On Optimal Scheduling in DEDS*

A. Ramírez J. Campos M. Silva

Dpto. Ingeniería Eléctrica e Informática
Universidad de Zaragoza
María de Luna 3, E-50015 Zaragoza (Spain)

Abstract

In this paper we present the Local Optimality (LO) property for Discrete Event Dynamic Systems modeled using Petri Nets (PN's). If a system is divided into independent subsystems and if the optimal behaviour of these subsystems guarantees the optimal behaviour of whole system, then this system possesses the LO property. The main advantage of this approach is that we can find an optimal schedule for this class of systems in less time, because we are determining the optimal behaviour of small problems. A characterization of systems exhibiting the LO property is given, as well as one procedure to divide them into subsystems.

1 Introduction

The scheduling problem deals with the allocation of resources and starting operation times. In Discrete Event Dynamic Systems (DEDS), there are schedules that optimize different criteria, for example some schedules are of minimum executing time, others are of minimum stock inventories, etc. In this paper, we are interested in schedules with minimum execution time; these schedules and their properties are studied in Scheduling Theory. One of the main challenges in this field is the *tractability* problem, it appears because there are too many permutations of the possible states of DEDS. This problem is NP-complete.

Petri Nets (PN's) are a well known tool to model DEDS. They are capable of capturing most of the features of DEDS in a simple manner [18, 23]. For example they capture precedence, synchronization and real concurrence of DEDS. PN's are a formal tool to describe DEDS, so they apport mathematical analysis, however there does not exist a general purpose algorithm to solve the scheduling problem in this field.

Many approaches have been proposed, and for special systems there exist exact solutions. For example, the Shortest Processing Time (SPT) rule minimizes the average flow-time on a single deterministic machine, the Johnson rule minimizes the maximum makespan on a tandem of two machines, and so on [6]. There are other kinds of rules (dispatching) that help us make good decisions, but they do not guarantee obtaining an optimal schedule [17]. There are also some mathematical approaches: Lagrange multipliers or integer programming have been used, but they have the same problem: time.

The scheduling problem has been studied in different fields: Operations Research [7, 14, 13], Artificial Intelligence [15, 20, 22], Computing Sciences, and Optimal Control. But all of them solve particular systems or give good, but not optimal, schedules.

Herein, we are proposing to take advantage of PN's to find new classes of systems in which we are capable to compute an optimal schedule in a reasonable time. In particular we present the Local Optimality (LO) property as well as independent subsystems, which are called cuts. These cuts do not correspond with real subsystems, but they provide one mechanism to help us to determines when a PN exhibits the LO property.

This approach has the advantage that solving small problems is quite fast. For example, if the solution is obtained in 2^N nanoseconds (where N is the data number), then for $N = 50$, the solution will be obtained in 130 days; but if the problem is divided in five subproblems of ten elements each one, then the solution will be found in 51 microseconds. Also, it is possible to find better heuristics for small problems, so the computing time could be smaller.

This paper is organized as follows. Section 2 briefly presents the PN definition and subsystems. Section 3 presents some previous results related with optimal scheduling and the the LO property in PN's. Section 4 presents the PN's having the LO property and, finally, section 5 presents an illustrative example.

*This work was partially supported by the Mexican CONACYT, the Spanish CICYT ROB91-0949 and the Aragonese CONAI P IT-6/91.

2 Petri Nets and Cuts

Although some basic concepts of PN's are presented in this section, we assume that the reader is familiarized with such models.

2.1 Petri Nets

An ordinary PN is a bipartite directed graph represented by the 4-tuple $N = (P, T, Pre, Post)$, where: $P = \{p_1, \ldots, p_n\}$ is a set of elements called *places*; $T = \{t_1, \ldots, t_m\}$ is a set of elements called *transitions*; $P \cap T = \emptyset$, $P \cup T \neq \emptyset$; *Pre* (*Post*) is the pre (post) *incidence function* representing the input (output) arcs to transitions, $Pre, Post : P \times T \to \{0, 1\}$.

Places are drawn as circles while transitions are depicted as bars or boxes.

