachability set, is constructed by using those firing sequences. In this set, any state can reach to the initial state. Furthemore, the algorithms are developed for the presented enforcement approach and implemented by using Matlab.

Key–Words: Discrete event systems, Petri nets, Invariants, Reversibility

1 Introduction

[1] Petri net model is frequently used for modeling and analysing discrete event systems ([1, 2, 5, 7]). Discrete event systems (DES) is that they consist of interacting nodes. Each node can be a system in itself and may be thought of as a component of the DES. These components can operate concurrently, i.e., a component can be performing one of its functions at the same time that another component is carrying out one of its respective functions. DES can be described in a precise, unambiguous manner by using Petri net model [6, 21, 22, 23].

Petri net model is introduced by Carl Adam Petri (1962). A Petri net consists of places, transitions, and arcs that connect them. There are other types of arcs, e.g. inhibitor arcs. Tokens are located into places; the current state of the modeled system (the marking) is given by the number of tokens in each place. Transitions are active components. They model activities which can occur (the transition fires), thus changing the state of the system (the marking of the Petri net). Transitions are only allowed to fire if they are enabled, which means that all the preconditions for the activity must be fulfilled (there are enough tokens available in the input places). When the transition fires, it removes tokens from its input places and adds some at all of its output places.

Petri nets are a promising tool for describing and studying systems that are characterized as being concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic. As a graphical tool, Petri nets can be used as a visual-communication aid similar to flow charts, block diagrams, and networks. In addition, tokens are used in these nets to simulate the dynamic and concurrent activities of systems. As a mathematical tool, it is possible to set up state equations, algebraic equations, and other mathematical models governing the behavior of systems.

One of the most important property of Petri nets is reversibility. When the reversibility is satisfied, the system can move back to an initial state from any reachable state. In this work reversibility and its enforcement is considered.

Many works which use reachability set to analyse reversibility (for example, [14, 13, 11, 8] and references therein) have been presented for several types of Petri nets. Since the computational complexity is, in general, exponentially related to number of places and transitions [11], construction of the reachability set needs an additional effort in the sense of computational time (see, [20]).

In [3], a relationship between reversibility and T-invariants, which is a structural analysis method, was given. Since a simple linear equation is solved to find T-invariants, the computational complexity for determination of T-invariants can be neglected. Moreover,

[1]This work is extended version of the paper ”Reversibility enformencent of Petri nets using T-invariants”, which is presented in ACMOS’2008.

this relationship is also valid for unbounded Petri nets. By this motivation, in this paper we used this relationship to analyse and enforce reversibility for bounded / unbounded Petri nets. Furthermore, a controller is used for reversibility enforcement.

2 Petri Net Model

A Petri net is denoted by a tuple $G(P, T, N, O, m_0)$, where P is the set of places, T is the set of transitions, $N : P \times T \rightarrow \mathcal{N}$ is the input matrix that specifies the weights of arcs directed from places to transitions, $O : P \times T \rightarrow \mathcal{N}$ is the output matrix that specifies the weights of arcs directed from transitions to places, where $\mathcal{N}$ is the set of non-negative integer numbers, and m_0 is the initial marking.

$M : P \rightarrow \mathcal{N}$ is a *marking vector*, $M(p)$ indicates the number of *tokens*, represented by black dots, assigned by marking M to place p. A transition $t \in T$ is *enabled* if and only if $M(p) \geq N(p, t)$ for all $p \in P$. Here, $N(p, t)$ corresponds, the element of the input matrix, to $p \in p$ and $t \in t$. An enabled transition $t \in T$ may *fire* at M, yielding the new marking vector:

$$M'(p_i) = M(p_i) + O(p_i, t) - N(p_i, t), \ \forall p \in P \quad (1)$$

where M' is new marking vectors which is obtained from the marking vector M, and $|*|$ denotes the number of elements of the set (*).

A marking M' is said to be reachable from M if there exists a firing sequence starting from M and yielding M' such as

$$M' = M + AU_g \quad (2)$$

where, $A := O - N$ denotes the incidence matrix, g indicates the sequence of enabled transitions, and $U_g : T \rightarrow \mathcal{N}$ denotes the firing count vector whose j^{th} element indicates how many times t_j is fired in g.

In this work, $\rho(M, g)$ denotes the transition function which gives yielded marking when the sequence g is fired starting from marking M, the reachability set, denoted by $R(G, m_0)$, is the set of all markings reachable from m_0.

The example Petri net is considered to explain the notation in this work. The example net is shown in Figure 1.

