Stock Optimization of a Kanban-based Assembly Line

M. Dub[1], G. Pipan[2] and Z. Hanzálek[1]

[1] Department of Control Engineering,
Czech Technical University, Faculty of Electrical Engineering,
Technická 2, 166 27 Prague 6, Czech Republic[1]

[2] XLAB
Teslova 30, SI-1000 Ljubljana, Slovenia

ABSTRACT

The objective of this paper is to describe a way to optimize the stock reserves for an existing assembly line where the parts are supplied according to the Kanban-method. The optimization is based on a simulation, for which a simulation tool was developed. The tool uses matrices resulting from a Stochastic Time Petri Net model of the assembly line (its size leads to a special kind of sparse matrices) and performs the simulation. The goal is to simulate the worst-case scenarios (e.g. delivery delays).

1. INTRODUCTION

The article deals with an existing assembly line, which produces cars. The parts needed for the assembly are delivered by the method Kanban. The intention was first to provide a simulation of production of an existing assembly line and then to optimize the numbers of the Kanbans for each part so that the stock reserves cover the production of a user-defined time length. This approach will be useful namely in the design of new assembly lines when simulations provide a convincing material for the manufacturer.

2. SYSTEM DESCRIPTION

2.1. KANBAN-METHOD

The term “Number of Kanbans”, as further used in this article, means the number of the Kanbans for one part type. The number of the tickets influences the system behavior: the higher the number is, the bigger the stores must be and the safer the system is (there are

[1] Phone (420) 2 2435-7434, Fax (420) 2 2435-7610, E-mail: dub@rtime.felk.cvut.cz, hanzalek@rtime.felk.cvut.cz , WWW: http://dce.felk.cvut.cz

more parts in the stores). The lower the number is, the more risk must be taken into account but the less expensive the production might be.

When the operator takes the first part away from the palette, he should also detach the Kanban from it and place it into the Kanban-container. At certain times the container is withdrawn and a reference list of Kanbans is made (needed parts). This reference list is then sent (typically 4 or 5 times a day) via fax to the main store (see Figure 1).

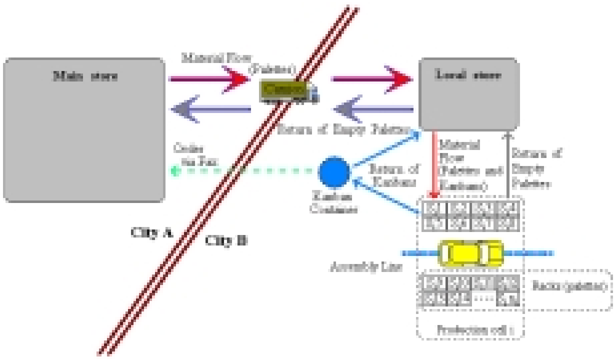

Figure 1: Kanban-method for the parts delivery

The major disadvantage of the Kanban method is that a missing Kanban might stop the whole production. Various researchers have studied the Kanban-method from the theoretical [4, 5, 7] and practical [1, 2, 8] point of view.

2.2. Assembly Line

The assembly line is based on a self-moving transporter, which carries the car bodies through a number of quite similar production cells that differ in the assembly actions. The time, which the car body spends in every production cell, is equal for all production cells and is given by the line rhythm (= constant). The product flow is linear – there are no places where the production splits or where the outgoing flow results from more incoming flows. Each production cell has some small stores (racks) where palettes with all parts specific to the particular production cell are to be found. In every cell there is a space to accommodate at maximum two palettes of each part type used there. One palette contains only the same parts e.g. “Oil Filter”, “Safety Belt-Front Left”, “Bumper”,...

