35th European Modeling & Simulation Symposium
20th International Multidisciplinary Modeling & Simulation Multiconference

2724-0029 © 2023 The Authors.
doi: 10.46354/i3m.2023.emss.041

Buffer Allocation Problem modeling by means of Timed Petri Nets

Joselito Medina-Marin[1,*], Maria Guadalupe Serna-Diaz[2], Norberto Hernandez-Romero[1] and Juan Carlos Seck-Tuoh-Mora[1]

[1]Área Académica de Ingeniería, Universidad Autónoma del Estado de Hidalgo, Carr. Pachuca-Tulancingo Km. 4.5, Col. Carboneras, Mineral de la Reforma, Hidalgo, 42184, México
[2]Área Académica de Química, Universidad Autónoma del Estado de Hidalgo, Carr. Pachuca-Tulancingo Km. 4.5, Col. Carboneras, Mineral de la Reforma, Hidalgo, 42184, México

[*]Corresponding author. Email address: jmedina@uaeh.edu.mx

Abstract

The Buffer Allocation Problem is a typical issue for production line designers. Several buffer slots must be distributed along the line in order to maximize throughput, taking into account the number of machines and their processing time. In this paper, Petri Net theory is proposed to model and simulate production lines in the Buffer Allocation Problem, where the processing time for each machine and the buffer's capacity are included in the extended Petri Net model. Numerical results show the feasibility of Petri Nets in representing the relationship between machines and buffers, as well as the use of state equations to calculate the throughput of the production line. Small, medium, and large-sized production lines are analyzed, with three different numbers of slots as restrictions. Moreover, two algorithms were implemented to create the Petri Net model from a production line configuration and to execute the simulation of the line with the Petri Net model.

Keywords: Buffer allocation problem; Throughput, Petri nets, State equation

1. Introduction

A typical problem in manufacturing system design is the assignment of the number of buffer slots in a production line. These slots are distributed among the machines to achieve the maximum system throughput. (figure 1). (Weiss, 2018)

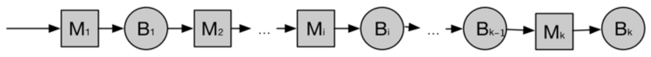

Figure 1. Serial flow in a production line composed of *K* machines (boxes) and *K* buffers of capacity B_i.

The combinatorial assignment challenge in manufacturing system design, known as the Buffer Allocation Problem (BAP), depends on the number of machines and the total number of slots.

The BAP can be defined as an optimization model with three different perspectives based on the objective function. The first perspective aims to optimize the maximum throughput of the system. The second perspective focuses on minimizing the buffer size to achieve a given performance rate. The last perspective aims to minimize the work in process inventory (Weiss, 2018).

Mathematical models for the BAP were presented by Papadopoulos et al. (2013) from three different problem perspectives.

In the Dual Problem (BAP-A), there are K machines and K − 1 buffer areas, with N buffer slots to be allocated. The solutions for this problem are represented as n = (N1, N2, ..., Nk-1). The objective is to achieve the highest performance rate in the production line with a fixed number of N buffer slots placed between each machine K. The mathematical formulation for the maximization function is denoted by Eq. 1 and Eq. 2.

$$\max X(n) = \max X(N_1, \dots, N_{k-1}) \tag{1}$$

s.t.

$$\sum_{i=1}^{K-1} N_i = N \quad N_i \geq 0, \forall i = 1, \dots, K-1 \tag{2}$$

The Primal Model (BAP-B) focuses on minimizing the number of allocated buffers between each machine K while ensuring a minimum throughput of X0 (Eq. 3), subject to the constraint shown in Eq. 4.

$$\min N = \sum_{i=1}^{K-1} N_i \tag{3}$$

s.t.

$$X(n) = X(N_1, \dots, N_{K-1}) \geq X_0 \tag{4}$$

The BAP-C is related to minimize the average work in process (WIP) given a minimum throughput X_0 (Eq. 5) and the restrictions denoted in equations 6 and 7.

$$\min L(n) = \min L(N_1, \dots, N_{k-1}) \tag{5}$$

s.t.

$$X(n) = X(N_1, \dots, N_{K-1}) \geq X_0 \tag{6}$$

$$\sum_{j=1}^{K-1} N_j \leq N \quad N_j \geq 0, j = 1, \dots, K-1 \tag{7}$$

where:

$L(n) = L(N_1, \dots, N_{K-1})$: Represents the mean of the WIP inventory which depends on the buffer size and the throughput X_0.