This net is described as $P = \{p_1, \ \ p_2\}$, $T = \{t_1, \ \ t_2\}$, $m_0 = [1 \ 1]^T$. In addition the input and output matrices are constructed as follows:

$$A = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

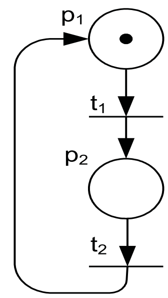

Figure 1: A Petri net

At initial marking, m_0, transition t_1 is enable, i.e $\rho(m_0, t_1) = M_1$:

$$\begin{aligned} M_1 &= m_0 + AU_{t_1} \\ M_1 &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{aligned}$$

Here U_{t_1} denotes the firing count vector of transition sequence $g = t_1$. First element of U_{t_1} denotes how many times t_1 is fired from the present marking. From marking M_1 not t_1 but t_2 is enabled, i.e $\rho(M_1, t_2) = M_2$:

$$\begin{aligned} M_2 &= M_1 + AU_{t_2} \\ M_2 &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{aligned}$$

Some properties of Petri nets are given as follow: Let us remember some behavioral properties related to the discussion of this work.

Definition 1: G is said to be K*-bounded*, if $M(p) \leq K(p)$, $\forall p \in P$, $\forall M \in R(G, m_0)$ ($K : P \rightarrow \mathcal{N}$), G is said to be *bounded* if it is K-bounded for some $K : P \rightarrow \mathcal{N}$. Otherwise G is *unbounded*.

Definition 2: A Petri net G is said to be reversible if $m_0 \in R(G, M)$, $\forall M \in R(G, m_0)$.

Definition 3: A Petri net G is said to be partial reversible if $m_0 \in R(G, M)$, for at least one $M \in R(G, m_0)$, and the reversible set is defined as $R_s := \{M \in R(G, m_0) \mid m_0 \in R(G, M)\} \subset R(G, m_0)$.

Definition 4: A Petri net G is said to be partial reversible if $m_0 \in R(G, M)$, for at least one $M \in R(G, m_0)$, and the reversible subset of the reachability set is defined as $R_s := \{M \in R(G, m_0) \mid m_0 \in R(G, M)\} \subset R(G, m_0)$.

Definition 5: Any nonzero solution of $AX = \hat{0}$, the vector X ($|T| \times 1$) is called as T-invariant. Here, $\hat{0}$ denotes zero vector ($|T| \times 1$), $|T|$ denotes the number of elements of T. If a T-invariant is not a linear combination of other T-invariants it is minimal T-invariant

In this work, the set of minimal T-invariants is determined by using the algorithm presented in [17] and it is denoted by $\mathcal{T}_m$.

Algorithm1

1. Construct $A' = [I \vdots A^T]$
2. j=$|T|$+1
3. Find the couples of nonzero elements in jth column such that sum of these elements are equal to zero.
4. Calculate the sum of the corresponding rows.
5. Each sum is appended to the bottom of A'.
6. Each addend are deleted from A'.
7. If $j = |P| + |T|$, go to 8th step, otherwise $j = j + 1$ and go back to $3rd$ step.
8. The first $|T|$ elements of the rows whose all elements are zero after $|P| + 1$ th column are the transpose of minimal T-invariants.

For example, consider a Petri net with the following incidence matrix:

$$A = \begin{bmatrix} -1 & 1 & 1 \\ 1 & -1 & -1 \\ 1 & -1 & 0 \end{bmatrix}$$

$$A^T = \begin{bmatrix} -1 & 1 & 1 \\ 1 & -1 & -1 \\ 1 & -1 & 0 \end{bmatrix}$$

$$A' = \begin{bmatrix} 1 & 0 & 0 & -1 & 1 & 1 \\ 0 & 1 & 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & 1 & -1 & 0 \end{bmatrix}$$

<u>Annulling 4th column</u>

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 1+2 \\ 1+3 \end{array} \quad \begin{bmatrix} 1 & 0 & 0 & -1 & 1 & 1 \\ 0 & 1 & 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & 1 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Delete 1st, 2nd and 3rd rows.

$$A' = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

<u>Annulling $5th$ row</u>

$5th$ row is annuled because of the calculations in the previous step.

<u>Annulling $6th$ row</u>

$6th$ row can not be annuled.

Minimal T-invariant is determined as $[1\ 1\ 0]^T$.

T-invariants are very important elements for Petri nets to analyse properties. If in all the T-invariant of the Petri net model of a manufacturing system, the same element is equal to zero, then we can claim that it is impossible to come back to the initial marking after firing sequence of transitions which contains the transition corresponding to this element. From a manufacturing point of view, this means that it will be impossible to come back to the initial state if we perform the operation represented by transition corresponding to the null elements of T-invariants.

Note that a sequence of transitions could be T-invariant without being firable. Let us assume that $\sigma_1, \sigma_2 ... \sigma_k$ are firable sequence of transitions and that X_{σ_i} are T invariants. If a management system proceeds by activating some of these sequences one or more times, then it guarantees that the state of the system is the same at the end of the process as it was at the beginning.