3. Modeling

3.1. One Part Assembly Model

First a model describing the assembly of one part was done (see Figure 2). The model is based on Stochastic Time Petri Nets [3, 6]. It is a general sub-model, which was later used for each single part of the assembly line. The sub-models differentiate in parameters corresponding to the particular part type:

- **Number of parts in one palette** (*PartsPal*) – arc T1 → P1 (e.g. 100)
- **Number of parts of the same type for one car** (*PartsCar*) – arc P1 → T2 (e.g. 2)
- **Number of Kanban-tickets** (*KTickets*) – P3 and P5. The max. number of the palettes in the rack is 2 (conservative component {P3, P4}). If *KTickets* > 2, the resting palettes stay in the local store (P5). P4 sets the limit – a token occurs there first when T1 is fired ⇒ there is one palette less in P3 ⇒ T3 may be fired. The initial marking of both P3 and P5 in the fig. is 2, which represents 4 Kanbans.

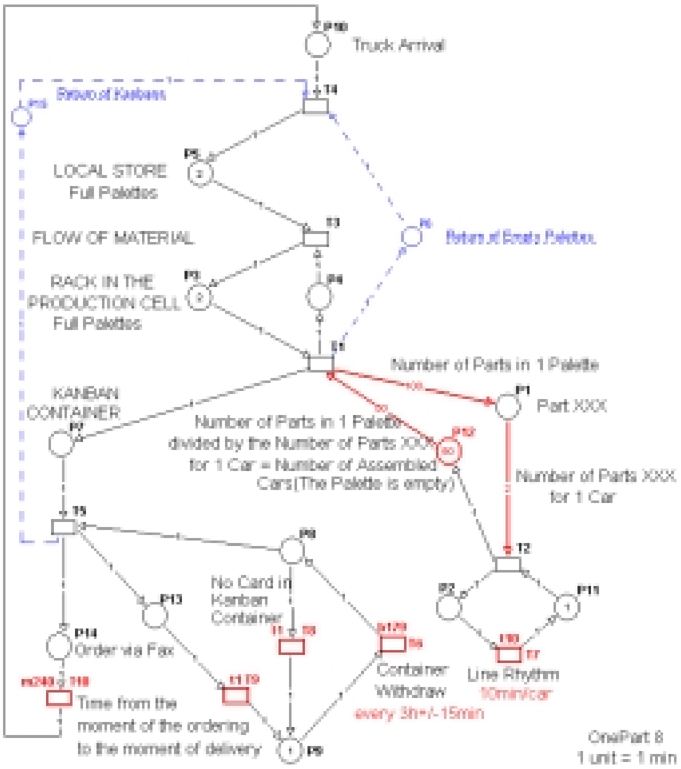

Figure 2: One part assembly PN-model (P6 and P15 are implicit places)

The parameters, which are supposed to remain the same for all parts:

- **Order execution** (the time interval from the order sending to the truck arrival) – T10. To describe this stochastic time transition a cumulative distribution function was prepared (based on the company database).
- **Line rhythm** (the time interval between the production of two consecutive cars; it may be also understood as the production speed) – transition T7 (e.g. 10 min) represents the assembly operation (the mounting of the part into the car).

- **Container withdraw** (the average time interval between two consecutive orders of the parts – here set to 180 min) is described by the transitions T5, T6, T8 and T9 and places P8, P9 and P13. The meaning is as follows: 179 min after the occurrence of a token in P9, the timed transition T6 is fired (initial marking of P9 is 1 ⇒ the firing is enabled; there is always one token in the conservative component {P8, P9, P13}) and a token appears in P8. If there is a Kanban in P7, T5 is immediately fired (the Kanban-container is withdrawn). If not, the timed transition T8 is fired (its delay as well as the delay of T9: 1 s). This assures that the Kanban-container is withdrawn every 180 min (the sum of the firing times in the both firing sequences T6, T5, T9 and T6, T8 is equal to 180min).
- Place P12 – number of parts in one palette divided by the number of parts needed for one car. This number also corresponds to the number of cars into which the parts of one palette might be mounted (e.g. 100 parts, 2 parts per car: M(P12)=50).

3.2. THE WHOLE ASSEMBLY LINE

Based on the one part assembly model, a model of the whole assembly line was done (see Figure 3). Let us remember that neither the movement of the cars along the assembly line according to the production cells nor the workforce distribution but only and only the number of the parts for the assembly are to be described.

