IEEE Int. Conference on Systems, Man and Cybernetics (SMC'94), San Antonio, TX, 2-5 October 1994, pp.1328-1333.
Copyright © 1994 IEEE (DOI 10.1109/ICSMC.1994.400029).

# Throughput Analysis of Manufacturing Cells Using Timed Petri Nets

W.M. Zuberek    W. Kubiak

Department of Computer Science    Faculty of Business Administration
Memorial University of Newfoundland
St. John's, Canada A1C–5S7

## Abstract

It is shown that a large class of flexible manufacturing cells can be modeled by timed Petri nets. Several net transformations are proposed to simplify the structure of the modeling nets while preserving their performance properties. The throughput of the model can be obtained by applying simple rules of operational analysis to simplified nets. Several extensions of the proposed approach are also indicated.

## 1. INTRODUCTION

Flexible manufacturing systems fill the gap between flow shops which manufacture a large quantity of similar products, and job shops which produce a large diversity of products in small quantities. They utilize versatile machines usually clustered in manufacturing cells and connected by automatic material–handling equipment such a robots which perform sequences of pickup, move, load, unload and drop operations, transporting the manufactured parts from one machine of the cell to another [SSS92, Cl83]. The throughput of the cell depends on the sequence of robot activities as well as on the sequence in which different parts enter the cell [DH90]. The problem of maximizing the throughput of a robotic cell can thus be considered as a scheduling problem.

The behavior of flexible manufacturing systems is represented by *events* and *activities*; an event corresponds to a change of system's state while an activity corresponds to an operation performed by a machine or a robot. Different sets of (simultaneous) activities determine the states of the system. In each state, several activities can occur concurrently, for example, several machines can perform their operations simultaneously and the robot can also transport a part. Petri nets provide a simple and convenient formalism for modeling systems that exhibit parallelism and concurrency [Mu89, Re85]. In fact, one of the very first applications of Petri net models was to analyze production schemata [Ha72]. However, the complexity of realistic manufacturing systems combined with not well–understood nature of concurrency, seem to be the reasons of limited popularity of these models for a number of years [Su85].

In order to study performance aspects of Petri net models, the duration of activities must also be taken into account and included into model specifications. Several types of Petri nets 'with time' have been proposed by assigning 'firing times' to the transitions or places of a net. In stochastic, the transition firings are instantaneous events, as in ordinary (i.e., 'untimed') nets, however, tokens are delayed in places for (exponentially distributed) time periods, determined by transitions connected with the place. In timed nets, transition firings are 'real–time' events, i.e., tokens are removed from input places at the beginning of the firing period, and they are deposited to the output places at the end of this period (sometimes this is also called a "three–phase" firing mechanism). The firing times may be either deterministic or stochastic, i.e., described by some probability distribution function. In both cases the concepts of state and state transitions have been formally defined and used in derivation of different performance characteristics of the model [Zu91].

Analysis of net models can be based on their behavior (i.e., the space of reachable states) or on the structure of the net; the former is called reachability analysis while the latter structural analysis. Structural methods eliminate the derivation of the state space, so they avoid the 'state explosion' problem of reachability analysis, but they cannot provide as much information as the reachability approach does. Quite often, however, all the detailed results of reachability analysis are not really needed, and more synthetic performance measures, that can be provided by structural methods, are quite satisfactory [Hi89].

This paper investigates the steady–state behavior of flexible manufacturing cells. Simple schedules, i.e., schedules in which exactly one part enters and one leaves the cell in each cycle, are modeled by conflict–free timed Petri nets. The throughput of the whole model is determined by reduction of the modeling net to an equivalent net with a very simple structure, for which the throughput can be directly determined from the net.

Section 2 recalls basic concepts of timed Petri nets. The definition and some properties of throughput in timed nets are presented in Section 3 while Section 4 introduced several useful net transformations which simplify net structure but preserve net performance properties. Timed net models of simple schedules and their throughput analysis are discussed in Section 5. Section 6 concludes the paper.

## 2. TIMED PETRI NETS

Only the few most important definitions of conepts related to timed Petri nets are recalled in this section. More detailed presentation can be found in [Zu88, Zu91].