3 Reversible set

In this section, our method which determines a reversible set of a Petri net by using minimal T-invariants is introduced. The set of T-invariants of a Petri net is given by, $\mathcal{T} = \{X \mid A\, X = \hat{0}, X \in \mathcal{N}^{|T|}\}$. Minimal T-invariants of the net is formed by the basis of $\mathcal{T}$. In this work, the set of minimal T-invariants is determined by using the algorithm presented in [17] and it is denoted by $\mathcal{T}_m$.

When U_g of a firing sequence g is equal to a minimal T-invariant, i.e. $U_g = X \in \mathcal{T}_m$, then $AU_g = \hat{0}$.

Hence, the second term at the right side of the equation (2) is zero. Thus, M' is obtained as $M' = m_0$ if the transitions in g are firable from m_0 respectively.

The method uses all minimal T-invariants to find firable firing sequences from m_0. In our approach, we use the sum of all minimal T-invariants for this purpose. This sum, C, is determined as

$$C = \sum_{X \in \mathcal{T}_m} X \tag{3}$$

Consequently, C is also a T-invariant.

The method finds the set of corresponding firing sequences of C. These firing sequences are different ordering of transitions in C. Note that each transition $t_i \in T$ are repeated $C(i)$ times in the sequences, where $C(i)$ is the i^{th} element of C. The sequences constructs the set $\hat{g}_C$, it consists of all possible permutations of transitions in C (it is possible to obtain this set by using $perms$ function in Matlab).

Since it is not necessary that all firing sequences in $\hat{g}_C$ are firable from m_0, the sequences firable from m_0 are searched through $\hat{g}_C$, after $\hat{g}_C$ is formed. If there exists, all markings obtained during the firing of these firable sequences from m_0 constructs the set R_s, in this case $R_s \neq \emptyset$.

If $A\ X = 0$ has only trivial solution, then there exists no firing sequence g such that $AU_g = 0$. Hence, reversible set is obtained as empty, $R_s = \emptyset$.

4 Reversible Set Algorithm

In this section, the algorithm of the method presented in the previous section is introduced. The algorithm, called as Reversible Set Algorithm (RSA), determines the set of firing sequences corresponding to the sum of the minimal T-invariants, firstly. Then, it checks the firability of the firing sequences from m_0, one by one and then fires the firable sequences from m_0 and constructs the set R_s with the marking vectors obtained during these firings.

RSA requires the definition G and the set of minimal T-invariants $\mathcal{T}_m$ of the considered Petri net (the definition G contains the set of places P, the set of transitions T, the input and output matrices N and O, and the initial marking m_0). Note that, for the construction of the set $\mathcal{T}_m$ the algorithm in [17] is implemented by using Matlab.

RSA calculates C, and constructs the set $\hat{g}_C$. The sequences in $\hat{g}_C$, are represented by appropriate ordered sets of transitions, i.e. firing sequence "$t_x t_y t_z$" is represented by the set $\{t_x,\ t_y,\ t_z\}$ in $\hat{g}_C$. Then, the algorithm takes firing sequences from the set $\hat{g}_C$ one by one.

In the algorithm RSA, notations are given as thereinafter: $\hat{g}_C$ denotes the set of all possible combinations (all possible sequences) of transitins in C, $\hat{g}_C^i$ denotes i^{th} ($i \in \mathcal{N}$) element of the set $\hat{g}_C$, $[g]_i$ denotes i^{th} transition of the firing sequence g, R_s denotes determined reversible set, $\mathcal{E}(G, M)$ is the function which finds the set of enabled transitions at marking M, $\rho(G, [g]_j)$ is the function which determines the obtained marking when transition $[g]_j$ is fired from marking M.

RSA checks whether $[\hat{g}_C^1]_1 \in \mathcal{E}(G, m_0)$. If $[\hat{g}_C^1]_1 \in \mathcal{E}(G, m_0)$, in other words the first transition of $\hat{g}_C^1$ is firable from m_0, then RSA calculates the yielded marking ${M_1}^1 = \rho(m_0, [\hat{g}_C^1]_1)$ and put it to set $\bar{R}$. Next continues with $[\hat{g}_C^1]_2$ if $[\hat{g}_C^1]_2 \in \mathcal{E}(G, M_1^1)$, then RSA calculates the yielded marking $M_1^2 = \rho(M_1^1, [\hat{g}_C^1]_2)$ and put it to set $\bar{R}$. This procedure keeps on likewise. If all transitions of $\hat{g}_C^1$ are respectively firable from m_0, all markings in $\bar{R}$ is transfered to the set R_s. $\bar{R}$ is cleared and the next firing sequence ($\hat{g}_C^2$) is taken from the set $\hat{g}_C$. And the procedure is repeated again. If a transition $[\hat{g}_C^i]_j$ is not firable from M_{j-1}^i during the procedure, algorithm stops checking firability of present $\hat{g}_C^i \in \hat{g}_C$ from m_0 and takes the next firing sequence from the set $\hat{g}_C$.

