9-th Symp. on Information Control in Manufacturing; Nancy–Metz, France, June 24–26, 1998, pp.423-428.

HIERARCHICAL DERIVATION OF SCHEDULES FOR MANUFACTURING CELLS

W.M. Zuberek

Department of Computer Science
Memorial University of Newfoundland
St.John's, Canada A1B 3X5

Abstract: An approach to systematic derivation of schedules for manufacturing cells is presented. The approach is based on model refinements in which simple model elements are replaced by more detailed submodels. For manufacturing cells, the refinements are used to introduce additional machines into a cell. The hierarchy of schedules is derived starting from a very simple cell, and increasing the complexity of schedules in consecutive steps. Timed Petri nets are used to represent the schedules and the refinements of models. *Copyright ©1998 IFAC.*

Keywords: manufacturing systems, modeling, hierarchical structures, Petri nets.

1. INTRODUCTION

Flexible manufacturing systems are typically composed of sequentially arranged, programmable manufacturing cells (or robotic cells), linked by a common transport system which moves the machined or assembled parts between the cells or between the cells and storage areas (Ayers and Butcher, 1993).

A manufacturing cell is composed of a number of machines and a robot (or a robotic arm) which moves the parts from one machine to another, from the input conveyor to the first machine and from the last machine to the output conveyor. The throughput of a cell depends on the sequence of robot moves as well as on the sequence in which parts enter the cell (Dixon and Hill, 1990). The (cyclic) sequence of operations performed by the robot is called a schedule. Maximizing the throughput of a robotic cell is thus equivalent to finding a schedule with minimal cycle time. Systematic generation of alternative schedules combined with efficient evaluation of the performance of each generated schedule provides a basis for performance optimization of robotic cells.

The behavior of a manufacturing cell is represented by 'events' and 'activities'; an activity corresponds to an operation performed by a machine or a robot, and an event corresponds to any change of the cell's activities. Different sets of activities determine the 'states' of the system, and in each state, several activities can occur concurrently, for example, an operation can be performed by one of the machines, and the robot can also transport a part from one machine to another. Petri nets provide a simple and convenient formalism for modeling system that exhibit concurrent activities (Reisig, 1985, Murata, 1989); they have been successfully used in modeling and analysis of manufacturing systems (DiCesare et al., 1993, Desrochers and Al-Jaar, 1995, Hillion, 1989).

Place/transition Petri nets are formal, abstract models of systems which contain concurrent, interacting components, with constraints on precedence and frequency of these interactions. Petri nets are composed of two types of elements, called 'places' and 'transitions', connected by directed arcs. The dynamic behavior is represented by 'tokens' which are assigned to places, and which – when some conditions are satisfied – can change their assignments. Transitions represent events, while places correspond to conditions; a condition is satisfied only if a place representing it has some tokens assigned to it. If all conditions of an event (i.e., places connected by directed arcs to a transition) are satisfied, the event (represented by this transition) can occur, and can change the distribution of tokens in the net, creating a new set of satisfied conditions, new events that can occur, and so on.

In order to study the performance aspects of Petri net models, the duration of activities must also be taken into account. Several types of Petri nets 'with time' have been proposed by assigning 'firing times' to transitions or 'enabling times' to places. In timed Petri nets, the events occur in 'real time', i.e., there is a (deterministic or stochastic) duration associated with each transition's firing, and different (concurrent) firings of transitions corre-

spond to (concurrent) activities in modeled systems. For timed Petri nets, the concept of 'state' and state transitions can be formally defined, and used to derive different performance characteristics of the model (Zuberek, 1991).

This paper describes an approach to systematic generation of schedules for manufacturing cells. This generation is based on refinements of timed Petri nets. Performance evaluation of generated schedules is not discussed here as it has beed addressed elsewhere (Zuberek, 1995, Zuberek and Kubiak, 1994). In net refinements, a net element (a transition in this case) is replaced by a subnet, creating a more complex model. Each such net refinement corresponds to adding a new machine to a manufacturing cell. The starting point for refinements is very simple, as it corresponds to an empty cell, i.e., a cell without any machine. The refinement process is performed systematically, so a complete set of schedules for a simpler cell is refined into a complete set of schedules for a more complicated cell. These derived schedules can be used for the derivation of schedules for an even more complicated cell, and so on.

