Scheduling Policies in Flexible Bernoulli Lines with Dedicated Finite Buffers☆

Kyungsu Park[a], Jingshan Li[b,1], Shaw C. Feng[c]

[a]*Department of Industrial & Management Systems Engineering, Dong-A University, Busan, South Korea*
[b]*Department of Industrial and Systems Engineering, University of Wisconsin, Madison, WI 53706, USA*
[c]*Engineering Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899, USA.*

Abstract

This paper is devoted to studying scheduling policies in flexible serial lines with two Bernoulli machines and dedicated finite buffers. Priority, cyclic and work-in-process (WIP)-based scheduling policies are investigated. For small scale systems, exact solutions are derived using Markov chain models. For larger ones, a flexible line is decomposed into multiple interacting dedicated serial lines, and iteration procedures are introduced to approximate system production rate. Through extensive numerical experiments, it is shown that the approximation methods result in acceptable accuracy in throughput estimation. In addition, system-theoretic properties such as asymptotic behavior, reversibility, and monotonicity, as well as impact of buffer capacities are discussed, and comparisons of the scheduling policies are carried out.

Keywords: Bernoulli reliability machine, flexibility, production rate, scheduling policy, dedicated buffer.

1. Introduction

To respond to rapid market change and customized demands, flexibility is becoming prevalent in modern manufacturing industry. Substantial efforts have been devoted by manufacturers to diversifying products and flexibilizing

☆This work is supported in part by NIST grant 70NANB14H260.
[1]Corresponding author: jingshan.li@wisc.edu

equipment, where multiple types of products are processed in the same production system. For example, vehicles with different styles, engines, colors, interior materials and other options are produced on one general assembly line. Customized computers or notebooks are assembled in the same production unit. Similar observations are found in other manufacturing systems as well.

In many flexible manufacturing systems, dedicated machines and buffers are used for specific type of products to avoid mismatch and disorder. For instance, in fuel injector production lines, components at different fabrication stages are stored in dedicated buffers in front of the central washers, waiting for cleaning. In motorcycle manufacturing, the transmission cases for multiple motor families are routed with separate conveyors specific to each family. In semiconductor manufacturing, multiple dedicated buffers are used to accommodate the diversity in physical configuration limits, temperatures, and avoid chemical contaminations. In many sequence based assembly lines, dedicated buffers could avoid sequence disruption due to scraps of defective parts. Similar examples can be found in many other flexible manufacturing systems.

Clearly, scheduling and control policies play an important role in such systems to ensure the desired productivity and quality. Numerous scheduling algorithms have been proposed and used on the factory floor. Among them, priority, cyclic, and work-in-process (WIP)-based policies are the prevalent ones due to their simplicity in control logic, while many other scheduling policies (e.g., processing time or due-day based and queue length based policies) can be equivalent into these policies. In addition, as one of the most important key performance indicators (KPIs), the production line throughput (or production rate) has been studied for decades (see, for instance, monographs [1]-[5] and reviews [6]-[8]). Similarly, manufacturing flexibility has also been addressed for a long time (e.g., reviews [9]-[14]). However, due to the complexity in flexible systems, analysis of KPIs (such as production rate) under different scheduling policies in flexible manufacturing systems still needs in-depth study, particularly in scenarios with unreliable machines and finite dedicated buffers.

The main contribution of this paper is in developing efficient analytical methods to study the scheduling policies of two-machine flexible lines with unreliable Bernoulli machines and dedicated finite buffers. Three scheduling regimes are studied: priority, cyclic and WIP-based policies. For small scale systems, a Markov chain method to derive exact solutions is presented. For larger ones, an iterative method is introduced based on decomposition of the system into multiple interacting serial lines. Numerical study shows that such a method leads to acceptable accuracy in production rate estimation without computation intensity. Ideas of extending the study to longer lines are explored. In addition, system-theoretic properties, such as monotonicity, reversibility, and asymptotic behaviors, are discussed analytically or based on experimental results. The impact of buffer capacity on line performance is investigated and comparisons between the scheduling policies are carried out.