When all firing sequences in $\hat{g}_C$ is checked for firability from m_0, RSA is terminated and returns the set R_s. If $R_s = \emptyset$ reversibility can not be enforced, otherwise $R_s \neq \emptyset$ and it is possible to enforce reversibility by the controller developed in [11].

General procedure of the method is given by Algorithm-2:

Algorithm2

1. $C = \sum_{X \in \mathcal{T}_m} X$
2. Construct the set $\hat{g}_C$ with the corresponding sequences of C.
3. Obtain firable sequences from m_0 in $\hat{g}_C$.
4. Fire enable transitions from m_0 and obtain new markings.
5. Construct R_s by using new markings.

The Pseudo code of the Algorithm2 is given as follow:

Pseudo Code of Reversible Set Algorithm (RSA)

$C = \hat{0}$
For $j = 1$ **to** $|\mathcal{T}_m|$
$C = C + [\mathcal{T}_m]_j$

End
$\hat{g}_C$=perm(C)
$R = \emptyset$, $G = \emptyset$, $\bar{R} = \emptyset$
For $t = 1$ **to** $|\hat{g}_C|$
  $M = m_0$
  $g = \hat{g}_C^t$
  For $j = 1$ **to** $|g|$
    same=0
    $T = \mathcal{E}(G, M)$
    For $k = 1$ **to** $|T|$
      If $[g]_j \in T$ **Then**
        same=1
        $\bar{M} = M + A\, U_{[g]_j}$
        $M = \bar{M}$
        $\bar{R} = \bar{R} \,\hat{\cup}\, M$
        break
      End
    End
    If $same = 1$ **Then**
      $R = R \,\hat{\cup}\, \bar{R}$
      break
    End
    If same=1 **Then**
      continue
    End
    If same=0 **Then**
      $\bar{R} = \emptyset$
      break
    End
  End
End
Return R_s

5 Controller

In [24], some algorithms and a controller have been presented to enforce boundedness, reversibility and liveness. In that work; initially, with an arbitrarily chosen bound vector, a bounded reachability set of an unbounded Petri net has been determined; then, the reversible subset of that bounded set is constructed by using developed algorithms. Since obtained reversible set may be empty, reversibility can not be enforced by the controller everytimes.

In this present work, we obtain R_s firstly. If $R_s \neq \emptyset$, reversibility is enforced by the controller approach developed in [24]. Because it is known that if a Petri net is partially reversible, the controller $c(M,t)$ below enforces reversibility of the net [24].

$$c(M,t) = \begin{cases} 1, & \text{if } \rho(M,t) \in R_s, \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

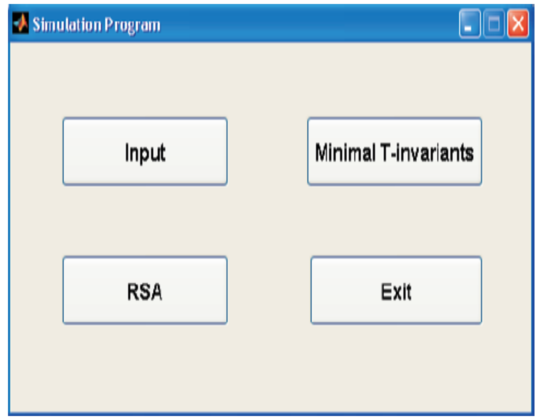

Figure 2: Program interface

where, $M \in R(G, m_0)$, $t \in \mathcal{E}(G, M)$. If $c(M,t) = 1$, then $\rho(M,t) \in R_s$ and firing transition t from marking M is allowed. If $c(M,t) = 0$, then $\rho(M,t) \notin R_s$ and firing transition t from marking M is forbidden.

6 Simulation Program

A program, called as RSA simulation program (RSA-SP) is developed to implement this algorithm. It is developed as a Matlab function. When RSA-SP is run, the interface shown in Figure 2 appears on the screen. Initially only "Input" button is activated. After this button is clicked the name of the input file (constructed by the user) must be entered. An example format of this file is given in Appendix B. After this operation, "Minimal T-invariants" button is activated. By clicking on this button the Matlab function developed for the algorithm introduced in [17] is run. Once, minimal T-invariant set is determined, the RSA button also becomes active. By clicking on this button, the Matlab function for the algorithym RSA executes and R_s is obtained.