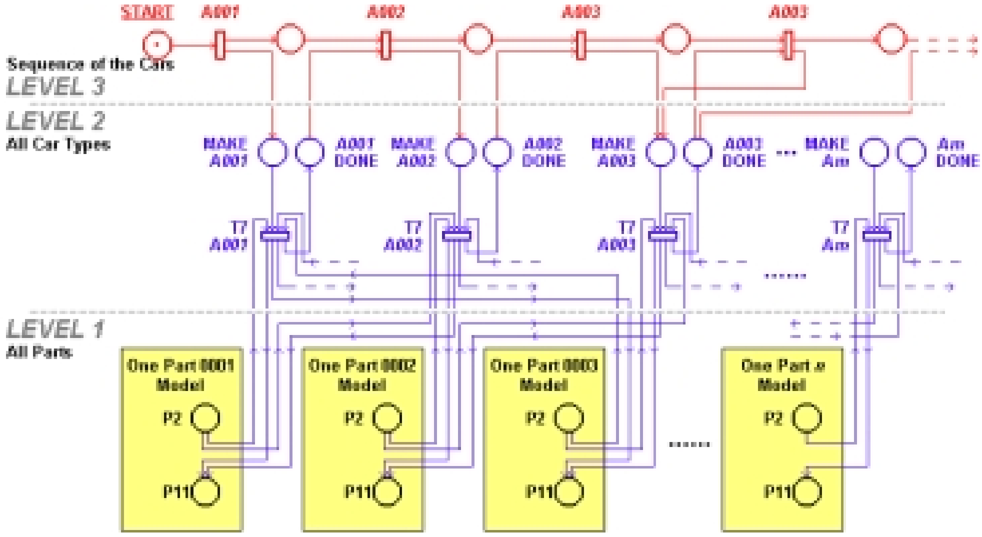

Figure 3: The whole assembly line model

The model has 3 levels:

Level 1: Parts definition. One Part Assembly Models (see the section above) of all the existing parts (approx. 1750) in the car assembly.

Level 2: Car types definition. For each car type selected parts are needed: they are connected via one transition (T7 A001), which describes the mounting of all the required parts into the selected car type (e.g. A001).

Level 3: Production plan i.e. the sequence of the cars on the assembly line.

3.3. MATRICES

The Petri Net model (see Figure 2 or Figure 3) was then translated into a series of matrices, which describe the Petri net. The matrices may be seen in Table 1.

Matrix	Meaning
Pre	Matrix of preconditions, m x n (rows: places, columns: transitions; the intersections show the value of the connection from the place to the transition)
Post	Matrix of postconditions, m x n (rows: places, columns: transitions; the intersections show the value of the connection from the transition to the place)
M0	Column vector of initial markings, m x 1 (rows: places)
TimeT	Column vector of times associated to the transitions, n x 1 (rows: transitions)
TypeT	Column vector of transition types, n x 1 (rows: transitions):

Table 1: Matrices describing a PN-model

4. SIMULATION METHODS

The result of the simulation should be stored in two matrices:

- *Sequence* – contains the numbers of transition that were fired and the corresponding time instants
- *Marking* – contains the markings of all places in the corresponding time instants, i.e. the first column describes the initial marking; the last column is then the marking at the end of the simulation.

Two different simulation approaches were used: MatLab and then C++ tool that was developed to enlarge the size of the simulated system.

4.1. BY MEANS OF A MATLAB FUNCTION

At the beginning MatLab was used because it seemed to be the easiest way to develop the simulation on the logical level and then to work with the matrices arising from the modeling (Table 1). This approach was limited by the computer memory size, which restricted both the simulated system size and the simulation length (ex.: for 27 parts 5000 min of production).

