A New Generic Petri Net Model for Design and Performance Evaluation of JIT Flexible Manufacturing and Assembly

Roongrat Seeluangsawat[1] and Erik L.J. Bohez[2]
[1] Ph.D student, Industrial System Engineering, School of Advanced Technologies,
Asian Institute of Technology,
P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
e-mail: muayjung@hotmail.com, Mobile Tel. +66-1-668-3131, Fax. +66-2-524-5697
[2] Industrial System Engineering, School of Advanced Technologies,
Asian Institute of Technology,
P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
e-mail: bohez@ait.ac.th, Tel. +66-2-524-5678, Fax. +66-2-524-5697

Abstract: This paper presents a new generic Petri net model for design and performance evaluation of a flexible assembly system with dual kanban. The architectural design of the model is derived from a generic bill of materials and the combination of FMS and FAS concepts. Two approaches for analysis and performance evaluation of the new model base on the theory of invariant analysis and linear programming are also introduced. Finally, the effect of different processing times between FMS-FAS on the number of kanban cards is investigated.

1. Introduction

The purpose of this paper is to propose a new generic black token timed Petri net (PN) model for design and performance analysis of a dual kanban flexible assembly system (FAS). Furthermore the kanban cards go through a series of event that can easily be included in the Petri net model. The kanban card in the Just-in-Time manufacturing is similar with the token in a place.

2. Literature Review

The flexible assembly system as an approach to automated assembly based on highly modular, reconfigurable and reusable process-centered technologies. The reconfigurable system is particularly well suited for today's fast-moving, global economy. Flexible Assembly System (FAS) is a valuable concept for any manufacturer who intends to remain competitive in the future. By combining highly automated, flexible, multitask systems with adaptive controls for real time automated decision making, manufacturer will be able to improve both cost and variability in the assembly process. Flexible Assembly Systems (FAS) is "assembly FMSs" (Andreasen and Ahm [1]). Cheng and Podolsky [3] give the basis meaning of just-in-time, it is producing the necessary units in the necessary quantities at the rigth time. The advantages of JIT are waste reduction, increased competitiveness, better working relations between employees, better working relations with suppliers, higher profits and improved customer satisfaction. Thus, there is no wonder why manufacturers around the globe have adopted the concept of flexible assembly system as their strategies to remain competitive. The outline of the paper is as follows in section 3. We briefly review how to build the generic model with dual kanban.

3. Petri Net Modeling with Dual Kanban in FAS

To build the generic model with dual kanban, we follow the following three steps: (i) build blocks modeling the assembly process with kanbans, (ii) integration of FMS and FAS petri net model and (iii) optimize the petri net model.

3.1 The Basic Building Block

A bill of materials (BOM) is a list of the items needed to produce an end product. Proth and Minis [4] have illustrate how to model the FAS as a petri net based on the BOM. In this paper, we extend Proth and Minis [4] Petri net model and Bohez [2] FMS kanban model by including dual kanban card in the assembly system. The symbolically are denoted, the t_{ij} represents an operation of machine j on part i. In the kanban system, PC_{ij} and MC_{ij} represent production card containers and move card containers respectively. One token placing in PC_{ij} represents one production card, whereas one token in MC_{ij} is represents one move card. Transition k_{ij1} is the act of detaching the move card from part i produced by machine j and then attach the production card to it. On the other hand, transition k_{ij2} is the act of detaching the production card from part i processed by machine j and attaching the move card to it. Place P_{ij3} contains parts with a production card. P_{ij2} contains part i attached with the move card. Finally, P_{ij1} contains part i with a production card attached (see figure 1).

3.2 Integration of FMS and FAS Petri Net Model

A FMS can be considered as a supplier to the FAS. We combine the FMS petri net model with FAS petri net model as shown in figure 2. From the figure, you can see that the FMS and FAS are linked by the transition BP12 where BP stands for "bridge to production". The idea stems from the fact that a production card should be initially exist from the subsequent process in order to enable the manufacturing process of a part. Next, the production card will flow through the path PC21-BP11-PC11 (see figure 2 for detail). Then, the FMS will sequentially produce part 1, part 2 and part 3 for the FAS in the process t11, t21 and t31. As a pull system, these manufacturing processes are enabled only when parts are required by the assembly process. For example, if there is an order for product P11, then part 1,part 2, and part 3 will be assembled together by assembly operation a21 and a11 respectively. A new naming system is defined in order to distinguish the FMS and FAS petri net model. All places containing parts with production card, move card and assembly card are named Aij3, Aij2 and Aij1 respectively. All kanban transition in the assembly process are named ak_{ij2} and ak_{ij1}. All assembly transitions are named a_{ij} representing the assembly process j performed at the station i. All places holding complete products (P1j in the BOM) are called A1j. All places containing the production card detached from component ij in the assembly process are referred to as APCij. All kanban *transitions detaching* assembly cards from product P1j are called sk_{ij3} (see the label in figure 3).