These problems have been addressed with exact methods in small production lines, and metaheuristic methods for production lines with many machines and lot of buffer slots to be distributed in the system. For each case, the authors use their own way to represent the production line and calculate the variables of interest. However, the mathematical tools of PNs have not yet been exploited for BAP. In this work, we propose the use of Petri nets (PNs) theory for modeling the relationship of machines and buffer slots, and obtain the throughput, work in process inventory and the cycle time.

The reminder of this paper is organized as follows. Section 2 are presented the works related to BAP. Next, PN basis are described in Section 3. The results and discussion are given in Section 4. Finally, the conclusions are presented in Section 5.

2. State of the art

The BAP has been studied since last century. The first published work was presented by Koenigsberg (1959). In this work, a review and analysis of the effective functioning of production lines was presented.

Recent published works regarding the BAP proposes different strategies to face it. Lopes et al (2020) proposed a mathematical model to evaluate the performance of production lines combining line balancing and buffer allocation problem. On the other hand, BAP has also been faced with evolutionary algorithms and simulation optimization in open serial production lines (Yelkenci et al, 2020). The authors focused on minimizing the total buffer space and maximizing the throughput.

The BAP has also been studied from an expert system perspective. Motlagh et al (2019) developed an expert system where they considered mean processing times of workstations and buffer capacities to obtain the production line performance. Moreover, the expert system includes genetic algorithms and simulation based on linear regression.

Due to some problems complexity, it is a good option to propose solutions divided in phases. A two-stage heuristic algorithm was developed by (Liang et al, 2020). The first stage obtains the resource allocation, and the second one is centered in determining the position of buffer slots.

As part of the proposals to solve the BAP, the use of PNs has been applied by many researchers. In 2011, Zhang et al defined two PN models for pre-allocating and partitioning buffers in a flexible manufacturing system. In 2016, Liu proposed the use of PNs for modeling resource allocation systems. The author's proposal addresses the deadlock problem via a general class of PNs to analyze shared resources and the process interaction. And one year later, Skolud et al (2017) proposed a tool based on PNs to perform production line simulations. The GPenSIM tool is used to identify problems in the production lines design and resource allocation. In these PN proposals, the author focused on graphical representation of BAP and PN simulation to analyze the problem. Nevertheless, the mathematical tools were not considered to find solutions of the resource allocation and buffer slots position.

The BAP has been modeled and optimized from different techniques and strategies, including PNs. Nevertheless, state equation and incidence matrix were not taken into advantage for throughput calculus. In this paper, mathematical tools, and graphic representation of PNs are used to model the production line, and obtain the production rate and throughput of the lines in the BAP.

3. Materials and Methods

The interaction of machines and buffers in the BAP is depicted as a Petri net (PN) model, and its mathematical tools are taken into advantage to calculate the throughput of the production line.

3.1. Petri nets concepts

A Petri net is a powerful graphical and mathematical tool for modeling discrete event driven systems (Peterson, 1977; Zhou). The graph is directed, *arc* weighted, and has two types of nodes: *places* (circles) and *transitions* (boxes).

In a PN, places are connected to transitions, and transitions to places, but nodes of the same type are not allowed to be connected. The system state is represented using *tokens*, which are put in the corresponding PN places.

A formal definition of a PN is showed in Table 1. (Murata, 1989)

Table 1. Formal definition of a PN.

A Petri Net is a 5-tuple, *PN=(P, T, F, W, M₀)* where:
$P = \{p_1, p_2, \ldots, p_m\}$ is a finite set of places.
$T = \{t_1, t_2, \ldots, t_n\}$ is a finite set of transitions.
$F \subseteq \{P \times T\} \cup \{T \times P\}$ is a set of arcs.
$W = F \rightarrow \{1,2,3, \ldots\}$ is a weight function.
$M_0 = P \rightarrow \{0,1,2, \ldots\}$ is the initial marking.
$P \cap T = \emptyset$ and $P \cup T \neq \emptyset$

The token game animation is used to represent the dynamical behavior of the system modeled through PNs. The change in the token position is based on the following transition firing rule: (Murata, 1989)

1. A transition $t \in T$ is enabled if each input place $p \in P$ of *t* contains at least *w(p,t)* tokens.
2. An enabled transition *t* will be fired if the event denoted by *t* occurs.
3. Once an enabled transition *t* is fired, *w(p,t)* tokens are deleted from each input place *p* of *t*, and *w(t,p)* tokens are put on every output place *p* of *t*.