RSA-SP determines the set of minimal T-invariants $\mathcal{T}_m$, the sum of these invariants C, corresponding firing sequence set $\hat{g}_C$. Then the set of transition squences firable from m_0 through the set $\hat{g}_C$. Finally, RSA-SP composes the set R_s. These results are written on output file Output.m. An example format of this file is given in Appendix C.

7 Example

We give an example to simply show how our algorithm works for the Petri net ([4] p.182) shown in Figure 3. The set of places is $P = \{p_1,\ p_2,\ p_3,\ r_1,\ r_2,\ r_3,\ p_4,\ p_5,\ p_6\}$, the set of transitions is $T = \{t_1,\ t_2,\ t_3,\ t_4,\ t_5,\ t_6\}$, and the initial marking is $m_0 = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T$. The input file of this example Petri net for RSA simulation program is given in Appendix A.

The example Petri net is not bounded, because the number of tokens in places p_2 and p_5 can increase. The set $\mathcal{T} = \{[1\,1\,1\,0\,0\,0]^T, [0\,0\,0\,1\,1\,1]^T\}$ is obtained by using the algorithm in [17]. Then C is determined as $C = [1\ 1\ 1\ 1\ 1\ 1\ 1]$.

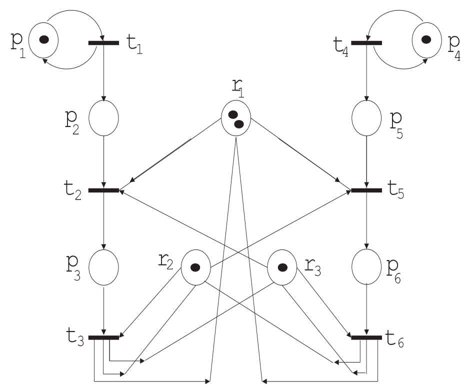

Figure 3: Example Petri net [4]

The Matlab function $perm(C)$ finds all possible sequences which consists of one t_1, one t_2, one t_3, one t_4, one t_5, one t_6. Output of the function $perm(C)$ has 720 firing sequences. Some of these sequences are given as fallow:

$$
\begin{array}{lllllll}
\hat{g}_C^1 = & \{t_1, & t_2, & t_3, & t_4, & t_5, & t_6\}, \\
\hat{g}_C^2 = & \{t_4,, & t_5, & t_6, & t_1, & t_2, & t_3\}, \\
\hat{g}_C^3 = & \{t_4,, & t_5, & t_1, & t_6, & t_2, & t_3\}, \\
\hat{g}_C^4 = & \{t_4, & t_1, & t_2, & t_3, & t_5, & t_6\}, \\
\hat{g}_C^5 = & \{t_4, & t_1, & t_5, & t_6, & t_2, & t_3\}, \\
\hat{g}_C^6 = & \{t_1, & t_4, & t_5, & t_6, & t_2, & t_3\}, \\
\hat{g}_C^7 = & \{t_4, & t_1, & t_5, & t_6, & t_2, & t_3\}, \\
\vdots & & & & & & \\
\hat{g}_C^{718} = & \{t_2, & t_4, & t_5, & t_6, & t_2, & t_3\}, \\
\hat{g}_C^{719} = & \{t_4, & t_1, & t_6, & t_5, & t_2, & t_3\}, \\
\hat{g}_C^{720} = & \{t_6, & t_4, & t_5, & t_1, & t_3, & t_2\}.
\end{array}
$$

Note that, 8 of these 720 transitions are firable from m_0. Let us analyse each of the sequences in order of occurence:

- $\hat{g}_C^{245} = \{t_4, t_5, t_6, t_1, t_2, t_3\}$

$\rho(m_0, t_4) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_1^{245}$
$\rho(M_1^{245}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_2^{245}$
$\rho(M_2^{245}, t_6) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_3^{245}$
$\rho(M_3^{245}, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_4^{245}$
$\rho(M_4^{245}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_5^{245}$
$\rho(M_5^{245}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{245} = m_0$

$$R_s = \{m_0, M_1^{245}, M_2^{245}, M_3^{245}, M_4^{245}, M_5^{245}\}$$

- $\hat{g}_C^{263} = \{t_4, t_5, t_1, t_6, t_2, t_3\}$

$\rho(m_0, t_4) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_1^{263}$
$\rho(M_1^{263}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_2^{263}$
$\rho(M_2^{263}, t_1) = [1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_3^{263}$
$\rho(M_3^{263}, t_6) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_4^{263}$
$\rho(M_4^{263}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_5^{263}$
$\rho(M_5^{263}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{263} = m_0$