The *incidence matrix* $C = [c_{ij}]$, $i = 1, \ldots, n$; $j = 1, \ldots, m$, is defined by $c_{ij} = Post(p_i, t_j) - Pre(p_i, t_j)$. Similarly, Pre and $Post$ can be represented by matrices.

A Marked Graph (MG) is an ordinary PN with $|{}^\bullet p| = |p^\bullet| = 1, \forall p \in P$. A State Machine (SM) is an ordinary PN with $|{}^\bullet t| = |t^\bullet| = 1, \forall t \in T$. A Free Choice (FC) net is an ordinary PN such that for all $p \in P$ with $|p^\bullet| > 1$ then ${}^\bullet(p^\bullet) = \{p\}$.

A function $M : P \to \{0, 1, \ldots\}$ (usually represented in vector form) is called *marking*. Markings represent (distributed) states. A *marked PN* $\langle N, M_0 \rangle$ is a PN N with an *initial marking* M_0.

2.2 Net Dynamics

A transition $t \in T$ is *enabled* at marking M iff $\forall p \in P : M(p) \geq Pre(p, t)$. An enabled transition can be *fired*. If transition t is fired, the marking changes to M' where $M'(p) = M(p) - Pre(p, t) + Post(p, t)$.

The fact that M' is obtained from M firing t is represented by $M[t\rangle M'$. A sequence of transitions $\sigma = t_1 t_2 \ldots t_n$ is a firing sequence of $\langle N, M_0 \rangle$ iff there exists a sequence of markings such that $M_0[t_1\rangle M_1[t_2\rangle \ldots [t_n\rangle M_n$, it can be written as $M_0[\sigma\rangle M_n$, and M_n is said to be reachable from M_0.

The *reachability set* $R(N, M_0)$ is the set of all reachable markings from M_0. $L(N, M_0)$ is the set of all firing sequences, $L(N, M_0) = \{\sigma | M_0[\sigma\rangle M;\ M \in R(N, M_0)\}$.

A marked net $\langle N, M_0 \rangle$ is *safe* iff $M(p) \leq 1, \forall p \in P, \forall M \in R(N, M_0)$.

All vectors X such that $CX = 0, X \geq 0$ are called *T-semiflows*; all vectors Y such that $Y^T C = 0, Y \geq 0$ are called *P-semiflows*.

Timed Petri Nets: A Timed Petri Net (TPN) is: $TPN = (PN, \Delta)$, where: PN is a marked PN and $\Delta : T \to \Re$ assigns one real number, representing the firing time, to each transition.

Through this paper we will work with deterministically timed, live and safe PN's. In order to avoid the coupling between resolution of conflicts and duration of activities, we suppose that transitions in conflict are immediate. Decisions for these conflicts are taken according to routing rates associated with immediate transitions in conflict. The routing rate imposes more constrains to the system, these constrains must be taken into account to find the optimal schedule.

2.3 Definition of Cuts

We are going to derive optimal scheduling of the whole net from the concatenation of the optimal scheduling of some particular subsystems, so we will introduce the concept of "cut". The cut is a subnet, with special properties. In our case the cuts belong to any PN class, so a broad kind of PN can be analyzed with this approach. One kind of cut will be presented, it is a Place-Place Cut (PPC), i.e., this cut has only one input place and only one output place. Formally:

Place-Place Cut (PPC)

A subnet $N' = (P', T', Pre', Post')$ of a net $N = (P, T, Pre, Post)$ (i.e., $P' \subseteq P$, $T' \subseteq T$, $Pre' = Pre|_{P' \times T'}$, $Post' = Post|_{P' \times T'}$) is a PPC iff:

- ${}^\bullet t$ *y* $t^\bullet \in P'$
- $\exists |\ p_{source} \in P'$ s.t. ${}^\bullet p_{source} \cap (T \setminus T') \neq \emptyset$ and it verifies that ${}^\bullet p_{source} \subseteq T \setminus T'$.
- $\exists |\ p_{sink} \in P'$ s.t. $p^\bullet_{sink} \cap (T \setminus T') \neq \emptyset$ and it verifies that $p^\bullet_{sink} \subseteq T \setminus T'$.
- if $x_i \in P' \cup T'$ then there exists a $Pre' \cup Post'$ directed path from p_{source} to x_i and another one from x_i to p_{sink}.
- $\forall t \in T', \exists p \in^\bullet t, Y \geq 0$ *s.t.* $Y^T C = 0, Y(p) = Y(p_{source}) = Y(p_{sink}) > 0$

Figure 1 shows a PPC. We describe only PPC cuts, but if we have other kinds of single input and single output cuts (place-transition, transition-place, or transition-transition) they can be transformed into a PPC one. It is necessary to expand the border transition in two transitions with one place in the middle. See figure 2. A cut is denoted by $X_1 X_2 C$, where X_1 is the input element to the cut and X_2 is the output element.