The remainder of the paper is organized as follows: In Section 2, related literature is briefly reviewed. Section 3 introduces the assumptions for formulates the problem. Sections 4 and 5 present solution methods for smaller scale and larger systems, respectively. Discussions on system properties and buffer impact are provided in Section 6, and conclusions are formulated in Section 7. All proofs are given in the Appendix.

2. Literature Review

During the last three decades, substantial studies on flexible manufacturing system have been conducted. A classical paper [9] reviews several analytical models of flexible manufacturing systems and provides guidance for research directions. In paper [10], more accumulated literature is reviewed by defining various concepts of flexibility in manufacturing, such as machines, processes, operations, products, routings, expansions and market flexibility. Monographs [2] and [11] investigate stochastic flexible manufacturing systems, while [1] and [12] analyze the systems from a deterministic perspective. The issues of performance analysis, optimal system design and production control, etc., are addressed. In reviews [8], [13] and [14], the concept and problems related to flexibility are discussed.

Since multiple types of products are processed on the same line in many flexible manufacturing systems, scheduling and control play an important role. For production lines with unreliable machines, references [15] and [16] apply a decomposition method to analyze the systems with a static priority rule to select the part type for production. The multi-product kanban like control systems are analyzed in [17], and the production capacity of flexible manufacturing systems with fixed production ratios is studied in [18]. Similarly, papers [19] and [20] present an analytical method with a general probabilistic constraint by decomposing the lines and aggregating states of machines, which are also used to model the priority rule. However, such models could not preserve the desired product composition (i.e., product mix ratio) in the system. More recently, paper [21] introduces the definition, problem and performance portrait of multi-job serial lines.

For cyclic rule, papers [22] and [23] address the performance of multi-product kanban systems with sequence-independent setup times using a decomposition method. A two-product polling model is introduced in [24] under different kinds of cyclic policy via both exact and decomposition methods. The studies in [25] and [26] extend the model from cyclic rule and compare the system performance under different scheduling policies. They also investigate the robustness of the policies and provide practical guidance for operation management. Paper [27] further extends the work to machines with arbitrary processing times.

In addition, for systems with constant work-in-process (CONWIP), paper [28] studies kanban assignment to multiple product types. A parametric decomposition method is provided in [29] for performance evaluation in closed queueing networks. Moreover, reference [30] presents an analysis of line production rate and average inventory level for each part type based on priority policy. Paper [31] considers a flexible manufacturing system consisting of common lines and dedicated branches to process different product types through addressing the split and merge behaviors. More recent works on multi-product lines appear in [32]-[34], where serial lines with shared (or non-dedicated) buffers are studied. Such works are extended to lines with setups and assembly systems in [35] and [36], respectively. Optimal production control has been investigated in [37] and [38] for partially flexible systems, where dedicated downstream lines are supplied by a flexible upstream line with batch operation and setups, using Bernoulli and geometric models, respectively.

In spite of these efforts, there is no available work to analyze different scheduling policies in flexible production lines with unreliable machines and dedicated finite buffers, investigate system properties and compare line performance. This paper intends to contribute to this end.

3. Assumptions and Problem Formulation

Consider a flexible two-machine line with finite dedicated buffers (see Figure 1, where the circles represent the machines and the rectangles are the buffers). The following assumptions define the product arrival, the machines, the buffers, their interactions and scheduling policies.

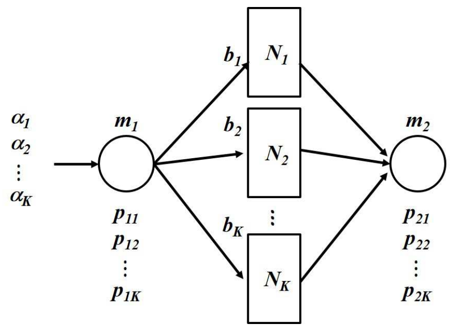

Figure 1: Two-machine production line with K product types and dedicated buffers

1) The production line can produce K types of products, denoted as types $1, 2, \cdots, K$.