$$R_s = R_s \cup \{M_1^{263}, M_2^{263}, M_3^{263}, M_4^{263}, M_5^{263}\}$$

- $\hat{g}_C^{350} = \{t_4, t_1, t_2, t_3, t_5, t_6\}$

$\rho(m_0, t_4) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_1^{350}$
$\rho(M_1^{350}, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_2^{350}$
$\rho(M_2^{350}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0]^T = M_3^{350}$
$\rho(M_3^{350}, t_3) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0]^T = M_4^{350}$
$\rho(M_4^{350}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_5^{350}$
$\rho(M_5^{350}, t_6) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{350} = m_0$

$$R_s = R_s \cup \{M_1^{350}, M_2^{350}, M_3^{350}, M_4^{350}, M_5^{350}\}$$

- $\hat{g}_C^{359} = \{t_4, t_1, t_5, t_6, t_2, t_3\}$

$\rho(m_0, t_4) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_1^{359}$
$\rho(M_1^{359}, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_2^{359}$
$\rho(M_2^{359}, t_5) = [1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_3^{359}$
$\rho(M_3^{359}, t_6) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_4^{359}$
$\rho(M_4^{359}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_5^{359}$
$\rho(M_5^{359}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{359} = m_0$

$$R_s = R_s \cup \{M_1^{359}, M_2^{359}, M_3^{359}, M_4^{359}, M_5^{359}\}$$

- $\hat{g}_C^{629} = \{t_1, t_4, t_5, t_6, t_2, t_3\}$

$\rho(m_0, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_1^{629}$
$\rho(M_1^{629}, t_4) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_2^{629}$
$\rho(M_2^{629}, t_5) = [1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_3^{629}$
$\rho(M_3^{629}, t_6) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_4^{629}$
$\rho(M_4^{629}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_5^{629}$
$\rho(M_5^{629}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{629} = m_0$

$$R_s = R_s \cup \{M_1^{629}, M_2^{629}, M_3^{629}, M_4^{629}, M_5^{629}\}$$

- $\hat{g}_C^{637} = \{t_1, t_4, t_2, t_3, t_5, t_6\}$

$\rho(m_0, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_1^{637}$
$\rho(M_1^{637}, t_4) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_2^{637}$
$\rho(M_2^{637}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0]^T = M_3^{637}$
$\rho(M_3^{637}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_4^{637}$
$\rho(M_4^{637}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_5^{637}$
$\rho(M_5^{637}, t_6) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{637} = m_0$

$$R_s = R_s \cup \{M_1^{637}, M_2^{637}, M_3^{637}, M_4^{637}, M_5^{637}\}$$

- $\hat{g}_C^{673} = \{t_1, t_2, t_4, t_3, t_5, t_6\}$

$\rho(m_0, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_1^{673}$
$\rho(M_1^{673}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_2^{673}$
$\rho(M_2^{673}, t_4) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0]^T = M_3^{673}$
$\rho(M_3^{673}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_4^{673}$
$\rho(M_4^{673}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_5^{673}$
$\rho(M_5^{673}, t_6) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{673} = m_0$

$$R_s = R_s \cup \{M_1^{673}, M_2^{673}, M_3^{673}, M_4^{673}, M_5^{673}\}$$

- $\hat{g}_C^{679} = \{t_1, t_2, t_3, t_4, t_5, t_6\}$

$\rho(m_0, t_1) = [1\ 1\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_1^{679}$
$\rho(M_1^{679}, t_2) = [1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]^T = M_2^{679}$
$\rho(M_2^{679}, t_3) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_3^{679}$
$\rho(M_3^{679}, t_4) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 1\ 0]^T = M_4^{679}$
$\rho(M_4^{679}, t_5) = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^T = M_5^{679}$
$\rho(M_5^{679}, t_6) = [1\ 0\ 0\ 2\ 1\ 1\ 1\ 0\ 0]^T = M_6^{679} = m_0$

$$R_s = R_s \cup \{M_1^{679}, M_2^{679}, M_3^{679}, M_4^{679}, M_5^{679}\}$$

R_s is a reversible set of Petri net. It has 40 elements. It is the largest reversible set that can be obtained by the sum of minimal T-invariants (it is possible to enlarge the set R_s using other linear combinations of minimal T-invariants). Note that, the output file of the RSA simulation program for this Petri net is given in Appendix B.

8 Conclusion

Petri net model is very useful modeling tool for discrete event systems. For analysing systems modeled by Petri nets reachability set approach is used frequently. But, because of the state explosion problem it is impossible to use reachability set approach for analysing big dimensional or unbounded systems.

In this work, we propose a method which does not construct reachability set and use a structural approach to analyse reversibility which is one of the most important properties of Petri net model.

