Resolving multiple conflicts (deadlocks) in the petri net model of flexible assembly system using marked Graph technique

M.A. Shouman
Operations Res. and Decision Support Sys. Dept., Faculty of Comp. and Infs., Zagazig University, Zagazig , Egypt
Profshouman@hotmail.com
Ibrahim M. Buseif
Faculty of Eng., Industrial Eng. Dept, University of Garyounis - Benghazi - Libya
ibrahimbuseif@yahoo.com

We presented in this paper an algorithmic procedure for resolution of 'multiple conflicts ' in Petri nets model of the Flexible Assembly System's (FAS's) constitutes an important part of a Computer Integrated Manufacturing System (CIMS) and its conveniently modelled with Petri nets. The flexibility of an FAS appears as ' multiple conflicts ' in a Petri net model of the FAS. The method proposed provides the transformation rules allowing to convert the set of production rules into the relevant Petri net model, which then is used for P-invariants analysis.

أنظمة التجميع المرن تعتبر جزء رئيسي من أنظمة التصنيع المرن حيث أن أنظمة التجميع المرن تحتوى علـى عـدة مراكـز لعمليات التجميع لأجزاء المنتج أو المنتجات، وميكانيكية نظام عمل هذه العملية هو مثل نظام نظرية الصفوف، حيث تصل أجزاء المنتج المراد تجميعة إلى مركز التجميع و تخزن بمخازن مؤقتة ثم يتم تجميعها باستخدام الإنسان الالى مما قد يسبب نـوع مـن الاختناقات لدى مراكز التجميع. هذه الورقة البحثية تقدم طريقة جديدة لمعالجة هذه الاختناقات وذلك بتحويـل قواعـد العمليـات الإنتاجية إلى نموذج شبكة بيترى يتم التأكد من وجود اختناقات بالنظام من عدمه، وفى حالة إيجاد الاختناقـات نـستخدم أسـلوب ماركت البياني التي تم اقتراحها لوضع الحل لهذه الاختناقات، هذا ولقد أعطت الطريقة المقترحة نتائج طيبة مـن خـلال البحـث المقدم.

Keywords: Automated manufacturing systems, Petri nets, Market graph model, Structured analysis, Computer control

1. Introduction

Flexible assembly systems are becoming a major part of computer integrated manufacturing systems, and they provide a significant potential for both productivity improvement and rapid response to market demands. An important feature of an FAS is its operational flexibility meaning that a given assembly operation may be performed by more than one operator and an operator can be assigned to more than one assembly operation. This type of flexibility usually promises an improved line performance, but it demands a more careful planning in the design of an FAS. Therefore, some forms of modelling tools for an efficient design and operational control of FAS are needed. Petri nets have been widely used in modelling systems which have serial and concurrent events with resource constraints. In its standard form, a Petri net model renders itself to a variety of analysis techniques Peterson [1], but does not have the notion of ' time ' which is an essential element in the modelling and analysis of FASs. There are two alternative approaches to associating time with the standard Petri net. Namely, the extension of Petri net with timed transition or timed places. In this paper a timed transition or timed places are not employed.

When modelling an FAS using Petri net, the operational flexibility appears on the net in the form of conflict, meaning that a ' place ' has more than one output transition. In this paper, it is called' single conflict'. When a transition in a single conflict is also involved in the other conflicts, these single conflicts constitute a' multiple conflict'. This multiple conflict phenomenon is an inherent feature of a Petri net model of FAS.

Introduced in this paper is an algorithmic procedure for identifying and resolving multiple conflicts (deadlocks) in Petri net model of FAS.

2. Building a Petri net model of an FAS

The Flexible Assembly System (FAS) considered in this paper consists of:
1. robots equipped with assembly tools;
2. work stations where the assembly operations are performed;
3. queues (in-process buffer storages); and
4. auxiliary facilities such as feeders.

