Performance Analysis of an Assembly System: a Case Study

Jean-Luc Cojan, Loic Plassart, Frank Singhoff, Philippe Le Parc

Performance Analysis of an Assembly System: a Case Study

Jean-Luc Cojan, Loïc Plassart, Frank Singhoff, Philippe Le Parc
LISyC - Laboratoire d'Informatique des Systèmes Complexes
Université de Bretagne Occidentale
CS 93837 - 29238 Brest Cedex 3 - France

Abstract

Petri nets are well suited for modelling production systems and analysis of their performance. In this paper we study a flowshop system driven by a set of local command units and a central controller, modelled with Timed Coloured Petri nets by means of CPN Tools. We show that Petri nets can be applied not only to improve its production rate by comparing various algorithms for the controller policy service, but also to analyse the significance of parameters as conveying and mechanical delays, maximum work-in-process or to understand problems appeared in the real system.

Keywords Petri nets, modelling, simulation, manufacturing systems, case study.

1 Introduction

The behaviour of a production system is not only conditionned by mechanical characteristics of the machines, but also by the equipment which ensures their control [4]. The designers of Livbag company[1], a worldwide leader in automotive safety, are confronted with this problem. The expected production targets are far from being met. Some problems, as apparent stoppings of the production, are even noticed. In [8] Plassart established the need for limiting the controller response time to messages from operative parts. He modeled the system with a FIFO policy service. We will extend this study to other policies. Moreover, we will show that Petri nets are also useful in the analysis of the significance of parameters such as conveying delays, maximum work-in-process or the steady state settling and may allow a better understanding about the origin of the encountered problems.

In this article, we assume the reader is familiar with Petri nets (see [7] for a general survey and [5] for coloured Petri nets). In section 2, we present the production system, the operating cycle of the machines and their modelling. In section 3, we settle bounds for mean inter-arrival delay and makespan to be compared with the simulation results shown in section 4. Due to the lack of space, this study turns only on linear flowshop with a single processor (see section 2.3).

[1] Société Livbag, groupe Autoliv - Route du Beuzit, 29590 Pont de Buis, France

2 Assembly system description, classification and modelling

In this section, we describe the architecture of the system and the operating cycle of the machines, then we propose a classification of the assembly lines according to their topology which allows to formalize a station by its characteristics.

2.1 System architecture

The considered production system is an automated assembly process with several machines called stations linked together by conveyors (figure 1). The stations work in an independent way from each other and execute their operating cycle. A station cannot retain and operate more than one part at a given moment. When a station is available (no assembly in progress) and a part is present at its entry, it starts its operating cycle. Storage capacity on conveyors and in the entry of the station is limited by means of sensors.

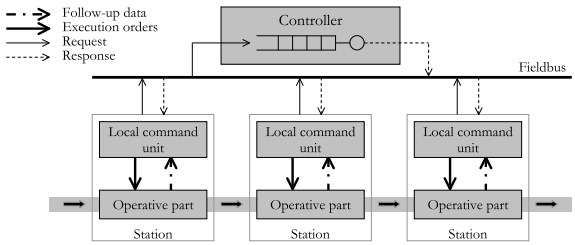

Figure 1: System architecture.

The control of the assembly line is ensured by a central controller which coordinates the various stations. Thus, it has to be considered as a shared resource of the system. In literature, many manufacturing control architectures are identified [2]. They are often declined in three main types from centralized over hierarchical to heterarchical control. Our control architecture is based on a typical hierarchical structure in which an upper level device coordinates the activities of a group of lower level devices in a master-slave manner [6].

In the present case study, the message exchanges are initiated by the local command units. They are operated according to a request transmission and a response reception. The stimulus is then bottom-up and more than one exchange can be running at the same time and then messages

are stored in a buffer. One of our aim is to evaluate the impact of the message service policy.

2.2 Operating cycle of the stations

The operating process of each station can be split up into five phases:

- identification phase, executed as soon as the part enters the station,
- status request phase. After a possible waiting time in the buffer, the request is processed by the controller and a response is sent back to the station. If the processing is not granted, the part is released, otherwise the process goes on,
- assembly phase, operating sequence completely controlled by a programmable logic controller and immediately executed on receiving the status reply,
- data reporting phase. A message with the necessary measurements for traceability purpose is sent to the controller,
- release phase, performed immediately on receiving the acknowlegment from the controller. This phase is also conditionned by the maximum capacity of storage of the next station or maximum work-in-process (WIP for short), which includes parts either on conveying or pending to be processed. In this paper, parts processed by a station are not considered as WIP.