4.2. BY MEANS OF A C++ TOOL

After developing the simulation on the logical level and moving to the real simulation environment, some of the MatLab drawbacks can still be recognized. When calculating small matrices in MatLab, the program execution is acceptable but when running on a realworld system, there are some disadvantages in comparison to a tool written in a native code. MatLab is an interpreter, meaning that it uses more memory and more time to compute the same problem than a native code. The new simulation tool was developed in C++ language, which guarantees shorter execution times and less memory usage than a MatLab function.

4.3. PN Matrices Representation

The biggest matrices used in the simulation are the matrices *Pre* and *Post* describing the PN architecture. Since there are only a few connections among the places and transitions, most of the values of the matrices *Pre* and *Post* are zeros (no connection). Therefore, a sparse matrix representation was considered.

The simulation results are stored in two matrices: *Sequence* and *Marking*. Both of them have as many rows as many simulation steps are made in the system, meaning that in longer period simulations, this could cause a serious problem with the memory consumption. That is why the simulation results are saved into a file after each calculation of predefined number of steps, which leads to minimization of the memory usage. The simulation length is then restricted only by the hard disc space.

4.4. Time and Space Complexity

When simulating a PN, in each cycle only the changes caused by firing of one transition are recalculated. With a column oriented computation, the time complexity of each cycle is *O(n),* where *n* is the number of the places (max. number of possible firings is *c*, *c* << *n*). From the same reason the space complexity can be evaluated also as *O(n).*

4.5. Representation in Memory

Knowing the system, each matrix was represented as an array of columns, where each column of the original matrix is encoded as an array of pairs of the row index and the value. In this way a semi-sparse matrix was created, which can be better described as "*an array of sparse vectors*" (Table 2). This kind of encoding reduces considerably the size of the matrices: the original matrix *Pre* describing one car type production (1750 parts): 24500 x 15750. The encoded semi-sparse matrix *Pre*: 28000 x 3 (approx. 4500 times less).

	Column 1	**Column 2**	**...**	**Column *m***
Element 1	R: 1, V: 1	R: 3, V: 2	...	R: 5, V: 3
Element 2	R: 4, V: 2	R: -1, V: -1	...	R: -1, V: -1
Element 3	R: -1, V: -1			
....				

Table 2: Memory representation as a semi-sparse matrix

4.6. Comparison of Both Simulation Approaches

Both the simulation system size and the simulation speed depend greatly on the computer. The comparison of both approaches presented in Table 3 and Table 4 describes the results achieved on similar computers.

	MatLab	C++ Tool
Maximum of parts	150	1200
Maximal simulation steps	100-1000 steps	No limit

Table 3: Comparison of the maximum size of the simulated system

	No. of parts	No. of sim. steps	Simulation time
MatLab	120	100	3 min
C++ Tool	900	500	3 min

Table 4: Comparison of the execution times of both simulations

5. Simulation Results

In the simulation results the markings of places corresponding to the store and the transition firing times defining the sending times of orders or to the delivery truck arrivals were searched. These values showed if the simulation corresponded to the practice.

Data from the company were obtained describing the production period of 31 days (2058 cars produced). Various simulations were performed that differed in the number of the Kanbans for each part and in the length of the simulated production interval. The most interesting results are explained in the following sections.

5.1. Case "The Real One"

This is the simulation with the real values. The number of the Kanbans in this case corresponds to the real situation in the production.

A simulation representing 5000 min of production was performed. For each part the following data were recorded (ex. may be seen in Table 5):

1. **Part #** – Part numbers (1 to 1750)
2. **Min** – Minimal counts of the palettes in the store during the simulation. It is obtained as the min(M(P3)+M(P5) for all the time instants).
3. **Max** – Maximal counts of the palettes in the store during the simulation = max(M(P3)+M(P5) for all the time instants).
4. **Sim.** – Ordered palettes in simulation (e.g. in 5000min) = firings of T10.
5. **Practice** – Ordered palettes in 31 days (2058 cars produced) – these are the numbers taken from the practice (company data).
6. **2058** – Necessary palettes to produce 2058 cars (not a simulation).
7. **Estimation** – An estimation of orders for the time period of 31 days based on the simulation results described in the column Sim., which was multiplied by a coefficient describing the difference in the production period between the practice and the simulation.