2) The production line consists of two machines, m_1 and m_2, and K buffers, b_1 to b_K, between the machines, each dedicated to one product type.

3) The arriving parts enter the system in a first come first serve (FCFS) manner, waiting to be processed by m_1. They follow a discrete distribution with probability α_j for product type j, $j = 1, \ldots, K$. In addition, $\sum_{j=1}^{K} \alpha_j = 1$.

   Remark 1. Assumption 3) implies that if the next part to be processed by machine m_1 is type j, but m_1 fails to process it, then m_1 cannot process another part type i, $i \neq j$. Similar assumption for machine m_2 is introduced. □

4) Both machines m_1 and m_2 have a constant and identical cycle time. The time axis is slotted with the duration of cycles.

5) The machines follow Bernoulli reliability model independently. In each cycle, machine m_i, $i = 1, 2$, is up with probability p_{ij} for product type j, $j = 1, \ldots, K$, and down with probability $1 - p_{ij}$.

6) Each buffer b_j, $j = 1, \ldots, K$, has a finite capacity, $0 < N_j < \infty$.

Remark 2. Assumptions 4)-6) introduce a Bernoulli reliability model of the line. Bernoulli models have been widely used in manufacturing systems studies (see monograph [5]). Such models are suitable for assembly type of machines whose average downtime is comparable to its cycle time. Bernoulli models have been successfully applied in automotive and many other industries (see case studies in and representative papers [34]-[37], [39]-[47]). In case of machines having different cycle times, a transformation can be introduced to make an equivalence of the original system into a Bernoulli line. Specifically, define $T_{up,i}$ and $T_{down,i}$ as the average up- and downtimes of machine m_i, respectively. Let c_i be the capacity or speed of machine m_i, and $c_{max} = \max_i c_i$. Then the Bernoulli machine parameter p_i can be calculated as

$$p_i = \frac{c_i}{c_{max}} \cdot \frac{T_{up,i}}{T_{up,i} + T_{down,i}}, \quad i = 1, 2.$$

In other words, the constant cycle time is defined by the shortest processing time ($1/c_{max}$). Parameter p_i represents the percentage or proportion of work m_i can finish within this cycle time. It can also be viewed as the probability or efficiency to produce a part during the cycle time. Since Bernoulli model is relatively easy to study (but still preserves the nature of the system), we start with Bernoulli model and plan to extend to other reliability models (such as geometric, exponential or general) in future work. □

7) The processing of parts at machine m_2 is determined by the following scheduling policies:

- Priority policy: The priority order is static, being a function of product type. For simplicity, we assume the part type with a smaller number has a higher priority to be processed by machine m_2. That being said, when m_2 is ready, type 1 is always selected first, and type j, $2 \leq j \leq K$, is selected only when all buffers with smaller numbers, i.e., b_1 to b_{j-1}, are empty.
- WIP-based policy: The product type that has the highest occupancy in its buffer will be selected first by machine m_2 when it is up for this type. If there are more than one product types satisfying the condition, either is selected equiprobably.

- Cyclic policy: Machine m_2 will select the product following the order of types $1, 2, \cdots, K$, and then back to type 1. A product type will be skipped in a cycle if its buffer is empty. If m_2 is down for type j in a cycle, then next cycle type $j+1$ will be selected.

Remark 3. In practice, there are many scheduling policies are used, some of which can be equivalent to the ones discussed here. For example, the policies based on due date or processing times can be characterized by priority policy, such as the part type with the longest processing time or earliest due date has the highest priority. The WIP-based policy has similar features to dynamic policies related to queue length, such as longest/shortest queue, largest/smallest available buffer space. For other policies not represented here, they will be investigated in future work. □

8) The machine status and the buffer status are updated at the beginning and the end of the time slot, respectively.