In addition, the BAP modeling needs the use of processing time. Thus, an extension of PN model with time in places is applied. A Timed Place Petri Net (TPPN) is a six-tuple $TPPN = \{P, T, F, W, M_0, D\}$, where $D = \{d_1, d_2, \ldots, d_m\}$ represents the processing time for every place $p \in P$. (Zhao et al., 2011). In this case, the first transition firing rule for PNs now considers not only the number of tokens in the input places, also the time d_i indicated for place p_i must be reached.

3.2. Incidence Matrix and state equation

In this paper, the matrix equation of PN theory is used to simulate the token game animation in the production line, obtain the total of products processed, and finally calculate the throughput of the system.

A PN with *n* transitions and *m* places can be denoted as a matrix of integers $A = [a_{ij}]$ with dimension $n \times m$. The a_{ij} values are calculated with the mathematical expression $a_{ij} = a_{ij}^{+} - a_{ij}^{-}$, where a_{ij}^{+} is the arc weight $w(t_i, p_j)$ and a_{ij}^{-} is the arc weight $w(p_j, t_i)$.

The state equation is helpful to determine the marking in a PN through the transition firings. This equation is shown in Eq. 8.

$$M_k = M_{k-1} \times A^T U_k, \qquad k = 1,2, \ldots \tag{8}$$

where U_k is a $n \times 1$ column vector of zeros and a unique one. The position of the one value indicates the transition t_j that will be fired. A^T is the transpose of the incidence matrix A. M_{k-1} is the marking before t_j is fired. And M_k is the marking reached after the firing of t_j.

4. Results and Discussion

In this work, we propose the use of PN theory to represent the BAP and obtain the performance rate of a production line, considering the number of stations and the buffer slots.

4.1. Timed Place Petri Net model

The PN structure depicted in figure 2 is used to represent every machine in the sequence of the workflow showed in figure 1. This structure with two places and two transitions is useful to model the availability of the stations and it contains the processing time assigned to every station. The buffers are represented with places between every PN pattern of figure 2.

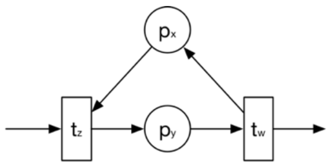

Figure 2. PN pattern to represent a station of the production line in the BAP.

Place $p_x \in P$ is used to indicate if a station is available or not, and place $p_y \in P$ is used to keep stored the token during d units of time to simulate the processing time.

To illustrate our proposal, the TPPN of a production line, with K=2 (two stations) and one buffer B between them with capacity N=5, is showed in figure 3.

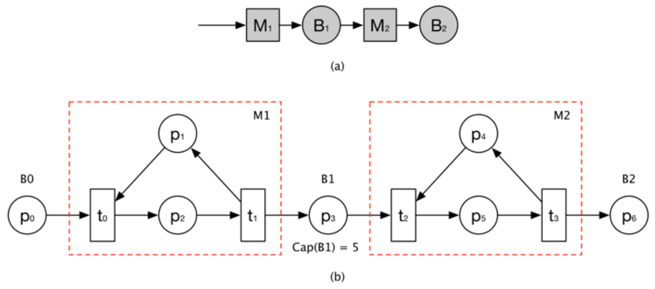

Figure 3. TPPN structure for modeling a production line with K=2 and N=5.

At the beginning, there is the place p_0 representing the raw material warehouse (B_0), with infinity capacity and a token is always available there. Then, machine 1 is depicted by $p_1, p_2 \in P$ and $t_0, t_1 \in T$. B_3 represents the buffer with a capacity to store five slots. Machine 2 is composed of $p_4, p_5 \in P$ and $t_2, t_3 \in T$. Finally, the storage B_2 for finished products is denoted by place $p_6 \in P$ and its capacity is also infinite.