A Petri net (or a Petri net structure) N is a triple N=$(P, T, A)$ where $P$ is a finite, nonempty set of places, $T$ is a finite, nonempty set of transitions, and $A$ is a set of directed arcs, $A \subseteq P \times T \cup T \times P$, such that for each transition there is at least one place connected with it. A marked Petri net $\mathcal{M}$ is a pair $\mathcal{M} = (\mathcal{N}, m_0)$ where $\mathcal{N}$ is a Petri net, $\mathcal{N} = (P, T, A)$, and $m_0$ is an initial marking function, $m_0 : P \to \{0, 1, ...\}$, assigning tokens places of the net.

A shared place $p$ is shared if it is connected to more than one transition. A net is conflict–free if it does not contain shared places. Only conflict–free net are considered in this paper since the models of schedules discussed here are conflict–free nets. A very similar approach can be used for more general class of free–choice nets.

A transition $t$ is enabled by a marking $m$ iff every input place of this transition contains at least one token. Every transition enabled by a marking $m$ can fire. When a transition fires, a token is removed from each of its input places and a token is added to each of its output places. This determines a new marking in a net, a new set of enabled transitions, and so on.

A conflict–free timed Petri net $\mathcal{T}$ is a pair $\mathcal{T} = (\mathcal{M}, f)$ where $\mathcal{M}$ is a conflict–free marked Petri net, $\mathcal{M} = (\mathcal{N}, m_0)$, $\mathcal{N} = (P, T, A)$, and $f$ is a firing time function which assigns the nonnegative (average) firing time $f(t)$ to each transition $t$ of the net, $f : T \to \mathbf{R}^{\oplus}$, and $\mathbf{R}^{\oplus}$ denotes the set of nonnegative real numbers.

The behavior of a timed Petri net can be represented by a sequence of 'states' where each 'state' describes the distribution of tokens in places and in firing transitions of the net; detailed definitions of states and state transitions are given in [Zu88, Zu91]. The states and state transitions can be combined into a graph of reachable states or simply state graph; this graph is a semi–Markov process defined by the timed net $\mathcal{T}$. The timed net is bounded iff its state graph is finite. Only bounded nets are considered in this paper.

The state graph for bounded conflict–free nets is either a simple (finite) path or a simple cycle. Net models of manufacturing cells have cyclic state graphs which represent the cyclic behavior of the manufacturing cells. The cycle time of the model determines the cycle time and the throughput of the modeled cell. It appears the the throughput of the model can easily be determined without the exhaustive analysis of the state space by using a few simple net transformations which simplify the structure of the net while preserving its performance properties.

## 3. THROUGHPUTS IN NETS

Intuitively, throughput of a place $p$ in a timed net $\mathcal{T}$, $\theta_{\mathcal{T}}(p)$, is equal to the average number of tokens entering $p$ in a unit time, or leaving $p$ (or $t$) in a unit time; in the steady–state of the net, the average numbers of tokens entering and leaving $p$ must be equal since no 'accumulation' of tokens can occur. Similarly, throughput of a transition $t$ in a net $\mathcal{T}$, $\theta_{\mathcal{T}}(t)$, is equal to the average number of new (or completed) transition's firings in a unit time. It should be noted that the throughput of a transition does not depend upon the number of incoming or outgoing arcs.

More formally, the throughput of a timed net $\mathcal{T}$ is defined as a function $\theta : P \cup T \to \mathbf{R}^{\oplus}$ which assigns a nonegative number to each place and each transition of the net in such a way that:

$$\forall (x \in P \cup T) \ \theta(x) = \lim_{n \to \infty} \frac{n}{\tau_n(x)}$$

where $\tau_n(x)$ denotes the time instant at which the $n$-th consecutive token enters (or leaves) the place $x$ or at which the transition $x$ initiates (or terminates) its $n$-th firing.