2. MANUFACTURING CELLS AND THEIR SCHEDULES

Fig.2.1 outlines a simple cell with three machines, $M1$, $M2$ and $M3$, an input conveyor In, an output conveyor Out, and a robot.

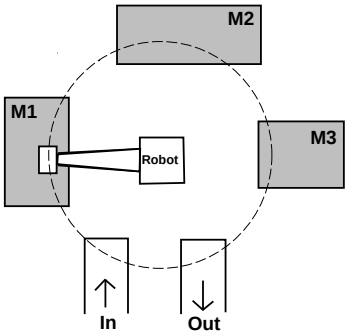

Fig.2.1. Layout of a 3–machine cell.

It is known (Smith at al., 1992) that for a cell with m machines, there are $m!$ different schedules. These schedules can be characterized by sequences of "move" operations (performed by the robot) which correspond to parts transported from one machine to another, from the input conveyor to the first machine, and from the last machine to the output.

Assuming (for simplicity) that all parts follow the same path through the cell, and that this path is from the input conveyor to the first machine, then from the first machine to the second, and so on, the six 3–machine schedules, denoted here A, B, ..., F, are as follows (in the schedules, element '01' denotes the operation of picking a new part from the input conveyor ('0'), moving it to the first machine ('1') and loading the part; similarly, element '12' denotes unloading the first machine, transporting the unloaded part to the second machine and loading the part, and element '30' – unloading the third machine, moving to the output conveyor and depositing the part there):

A: 01 – 12 – 23 – 30
B: 01 – 12 – 30 – 23
C: 01 – 23 – 12 – 30
D: 01 – 23 – 30 – 12
E: 01 – 30 – 12 – 23
F: 01 – 30 – 23 – 12

Since all schedules are cyclic, it is (arbitrarily) assumed that a uniform 'beginning' of all schedules is the operation of picking a new part from the input, transporting it and loading on the first machine of the cell (element '01').

The schedules for a 3–machine cell can conveniently be derived from the two schedules for a 2–machine cell:

01 – 12 – 20
01 – 20 – 12

by first replacing entries '20' by '23' (which is due to the additional machine in the cell), and then inserting the new entry '30' in all three possible positions of each 2–machine schedule, i.e., after the third entry, between the second and the third entries, and between the first and the second entries (the first entry must remain '01' as the 'standard beginning'); so, "01 – 12 – 20" expands into schedules:

01 – 12 – 23 – 30 (schedule A)
01 – 12 – 30 – 23 (schedule B)
01 – 30 – 12 – 23 (schedule E)

and "01 – 20 – 12" into:

01 – 23 – 12 – 30 (schedule C)
01 – 23 – 30 – 12 (schedule D)
01 – 30 – 23 – 12 (schedule F)

In the same way, the 3–machine schedules can be expanded into 4–machine ones (there are 24 such schedules), and the 2–machine schedules can be derived from the single 1–machine schedule:

01 – 10

Each of these schedules can easily be transformed into a complete sequence of robot's operations by adding all those robot's moves which are necessary to perform the required transport functions. For example, the 3–machine schedule D, "01 – 23 – 30 – 12", requires that the robot, after loading the new part on machine $M1$ (element '01'), move to machine $M2$ to unload it, transport the unloaded part to machine $M3$ and load the part (element

'23'); then the robot waits until the $M3$'s operation is finished, unloads the part and transports is to the output conveyor and drops it there (element '30'), after which it moves to machine $M1$ to unload it and transport the unloaded part to machine $M2$ (element '12'), and finally moves to the input conveyor to start another cycle.