There is a growing demand from the designers of production lines for increasing the number of messages during a cycle, in order to avoid hazardous manipulations or to save raw materials in case of failure during one of the step of the assembly phase for example. Hence, we extend the operating cycle of a station to $M+1$ mechanical treatments with M messages exchanges in-between. It can be depicted for the processing of one part in figure 2.

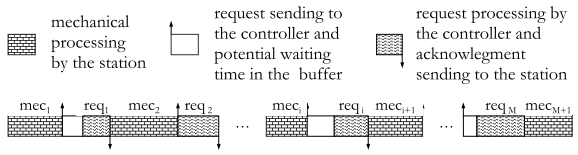

Figure 2: Operating cycle of a station.

The durations of mechanical and message processings are specific to each station. In this paper our approach consists in considering these delays constant. To these delays, we have to add waiting time in the message buffer, which depends on the scheduling policy and therefore vary from a part to another.

2.3 Assembly systems modelling

An assembly system, composed by a set $\mathcal{S} = (S_i)_{1 \leq i \leq N_S}$ of stations and a set $\mathcal{C} = (C_j)_{1 \leq j \leq N_C}$ of conveyors, can be viewed as an acyclic oriented graph, whose nodes symbolize stations and edges, conveyors transporting parts from a station to another one. Thus, an assembly system is characterized by a relation σ from $\mathcal{S}$ in $P(\mathcal{S})$ which links each station with its successors list. The set of successors (resp. predecessors) of a station S_i is denoted $\sigma^+(S_i)$ (resp. $\sigma^-(S_i)$).

We only consider in this study systems with single input and output station. This assumption does not imply any restriction. Indeed, the behaviour of a system with multiple inputs or outputs is not modified by the addition of a head or tail station with processing delays equal to zero.

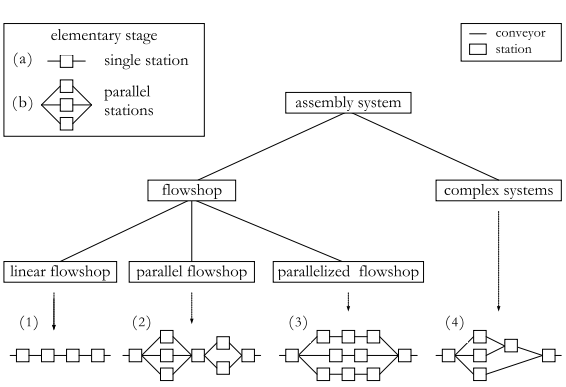

Figure 3: Typology of assembly systems.

We propose a typology of assembly systems, inspired from [9], according to the stages (steps corresponding to identical operations) and the number of stations performing these operations as shown in figure 3.

2.4 Characteristics of a station

For a given assembly system, any station S_s with M_s requests to the controller can be modeled by a 4-tuple, called characteristics of the station,

$$\Big((\theta_{conv_{s,j}}),\ \varpi_s,\ (\theta_{mec_{s,k}}),\ (\theta_{req_{s,l}}) \Big)$$

where

- $(\theta_{conv_{s,j}})$ is a matrix with the conveying delays from the upstream stations of S_s $\quad (S_j \in \sigma^-(S_s))$,
- ϖ_s is the maximum work-in-process of S_s,
- $(\theta_{mec_{s,k}})$ is a matrix with the mechanical delays $(1 \leq k \leq M_s + 1)$,
- $(\theta_{req_{s,l}})$ is a matrix with delays of request processing by the controller $(1 \leq l \leq M_s)$.

In order to simplify the notation, we will omit some subindices in forthcoming equations, where the context allows. Moreover, we denote the sums of these different delays as follows:

$$\theta_{mec_s} = \sum_{k=1}^{M_s+1} \theta_{mec_{s,k}} \qquad \theta_{req_s} = \sum_{l=1}^{M_s} \theta_{reqs,l}$$

$$\theta_{sta_s} = \theta_{mec_s} + \theta_{req_s} \qquad (station\ delay)$$

$$\theta_{ctrl} = \sum_{all\ \ stations\ \ s} \theta_{req_s} \quad (controller\ delay)$$

2.5 Modelling of a station

A station can be modelled using a timed Petri net shown in figure 4, where the timed transitions are depicted with a blank bar. The places STA, CPU, WIP model the availability of their corresponding resource, i.e. the station, the processor(s) and the conveyor(s).