8. **Difference** – Corresponds to the difference (2058 – Estimation) and indicates the difference between the simulation and the reality.

Part #	Min	Max	Sim.	Practice	2058	Estimation	Difference
1	11	14	82	374	343	337	6
4	3	6	37	168	159	152	7
9	0	2	24	93	103	98	5
24	2	3	17	67	69	69	0
25	3	5	15	61	59	61	-2

Table 5: Case "The Real One" (all values except the first column represent the numbers of palettes)

It was observed that the longer the simulation runs, the closer to the practice the simulation results are. One of the reasons is that the simulation does not take into consideration the initial state of the store and the influence of this difference is reduced in longer simulations. The time for the simulation does not influence all the simulation results: the minimal and maximal numbers of the palettes in the store remain the same and do not depend on the simulation time. It means shorter periods of time are sufficient for the optimization. Ex. in Figure 4 describes the evolution of the number of one of the parts in the store.

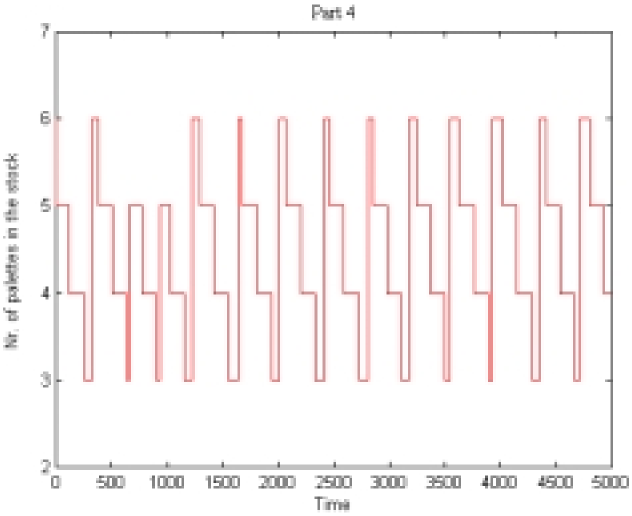

Figure 4: Stock evolution in time for Part 4 and the case "The Real One"

5.2. *CASE "THE SAFE ONE"*

To optimize the number of Kanbans for specific part, the following formula was used:

$$KTickets_{SAFE} = MAX - MIN + NrPalettes_{SAFETY} \quad [1]$$

where $MAX - MIN$ defines the number of Kanbans necessary for the continuous production. $NrPalettes_{\mathrm{SAFETY}}$ (differs part to part) corresponds to the number of cars that can be furthermore produced after the delivery pauses:

$$NrPalettes_{\mathrm{SAFETY}} = \mathrm{ceil}((SafetyTime * PartsCar) / (LineRhythm * PartsPal)) \qquad [2]$$

where *ceil* rounds a number towards infinity, *SafetyTime* is the time interval after a delivery stop for which the production should run (in minutes, user-defined) and *LineRhythm* is the production speed in minutes.

5.3. DELIVERY DELAY SIMULATION

To examine the assembly line behavior in an "emergency case", a simulation of a delivery delay was performed. According to the company routine, the delay was set to 120 min, which should be still covered by the stock reserves. However, in the case "The Real One" the production stopped because there was not enough of Parts 9. The stoppage lasted for 73 min, which represents a lost of 7 cars (production speed 6 cars/h). The circled values in Figure 5correspond to the following situations:

① The truck is loaded and ready to leave the main store (T5 fired).

② The truck accident.

③ The parts are finally delivered to the local store (T10 fired).