The method calculates all possible transition sequences of minimal T-invariants. Then, it checks these sequences one by one. If transitions of a sequence are respectively firable from m_0, all markings obtained during this firing are put into the set R_s. At the end RSA returns R_s. If R_s is not empty reversibility can be enforced by using the controller in [11]. This controller guarantees that the reachability set of the controlled Petri net is reversible .

A Petri net model which was given by [5] is considered to show the advantage of the presented algorithm. This Petri net model has 32 places and 20 transitions. For this example, the reversible set was constructed by [11] depending on the reachability set. The centralized method took about 32 min. (on a PC with a Pentium-4 microprocessor running at 3.0 GHz and has 768MB RAM). The number of elements of this set is 3802. The presented algorithm took 30 sec. and the reversible set has 450 elements. The reason, which causes this difference, is explained such that, in [11], all markings in the reachability set are searched to find the reversible markings. Thus, although the largest reversible set is obtained, the duration takes a lot of time. In our presented work, reversible set is found by only minimal T-invariants. If other linear combinations of T-invariants are used, the largest reversible subset is constructed.

Overlapping decompositions and expansions (see, [19]) may be used and an algorithm which is based on local T-invariants may be developed for further work. Another direction for further research may be developing the structural controller for reversibility enforcement (see, [15]).

APPENDIX

In this section, the input file of the program RSA-SP for the definition of example net and the output file of RSA-SP which contains the minimal T-invariants, the sequences from m_0 and the elements of R_s for the example net are given in Appendix A and Appendix B, respectively.

Appendix A : Input file of the example

```
N=[1 0 0 0 0 0
   0 1 0 0 0 0
   0 0 1 0 0 0
   0 1 0 0 1 0
   0 0 1 0 1 0
   0 1 0 0 0 1
   0 0 0 1 0 0
   0 0 0 0 1 0
```

```
   0 0 0 0 0 1]


O=[1 0 0 0 0 0
   1 0 0 0 0 0
   0 1 0 0 0 0
   0 0 1 0 0 1
   0 0 1 0 0 1
   0 0 1 0 0 1
   0 0 0 1 0 0
   0 0 0 1 0 0
   0 0 0 0 1 0]


m0=[1 0 0 2 1 1 1 0 0]
```

Appendix B : Output file of the example

```
Minimal T-invariants:
1   1    1    0    0     0

0   0    0    1    1     1

C:
1    1     1     1     1     1


Firable  sequences:
{t4,  t5,  t6,  t1,  t2,  t3}

{t4,  t5,  t1,  t6,  t2,  t3}

{t4,  t1,  t2,  t3,  t5,  t6}

{t4,  t1,  t5,  t6,  t2,  t3}

{t1,  t4,  t5,  t6,  t2,  t3}

{t1,  t4,  t2,  t3,  t5,  t6}

{t1,  t2,  t4,  t3,  t5,  t6}

{t1,  t2,  t3,  t4,  t5,  t6}

RS:
1  0  0  2  1  1  1  1  0

1  0  0  1  0  1  1  0  1

1  0  0  2  1  1  1  0  0

1  1  0  2  1  1  1  0  0

1  0  1  1  1  0  1  0  0

1  0  0  2  1  1  1  1  0

1  0  0  1  0  1  1  0  1

1  1  0  1  0  1  1  0  1

1  1  0  2  1  1  1  0  0

1  0  1  1  1  0  1  0  0

1  0  0  2  1  1  1  1  0

1  1  0  2  1  1  1  1  0

1  0  1  1  1  0  1  1  0

1  0  0  2  1  1  1  1  0

1  0  0  1  0  1  1  0  1

1  0  0  2  1  1  1  1  0

1  1  0  2  1  1  1  1  0

1  1  0  1  0  1  1  0  1

1  1  0  2  1  1  1  0  0

1  0  1  1  1  0  1  0  0

1  1  0  2  1  1  1  0  0

1  1  0  2  1  1  1  1  0

1  1  0  1  0  1  1  0  1

1  1  0  2  1  1  1  0  0

1  0  1  1  1  0  1  0  0

1  1  0  2  1  1  1  0  0

1  1  0  2  1  1  1  1  0

1  0  1  1  1  0  1  1  0

1  0  0  2  1  1  1  1  0

1  0  0  1  0  1  1  0  1

1  1  0  2  1  1  1  0  0

1  0  1  1  1  0  1  0  0
```

1	0	1	1	1	0	1	1	0
1	0	0	2	1	1	1	1	0
1	0	0	1	0	1	1	0	1
1	1	0	2	1	1	1	0	0
1	0	1	1	1	0	1	0	0
1	0	0	2	1	1	1	0	0
1	0	0	2	1	1	1	1	0
1	0	0	1	0	1	1	0	1

References:

[1] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, Kluwer Academics, Norwell, MA; 1999.