It follows immediately from the definition of throughput that:

- the throughput of a place $p$ is equal to the sum of throughputs of its input transitions as well as the sum of throughputs of its output transitions:

  $\forall (p \in P) \ \theta(p) \quad = \sum_{t_i \in Inp(p)} \theta(t_i)$
  $= \sum_{t_j \in Out(p)} \theta(t_j)$

- for each (non–shared) place $p$, the throughput of $p$'s output transition is equal to the throughput of $p$:

  $\forall (p \in P) \ Out(p) = \{t\} \Rightarrow \theta(t) = \theta(p),$

An elementary net is a connected net in which there is exactly one input place and exactly one output place for each transition of the net, and one input transition and one output transition for each place of the net. In other words, the (directed) graph of an elementary net is a (simple) cycle. It follows from property 1 that in elementary nets the throughputs of all transitions and all places are the same. To determine the value of these throughputs, the Little's law [Ki90] can be applied to an elementary net considered as an open system (by braking one of the arcs) in which the mean response time is equal to the sum of (the average) firing times of all transitions.

For a timed elementary net $\mathcal{T}$:

$$\forall(x \in P \cup T)\ \theta(x) = \frac{\sum_{p \in P} m_0(p)}{\sum_{t \in T} f(t)}$$

Analysis of nets in which transitions can have more than one input arc must take into account 'synchronization delays' which do not exist in state graphs (and elementary nets). Since firing of a transition removes (single) tokens from all input places simultaneously, some tokens may wait (in places) for the enabling of a transition. Such waiting times will affect the response time of timed models. However, there are simple solutions to some special cases of nets.

An omega net is a net composed of two elementary nets which have exactly one common transition. Consequently, the throughput of all elements is the same, and is equal to

$$\theta(\mathcal{T}) \ = \ \min(\theta(\mathcal{T}_1), \theta(\mathcal{T}_2))$$

where $\theta(\mathcal{T}_1)$ and $\theta(\mathcal{T}_2)$ are throughputs of the first and the second elementary subnets, respectively (when considered independently of the other).

## 4. NET TRANSFORMATIONS

It can easily be shown that different timed nets may have isomorphic state graphs, which means that the behavior of such nets is "equivalent" in the sense of stationary probabilities of states and performance properties which can be derived from these stationary probabilities [Zu82]. Some of net transformation which preserve such "equivalence" are as follows:

**(a) Backward split:** A simple transition followed by a "fork" one, as shown in Fig.1(a), is equivalent to a simple transition replaced by a "fork" with parallel paths.

**(b) Forward split:** A "join" transition followed by a simple one, as shown in Fig.1(b), is equivalent to the simple transition replaced by a "join" and two parallel transitions replacing the original "join".

**(c) Removal of parallel paths:** In the case of parallel paths (as shown in Fig.1(c)), the first transition has the firing time which is equal to the smaller of the two original firing times, and the the second transition has the firing time equal to the difference of the two original firing times. The transformation can be applied not only to simple parallel transitions but any parallel paths. In particular, if one of parallel paths contains only a place, the path can be removed without affecting the performance of the model (Fig.1(d)).

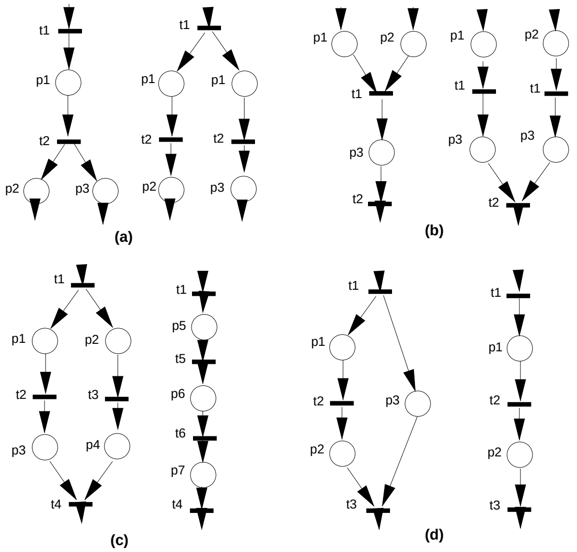

Fig.1. Net transformations.