3.3 Optimize the Petri Net Model

Elementary circuits with cycle time greater than the cycle time of the sequencing circuit are selected for the optimization. The initial marking will be determined in such a way that there is one token in each elementary circuit. This condition guarantees the liveness of the PN. Physically this corresponds to a deadlock free FMS-FAS. Some simple rules to do this can be given: all the kanban/part circuits should contain one token. The rule of thumb here is put a token in a kanban card place instead of a part place because this does not increase the work in process. In order to perform an analysis, we used two software tools; Integrated Net Analyzer (INA) and LINGO. INA is used as a tool for finding all the elementary circuits [5]. On the other hand, LINGO is used as a tool to solve the model. Two methods for optimizing the petri net model are suggested as the following:

Method I: This method solves the optimal solution without initial marking in the model. The model is given below:
The minimum work in process can be formulated as a linear programming problem (LP).

$$MIN\sum_{i,j} P_{ij3} + P_{ij2} + P_{ij1} + A_{ij3} + A_{ij2} + A_{ij1}; \quad (1)$$

Subject to $M(\gamma) \geq \frac{\tau(\gamma)}{C(\gamma_C)}$ at all γ (2)

$\forall$ P_{ij3}, P_{ij2}, P_{ij1}, MC_{ij}, PC_{ij}, C_{ij}, Integers ≥ 0

Remark: γ_c = bottleneck sequencing circuit

$C(\gamma) = \frac{\tau(\gamma)}{M(\gamma)}$ we can write the constraints as $M(\gamma) \geq \frac{\tau(\gamma)}{C(\gamma_C)}$

which are linear because $\tau(\gamma)$ and $C(\gamma_C)$ are constants due to the fact that the number of servers was fixed. $M(\gamma)$ is the sum of tokens in the places of the circuit γ.

Equation (2) also includes the liveness condition of $M(\gamma) \geq 1$, all variables are integer.

Method II: The method II gives the optimal solution with initial marking. The result of this model is based on previous initial marking. The maximum throughput with minimum work in process can be formulated as a linear programming problem (LP).

$$MIN\sum_{i,j} P_{ij3} + P_{ij2} + P_{ij1} + A_{ij3} + A_{ij2} + A_{ij1};$$

(Place names = number of tokens in that place) subject to the $C(\gamma) \leq C(\gamma_c)$ $\forall\gamma$ with $C(\gamma) \geq C(\gamma_c)$ after initial marking stage

γ_c = bottleneck sequencing circuit

$\forall$ P_{ij3}, P_{ij2}, P_{ij1}, MC_{ij}, PC_{ij}, C_{ij}, Integers ≥ 0

For both method I and method II, we must include tokens (or # of servers) of the sequencing circuits in the constraints in order to find the global optimum. Changing the problem to a mixed integer-programming problem. No assignment of tokens in the sequencing places C_{ij} is done and the following constraints are added to the above linear programming problem.

$\forall$ ij $C_{ij} = X_{ij}$; X_{ij} = 0, 1, 2 .. # servers in station j;

$\forall$j $\sum_i X_{ij}$ = # servers of machine j; (3)

4. Effect of Processing Times in FMS-FAS on Kanban Cards

This section focuses on the effect of different processing time between FMS-FAS on the number of kanban card. In our case, there are two stations, which are FMS and FAS station. The first operation start at FMS station (this case FMS station has one machine) which produce three parts. The parts are send to assembly. The assembly *operation is* following BOM in figure 1-a. Based on the model in figure 3. The model consists of five main processes. The processing time of three parts in FMS (t11, t12, t13) and the assembly time in the two level BOM in FAS (a11, a12). To investigate the effect of the number of kanban card on the bottleneck station, the first thing we have to do is studying the balanced system. The balanced system has the same total processing time in the FMS station and the FAS station. The cycle time is equal to $\sum_{i,j} a_{ij} or \sum_{i,j} t_{ij}$. Thus, the scenario 1 is a balanced system. Next, we investigate the unbalanced system influence by the number of kanban card. One scenario is when FMS station is the bottleneck station ($\sum_{i,j} a_{ij} < \sum_{i,j} t_{ij}$) (scenario 2). And another one is when FAS station is the bottleneck station ($\sum_{i,j} a_{ij} > \sum_{i,j} t_{ij}$) (scenario 3) (see table 1.). The cycle time of the system is the cycle time of bottleneck station. In scenario 2, the cycle time is $\sum_{i,j} t_{ij}$.