The detailed schedule of the robot is a sequence of moves in which the robot carries a part (this moves are denoted by $\Rightarrow$); the robot also performs 'empty' moves, without carrying a part (denoted by $\rightarrow$):

A: $In \Rightarrow M1 \Rightarrow M2 \Rightarrow M3 \Rightarrow Out \rightarrow In$

B: $In \Rightarrow M1 \Rightarrow M2 \rightarrow M3 \Rightarrow Out \rightarrow M2 \Rightarrow M3 \rightarrow In$

C: $In \Rightarrow M1 \rightarrow M2 \Rightarrow M3 \rightarrow M1 \Rightarrow M2 \rightarrow M3 \Rightarrow Out \rightarrow In$

D: $In \Rightarrow M1 \rightarrow M2 \Rightarrow M3 \Rightarrow Out \rightarrow M1 \Rightarrow M2 \rightarrow In$

E: $In \Rightarrow M1 \rightarrow M3 \Rightarrow Out \rightarrow M1 \Rightarrow M2 \Rightarrow M3 \rightarrow In$

F: $In \Rightarrow M1 \rightarrow M3 \Rightarrow Out \rightarrow M2 \Rightarrow M3 \rightarrow M1 \Rightarrow M2 \rightarrow In$

3. PETRI NET MODELS

This section recalls basic concepts of timed Petri nets and net refinements. A more detailed discussion can be found in (Murata, 1989, Reisig, 1985, Zuberek, 1991).

A place/transition net $\mathbf{N}$ is a triple $\mathbf{N} = (P, T, A)$ where P is a finite, nonempty set of places, T is a finite, nonempty set of transitions, A is a set of directed arcs, and $A \subseteq P \times T \cup T \times P$, such that for each transition there exists at least one place connected with it. For each place p (and each transition t) the input set, $Inp(p)$ (or $Inp(t)$), is the set of transitions (or places) connected by directed arcs with p (or t). The output sets, $Out(p)$ and $Out(t)$, are defined similarly.

A marked Petri net $\mathbf{M}$ is a pair $\mathbf{M} = (\mathbf{N}, m_0)$ where $\mathbf{N}$ is a Petri net, $\mathbf{N} = (P, T, A)$, and m_0 is an initial marking function, $m_0 : P \rightarrow \{0, 1, ...\}$ which assigns a (nonnegative) number of tokens to each place of the net.

Let any function $m : P \rightarrow \{0, 1, ...\}$ be called a marking in a net $\mathbf{N} = (P, T, A)$.

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. The set of all markings that can be derived from the initial marking is called the set of reachable markings. If this set if finite, the net is bounded.

A place p is shared iff it is an input place for more than one transition. A net is free–choice if the input sets of all transitions sharing the same place are identical. A net is (structurally or statically) conflict–free is it does not contain shared places. It is (dynamically) conflict–free if for any marking in the set of reachable markings, and for any shared place, at most one of transitions sharing this place is enabled. Only bounded conflict–free nets are considered in this paper.

Refinements in Petri nets can be defined in several ways; a convenient approach, proposed in Zuberek and Bluemke, 1996, refines a net by replacing a single element (a transition or a place) by a subnet connected to the input and output sets of the replaced element.

More formally, a refinement system $\mathcal{R}$ is defined as a 5–tuple, $\mathcal{R} = (\mathbf{M}_0, \mathcal{M}, \rho, \phi, \psi)$, where:

- $\mathbf{M}_0$ is a marked (initial) place/transition net, $\mathbf{M}_0 = (P_0, T_0, A_0, m_{0,0})$,
- $\mathcal{M}$ is a family of (marked) place/transition refinement nets, $\mathcal{M} = \{\mathbf{M}_1, ..., \mathbf{M}_k\}$,
- ρ is a (partial) refinement function which associates elements of P_0 (place refinements) and T_0 (transition refinements) with nets from $\mathcal{M}$, $\rho : P_0 \cup T_0 \rightarrow \{1, ..., k\}$, so each place $p \in P_0$ is refined by the net $\mathbf{M}_{\rho(p)}$ (if $p \in Dom(\rho)$, otherwise p remains a simple place), and each transition $t \in T_0$ is refined by $\mathbf{M}_{\rho(t)}$ (if $t \in Dom(\rho)$).
- ϕ and ψ are (input and output) interface functions which define the interconnections between the input and output sets of a place (or transition) and its refinement determined by ρ; for each $p \in P_0$, if $p \in Dom(\rho)$, then $\phi(p) : T_0 \rightarrow 2^{P_{\rho(p)}}$ and $\psi(p) : T_0 \rightarrow 2^{P_{\rho(p)}}$; similarly, for each $t \in T_0$, if $t \in Dom(\rho)$, then $\phi(t) : P_0 \rightarrow 2^{T_{\rho(t)}}$ and $\psi(t) : P_0 \rightarrow 2^{T_{\rho(t)}}$.