3 Optimal Scheduling for Petri Net Models

3.1 Fundamental Concepts

Feasible Schedule: It is a sequence of ordered pairs of $(i_k, \phi_k), k \geq 1$, where $\sigma = t_{i_1} t_{i_2} \ldots t_{i_k} \ldots$ is a firing

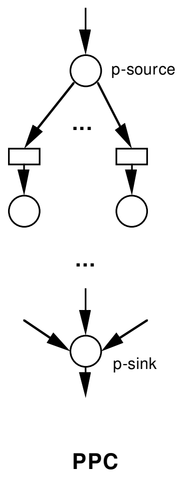

Figure 1: A Place-Place Cut

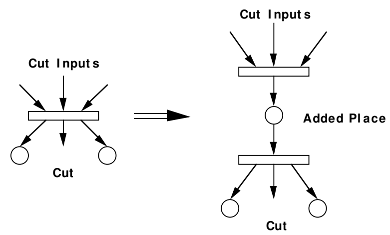

Figure 2: Expanding a Transition

sequence of the considered marked PN that is compatible with the routing rates associated with immediate transitions in conflict and ϕ_k is the starting firing time of transition t_{i_k} in σ.

Throughput: For a transition, it is the average number of transition fires per time unit, in the limit. The cycle time is the inverse of the throughput.

Optimal Schedule: It is a feasible schedule that produces the optimal behaviour of the net, i.e., performing this schedule, the throughput of the net is maximum.

3.2 Previous Results

In the literature of PN's there are some works dealing with the scheduling problem. Earlier works are related with bounds and cycle time of PN's; [19, 11] developed these bounds. Specially, Ramamoorthy and Ho proved that for MG's the bound is reachable, and the proof of this fact gives an algorithm to obtain the starting firing time of each transition. Later, [3, 2] proved that for MG's, the rule *fire transitions as soon as they are enabled* obtains an optimal schedule for this kind of nets. There are several approaches combining artificial intelligence and PN's: [8] uses PN's and an expert system to decide which priority rule to apply; [12] proposes a method to model a special class of systems using PN, then a modified beam-search algorithm and an expert system are used to perform the search; [10] uses a PN model to generate the reachability graph, one A* like search algorithm is used and a comparison between four heuristic functions is given; [4] uses PN's to model Kanban systems and later apply JIT politics; [16] gives an algorithm to find out a good schedule for all kinds of PN, it is based on Sifakis bound and uses the state equation as a continuous one; [5] derives an algorithm to find out the optimal cycle time for a special class of PN; [1] generalizes the cycle time bounds for more general PN's and [9] gives an algorithm to find out an optimal marking for an MG when the cycle time is known.

Although there are some more works, these are the most representative in the scheduling field using PN models.

Marked Graphs: In MG's an optimal schedule can be found in polynomial time. The reason is that in MG there are no decisions, so it is enough to *fire each transition as soon as it is enabled.* Using this heuristic we obtain one optimal schedule (there could be more than one).

State Machines: In SM's there are conflicts, and some local policies are given to solve them. Using these policies it is possible to find a "visit ratio" vector solving the following equation system [1]:

$$\begin{pmatrix} C \\ R \end{pmatrix} v = 0 \tag{1}$$

where R reflects the policies taken in the net conflicts. The optimal schedule is any that preserves the visit ratio vector. One solution could be taken probabilistically.

Free Choice Nets: For FC nets (and more general nets) there are no results. Furthermore, finding out the optimal throughput (maximum) seems to be an NP-Complete problem [11].