The base component seated on a pallet is moved into a work station where one or more assembly components fed from feeders are assembled on to the base component by a robot. Then, the pallet is moved to the next work station and so on. A VTR deck assembly line is a typical example of such as FAS. In this section, we propose an informal method (An informal model is a verbal and/or graphical description of the system under consideration, such a model lacks formal semantics) of building a Petri net model of the FAS [2-3].

A portion of an FAS is considered for explanatory purposes and is depicted in fig. 1- a. There are three work stations (WS_1, WS_2, WS_3), two robots (R_1, R_2), and two queues (Q_1, Q_2) each having a capacity of 3. Two assembly operations (A_1, A_2) are performed at WS_1 by R_1; one assembly operation (A_3) is performed at WS_2 either by R_1 or by R_2; and another assembly operation (A_4) at WS_3 by R_2.

Now we propose an informal model description method for the FAS. The proposed scheme is similar to the Activity Cycle Diagram (ACD). The ACD-like model shown in fig. 1-b is obtained as follows [4-7]:
1. An assembly operation A_1 is denoted as a (rectangular) assembly node.
2. A queue space Q_1 is denoted as a (circular) queue node.
3. A robot R_1 is denoted as a (circular) robot node.
4. A (circular) wait node is provided in-between consecutive assembly nodes.
5. The assembly nodes, queue nodes, and wait nodes are connected by solid arrows (i.e. directed arcs) along the flow of the assembly line.
6. The relations between assembly operations and robots are indicated dashed arrows.
7. Assembly nodes in a work station WS_1 are connected as a work station node.

The mapping from the FAS components to the ACD-like symbols is very straight-forward and the resulting model description is quite clear.

It is now necessary to convert the ACD-like model to standard Petri net model. The conversion rules are as follows see fig. 2:
1. A robot node becomes a place having a token and dashed arrow becomes a pair of (input/output) arcs connected to the places fig. 2-a.
2. A wait node becomes an empty place and solid arrow an arc fig. 2-b.
3. An assembly node associated with a single robot is bounded by a pair of primitive transitions that are connected by a place fig. 2-c.
4. An assembly node shared by n robots is bounded by a pair of primitive transitions that are connected by a place fig. 2-d.
5. A work station node (i.e. transitions belonging to the work station) is bounded by a pair of primitive transitions connected by a place having one token fig. 2-e.
6. A queue node becomes a place bounded by a pair of primitive transitions that are connected by a place having a number of tokens equal to the queue capacity fig. 2-f.

In modelling the queue nodes, it is assumed that the transport times passing through the queue spaces are negligible (meaning that they do not affect the system performance) in order to simplify the discussion somewhat. It is not a crucial assumption, however.

By applying the conversion rules of fig. 2, the ACD-like model given in fig. 1-b would become the Petri net shown in fig. 3. Notable features of the standard Petri net model are as follows:
- There are eight transitions (A_{11}, A_{12}, A_{21}, A_{22}, A_{31}, A_{32}, A_{41}, A_{42}) for assembly operations.
- There are two places for robots (R_1, R_2), two for queues (Q_1, Q_2), three for work stations (WS_1, WS_2, WS_3), one for wait (W), and eight dummy places.