For example, Fig.3.1(a) shows a net that can be refined by using the net shown in Fig.3.1(b) as a replacement of transition t_2; i.e., first t_2 and all arcs incident with it are removed from the net in Fig.3.1(a), and then the remaining net is connected to the net in Fig.3.1(b) by new arcs: from p_4 to t'_3, from t'_5 to p_3, and from t_3 to p_1. The resulting net is shown in Fig.3.2 with the new connecting arcs indicated by dashed lines.

For this example, $\mathbf{M}_0$ is the net in Fig.3.1(a), $\mathcal{M} = \{\mathbf{M}_1\}$, where $\mathbf{M}_1$ is the net in Fig.3.1(b), and the partial functions ρ, ϕ and ψ are as follows:

$$\forall x \in P_0 \cup T_0 : \rho(x) = \begin{cases} 1, \text{if } x = t_2; \\ \text{undefined otherwise;} \end{cases}$$

$$\forall p \in P_0 : \phi(t_2)(p) = \begin{cases} \{t'_3\}, \text{if } p \in Inp(t_2) = \{p_4\}; \\ \text{undefined otherwise;} \end{cases}$$

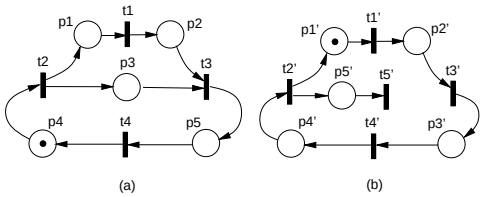

Fig.3.1. Petri net (a) and its refinement (b).

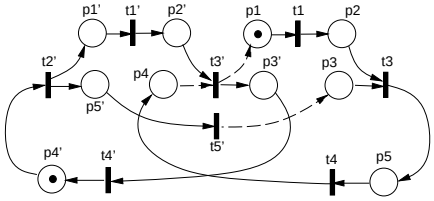

Fig.3.2. Refined net.

$$\forall p \in P_0 : \psi(t_2)(p) = \begin{cases} \{t_3'\}, \text{if } p \in Out(t_2) \wedge p = p_1; \\ \{t_5'\}, \text{if } p \in Out(t_2) \wedge p = p_3; \\ \text{undefined otherwise.} \end{cases}$$

In timed Petri nets each transition takes a 'real time' to fire, i.e., there is a 'firing time' associated with each transition of a net which determines the duration of transition's firings. Firing times are used to evaluate the performance of net models, an aspect which has been explored elsewhere (Zuberek and Kubiak, 1994, Zuberek, 1995).

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 firing transitions of the net (Zuberek, 1991). The states and state transitions can be combined into a graph of reachable states; for nets with deterministic firing times, this graph is a semi–Markov process defined by the timed net. For timed conflict–free nets, the reachability graphs are simple cycles which represent the cyclic behavior of such nets. Each such timed Petri net contains a basic subnet with the cycle time equal to the cycle time of the whole net. Moreover, all other basic subnets have cycle times which are not greater than the cycle time of the net; the cycle time of the net is thus equal to the maximum cycle time of its basic invariant subnets (Zuberek, 1995).

Timed Petri net models of manufacturing cell schedules can be derived from the detailed descriptions of schedules. For a cell with m machines, the model will contain m sections representing the machines, each section composed of a single transition representing the operations of the machine, i.e., with firing time equal to the (average) time of the operation, and with a single input place (representing the condition "part loaded") and a single output place (representing the condition "machine's operation completed"). The remaining part of the model is the representation of the sequence of robot's operations. For example, the net model of schedule B (for a 3–machine cell from the previous section) is shown in Fig.3.3.