3.3 Local Optimality Property

In this work each PPC will be used as a subsystem or subnet, it can be any type of net. The optimal schedule for the cut can be obtained using enumerative techniques or some heuristics. Given that the number of possible schedules to analyze is lower than in the original net, the time consumed to solve the problem decreases. Cuts are reduce to two places (p'_{source}, p'_{sink}) and one transition between these places (t_c).

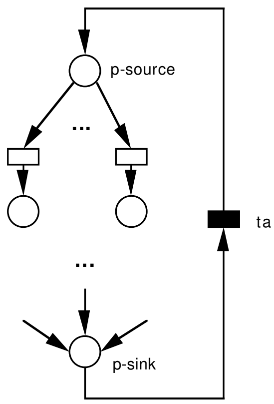

Figure 3: Strongly connected cuts

Local Optimality Property: When a system can be divided into independent subsystems, if the optimal behaviour of these subsystems guarantees the optimal behaviour of the whole system, then this system has the *Local Optimality* property.

4 Nets Having LO Property

If a net has LO property, it can be split into simple cuts and each cut optimized separately. This procedure saves a lot of computation time; the problem to solve herein is to determine when the net has this property.

The analysis of isolated cuts is performed adding one transition t_a from p_{sink} to p_{source} in a PPC, see figure 3.

With the added element, the cut becomes strongly connected. Furthermore we can perform some computations such as the Sifakis bound, the visit ratio vector and to obtain an optimal schedule. If the cut is quite simple (e.g., an MG) then the optimal schedule is easy to obtain, but if the cut is a general net, then enumerative techniques should be used (with or without heuristics), but any way this cut has less elements, so the consumed time should be less than the original net.

4.1 Results

Not all nets have LO property, but some of them exhibit it; the characteristics of nets exhibiting this property will be presented.

Property 4.1 *If a State Machine is obtained when all cuts are reduced, then this net has the local optimality property.*

Proof: For the proof we compute PPC, Reduced Net and Global Net visit ratios, then all feasible schedules for PPC and Global Net are computed. We prove that the optimal schedule for the Global Net is obtained when individual schedules of PPC are optimized, under assumption that the Reduced Net is a state machine. We denote GN the whole net; by SM the reduced net, i.e. when PPC are reduced.

• SM has v_{sm} as visit ratio vector. Also each cut has its own visit ratio vector v_{ci}. The visit ratio vector is any one verifying (1) of minimum norm with all its components integers.

In a cut C_i, q_{ci} is the entry in v_{ci} for t_a. The visit ratio vector v for the global net can be computed by: $v' = \Pi q_{ci} \cdot v_{sm}$ for transitions belonging to SM,$v' = (\Pi q_{ci} \cdot v_{sm})/q_{ci}$ for transitions belonging to C_i and $v = v'/k$; where k is the maximum common divisor of v' elements.

It is true because in the incidence matrix C, the cuts are isolated blocks of information, so they have no relationship with each other. They are communicated with the reduced net by two columns, and this value is 1 or -1, so the result is exactly scaled by the visit ratio of added transition.

• Several possible schedules exist for cut c_i. Some of them must be performed to fulfill the visit ratio vector. Suppose that the set:

$S_i = \{s_1^i, s_2^i, \ldots, s_k^i\}$

contains all feasible schedules of cut c_i. (according to the visit ratio and net dynamics). In each schedule:

s_j^i, t_a (PPC) appears q_{c_i} times because it has to fulfill visit ratio constraints. Each feasible sequence s_j^i is decomposable into several concatenated subsequences:

$s_j^i = s_{j_1}^i \oplus t_a \oplus s_{j_2}^i \oplus t_a \oplus \ldots \oplus s_{j_{ki}}^i \oplus t_a$ ($\oplus$ means concatenation).

Suppose that the optimal schedule for this cut is:

$s_j^{i*} = s_{j_1}^{i*} \oplus t_a \oplus s_{j_2}^{i*} \oplus t_a \oplus \ldots \oplus s_{j_{ki}}^{i*} \oplus t_a$

• In the GN, the cut schedules are interleaved because t_a does not exist, and the cuts must be executed in sequence. For example, a GN sequence could be:

$s_{j_1}^i \oplus t_x \oplus s_{k_1}^i \oplus t_y \oplus \ldots$

It indicates that cut c_i starts to fire; when it finishes t_x starts to fire (it belongs to no cut, it is free), then cut c_k is fired and so on until all cut schedules and free transitions are fired according to the visit ratio constraints.