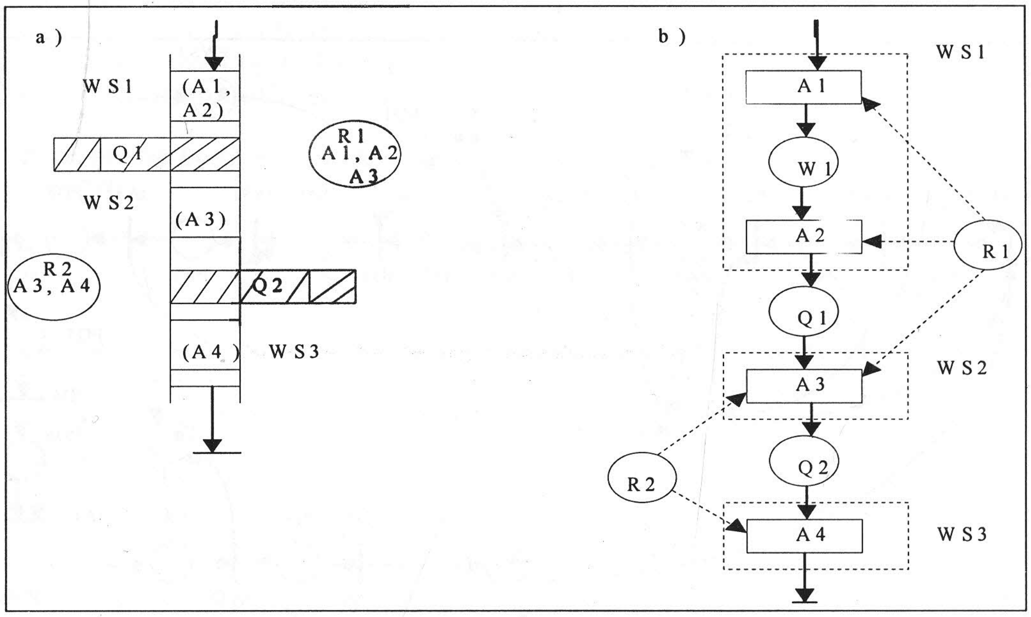

Fig. 1. ACD-like modelling o af FAS.