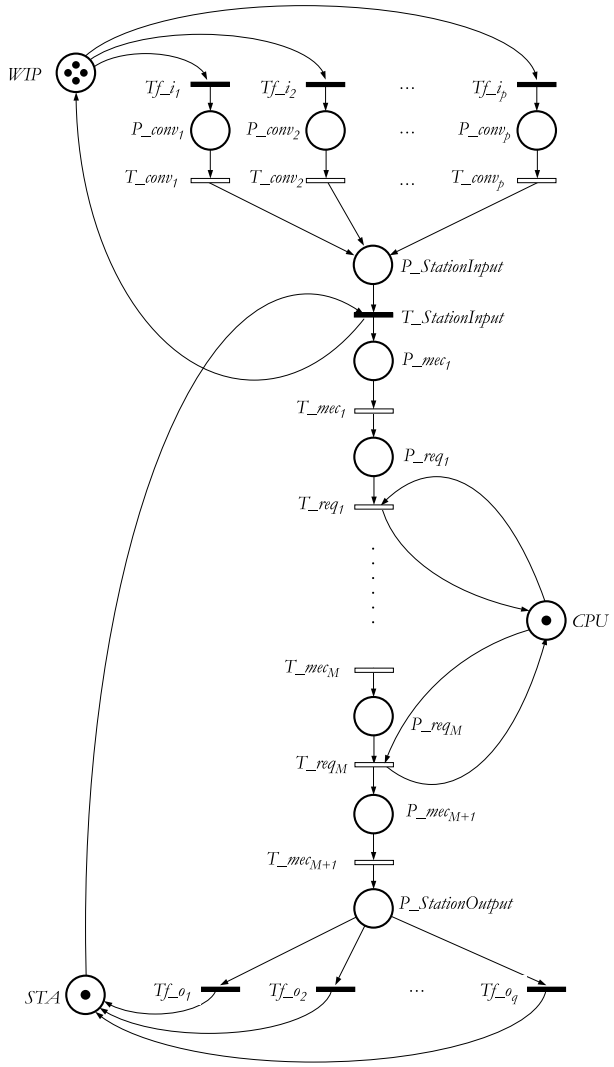

Figure 4: Petri net modelling a station with M requests, p upstream and q downstream stations.

Hence, an assembly system is modelled by connecting the Petri nets corresponding to each station, merging the place CPU and the transitions Tf_i et Tf_o according to its topology.

The aim of the next section is to determine the theoretical optimum production rate of an assembly system according to characteristics of its stations. Then, we can deduce the minimum makespan, which will be compared with simulation results in section 4.

3 Bounds for the throughput

System performance is defined as the maximum rate a system can achieve. It can be expressed as the highest number of parts produced per unit of time or, inversely, by the lowest inter-arrival delay between the production of two successive parts. Other way to measure it is the makespan defined as the duration between the input date of the first part and the output date of the last completed part. The bound study of these quantities is significant, insofar they allow to quantify the quality of the policies we will test. Moreover, the obtained bounds show the importance of some parameters we should not have taken into account without their prior study.

3.1 Bounds of the inter-arrival delay

There is an intuitive relation between the ready-state behaviour of a system and the notion of repeatable firing sequences, and consequently with the T-semiflows. In [1], Campos *et al.* give, for any timed Petri net and for any probability distribution function of firing transition delays, the following lower bound Γ_i for the mean cycle time in steady-state associated with a transition t_i

$$\Gamma_i \geq \max_{Y \in \{P-semiflow\}} Y^T.PRE.D.F_i \quad subject\ to\ Y^T.M_0 = 1 \qquad (1)$$

where

- Y is a P-semiflow (i.e. $Y^T.C = 0$, $Y \geq 0$, $Y \neq 0$, with C the global incidence matrix),
- PRE is the Pre-incidence matrix,
- D is the diagonal matrix with the mean value of the delays assigned to the transitions,
- F_i is a T-semiflow (i.e. $C\ X = 0$, $X \geq 0$, $X \neq 0$) whose component corresponding to t_i is equal to 1,
- M_0 is the initial marking.

In this study, Γ_i corresponds to our minimal inter-arrival delay, where t_i is the last transition in the net modelling the tail station. We have to note that reachability of this bound is not ensured.