The incidence matrix for this PN, with four transitions and seven places, is as presented in Eq. 9.

$$A = \begin{bmatrix} -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 \end{bmatrix} \tag{9}$$

Moreover, a vector τ with the processing times is defined. Only the places used to simulate the delay in the station will contain either a constant value or a probabilistic distribution function to generate a random value. For this example, an exponential distribution is used for emulating the processing time in the corresponding places and the value of $\mu = 1$.

$$\tau = \begin{bmatrix} 0 \\ 0 \\ Exp(1) \\ 0 \\ 0 \\ Exp(1) \\ 0 \end{bmatrix} \tag{10}$$

The initial marking M_0 contains a token in the initial place p_0, and in places p_1 and p_4 to indicate the availability of machines 1 and 2, respectively (Eq. 11).

$$M_0 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{11}$$

Figure 4 shows graphically the initial marking, the small black filled circles inside the places correspond to the values in M_0.

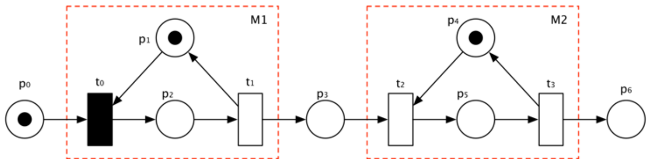

Figure 4. Initial marking of the production line, with the raw material ready to be processed, and the stations waiting for raw material.

Furthermore, it is necessary to know the storage capacity for every place due to the transition firing rule. Above was mentioned that to trigger a transition, its output place must have space to receive the incoming token. Therefore, it is important to define the capacity vector κ to store the capacity of each place. For this example, vector C contains the values shown in Eq. 12.:

$$\kappa = \begin{bmatrix} inf \\ 1 \\ 1 \\ 5 \\ 1 \\ 1 \\ inf \end{bmatrix} \tag{12}$$

The next subsection presents the description of the algorithm implemented to create the TPPN incidence matrix A, vector τ, initial marking M_0 and the vector κ. Moreover, the algorithm that executes the simulation to calculate the throughput of the system is described.

4.2. Algorithms

The first algorithm was implemented to create the TPPN and all the necessary elements to model the BAP. This algorithm requires the number of stations (K), the mean value (μ) for the probabilistic distribution function used to emulate the processing time, and the list of buffers capacity B_i.

Algorithm 1 Create TPPN
Input: K, μ, Capacity
Output: A, τ, κ, M_0

```
$row \leftarrow 2 * K$
$col \leftarrow 3 * K + 1$
$A \leftarrow [0]_{row,col}$
$\tau \leftarrow [0]_{1,col}$
$\kappa \leftarrow [1]_{1,col}$
$pos_p \leftarrow 0$
$pos_b \leftarrow 0$
for i=1 to K do
    /* First transition */
    $pos_t \leftarrow 2 * i$
    $A[pos_t, pos_p] \leftarrow -1$
    $A[pos_t, pos_p + 1] \leftarrow -1$
    $A[pos_t, pos_p + 2] \leftarrow 1$
    $T[pos_p + 2] \leftarrow \mu$
    $pos_p \leftarrow pos_p + 1$
    /* Second transition */
    $pos_t \leftarrow 2 * i + 1$
    $A[pos_t, pos_p] \leftarrow 1$
    $A[pos_t, pos_p + 1] \leftarrow -1$
    $A[pos_t, pos_p + 2] \leftarrow 1$
    $pos_p \leftarrow pos_p + 2$
    $C[pos_p] \leftarrow cap[pos_b]$
    $pos_b \leftarrow pos_b + 1$
end for
/* Initial marking */
$M0 \leftarrow [0]_{col,1}$
$M0_0 \leftarrow 1$
```

```
28: for m=1 to col/3 do
29:     $pos_M \leftarrow 3*m+1$
30:     $M0_{pos_M} \leftarrow 1$
31: end for
32: /* Delete places with capacity 0 */
33: $res \leftarrow$ positions of B whose Capacity is 0
34: for $x \in res$ do
35:     $pos_{place} \leftarrow 3*x+3$
36:     $res2 \leftarrow$ input and output transition for place with capacity 0.
37:     $pos_{tran} \leftarrow res2[0]$
38:     $A_{pos_{tran}} \leftarrow A_{pos_{tran}} + A_{pos_{tran+1}}$
39:     Delete place from $A$, $\tau$, $\kappa$, and $M_0$
40: end for
```

This algorithm returns the incidence matrix A denoting the TPPN, the processing time τ for each station, the buffer capacity κ, and the initial marking M0.