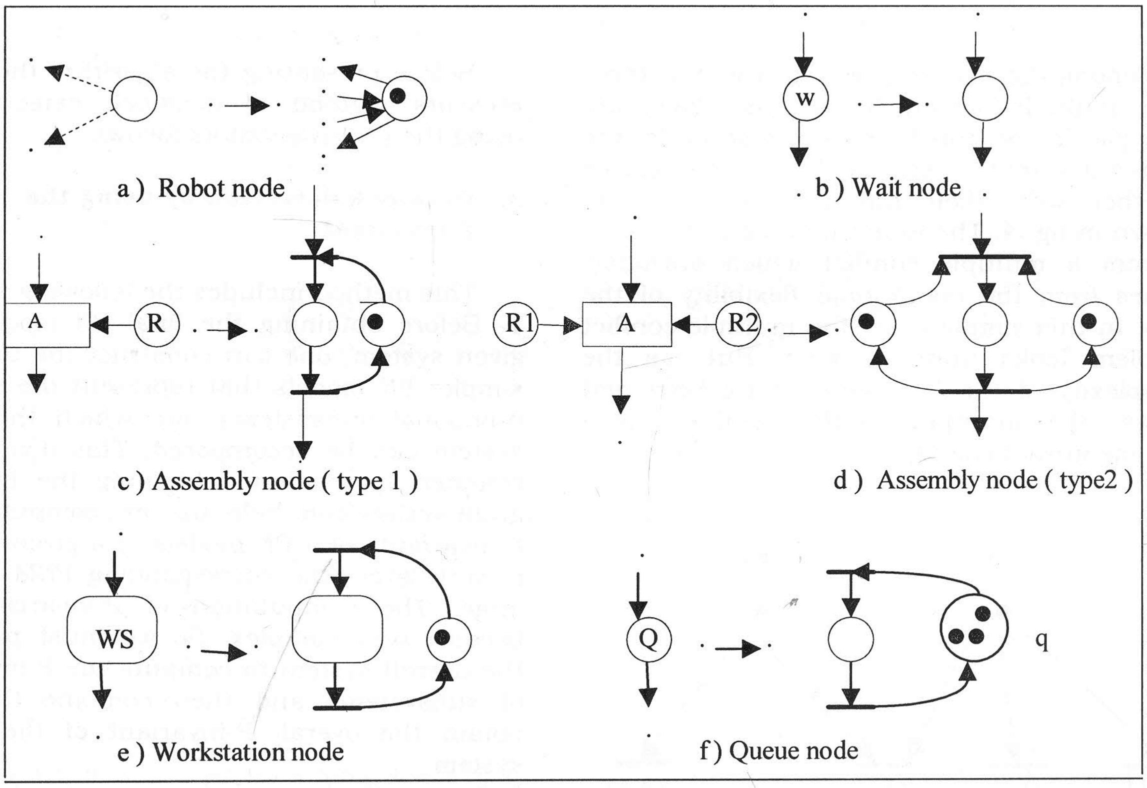

Fig. 2. Conversion from ACD-like to Petri net model.

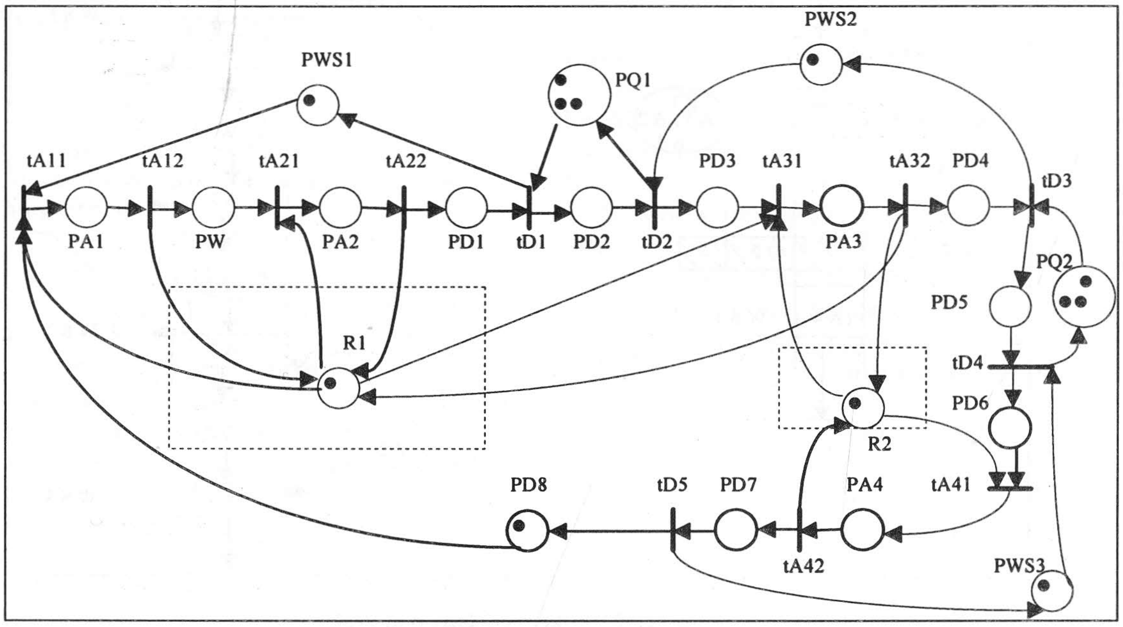

Fig. 3. Standard Petri net (PN) model of the FAS in fig.1.

Among the twenty places in the net, three have multiple output transitions. There are two places for robots corresponding to the shared assembly operation A_3. The two places together with their output transitions are shown in fig. 4. The four transitions in the fig. 4 from a multiple conflict which obviously comes from the operational flexibility of the FAS. In this simple case, the multiple conflict problem looks quite obvious. But, as the complexity of the FAS (and of its Petri net) grows, the multiple conflict problem may become intractable [8].

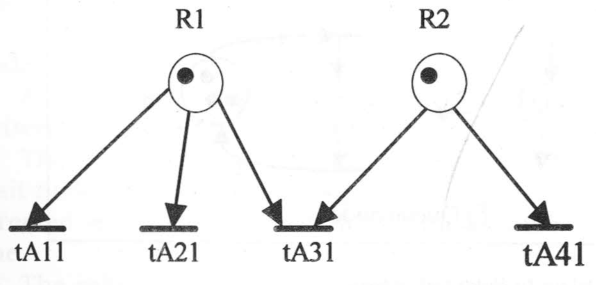

Fig. 4. Multiple conflict in the Petri net of fig. 1.