## 5. SIMPLE SCHEDULES AND THEIR THROUGHPUTS

Simple schedules are schedules in which exactly one part enters and one leaves the cell in each cycle. A detailed description of timed Petri net models of simple schedules is given in [ZK94]; only main aspects are recalled here. Timed net models of manufacturing cells are composed of:

- transitions representing machine operations; firing times associated with these transitions are average operation times; for each machine there are also two places representing conditions "part ready for operation" (i.e., loaded) and "operation finished" (and the part ready for unloading),

- a cyclic directed path modeling the robot's schedule; this path is composed of transitions and places connected by directed arcs; the transitions model robot's operations "pick–move–load", "unload–move–load" and "unload–move–drop" with firing times representing durations of the operations; the places represent "readiness for another operation".

The robot actions are 'naturally' synchronized with machine operations; a machine cannot start its operation unless it is loaded; the robot cannot unload a machine until its operation is finished, etc. All such synchronizations are represented by transitions with two input places (robot and machine).

A simple manufacturing cell composed of three machines and a robot is sketched in Fig.2 [SSS92]; the machines are denoted by $M_1$, $M_2$ and $M_3$, *In* represents a conveyor bringing the parts in, while *Out* a conveyor for outgoing parts. It is assumed (for simplicity) that each part has to go through $M_1$, $M_2$ and $M_3$, in that order before leaving the cell.

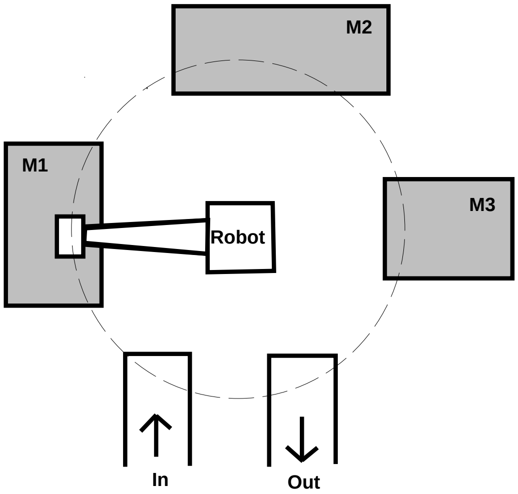

Fig.2. An outline of a three–machine cell.

There are six possible robot's simple schedules [SSS92]; the schedules differ in the order of robot's actions. Denoting the robot moves from $X$ to $Y$ by $X \Rightarrow Y$ if the robot carries a part and by $X \rightarrow Y$ otherwise, the schedules are as follows:

A: $In \Rightarrow M_1 \Rightarrow M_2 \Rightarrow M_3 \Rightarrow Out \rightarrow In$
B: $In \Rightarrow M_1 \Rightarrow M_2 \rightarrow M_3 \Rightarrow Out \rightarrow M_2 \Rightarrow M_3 \rightarrow In$
C: $In \Rightarrow M_1 \rightarrow M_3 \rightarrow Out \rightarrow M_1 \Rightarrow M_2 \Rightarrow M_3 \rightarrow In$
D: $In \Rightarrow M_1 \rightarrow M_2 \Rightarrow M_3 \rightarrow M_1 \Rightarrow M_2 \rightarrow M_3 \Rightarrow Out \rightarrow In$
E: $In \Rightarrow M_1 \rightarrow M_2 \Rightarrow M_3 \Rightarrow Out \rightarrow M_1 \Rightarrow M_2 \rightarrow In$
F: $In \Rightarrow M_1 \rightarrow M_3 \Rightarrow Out \rightarrow M_2 \Rightarrow M_3 \rightarrow M_1 \Rightarrow M_2 \rightarrow In$

A timed Petri net model of the cell from Fig.2 with the schedule A is shown in Fig.3. The three machines (or rather machine operations) are represented by $t_1$, $t_2$ and $t_3$, each transition with its input and output place (for the conditions 'part loaded' – $p_{10}$, $p_{21}$ and $p_{32}$ – and 'machine operation finished' – $p_{12}$, $p_{23}$ and $p_{34}$). The firing times associated with these transitions, $f(t_1) = o_1$, $f(t_2) = o_2$ and $f(t_3) = o_3$, represent the average times of performing the operations on machines $M_1$, $M_2$ and $M_3$, respectively.