The cycle time of GN is a sum of individual schedule time:

$\pi = \tau(s_{j_1}^i) + \tau(t_x) + \tau(s_{k_1}^i) + \tau(t_y) + \ldots$

where $\tau(x)$ means consumed time by schedule x. Since cut executions do not overlap, we conclude that:

If we optimize each cut separately, we have the sum of minimum value terms, and the sum result is minimum.

Each cut can be optimized separately to obtain a global minimum and this result holds for any sequence permutation; so the LO property follows.

Q.E.D.

Observing the proof, there are two basic reasons why the LO property holds. There are no *min* or *max* operations, and the sum is the same for each permutation.

Property 4.2 *If the net has only PPC's that have one T-semiflow, with all its elements equal to one (for example a marked graph or a marked graph with monitors):*

1. *When these cuts are reduced and the resulting net is a state machine, then it has the LO property.*

2. *When these cuts are reduced and the net is a marked graph, then it has LO property.*

Proof: Using property 4.1 the part (1) is proved.

Part (2): as the cut has only one *T-semiflow* with its elements equal to one, if the input transition (t_{entry}) is fired one time then the output transition (t_{output}) and each cut transition is fired one time. This means that the cut is executed completely by each t_{entry} fire. The behaviour of this cut is not affected by other cuts (it only has two border elements and all transitions are executed in one step), so it can be optimized separately, given a global optimal schedule, and the LO property follows. **Q.E.D.**

5 Example

Figure 4 shows an assembly cell layout. It is composed by one input store, and three subcells C_1, C_2, C_3.

C_1 is composed of two twin machine stations, one disassembly section and one assembly section.

The disassembly section splits the arriving parts into two separate parts. Each one of these parts is loaded into one machine station. Inside of the machine station, the part can be routed to machine M_1 or machine M_2 (it depends on local policies).

Machine M_1 produces two parts, one called part A, and the other is a non-terminal part. This last part is loaded into machine M_3. Machine M_2 produces one terminal part called B, and one non terminal part which is loaded into machine M_4. M_3 produces part B and M_4 produces part A. Finally product A and B are assembled to obtain the final product.

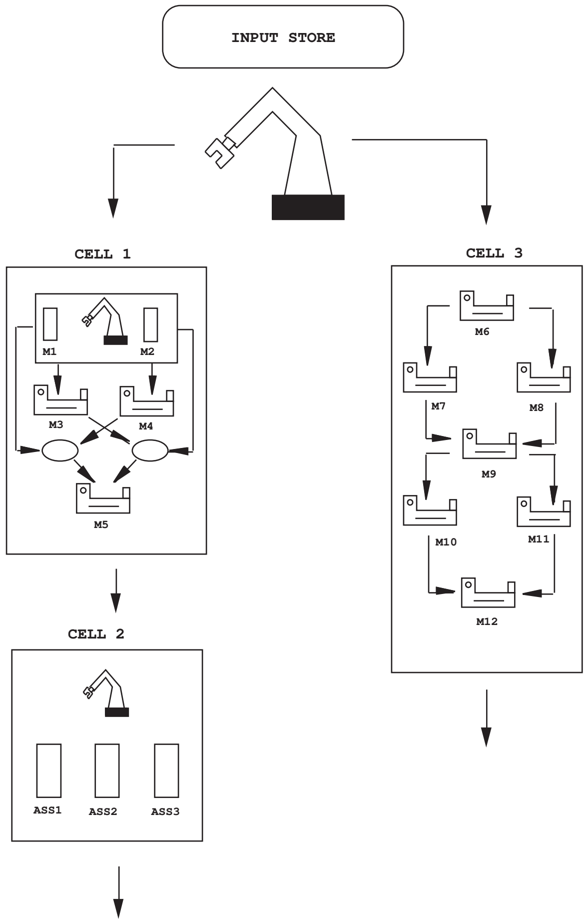

Figure 4: The layout of the assembly cell

In subcell C_2, the ending products of C_1 are processed. This cell has three machines, one product can be processed by any machine.