Before presenting the algorithm the paper presents method of deadlock detection by using the P - Invariant as follows.

3. Deadlock detection by using the P-Invariant

This method includes the following steps:
1- Before obtaining the final PN model of a given system, one can construct the union of simpler PN models that represent the various functional subsystems into which the given system can be decomposed. This divide-and-conquer approach to obtaining the PN of a given system can help us in computing the P-invariants of a PN models. If a given system is very large, the corresponding PNM is also large. The computation of P-invariant will become very complex. So we must partition the overall system to compute the P-invariant of subsystems and then combine them to obtain the overall P-invariant of the entire system.
2- The method can help us compute the P-invariant of the union of two PNs when the P-invariant of individual nets are known and the nets satisfy the conditions of the theorem.

3- Basically the method uses the incidence matrix representation D of a PN to compute the P-invariant characterised by U through the equation $U * D = 0$. Again from the equation

$\mu = \mu_o + D^*Y$, multiplying by U, we obtain:
$U^*\mu = U^*\mu_o + U^*(D^*Y)$

where $U^*D = 0$
Then, $U^*\mu = U^*\mu_o$

4- From step (3), we get some equations and then choose some states to verify if these states are deadlocked. Then one can say if the original system is deadlocked or not.

We are applying our problem fig. 3 to detect the deadlock (conflict points) of a PN model by using P-invariant. First we calculate the incidence matrix by using this equation $D = D^+ - D^-$, where D is the incidence matrix and D^+ is output matrix and D^- is an input matrix. For example, transition ($tA11$) has only one place ($PA1$) output and three places ($PWS1$, $R1$, and $PD8$) input.

$$D = \begin{array}{r|ccccccccccccc|}
PA1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PW & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PA2 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PD1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PWS1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
R1 & -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PD2 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PQ1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
PD3 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
PWS2 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
PA3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
R2 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 \\
PD4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
PD5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\
PQ2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\
PD6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
PA4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\
PWS3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\
PD7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\
PD8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}$$

The relation $U * D = 0$, $u_i \geq 0$ can rewritten as:

$$\begin{array}{lr}
u_1 - u_5 - u_6 - u_{20} & = 0 \\
-u_1 + u_2 + u_6 & = 0 \\
\vdots & \vdots \\
u_{18} - u_{19} + u_{20} & = 0
\end{array}$$

When we solve these equations, there are some values of $u_i < 0$, then this means that the PN model is not deadlock-free.

If we want to implement the above method in computers we may face some difficulties. The first one is how to solve the equation $U^* D=0$. Because there are no exact solutions of the equation, we must assume some unknown integers to represent these variables. Using a computer it is difficult to implement the process. The second problem is in step (4), where it is difficult to choose a special marking the deadlock property. The computer is not intelligent enough to know the event [2-5].

The completeness and consistency examination are closed linked to the reachability problem. In order to illustrate this fact let us introduce the concept of a marked graph (MG) which is a Petri net $C = (P, T, I, O, \mu_0)$ [9].

4. Marked graph

A marked graph is a PN in which each place is an input for exactly one transition and an output for exactly one transition. Alternatively, we can say that each place exactly one input and one output [1, 4].

A marked graph is a PN $C = (P,T,I,O)$ such that for each $p_i \in P, |I(pi)| = |\{t_j / p_i \; O(t_j)\}| = 1$ and $|O(pi)| = |\{t_j / p_i \; I(tj)\}| = 1$.

Marked graphs can model concurrence and synchronisation but cannot model conflict or data-dependent decisions. The properties which have been investigated for marked graphs have been liveness, safeness, and reachability.