The sequence of robot actions is represented by the path $t_{01}$, $t_{12}$, $t_{23}$, $t_{34}$, $t_{40}$ with the following 'execution' times ('$a$' denotes the pickup time, '$b$' the unload time, '$c$' the load time, '$d$' the drop time and '$e$' the 'travel' time between machines; it is assumed, for simplicity, that all travel times between two adjacent machines are the same, and are also the same for *Out* to *In* move, *In* to $M_1$ as well as $M_3$ to *Out* move):

| | *operation* | *time* |
|---|---|---|
| $t_{01}$ | pick from *In*, move to $M_1$, load | $a+c+e$ |
| $t_{12}$ | unload $M_1$, move to $M_2$ and load | $b+c+e$ |
| $t_{23}$ | unload $M_2$, move to $M_3$ and load | $b+c+e$ |
| $t_{34}$ | unload $M_3$, move to *Out*, drop | $b+d+e$ |
| $t_{40}$ | move from *Out* to *In* | $e$ |

It is assumed that there is always an available part in *In* and that *Out* removes manufactured parts so quickly, that there is never any waiting; consequently, *In* and *Out* are not shown in the model although they could easily be added if needed.

It can be observed that the arcs incident with $p_{11}$, $p_{22}$ and $p_{33}$ form parallel paths to transitions $t_1$, $t_2$ and $t_3$, respectively. Removal of all these arcs results in an an elementary net, so the throughput for schedule A is simply:

$$\theta_A \ = \ 1/(o_1 + o_2 + o_3 + a + 3b + 3c + d + 5e)$$

A model of the robot's schedule B is shown in Fig.4; $t_1$, $t_2$ and $t_3$ represent the machine operations, as in Fig.3, and the remaining transitions correspond to the following robot's actions:

| | *operation* | *time* |
|---|---|---|
| $t_{01}$ | pick from *In*, move to $M_1$, load | $a+c+e$ |
| $t_{12}$ | unload $M_1$, move to $M_2$ and load | $b+c+e$ |
| $t_{23}$ | unload $M_2$, move to $M_3$ and load | $b+c+e$ |
| $t_{30}$ | move from $M_3$ to *In* | $2e$ |
| $t_{32}$ | move from $M_2$ to $M_3$ | $e$ |
| $t_{34}$ | unload $M_3$, move to *Out*, drop | $b+d+e$ |
| $t_{42}$ | move from *Out* to $M_2$ | $2e$ |

The initial marking function corresponds to the distribution of parts in the cell; for the schedule B (Fig.4), the initial token in $p_{03}$ represents the robot ready to pick a part from the input conveyor, while the token in $p_{32}$

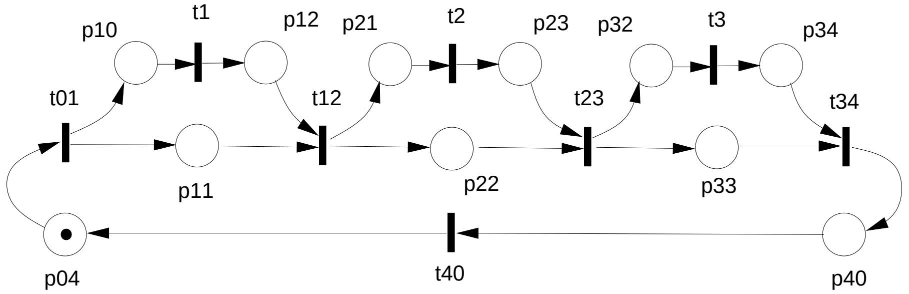

Fig.3. Net model of schedule A.