Once the TPPN is obtained, the simulation of the production line can be executed. The steps to carry out the simulation are shown in Algorithm 2, which receives as input all the parameters obtained in Algorithm 1 plus the simulation time.

Algorithm 2 Run the simulation with the TPPN
Input: A, τ, κ, M_0, Total simulation time (SimTime)
Output: Throughput

```
1: $st \leftarrow 0$
2: $vT \leftarrow [0]_{|P|}$
3: while $st \leq SimTime$ do
4:     $U_k \leftarrow$ get enabled transitions
5:     /* State equation operation */
6:     $M_k \leftarrow M_{k-1} + A^T * U_k$
7:     /* Update times in output places */
8:     $FT \leftarrow$ fired transitions
9:     for $t \in FT$ do
10:        for $p \in t^{\bullet}$ do
11:            $vT(p) \leftarrow \tau(p)$
12:        end for
13:    end for
14:    /* Put a new token in the initial place $p_0$ */
15:    $M_k(p_0) \leftarrow 1$
16:    /* Advance to the next simulation time step */
17:    $minTime \leftarrow min(vT)$
18:    $vT \leftarrow vT - minTime$
19:    /* Update condition variable $st$ */
20:    $st \leftarrow st + minTime$
21: end while
```

This second algorithm returns the production line throughput for the buffer slots distribution specified in k.

4.3. Results

Three categories are considered in the numerical experiments, small, medium, and large production lines with k=7, 21, and 70 machines, respectively. To analyze the behavior of these production lines, three different values for N are proposed, according to equation 10.

$$N = \begin{cases} truncate\left(\frac{k}{2}\right) \\ k+1 \\ 2k \end{cases} \tag{10}$$

Thus, the values used in the experimental are shown in Table 2.

Table 2. Number of slots (N) for every production line category.

	$truncate\left(\frac{k}{2}\right)$	$k+1$	$2k$
Small (k=7)	3	8	14
Medium (k=21)	10	22	42
Large (k=70)	35	71	140

Moreover, the Algorithm 2 was executed 100 times to determine the slots distribution between each machine for a simulation time of 100 units of time. In a first stage, an infinity capacity for each buffer was considered. The maximum number of tokens during the simulation represents the maximum number of pieces in every buffer. Thus, a percentage value was assigned to each buffer, which was useful to distribute the N slots available in the whole production line.

The throughput obtained for each category, considering infinity capacity in all buffers, is shown in Table 3.

Table 3. Production line throughput with infinity capacity buffers.

	Small (k=7)	Medium (k=21)	Large (k=70)
Maximum	0.7699	0.4599	0.0900
Minimum	0.4896	0.2698	0.0200
Average	0.6444	0.3684	0.0525

It can be observed in Table 3 that the smaller is the line the better production line throughput is obtained. Small lines are composed of k=7 machines, and every product is processed only seven times. On the other hand, large lines contain 70 machines, thus every product needs to be processed by seventy machines, plus the waiting time in the buffer when the machine is busy with another piece of work.

Once the slots distribution is known, i.e., how many slots are needed for every buffer depending on the number of available slots (N), and the obtained percentage, we proceed to create 9 TPPNs (3 categories with 3 different values for N), and the Algorithm 2 was executed again 100 times with these new TPPNs. The maximum, minimum, and average throughput obtained in the TPPN simulation is shown in table 4.

Table 4. Production line throughput with finite capacity buffers.

N		Small (k=7)	Medium (k=21)	Large (k=70)
$truncate\left(\frac{k}{2}\right)$	Max	0.5377	0.3696	0.0999
	Min	0.3599	0.2195	0.0100
	Ave	0.4570	0.2956	0.0502
$k+1$	Min	0.6376	0.3999	0.0800
	Max	0.4491	0.2499	0.0100
	Ave	0.5382	0.3342	0.0513
$2k$	Min	0.6687	0.4394	0.0900
	Max	0.4389	0.2698	0.0100
	Ave	0.5738	0.3453	0.0523

From the 100 executions for every TPPN in large, medium and small lines, we obtained the maximum, minimum and average of the production line throughput.