In scenario 3, the cycle time is $\sum_{i,j} a_{ij}$.

The initial makings of petri nets are the same because the elementary circuits are the same. However when we change the processing time, we can change it directly in a spreadsheet package (Microsoft Excel). In each spreadsheet, the critical circuit can be identified. However in each scenario the performance can be improved by relaxing the LP model to an Integer programming model. The method I is solves the model without initial marking in the model. The method II solves the model with initial marking.

4.1 Determine the Optimal Model in Each Scenario

The optimal result (figure 4) is found by solving the mixed integer program (1) and (2). From INA, there are 31 elementary circuits in this petri net model. 31 circuits result in 31 constraints. The right side of the constraints is the total processing time of each elementary circuit divided by the cycle time of critical sequencing circuit.

In figure 5 we can see that there is still one elementary circuit (circuit 19) with the cycle time (circuit 14) greater than the cycle time of the sequencing circuit of the assembly (circuit 12). The ideal condition (the maximum output at the minimum work-in-progress) can be mathematically represented as a linear programming problem (LP):

MIN P113+P112+P111+P213+P212+P211+P313+P312+P311 +A313+A312+A311+A323+A322+A321+A213+A212+A211+A 223+A222+A221+A11

Each place name represents the amount of tokens (the work-in-progress) in that place.

Table 1: The three different scenario of processing time

Scenarios	FAS station		FMS station			Compare the processing time between two station
	a11	a12	t11	t12	t13	
1	6	6	3	3	6	$\sum_{i,j} a_{ij} = \sum_{i,j} t_{ij}$ = 12 min
2	3	3	3	3	3	$\sum_{i,j} a_{ij} < \sum_{i,j} t_{ij}$ (6 min< 9 min)
3	9	9	3	3	3	$\sum_{i,j} a_{ij} > \sum_{i,j} t_{ij}$ (18min>9 min)

The cycle time of elementary circuit 19 (of which the total processing time equals to circuit 14) requires to be less or equal to twelve, the new amount of tokens to be added to this circuit can be determined by: number of tokens = total processing time / cycle time = 14 / 12 = 1.16

Elementary circuit 19 should totally hold tokens equal to or more than 1.16.

$A211 + CA21 + A213 + A212 \geq 1.16$

The linear programming problem (LP) above does not deal with the tokens in the sequencing circuits themselves. To resolve this, the problem is changed into a mixed integer programming problem (MI) by modifying the LP LINGO model above to:

C11 = X11, C21 = X21, C31 = X31 and CA11 = XA11, CA21 = XA21, where X11, X21, X31, XA11, XA21 can be only 0 or 1, and X11 + X21 + X31 = 1 and XA11 + XA21 = 1

The result (figure 6) from the mixed integer programming problem suggests to move the token from sequencing place C11 to C31 and place CA11 to CA21.

4.2 Result

The performance analysis of senario 2 and 3, is found in the same way as in section 4.1. The optimal result from method I and method II is shown in figure 6.

4.3 Effect of Processing Time on the Number of Kanban Cards

The scenario 1 and 3 show the same result. The scenario 2 is different. From figure 6, scenario 1 and 3 have a token in A212, which means the assembly is authorized to move on but in scenario 2, the move card is still in move card container. We can conclude that in scenario 1 and 3, most of the time the move card is attached to the part.. In figure 4.4, when we optimize the model based on the initial marking, the result shows that one more move card is added in place A212, in scenario 1 and 3, which mean we have one more move card attached with the assembly part. Using two ways to optimize the model, the results show that the different methods (with/without initial making) give different results but give the same conclusion. The effect of different processing times between FMS-FAS on the number of kanban cards is the same. In other words, we can conclude that when the utilization of the FMS cell is much higher than the FAS cell (FMS is bottleneck station), the number of move cards have no effect on the performance. The move cards are not necessary to regulate the transfer part. When the FMS cell is much faster than FAS cell, the move cards do play an important role in the performance. When the model is balanced, the move cards play an important role in the performance an are also necessary.