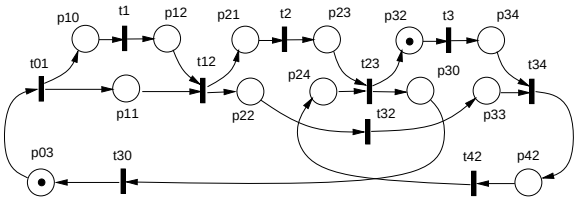

Fig.3.3. Petri net model of schedule B.

The three machines are represented by t_1 (with p_{10} and p_{12}), t_2 (with p_{21} and p_{23}), and t_3 (with p_{32} and p_{34}). The remaining part is the representation of the robot's moves:

robot	*net elements*
$In \Rightarrow M1$	p_{03}, t_{01}, p_{11}
$M1 \Rightarrow M2$	p_{11}, t_{12}, p_{22}
$M2 \rightarrow M3$	p_{22}, t_{32}, p_{33}
$M3 \Rightarrow Out$	p_{33}, t_{34}, p_{42}
$Out \rightarrow M2$	p_{42}, t_{42}, p_{24}
$M2 \Rightarrow M3$	p_{24}, t_{23}, p_{30}
$M3 \rightarrow In$	p_{30}, t_{30}, p_{03}

Operations represented by the transitions and estimation of their (average) firing times is as follows:

t	*robot's operations*	*time*
t_{01}	pick from In, move to M_1, load	$u + w + y$
t_{12}	unload M_1, move to M_2, load	$v + w + y$
t_{23}	unload M_2, move to M_3, load	$v + w + y$
t_{30}	move from M_3 to In	$2y$
t_{32}	move from M_3 to M_2	y
t_{34}	unload M_3, move to Out, drop	$v + x + y$
t_{42}	move from Out to M_2	$2y$

where:

u denotes the (average) pickup time,
v – the (average) unload time,
w – the (average) load time,
x – the (average) drop time and
y – the average 'travel' time between two adjacent machines (assuming, for simplicity, that this time is the same for all adjacent machines, and also the same for $M3$ to Out, Out to In and In to $M1$ moves).

4. HIERARCHICAL DERIVATION OF SCHEDULES

The schedules can be systematically derived from the 'empty' schedule, i.e., the schedule for a 0–machine cell, which has only two operations (represented by two transitions): 'pick a part from the input conveyor' and 'deposit the part on the output conveyor'. The net model of this schedule is shown in Fig.4.1.
The refinement operations will use five different nets, all representing a single machine, but with slightly different 'connections' to other elements of the refined model. These five refining nets are shown in Fig.4.2.

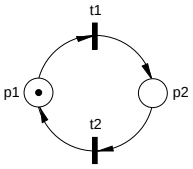

Fig.4.1. Initial model.

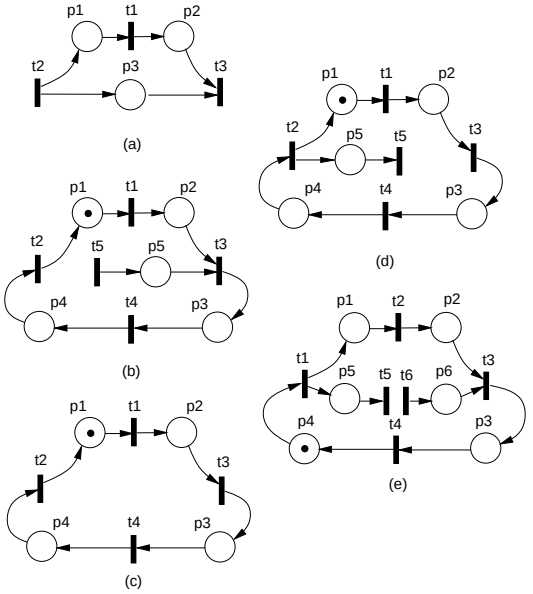

Fig.4.2. Refining nets: (a) R1, (b) R2, (c) R3, (d) R4, (e) R5.

Application of net R1 to transition t_1 of the initial net creates a net model of the single schedule for a 1–machine cell; this model is shown in Fig.4.3.