Cell C_3 is a sequence of machines. One machine is shared by two different operations.

The net which models this assembly cell is shown in figure 5. The numbers depicted near of transitions are the processing times of the respective machines. The dashed boxes show the different cuts. When these cuts are reduced the net becomes a state machine, so the net exhibits the LO property. For Cut A (C_A) the visit ratio of t_{aA} is: $v_{aA} = 2$; for C_B, $v_{aB} = 3$ and for C_C, $v_{aC} = 1$. Transitions not belonging to any cut have $2 \cdot 3 \cdot 1$ visit ratio; belonging to C_A, $3v_{C_A}$; belonging to C_B, $2v_{C_B}$; and belonging to C_C, $6v_{C_C}$.

The global net, and the cuts are optimized using the algorithm reported in [21] running on a SPARC processor. This algorithm is an enumerative one using some heuristics to reduce the complexity of the problem. The results are shown in table 1.

Net	Time
Global	7.86 min
C_1	15.8 sec
C_2	0.8 sec
C_3	2.6 sec
Reduced	0.85 sec

Table 1: Time consumed by different cuts and whole net

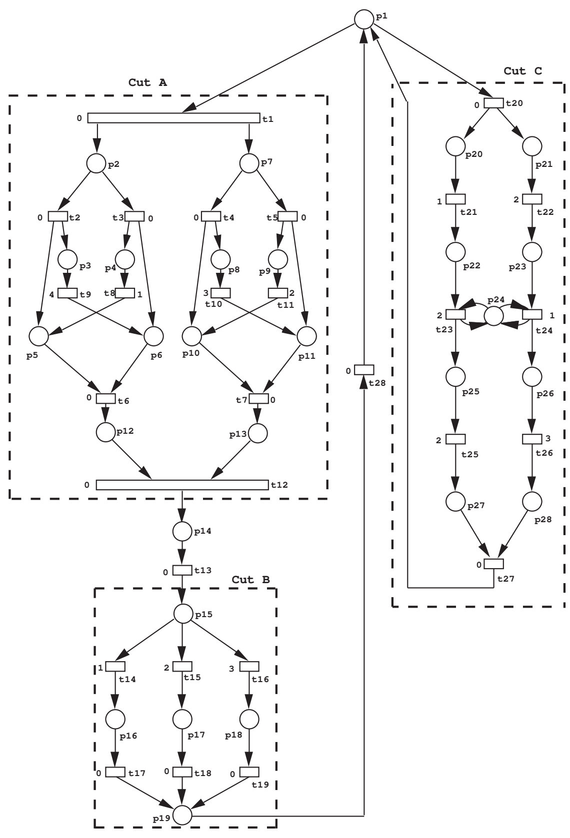

Figure 5: A Petri Net exhibiting LO property

Using cuts the consumed time is 20.05 sec. This analysis is faster than the global net analysis (7.86 min.). The schedule is not the same, but both of them are optimal.

6 Conclusions

In this paper we presented the Local Optimality property for DEDS, which guarantees that a system has an optimal behaviour when its subsystems have an optimal behaviour. If one system has this property then it is possible to obtain an optimal schedule by optimizing its subsystems. The most important characteristic of this property is that the number of the input data is lower and the computing time is also lower.

Although it is a good improvement, there is much work to do. For example, if the cut has too many reachable states, then the computing time is still too high. One way to eliminate this problem is to find new heuristics for these cuts. The new heuristics should be easier to obtain because the cuts have special characteristics. Up to now we have some heuristics for PPC with one safe and live Free Choice net in the middle of its border transitions.

In any situation, the computation time is lower because the number of possible combinations of reachable states is lower.

References

[1] J. Campos, G. Chiola, and M. Silva. Properties and performance bounds for closed free choice synchronized monoclass queueing networks. *IEEE Trans. on Autom. Cont.*, 36(12):1368–1382, December 1991.

[2] J. Carlier and Ph Chretienne. Timed Petri net schedules. In G. Rozenberg, ed., *Advances in PN's 1988*, vol. 340 of *Lecture Notes in Computer Sciences*, pp. 62–84. Springer-Verlag, Berlin, Germany, 1988.