5. Conclusions and Recommendations

A new generic deterministic petri net model of FAS with dual kanban card was proposed. Firstly, invariants are determined (INA). Based on the invariants, the performance is optimized using a spreadsheet (Microsoft Excel) and LINGO (solving a linear programming problem and mixed integer programming problem). One way is optimize the number of tokens in the petri net model without initial marking. Another one is optimize the number of tokens in the model with initial marking. Finally, the effect of different processing times between FMS-FAS on number of kanban card is investigated. The results show that the move cards are necessary when the upstream cell (FMS) is much faster than down stream cell (FAS). The basic building block proposed by the study can deal with an increase in the level of the tree-structured BOM.

References

[1] Andreasen M. M. and Ahm T., 1988, "Flexible Assembly System, UK:IFS Publications.

[2] Bohez E. L. J., 2002, " A New Generic Time Petri Net Model for Performance Analysis of a Dual Kanban FMS", to a appear in Production Planning and Control.

[3] Cheng T. C. E. and Podolsky S., 1996, "Just-In-Time Manufacturing: An Introduction (2nd edition)", Great Britain: Chapman&Hall.

[4] Proth J.M. and Minis I., 1995,"Planning and Scheduling Based on Petri Nets", in Petri Nets in Flexible and Agile Automation (ed. M. Zhou). USA:Kluwer Academic Publishers.

[5] Starke H.P., Integrated Network Analyser (INA), http://www.informatik.huberlin.de/lehrstuehle/automaten/ina/

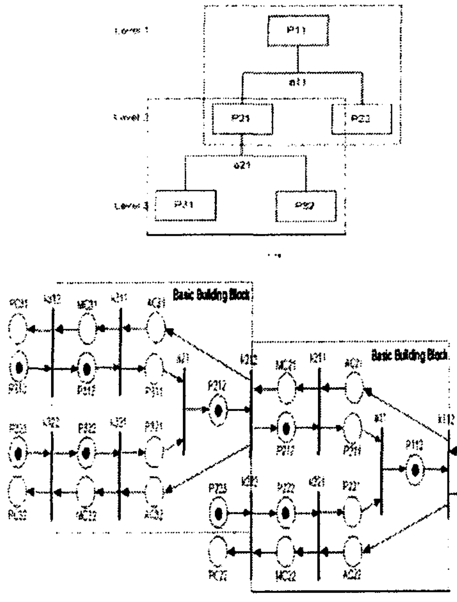

Figure 1: Bill of materials with three levels and its petri net model

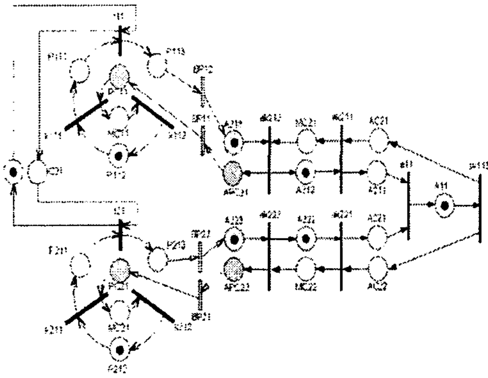

Figure 2: New naming system for the integrated model.