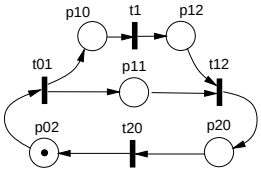

Fig.4.3. Model of the 01–10 schedule.

One of the two schedules for 2–machine cells can be obtained by refining one of transitions t_{01} or t_{12} in Fig.4.3 using the net R1 of Fig.4.2; the two refined nets are isomorphic and correspond to the model shown in Fig.4.4.

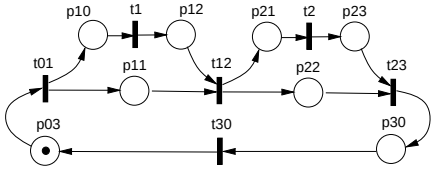

Fig.4.4. Model of the 01–12–20 schedule.

The second 2–machine schedule can be obtaining by refining t_{12} in Fig.4.3 using the net R2 of Fig.4.2, or by refining t_{01} in Fig.4.3 using the net R4 of Fig.4.2; again, these two refined nets are isomorphic and correspond to the model shown in Fig.4.5.

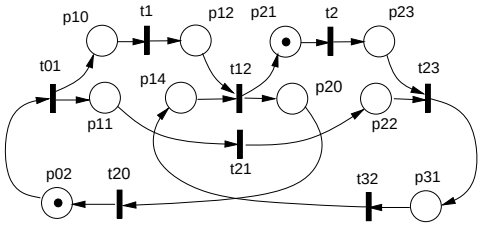

Fig.4.5. Model of the 01–20–12 schedule.

For 3–machine cells, the six schedules can be obtained from the two 2–machine schedules as shown in Tab.4.1.

Tab.4.1. Derivation of 3–machine schedules.

schedule	*refined transition*	*refining net*
A	t_{01} in Fig.4.4, or	R1
	t_{12} in Fig.4.4, or	R1
	t_{23} in Fig.4.4	R1
B	t_{23} in Fig.4.4, or	R2
	t_{01} in Fig.4.5	R1
C	t_{12} in Fig.4.4	R5
D	t_{01} in Fig.4.4, or	R4
	t_{23} in Fig.4.5	R1
E	t_{12} in Fig.4.5	R1
F	t_{12} in Fig.4.5	R3

The first five models are shown in Fig.4.6, Fig.3.3, and Fig.4.7 to Fig.4.9.

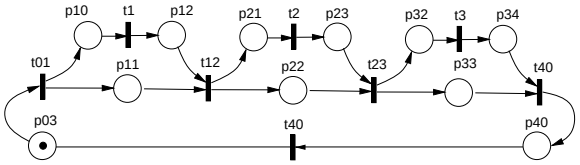

Fig.4.6. Model of the schedule A.

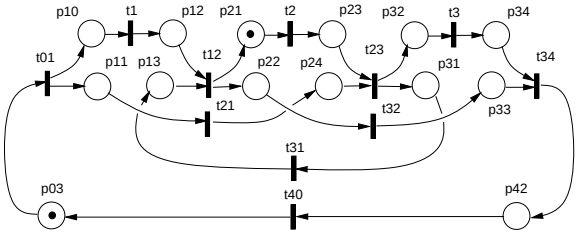

Fig.4.7. Model of the schedule C.

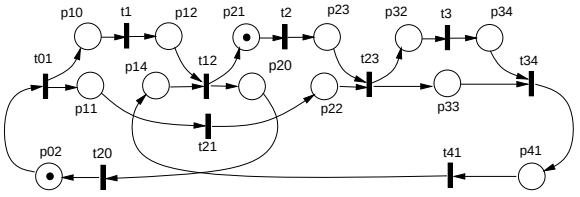

Fig.4.8. Model of the schedule D.

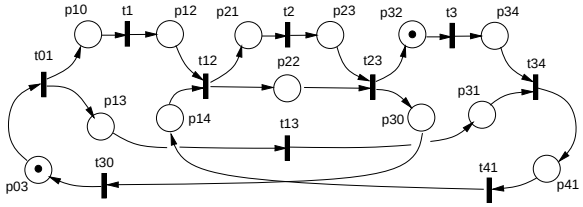

Fig.4.9. Model of the schedule E.