[2] Y. (Ed.) Ho, *Discrete Event Dynamic Systems: Analyzing Complexity and Performance in the Modern World*, A Selected Reprint Volume, The Institute of Electrical and Electronics Engineers, Inc., Newyork; 1992.

[3] A. A. Desrochers and R. Y. Al-Jaar, *Applications of Petri Nets in Manufacturing Systems*, The Institute of Electrical and Electronics Engineers Inc., New York; 1995.

[4] J. Proth and X. Xie, *Applications of Petri Nets in Manufacturing Systems*, John Wiley & Sons, West Sussex; 1996.

[5] M. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing SystemS*, Norwell, MA; 1993.

[6] Q. Ye and M. H. MacGregor, ”Combining Petri Nets and ns-2: a hybrid method for analysis and simulation”, *Proceedings of the 4th Annual Communication Networks and Services Research Conference*, 2006, pp. 21-28.

[7] B. Hruz, ”Solution of the Manufacturing Transport Control Using Petri Nets”, *Proceedings of IEEE International Conference on Systems Man and Cybernetics*, 2003, pp. 4736-4741.

[8] H. Apaydın and A. Aybar, ”Reversibility Enforcement for Unbounded Petri Nets”, *International Conference on Informatics in Control, Automation and Robotics*, 2005, pp. 181-186.

[9] H. Apaydın, A. Manay, A. Aybar and A. İftar, ”A Program for Analysis and Control of Petri Nets”, *Proc. CD-ROM of IEEE International Conference on Computational Cybernetics*, 2004, pp. 309-314.

[10] A. Aybar and A. İftar, Decentralized Supervisory Controller Design to Avoid Deadlock in Petri nets, *International Journal of Control*, vol. 76, 2003, pp. 1285-1295.

[11] A. Aybar, A. İftar and H. Apaydın, Centralized and Decentralized Supervisory Controller Design to Enforce Boundedness, Liveness, and Reversibility in Petri Nets , *International Journal of Control*, vol. 78, 2005, pp. 537-553.

[12] T. Murata, Petri Nets: Properties, Analysis and Applications, *Proceeding of the IEEE*, vol. 77, 1989, pp. 541-580.

[13] M. D. Jeng and M. Y. Peng, Augmented Reachability Trees for 1-Place-Unbounded Petri Nets, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 29, 1999, pp. 173-183.

[14] L. Ferrarini, On the Reachability and Reversibility Problems in a Class of Petri nets, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 29, 1994, pp. 1474-1482.

[15] A. Aybar, Decentralized Structural Control Approach for Petri Nets, *Control and Cybernetics*, vol. 36, 2007, pp. 143-159.

[16] R. Bou and A. E. Moudni, On the Analysis of Some Structural Properties of Petri Net, *In Applications and Theory of Petri Nets*, vol. 35, 2005, pp. 784-794.

[17] J. Martinez and M.Silva, A Simple and Fast Algorithm to Obtain All Invariants of a Generalized Petri Net, *Comput. J.*, vol. 52, 1982, pp. 301-311.

[18] K. El-Fakih, H. Yamaguchi, G. V. Bochmann and T. Higashino, Petri Net Based Protocol Synthesis with Minimum Communication Costs, *Journal of the Franklin Institute*, vol. 343, 2006, pp. 501-520.

[19] A. Aybar and A. İftar, Overlapping Decompositions and Expansions of Petri nets, *IEEE Transactions on Automatic Control*, vol. 47, 2002, pp. 511-515.

[20] J. Esparza and M. Nielsen, Decidability Issues in Petri Nets, *BRICS Report Series, RS-94-8*

Department of Computer Science, University of Aarhus, Aarhus, Denmark, 1994.

[21] F. Zuniga, A. Alcala, I. Piza and F. Ramos, A Formal Method to Tranform an Elementary Object Systems into a Colored Petri Net *WSEAS Transactions on Information Science and Applications*, vol. 1, 2004, pp. 1082-1088.

[22] T. Chiang and L. Dung, System Level Verification on High-Level Synthesis of Dataflow Algorithms using Petri Net *Wseas Transactions On Circuits And Systems*, vol. 5, 2006, pp. 790-797.

[23] J. Tu, Accomplishing a Petri-net Simulator Use Fuzzy Rule-Based *Wseas Transactions On Systems* , vol. 6, 2007, pp. 1073-1082.

[24] Aybar, A., A. Iftar and H. Apaydin-zkan, Centralized and Decentralized Supervisory Controller Design to Enforce Boundedness, Liveness, and Reversibillity in Petri Nets, *Int. Journal of Control*, vol. 78, 2005, pp. 537-553.