[3] J. Carlier, Ph Chretienne, and C. Girault. Modelling scheduling with timed Petri nets. In G. Rozenberg, H. Genrich, and G. Roucairol, eds., *Advances in PN's 1984*, vol. 188 of *Lecture Notes in Computer Sciences*, pp. 62–82. Springer-Verlag, Berlin, Germany, 1984.

[4] M. Di Mascolo. *Modélisation et Évaluation de Performances de Systèmes de Production Gérés en Kanban*. PhD thesis, Institut National Polytechnique Grenoble, France, 1990. In French.

[5] P. Freedman. Time, Petri nets, and robotics. *IEEE Trans. on Robot. and Autom.*, 7(4):417–433, August 1991.

[6] S. French. *Sequencing and Scheduling, An Introduction to the Mathematics of the Job-Shop*. John Wiley & Sons, New York, NY, USA, 1982.

[7] B. Gillett. *Introduction to Operation Research, A Computer-Oriented Algorithmic Approach*. McGraw-Hill, USA, 1976.

[8] I. Hatono, K. Yamagata, and H. Tamura. Modeling and on-line scheduling of flexible manufacturing systems using stochastic Petri nets. *IEEE*

Trans. on Soft. Eng., 17(2):126–132, February 1991.

[9] H. Hillion and J. Proth. Performance evaluation of job-shop systems using timed event-graphs. *IEEE Trans. on Autom. Cont.*, 34(1):3–9, January 1989.

[10] D. Y. Lee and F. DiCesare. FMS scheduling using Petri nets and heuristic search. In *Proc. of Intern. Conf. on Robot. and Autom.*, pp. 1057–1062, Nice, France, May 1992.

[11] J. Magott. New NP-complete problems in performance evaluation of concurrent systems using Petri nets. *IEEE Trans. on Soft. Eng.*, SE-13(5):578–581, May 1987.

[12] H. Martins and T. Sekiguchi. A timed Petri net and beam search based on-line fms scheduling system with raouting flexibility. In *Proc. of Intern. Conf. on Robot. and Autom.*, pp. 2548–2553, Sacramento, California USA, April 1991.

[13] K. Neumann. *Stochastic Project Networks, Temporal Analisis, Scheduling and Cost Minimization*, vol. 344 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, Germany, 1990.

[14] K. Neumann and U. Steinhardt. *GERT Networks, and the Time-Oriented Evaluation Projects*, vol. 172 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, Germany, 1979.

[15] S. J. Noronha and V. V. S. Sarma. Knowledge-based approaches for scheduling problems: A survey. *IEEE Trans. on Knowl. and Data Eng.*, 3(2):160–171, June 1991.

[16] K. Onaga, M. Silva, and T. Watanabe. On periodic schedules for deterministically timed Petri net systems. In *Proc. of Fourth Intern. Workshop on PN's and Performance Models*, pp. 210–215, Melbourne, Australia, December 1991.

[17] S. Panwalkar and W. Iskander. A survey of scheduling rules. *Operations Research*, 25(1), January-February 1977.

[18] J.L. Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981.

[19] C. Ramamoorthy and G. Ho. Performance evaluation of asynchronous concurrent systems using Petri nets. *IEEE Trans. on Soft. Eng.*, 6(5):440–449, September 1980.

[20] K. Ramamritham and J. Stankovic. Efficient scheduling algorithms for real-time multiprocessor systems. *IEEE Trans. on Paral. and Distr. Syst.*, 1(2):184–194, April 1990.

[21] A. Ramírez, J.L. Villarroel, M. Silva, and P.R. Muro. Scheduling en celdas autónomas de ensamblaje basado en un algoritmo de tiempo mínimo en RdP. *To appear in Revista de Robótica y Automatización*, 1992. In Spanish.

[22] F. Rodammer and P. White. A recent survey of production scheduling. *IEEE Trans. on Syst. Man, and Cybern.*, 18(6), November-December 1988.

[23] M. Silva. *Las Redes de Petri en la Automática y la Informática.* AC, Madrid, España, 1985. In Spanish.