Solving the linear programming problem associated with 1, this bound can be computed. Concerning linear flowshop with a single processor, we deduced the following results.

Property 1 (Single station) *A single station (with single input and output, figure 3-(a)) with characteristics:*

$$\left((\theta_{conv}), \varpi, (\theta_{mec_k}), (\theta_{req_l})\right)$$

has the following inter-arrival delay lower bound:

$$\tau_d = max\left\{\frac{\theta_{conv}}{\varpi}\ ,\ \theta_{sta}\right\} \qquad (2)$$

Property 2 (Linear flowshop) *A linear flowshop (figure 3-(1)) compound of N_S stations with respective characteristics:*

$$\left((\theta_{conv_S}), \varpi_s, (\theta_{mec_{S_k}}), (\theta_{req_{S_l}})\right)$$

has the following inter-arrival delay lower bound:

$$\tau_d = \max\left\{\max_{s\in[1,N_S]}\left\{\frac{\theta_{conv_S}}{\varpi_s}\ ,\ \theta_{sta_S}\right\},\ \sum_{s=1}^{N_S}\theta_{req_S}\right\} \qquad (3)$$

3.2 Bounds of makespan

In the case of a linear flowshop, we can deduce from (3) lower bounds of the makespan, to be later compared with the simulation results. Two situations arise according to the value of τ_d:

- $\tau_d = \frac{\theta_{conv_S}}{\varpi_S}$ or θ_{sta_s}, respectively called conveyor and station bound of the system, where s is the bottleneck station or conveyor. The best case occurs when this resource never waits for a part. Hence, the minimal makespan for N parts is

$$\tau_m = \sum_{s=1}^{N_S} (\theta_{sta_s} + \theta_{conv_s}) + (N-1) \times \tau_d \qquad (4)$$

- $\tau_d = \theta_{ctrl}$, called controller bound of the system. The best case occurs when the controller processes continually the received messages, since the initial one for the first part until the final one for the last part. In this case, the minimal makespan for N parts is obtained by adding $N \times b$ to the mechanical delays before and after the first and last messages, which depends on station characteristics.

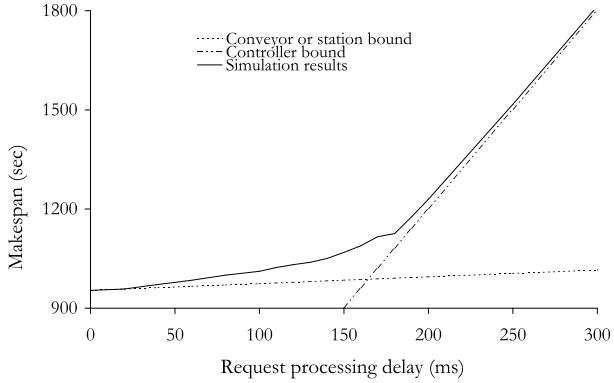

Figure 5: Expected simulation results according to request proccessing delays compared with makespan bounds.

We have to note that these assumptions are only realistic when the controller bound is far enough from the station or conveyor bound. Therefore, we should get simulation results looking like those depicted in figure 5.

To conclude this section, we obtain theoretical optimum for makespan in order to compare them with the simulation results. Moreover the bounds obtained in (3) correspond to either one conveyor or station delay station, or the controller delay. Therefore, we can distinguish three situations and parameters to analyse:

Bottleneck resource	Parameters
Conveyors	Conveying delays and maximum WIP
Controller	Controller delay
Stations	Station delays

4 Flowshop simulation results

The notion of time inherent in the system and the similarity of the station behaviour led us naturally to use timed coloured Petri nets. The tool used to perform simulation is CPN Tools[2] , maintained by the CPN Group, University of Aarhus, Denmark. It allows edition, simulation and analysis of such class of Petri nets [3]. First, we present the characteristics of the modelled system and the various service policies we tested and then we show the first simulation results and conclusions we draw from them.

[2]www.daimi.au.dk

4.1 Considered system characteristics

For a first set of tests, data were extracted from readings of existing configurations by Livbag, corresponding to the five phases depicted in section 2.2. The delays (expressed in ms) are constant for all stations, except for the assembly phase:

phase	delays
conveying	3000
maximum WIP	3
pre-assembly	730
request processing	600
assembly	*see figure 6*
post-assembly	60

Table 1: Station characteristics.