With this strategy for assigning slots according to the percentage values, the throughput tends to be closer to the values obtained when the buffer capacity is infinity.

5. Conclusions

BAP is a NP-Hard Problem where the number of combinations spreads when the number of machines, buffers and slots also increase. An important issue in BAP is the modeling technique to denote all the significative features. Despite many of the research papers regarding the BAP, no one of them applies the mathematical tools of PNs to solve this problem.

In this paper, we propose the use of an extended Petri net model, called Timed Place Petri Net (TPPN). This PN can represent the production line and the buffer capacity among each workstation. It can be used to simulate the behavior of the production line with different configurations.

Three categories were analyzed. Small, medium, and large production lines were depicted by TPPNs. Moreover, the simulation for the TPPN was executed to find the best buffer slots distribution. With this strategy, the algorithm 2 provides the number of slot s assigned to each buffer.

The limitation of this proposal, as many of PN models, is that the number of places and transitions will depend on the number of machines and buffers. In other words, if for every machine we need 2 places and 2 transitions, and for each buffer we need a place, then the incidence matrix dimensions will be 3*k+1 places, and 2*k transitions, in a production line with k machines.

As further work, the proposed TPPN can be used as a modeling tool in the optimization of BAP, where the incidence matrix will allow the calculus of the production line throughput by means of the state equation.

Funding

This work was supported by the Consejo Nacional de Humanidades, Ciencia y Tecnología (Conahcyt), under Project CONACYT CB-2017-2018-A1-S-43008

References

Koenigsberg, E. (1959). Production lines and internal storage—A review. Management Science, 5(4), 410-433.

Liang, Y., Cui, N., Hu, X., & Demeulemeester, E. (2020). The integration of resource allocation and time buffering for bi-objective robust project scheduling. International Journal of Production Research, 58(13), 3839-3854.

Liu, G. (2016). Complexity of the deadlock problem for Petri nets modeling resource allocation systems. Information Sciences, 363, 190-197.

Lopes, T. C., Sikora, C. G. S., Michels, A. S., & Magatão, L. (2020). An iterative decomposition for asynchronous mixed-model assembly lines: combining balancing, sequencing, and buffer allocation. International Journal of Production Research, 58(2), 615-630.

Motlagh, M. M., Azimi, P., Amiri, M., & Madraki, G. (2019). An efficient simulation optimization methodology to solve a multi-objective problem in unreliable unbalanced production lines. Expert Systems with Applications, 138, 112836.

Murata, T. (1989). Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 77(4), 541-580.

Papadopoulos, C. T., O'Kelly, M. E. J., & Tsadiras, A. K. (2013). A DSS for the buffer allocation of production lines based on a comparative evaluation of a set of search algorithms. International Journal of Production Research, 51(14), 4175-4199.

Peterson, J. L. (1977). Petri nets. ACM Computing Surveys (CSUR), 9(3), 223-252.

Skolud, B., Krenczyk, D., & Davidrajuh, R. (2017). Solving repetitive production planning problems. an approach based on activity-oriented Petri nets. In International Joint Conference SOCO'16-CISIS'16-ICEUTE'16: San Sebastián, Spain, October 19th-21st, 2016 Proceedings 11 (pp. 397-407). Springer International Publishing.

Weiss, S., Schwarz, J. A., & Stolletz, R. (2019). The buffer allocation problem in production lines: Formulations, solution methods, and instances. IISE Transactions, 51(5), 456-485.

Yelkenci Kose, S., & Kilincci, O. (2020). A multi-objective hybrid evolutionary approach for buffer allocation in open serial production lines. Journal of Intelligent Manufacturing, 31, 33-51.

Zhang, Z., & Wu, W. (2011). Combined buffer pre-allocation and siphon control for deadlock prevention in Petri nets. International journal of production research, 49(20), 6125-6154.

Zhao, Z., Zhang, G., & Bing, Z. (2011, August). Scheduling optimization for FMS based on Petri net modeling and GA. In 2011 IEEE International Conference on Automation and Logistics (ICAL) (pp. 422-427). IEEE.

Zhou, M. (Ed.). (2012). Petri nets in flexible and agile automation (Vol. 310). Springer Science & Business Media.