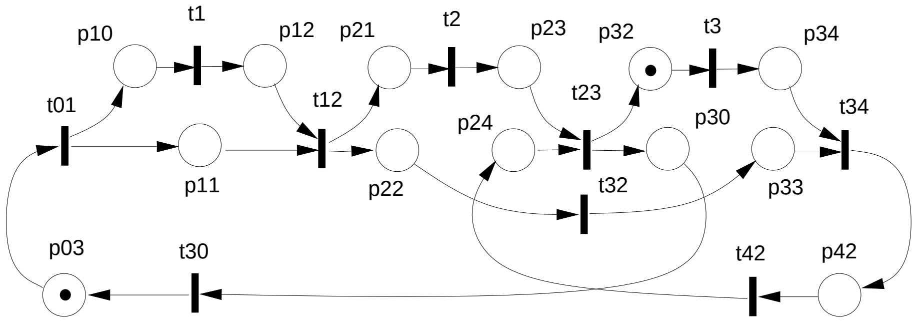

Fig.4. Net model of schedule B.

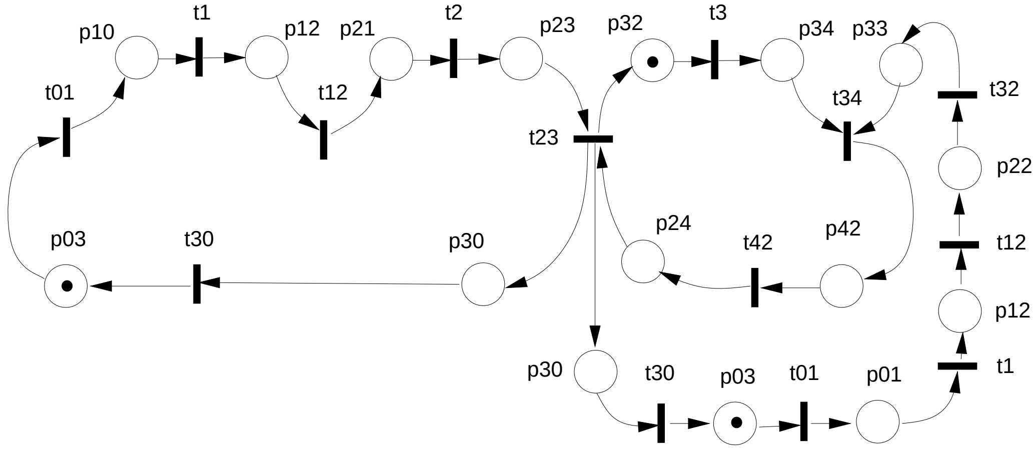

Fig.5. Transformed model of schedula B.

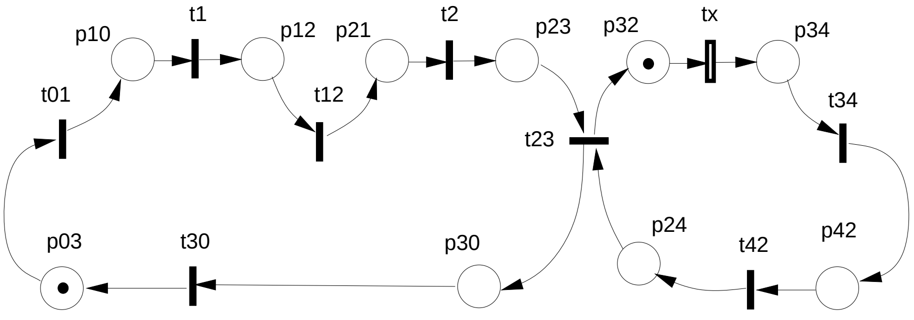

Fig.6. Simplified model of schedule B.

$f(t_x) = \max(f(t3), f(t_{30}) + f(t_{01}) + f(t_1) + f(t_{12}) + f(t_{32}))$ creates an omega net shown in Fig.6. Its throughput is:

$$\begin{aligned}\theta_B \;&=\; \min(1/(o_1 + o_2 + a + 2b + 3c + 5e),\\ &\quad 1/(2b + c + d + 4e + \max(o_3, o_1 + a + b + 2c + 5e)))\end{aligned}$$

Similar models can easily be derived for other simple schedules.

## 6. CONCLUDING REMARKS

It has been shown that timed Petri nets can conveniently model (at least a class of) flexible manufacturing cells. For simple schedules, the modeling nets are composed of conflict–free nets, and the throughput can easily be determined by reducing the original model to a simpler one, for which the solution can be obtained directly. The solution is obtained in an analytical (or symbolic) form, in which all times of basic operations (like ”load”, “unload”, “move”) are parameters.