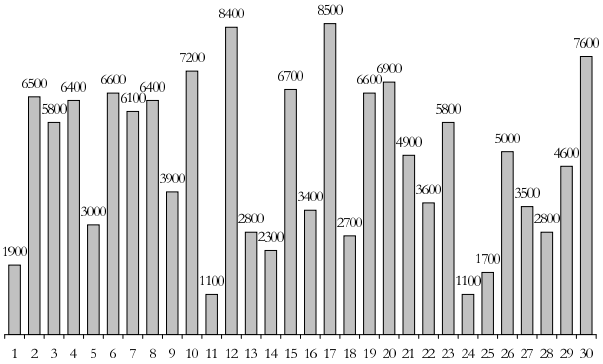

Figure 6: Assembly delays expressed in ms.

Unless explicit mention, below refered simulation results correspond to the production of 1000 parts by lines compound of the n^{th} first stations with these characteristics ($5 \leq n \leq 30$). The quantity measured is the absolute or relative deviation from the observed makespan to its corresponding theoretical lower bound.

4.2 Service policies

The message exchanges are initiated by the stations. The requests are stored in a buffer and the stations remain locked until the response. Hence, the service policy may have a significant impact on the makespan. In [8], only FIFO was taken into account. Here we extend our study to following algorithms:

Acronym	Priority criteria[3]
RAN	random
FIFO	first-in first-out
LIFO	last-in first-out
FSF	fastest station delay first
SSF	slowest station delay first
Push	closest to the head station
Pull	closest to the tail station

Table 2: Tested service policies.

The first results brought us to add another criterion: lowest work-in-process in the next station. These algoritms are denoted by LWxx, where xx is one of the above

[3]In case of equality, the secondary criterion is: closest to the tail station

acronyms. We modeled these policies by means of lists. The existence of various list fonctions in CPN Tools allowed us to specify easily the priority between tokens.

4.3 Considered system simulation results

We perform simulations for the configuration expound in section 4.1 with request processing delays varying from 0 ms to around 2000 ms, with especial attention to values close to the station bound. For 30 stations its value is 160 ms. The figure 7 depicts the results for some service policies with this configuration. With such a graph the quality of the different algorithms can be compared. For example, the assumptions done in section 3.2 are very strong for close station and controller bounds is confirmed.

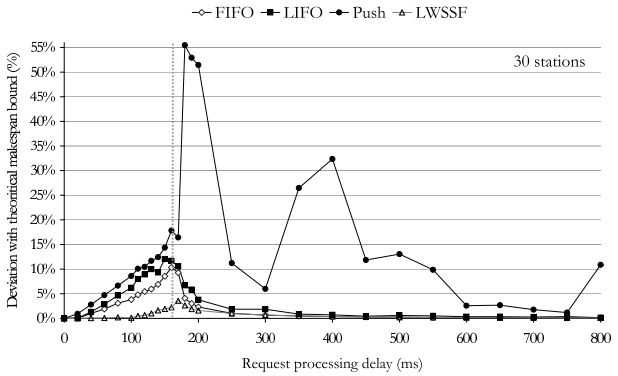

Figure 7: Relative deviation with theoretical makespan bound (30 stations and request processing varying delays).

The table 3 shows the maximum deviation from the theoretical makespan bound with configurations from 5 to 30 stations (expressed in percentage):

From these simulations, some points emerge:

- the service policy have a great impact on the performance. For example, with a configuration of 12 stations with a message process duration of 420 ms, the makespan got with FSF algorithm is 52% higher than the one got with Pull,
- among the policies which do not take into account the work-in-process of the downstream station, a more detailed analysis shows that SSF and Pull are the most performant when the bottleneck is a station. FIFO or Random are better when the controller is overloaded,
- for configurations above 8 stations, algorithms taking into account the work-in-process amount are clearly more performant unlike for lighter configurations. This is probably due to the relative repartition homogeneity of the stations 2 to 8. Indeed, more recent simulations with stations of equal assembly delay show the poor quality of this class of algorithm with such configuration.

However, neither algorithm is really the most performant (even random is far from being the worst). This leads us to conclude that the most appropriate way to get the best policy is simulation, especially for configurations more complex than those we analyse in this paper.