In the investigation of these properties, the major structural parts of a marked graph of interest are its cycles. A cycle in a marked graph is a sequence of transitions $t_{j1}t_{j2}\ldots t_{jk}$ such that for each t_{jr} and t_{jr+1} in the sequence there is a place p_{ir} the $p_{ir} \in O(t_{jr})$ and $p_{ir} \in I(t_{jr+1})$ and $t_{j1} = t_{jk}$.

A cycle is such a closed path from a transition back to that same transition. The importance of cycles for marked graphs derives from a number of theorems that are covered in the papers [4, 10].

The modified standard Petri net after applying some rules of transformation is shown in fig. 5.

4.1. Computer implementation of the corrected algorithm

A flow chart for computing P - invariants is given in fig 6. This is to check for deadlocks and determine the other properties of PN.

The paper illustrates the proposed approach through an example.

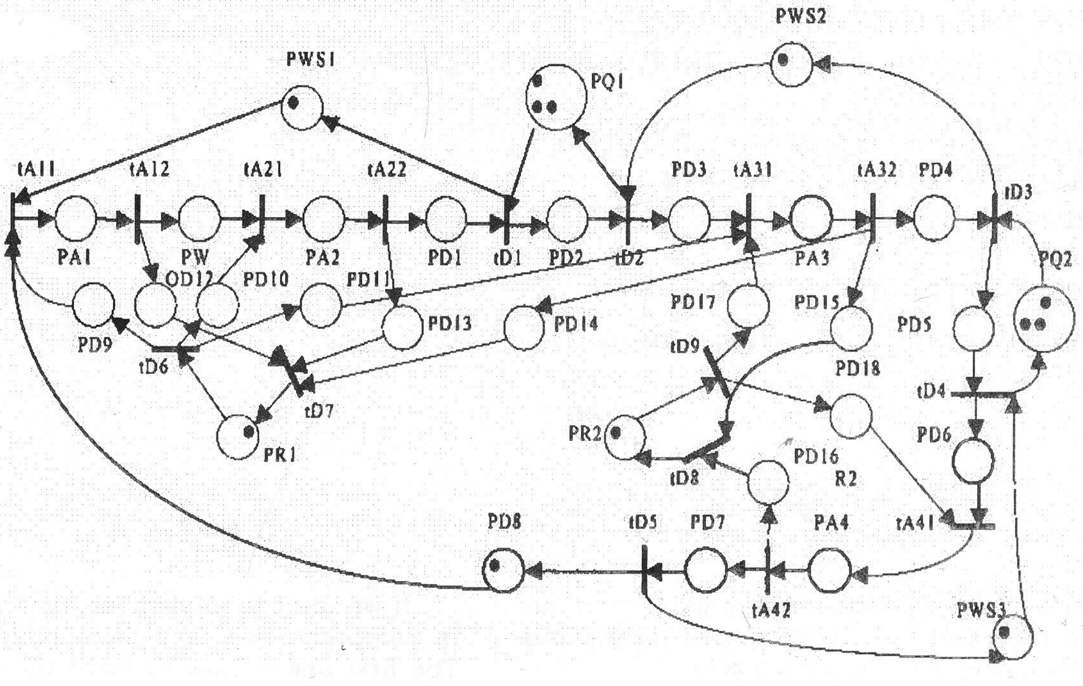

Fig. 5. New standard Petri net (PN) model of the FAS in fig. 1. after applying R1 and R2 rules.