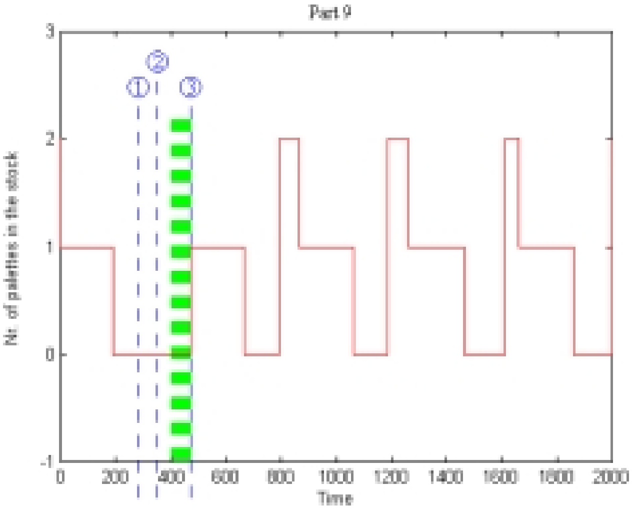

Figure 5: Stock evolution in time for Part 9 and the case "The Real One With A Truck Accident"

It may be seen that if a truck delivery failed and the withdrawn parts were delivered with a shift of 120 min, the assembly line would stop (case "The Real One", the marked time interval in Figure 5). Our proposition (case "The Safe One") is able to cover such a shock without an excessive stock (see the Equation [1] and [2]). The case "The Real One" has also a delivery delay limit that can be covered (47 min) but the stock reserves are not optimized:

for some parts they are overestimated (e.g. Part 1: approx. 6 h) while for some other ones they are underestimated (e.g. Part 9: 47 min).

6. CONCLUSION

This article shows how the stock reserves could be improved with respect to the involved risk and store space. Simulation results of two cases were compared: case “The Real One” describing the real production and the case “The Safe One” optimizing the number of the Kanbans for each part.

It was shown that the total maximal number of the palettes in the case “The Real One” is 119 (= 100%) while in the case “The Safe One” it is only 77 palettes (= 65%). If the delivery of the parts stopped, the stock reserves would cover only 47 min in the case “The Real One”, while in the case “The Safe One” it would be at least 120min of production. With the new approach the store space was saved by 35% covering a longer period of “unsupplied” production at the same time.

For simulations of even bigger systems, a parallel version of the C++ simulation tool might be used. If the places were grouped in the right way, the network traffic would be lowered, enabling the system speeding-up, too. This might be the subject of the future development.

Acknowledgements: this research work has been supported by the research program No. J04/98:212300013 "Decision Making and Control for Manufacturing" of the Czech Technical University in Prague (sponsored by the Ministry of Education, Youth and Sports of the Czech Republic).

8. REFERENCES

[1] Aytug H., Dogan C. A.: “A framework and a simulation generator for kanban-controlled manufacturing systems”, Computers ind. Engng. 34 (2), pp. 337-350, 1998

[2] Chaouiya C., Liberopoulos G., Dallery Y.: “The extended Kanban System for Production Control of Assembly Systems”, Technical Report, MASI, January 1997

[3] David R., Alla H.: “Du Grafcet aux Résaux de Petri”, Deuxième édition revue et augmentée, Editions Hermès, Paris, 1992

[4] Di Mascolo M.: “Evaluation des Performances des Systèmes de Production”, Journées Flexibilité des Systèmes de Production et Compétitivité de l’Entreprise, IMACS-IEEE/SMC CCESA ’96, Lille, France, July 1996

[5] Karaesmen F., Dallery Y.: “A performance comparison of pull type control mechanisms for multi-stage manufacturing”, International Journal of production economics 68 (2000), pp. 59-71, 2000

[6] Marsan A., Balbo G., Conte G., Donatelli S., Franceschinis G.: “Modeling with Generalized Stochastic Petri Nets”, Wiley, Chichester, England, 1995

[7] Panayiotou Ch. G., Cassandras Ch. G.: “Optimization of kanban-based manufacturing systems”, Automatica 35 (1999), pp. 1521-1533, 1999

[8] Savsar M.: “Simulation analysis of a pull-push system for an electronic assembly line”, International Journal of production economics 51 (1997), pp. 205-214, 1997