	RAN	FIFO	LIFO	FSF	SSF	Push	Pull
5	11,7	12,0	10,0	11,5	16,9	17,1	12,1
6	11,3	14,0	17,3	40,3	16,0	40,6	13,9
8	10,2	10,1	10,8	34,3	16,1	36,5	13,2
10	13,7	9,4	16,6	48,4	12,9	44,0	11,1
12	13,3	11,8	18,4	60,2	12,3	48,0	7,1
15	11,5	11,5	17,5	51,4	8,7	38,0	5,6
18	10,2	11,1	15,3	42,5	13,5	48,2	5,8
20	10,8	11,0	13,8	40,2	9,0	53,0	5,7
22	10,3	11,8	13,1	39,8	7,5	40,9	6,1
25	9,7	11,2	13,9	37,6	7,8	27,3	6,8
28	10,3	10,8	13,0	36,2	6,9	21,8	5,8
30	8,7	10,3	12,0	32,6	6,8	55,5	6,1
		LW FIFO	LW LIFO	LW FSF	LW SSF	LW Push	LW Pull
5		12,1	20,7	11,5	20,8	18,2	12,0
6		12,8	12,7	11,5	15,5	24,8	12,0
8		20,1	17,4	12,4	21,3	19,3	12,5
10		17,7	6,8	6,9	17,1	20,0	8,4
12		7,3	5,4	10,9	8,8	5,9	5,2
15		6,1	4,1	4,8	7,8	6,0	6,7
18		4,2	3,7	4,6	4,7	3,8	4,1
20		4,8	5,0	4,0	11,2	3,5	4,5
22		3,0	2,8	3,0	6,0	3,2	6,6
25		3,0	4,0	3,8	3,4	3,1	3,7
28		3,3	3,6	4,5	3,6	3,2	5,5
30		3,1	3,4	5,1	3,6	3,2	6,0

Table 3: Simulation results.

4.4 Other studies

We also studied other problems, such as significance of conveying delays or steady state settling.

4.4.1 Conveying delays and maximum work-in-process

The inequation $\tau_d \geq \frac{\theta_{conv_S}}{\varpi_S}$ shows that the conveyors delays cannot be disregarded. Althought the maximum work-in-process only has to be increased to prevent a conveyor to be a bottleneck resource, the significance of conveying delays and work-in-process must be analysed.

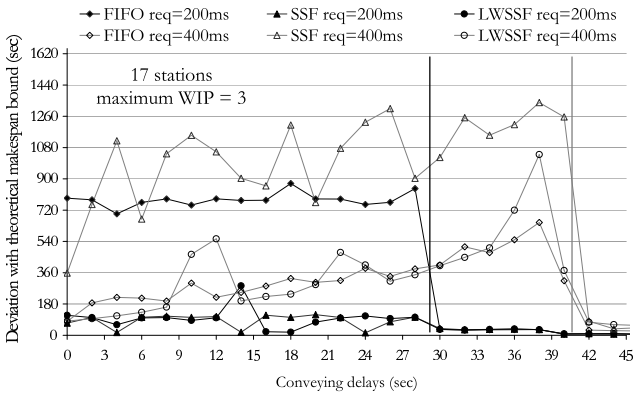

Figure 8: Absolute deviation with theoretical makespan bound (extract from 0 to 45 sec).

The figure 8 exhibits the results of two sets of simulation on 17 stations, 3 as maximum work-in-process and conveying delay varying from 0 to 60 seconds. The first set when the slowest resource is a station (request processing time = 200 ms) and the second when the bottleneck is the controller (request processing time = 400 ms). The respective thresholds for conveying delay to become the penalising

resource are 29.07 and 40.80 seconds and are depicted as vertical lines in the figure.

The results on other configurations are quite similar. Hence, we can deduce that, in case the bottleneck resource is a station, an increase in the conveying delay implies a nearly equal increase of the makespan. On the contrary if the controller is the slowest resource, a worsening of the inter-arrival delay is noticed. On the other hand, simulations showed that increasing the maximum work-in-process does not improve the makespan. Thus, we can sum up our conclusions as follows:

Bottleneck resource	Conclusion
Station	Conveying delays have few significance
Controller	Reduce the conveying delays
Stations	Increase the maximum WIP and act according to the new bottleneck

4.4.2 Steady state settling

We also took an interest in the steady state settling. The detection of some periodicity in the inter-arrival delays appeared us difficult. So we tackled the problem by studying work-in-process total amount. Indeed, in addition to the proper interest of this quantity, its stability seems intuitively a sufficient condition for the steady state settling. We got results we can summarize as in figure 9 for 500 parts production with 30 stations and 800 ms as resquest processing time (controller is the bottleneck resource).