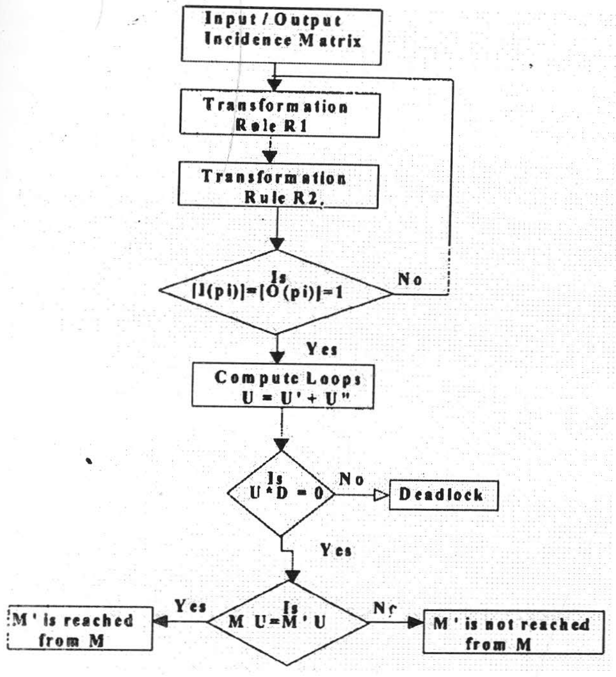

Fig. 6. Flow chart of computing P- invariants.

5. Concluding remarks

The conflict resolution scheme has been focused on the Petri net model of a FAS's, but it is applicable to a general Petri net having multiple conflicts. In general the multiple conflict problem is inherent in the most' flexible systems'. The proposed approach provides the basis for the development of computer-based methods allowing to verify rule-based data bases. Many tasks, however, have to be resolved in order to obtain such tools.

Petri net modelling is a powerful tool for modelling and analysis of asynchronous, 67 concurrent systems which are difficult to model using simulation and QNs. It is convenient to model non-product form characteristics, such as multiple workstation holding, blocking, synchronisation, concurrent, and prioritisation, which are common in a computer integrated manufacturing CIM.

References

[1] J. Peterson, PETRI NET Theory and the Modelling of System, Prentice Hall, New Jersey, U.S.A (1981).

[2] Fu-Shiung. Hsieh, "Model and Control Holonic Manufacturing Systems based on Fusion of Contract Nets and Petri Nets", In: Automatica, (40), pp. 52-57 (2004).

[3] M. Uzam, "An Optimal Deadlock Prevention Policy for Flexible Manufacturing Systems using Petri net Models with Resources and the Theory of Region", International Journal of Advanced Manufacturing Technology Vol. 19 (3), pp. 192 – 208 (2002).

[4] I.M. Buseif, "An Approach to FMS Design Using SADT and PN Tools", Thesis, Ph. D Warsaw University of Technology, Faculty of Production Engineering, Warsaw-Poland (1997).

[5] F. Dicesare, G. Harhalakis, J.M. Proth, M. Silva and F.B. Vernadat, Practice of PETRI NETS in Manufacturing, Chapman and Hall, London (1993).

[6] S. Daniel. Miryam, "Flexibility in Manufacturing Systems: Definitions and PETRI NET Modelling", INT. J. PROD. RES., Vol. 26 (2), pp 237-248 (1988).

[7] J. Sifakis, Performance Evaluation of Systems Using Nets, in: W. Brauer (ed), Net Theory and Applications, Lecture Notes in Computer Sci., 84 Springer Verlag, pp. 307-319 (1980).

[8] K. Santarek and I.M. Buseif, An Approach to Manufacturing Systems Design Using SADT and Petri Net Tools. Industrial Control and Management Methods Theory and Practice, Preprints of the Dycoman's Workshop (Prague) Czech Republic October, pp. 5-7 (1995).

[9' K. Santarek, I.M. Buseif, An Approach to AMS Modelling and Design and its Application for a Protorype CIM Solution, International Conference on Computer Integrated Manufacturing, Zakopane-Poland, pp. 14-17 (1996).

[10] Z. Banaszak and K. Jedrzejek, "Rule-Based Knowledge Verification Using Petri Nets". The Third Turkish Symposium on AI and Networks, pp. 19-28 (1994).

Received April 10, 2005
Accepted March 2, 200