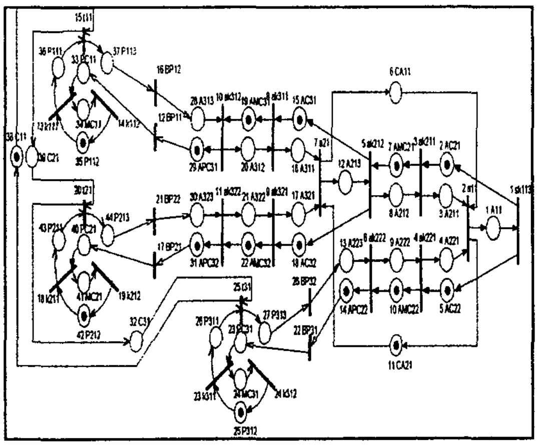

Figure 3: The initial making of FMS-FAS model with two assembly processes

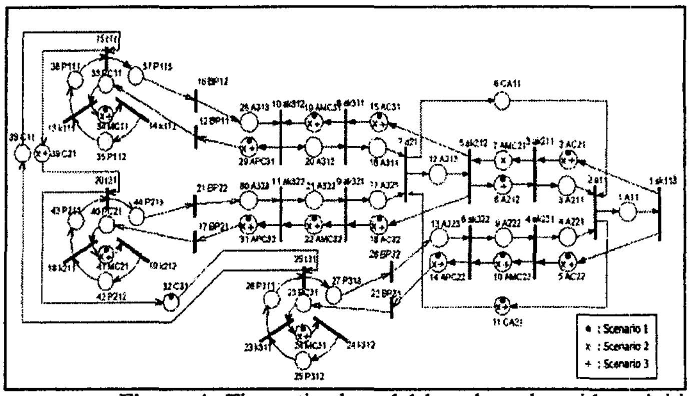

Figure 4: The optimal model based on the without initial marking

Transition Time	1	1	6	6	1	6	1	1	1	1	6	1	1	0	1	6	6	1	1	1	1	1	1	1	1	6	0	1	0	1	0	6	1	1	1	3	0	3	3	1	1	1	3	0				
Transition	[illegible]	[illegible]	A11	A11	[illegible]	A11	AK212	AK211	AK221	AK222	A21	AK213	AK223	BP31	AK311	A21	A31	AK321	AK312	AK311	AK321	AK322	K311	K312	K311	T31	BP32	AK312	BP11	AK322	BP21	T31	K111	K112	K111	T11	BP12	T11	T21	K211	K212	K211	T21	BP22			Max	
# Token	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	0	1	0	0			14	
Place	A11	AC21	A211	A221	AC22	CA11	AMC21	A212	A222	AMC22	CA21	A313	A323	APC22	AC31	A311	A321	AC32	AMC31	A312	A322	AMC32	PC31	MC31	P312	P311	P313	A313	APC31	A323	APC32	C31	PC11	MC11	P112	P111	P113	C11	C21	PC21	MC21	P212	P211	P213				
Nr.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	M(c)	t(c)	c(c)	t(c)/12
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	2	2.00	0.167
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
4	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
5	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8	8.00	0.667
6	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8	8.00	0.667
7	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
8	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8	8.00	0.667
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2.00	0.167
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	5	5.00	0.417
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	5	5.00	0.417
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	12	12.00	1
14	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	12	12.00	1
15	1	0	0	0	1	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	4	30	7.50	2.5
16	1	0	0	0	1	1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	4	27	6.75	2.25
17	1	1	0	1	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	5	26	5.20	2.167
18	1	1	0	1	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	0	5	23	4.60	1.917
19	0	0	1	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	14	14.00	1.167
20	1	1	0	0	0	1	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	16	5.33	1.333
21	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8	8.00	0.667
22	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	16	5.33	1.333
23	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8	8.00	0.667
24	1	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0	0	0	1	5	26	5.20	2.167
25	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0	0	0	1	3	18	6.00	1.5
26	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1	4	32	8.00	2.667
27	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0	1	1	0	1	0	0	1	0	6	32	5.33	2.667
28	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0	1	1	0	1	0	0	1	0	4	24	6.00	2
29	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	4	29	7.25	2.417
30	0	0	0	1	0	0	0	0	1	0	1	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	4	30	7.50	2.5
31	0	0	0	1	0	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	0	4	27	6.75	2.25

Figure 5: Initial number of token and the cycle time calculation of scenario 1

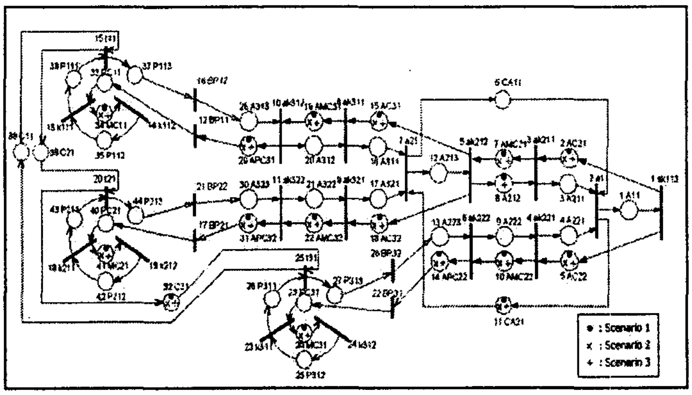

Figure 6: The optimal model based on the kanban rule initial marking