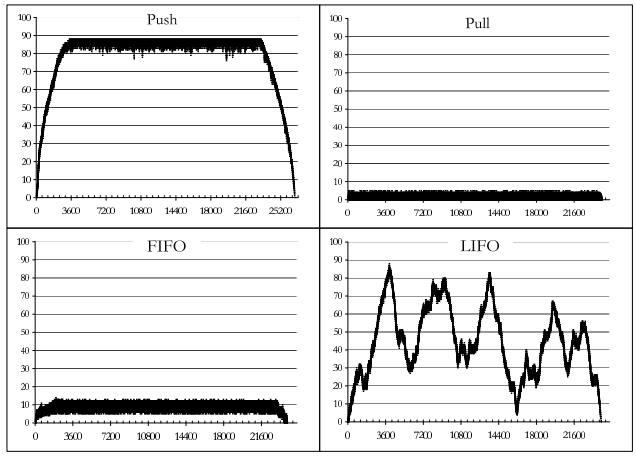

Figure 9: Work-in-process total amount in relation with processing time in seconds.

With this configuration, the Pull and, to a lower extent, the FIFO policy limit the WIP amount whereas the Push one makes it almost maximum. The steady state is established before at least half an hour (resp. an hour) for FIFO (resp. Push) policy.

The case of LIFO is more amazing. Analysing the maximum inter-arrival delays with this policy, we founded values as 11 minutes for request processing delays of 400 ms or 36 minutes for 800 ms. This situation corresponds to the apparent production stoppings mentionned in introduction. Howewer the observed unstability is not translated into a significative productivity loss, scarcely 2 minutes for a 5 hours production.

5 Conclusion and future works

The Petri nets allows an efficient modelling, performance analysis and behaviour comprehension of manufacturing processes. By their solid mathematical basis, theoretical results can be proved. Solving the linear programming problems associated with (1) gives us formal optimum throughput bounds for linear flowshops. Similar but more complicated bounds can be deduced for parallel and parallelized flowshops. The existence of numerous tools (CPN Tools in our case) permits the comparison of simulation results with these optima. Although none of tested service policies did not proved to be the most performant, some trends can be drawn from this study. Behaviour as apparent freezed production have been also explained.

Our future works are based on two distinct angles. On the one hand, we will extend the study to other service policies and more complex systems (figure 3). On the other hand, we will develop a tool which would allow to an user without specific knowledge of Petri nets to perform automatic simulation of his system behaviour.

References

[1] J. Campos, G. Chiola, and M. Silva. Ergodicity and throughput bounds for petri nets with unique consistent firing count vector. *IEEE Transactions on Software Engineering*, 17(2):117–125, 1991.

[2] D.M. Dilts, N.P. Boyd, and H.H. Whorms. The evolution of control architectures for automated manufacturing systems. *Journal of Manufacturing Systems*, 10:79–83, 1991.

[3] Ratzer et al. CPN Tools for Editing, Simulating, and Analysing Coloured Petri Net. In *Proceedings of Applications and Theory of Petri Nets 2003*, pages 450–462. Springer-Verlag LNCS 2679, 2003.

[4] A. Grieco and al. A review of different approaches to the FMS loading process. *The International Journal of Flexible Manufacturing Systems*, 13:361–384, 2001.

[5] Kurt Jensen. An introduction to the theoretical aspects of coloured petri nets. *Lecture Notes in Computer Science; A Decade of Concurrency*, 803:230–272, 1993.

[6] A. Jones, E. Barkmeyer, and W. Davis. Issues in the design and implementation of a system architecture for computer integrated manufacturing. *Journal of Computer Integrated Manufacturing*, 2:65–76, 1989.

[7] Tadao Murata. Petri Nets: Properties, analysis and applications. In *Proceedings of the IEEE,* Vol. 77, num. 4, April 1989.

[8] L. Plassart, P. Le Parc, F. Singhoff, and L. Marcé. Modelling and Simulation of Interactions Between the Local Command Units and the Supervisor of an Automated Assembly Line: a Case Study. *XVI Workshop on Supervising and Diagnostics of Machining Systems*, 1-2 chap. 5:210–221, 2005.

[9] Olivier Telle. *Gestion de chaînes logistiques dans le domaine aéronautique : Aide à la coopération au sein d'une relation Donneur d'Ordres/Fournisseur.* PhD thesis, Ecole nationale supérieure de l'aeronautique et de l'espace, 2003.