The evaluation of the net models discussed in Sections 5 can be used for the determination of optimal schedules. For example, for the manufacturing cell of Fig.2, there are six different simple schedules (models of two of them are shown in Fig.3 and Fig.4); the optimal schedule in this case is the schedule which maximizes the throughput:

$$\theta_{opt} = \max(\theta_A, \theta_B, ..., \theta_F)$$

The optimization procedure (based on systematic analysis of feasible schedules) can be automated. However, the number of schedules can grow very quickly with the number of cell components. This large number can be significantly reduced if specific values of (some) parameters are taken into account.

Several simplifying assumptions were used in the discussion presented in previous sections, e.g., all parts are identical, robot ‘travel times’ are equal, etc. It should be noted, however, that all these assumptions were made to simplify the discussion and they can easily be removed by straightforward modification of the presented approach because all these assumptions do not affect the structure of the model.

Only simple schedules are discussed in this paper, but a similar approach can be developed for schedules in which several parts enter (and leave) the cell within a single cycle. All such ‘composite’ schedules can be obtained from of a number of simple schedules combined together. A systematic method of ‘consistent’ compositions of simple schedules, combined with a similar method of analysis can be applied to composite schedules.

## References

[Cl83] Claybourne, B.H.: “Scheduling robots in flexible manufacturing cells”; CME Automation, vol.30, no.5, pp.36–40, 1983.

[DH90] Dixon, C., Hill, S.D.: “Work–cell cycle–time analysis in a flexible manufacturing system”; Proc. Pacific Conf. on Manufacturing, Sydney–Melbourne, Australia, vol.1, pp.182–189, 1990.

[Ha72] Hack, M.: “Analysis of production schemata by Petri nets”; Project MAC Technical Report TR–94, 1972.

[Hi89] Hillion, H.P.: “Timed Petri nets and application to multi-stage production system”; in: Advances in Petri Nets 1989 (Lecture Notes in Computer Science 424); pp.281–305, Springer Verlag 1989.

[Ki90] King, P.J.B.: “Computer and communication systems performance modeling”; Prentice–Hall 1990.

[Mu89] Murata, T.: “Petri nets: properties, analysis and applications”; Proceedings of IEEE, vol.77, no.4, pp.541–580, 1989.

[Re85] Reisig, W.: “Petri nets - an introduction” (EATCS Monographs on Theoretical Computer Science 4); Springer Verlag 1985.

[SSS92] Sethi, S.P., Sriskandarajah, C., Sorger, G., Blazewicz, J., Kubiak, W.: “Sequencing of parts and robot moves in a robotic cell”; Int. Journal of Flexible Manufacturing Systems, vol.4, pp.331–358, 1992.

[SV89] Silva, M., Valette, R.: “Petri nets and flexible manufacturing”; in: “Advances in Petri nets 1989” (Lecture Notes in Computer Science 424), pp.374–417, Springer Verlag 1989.

[Su85] Suri, R.: “An overview of evaluative models for flexible manufacturing systems”; Annals of Operations Research, vol.3, no.1, pp.3–21, 1985.

[Zu82] Zuberek, W.M.: “Application of timed Petri nets to analysis of multiprocessor realizations of digital filters”; Proc. 25–th Midwest Symp. on Circuits and Systems, Houghton MI, pp.134–139, 1982.

[Zu88] Zuberek, W.M.: “D–timed Petri nets and modeling of timeouts and protocols”; Transactions of the Society for Computer Simulation, vol.4, no.4, pp.331–357, 1988.

[Zu91] Zuberek, W.M.: “Timed Petri nets – definitions, properties and applications”; Microelectronics and Reliability (Special Issue on Petri Nets and Related Graph Models), vol.31, no.4, pp.627–644, 1991.

[ZK94] Zuberek, W.M., Kubiak, W.: “Modeling simple schedules of manufacturing cells using timed Petri nets”, Proc. Int. Workshop on Intelligent Systems and Innovative Computations, Tokyo, Japan, 1994.