The 24 schedules for 4–machine cells can be derived from the six schedules for 3–machine cells, and so on.

5. CONCLUDING REMARKS

The systematic approach to generating schedules of manufacturing cells proposed in this paper is based on 'local' refinements of elements (in this case, transitions) of Petri nets in the sense that the refining net is connected only to the input and output elements of the refined element. A more general refinement scheme can easily be derived in which more than one element can be replaced during a single step of refinement, and in which the effects of refinements are not restricted to local environments.

The models of manufacturing cells discussed in this paper are very simple, with many simplifying assumptions. Many of these assumptions can easily be removed by some complications of the model. For example, it was assumed that all part are identical, that they follow the same path through the cell, etc. It appears that more general schedules can be defined for cases when several parts enter (and several leave) the cell in each cycle (such schedules are called composite schedules in Zuberek, 1995). It is believed that, with some modifications, the hierarchical approach proposed in this paper can be adopted to composite schedules as well.

Systematic generation of schedules can be important for schedule optimization. For any given set of parameters describing the operation times (i.e., the operations performed by machines as well as the robot), there is a schedule with the smallest cycle time, which means the highest throughput. The evaluation of the cycle time can be performed efficiently using net invariants (Zuberek and Kubiak, 1994), so if there is an efficient procedure generating the schedules, the optimization process can be done by evaluating the schedules and selecting the one with the best performance.

An interesting aspect of the hierarchical approach is to use this approach for efficient performance evaluation of generated schedules, i.e., to "reuse" the performance characteristics of simpler schedules for evaluation the performance characteristics of more complex, refined models. Although some preliminary results have been reported (Basten and Voorhoeve, 1995, Buchholz, 1995), further research is needed in this area.

REFERENCES

Ayres, R.U., Butcher, D.C. (1993). "The flexible factory revisited"; American Scientist, vol.81, no.5, pp.448–459.

Basten, T., Voorhoeve, M. (1995). "An algebraic semantics for hierarchical P/T nets"; Proc. 16-th Int. Conf. on Applications and Theory of Petri Nets 1995 (Lecture Notes in Computer Science 935), Torino, Italy, pp.45–65.

Buchholz, P. (1995). "Hierarchical Markovian models: symmetries and reduction"; Performance Evaluation, vol.22, no.1, pp.93–110.

Desrochers, A.A., Al-Jaar, R.Y. (1995). **Applications of Petri nets in manufacturing systems**; IEEE Press.

DiCesare, F., Hahalakis, G., Orith, J.M., Silva, M., Vernadat, F.B. (1993). **Practice of Petri nets in manufacturing**; Chapman & Hall.

Dixon, C., Hill, S.D. (1990). "Work–cell cycle–time analysis in a flexible manufacturing system"; Proc. Pacific Conf. on Manufacturing, Sydney–Melbourne, Australia, vol.1, pp.182–189.

Hillion, H.P. (1989). "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.

Murata, T. (1989). "Petri nets: properties, analysis and applications"; Proceedings of IEEE, vol.77, no.4, pp.541–580.

Reisig, W. (1985). **Petri nets - an introduction** (EATCS Monographs on Theoretical Computer Science 4); Springer Verlag.

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

Zuberek, W.M. (1991). "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.

Zuberek, W.M. (1995). "Application of timed Petri nets to modeling and analysis of flexible manufacturing cells"; Technical Report #9503, Department of Computer Science, Memorial University of Newfoundland, St.John's, NF, Canada A1B 3X5.

Zuberek, W.M., Bluemke, I. (1996). "Hierarchies of place/transition refinements in Petri nets"; Proc. 5-th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA'96), Kauai, Hawaii, pp.355–360.

Zuberek, W.M., Kubiak, W. (1994). "Modeling simple schedules of manufacturing cells using timed Petri nets", Proc. Int. Workshop on Intelligent Systems and Innovative Computations, Tokyo, Japan, pp.38–47.