9) Machine m_1 is never starved, but can be blocked for product type j if it is up for type j, buffer b_j is full, and machine m_2 does not take a part from b_j. Machine m_2 is never blocked, but it is starved if it is up and all buffers are empty.

The above assumptions define the system under consideration. To study its performance, define PR_j, $j = 1, \ldots, K$, as the line production rate of type j parts, i.e., the probability to produce a type j part by m_2 during a cycle. Then the problem to be addressed is formulated as follows: *Given production system 1)-9), develop a method for evaluating the line production rate as a function of machine and buffer parameters and scheduling policies, and investigate system-theoretic properties.*

The solutions to the above problem are given in Sections 4 and 5 below. First, an exact method using Markov chain models is developed for small scale systems. Then an approximation method based on decomposition and iteration is introduced for larger ones.

4. Markov chain Method for Small Systems

In this section, we derive exact equations for performance analysis using Markov chain models. Such a method is suitable for small scale systems, i.e., lines with small buffer capacities and a limited number of product types. First, we define the state space and transition probabilities.

4.1. State Space and Transition Probability

4.1.1. Priority and WIP-based policies

The state definition of the systems can be the same under these two policies. Let h_j denote the occupancy of product type j in buffer b_j, $0 \leq h_j \leq N_j$, $j = 1, \ldots, K$, and u be the product type to be processed in machine m_1, $1 \leq u \leq K$. Then the system state can be characterized by $S = (h_1, \ldots, h_K, u)$. The total number of states in the system, M, can be calculated as

$$M = K \cdot \prod_{i=1}^{K} (N_i + 1).$$

By considering the transitions between two effective states, s_1 and s_2, the state transition probabilities P_{s_1,s_2} can be obtained. Detailed derivation process is illustrated in Appendix A.

4.1.2. Cyclic policy

In this policy, the product type to be processed at machine m_2 should be included in state definition, i.e., the state of the system is characterized by $S = (h_1, \ldots, h_K, u, v)$. Note that $h_v = 0$ and $v > 0$ cannot appear at the same time in a state, since a type v product cannot be processed at machine m_2 due to the empty buffer. When all buffers are empty, we use $v = 0$. In order to simplify the notation, assume that $j + K := j$ so that $(h_1, \ldots, h_K, u, v) := (h_1, \ldots, h_K, u, v + K)$ when $v \neq 0$. Then, the number of effective states for cyclic policy, M, can be calculated as

$$\begin{aligned} M &= \prod_{i=1}^{K} (N_i + 1) \cdot K \cdot K - \sum_{i=1}^{K} \prod_{\substack{j=1 \\ j \neq i}}^{K} (N_j + 1) \cdot K + K \\ &= K \cdot \left\{ \prod_{i=1}^{K} (N_i + 1) \cdot \left[K - \sum_{j=1}^{K} \frac{1}{N_j + 1} \right] + 1 \right\}, \end{aligned}$$

where $\sum_{i=1}^{K} \prod_{\substack{j=1 \\ j \neq i}}^{K} (N_j + 1) K$ represents the ineffective cases $h_v = 0$ but $v > 0$, and the last term K characterizes the $v = 0$ scenario.

Again, by enumerating effective states, P_{s_1,s_2}, the transition probabilities between states s_1 and s_2 can be derived. The details are presented in Appendix A.

4.2. Performance Analysis

Based on the transitions derived in above subsection, the transition matrix P with dimension $M \times M$ can be constructed. Let ψ_i be the steady-state probability associated with state s_i, where $s_i = (h_1, \ldots, h_K, u)$ for priority and WIP-based policy, and $s_i = (h_1, \ldots, h_K, u, v)$ for cyclic policy.

$$\Psi = [\psi_1; \psi_2; \ldots; \psi_M].$$

Then the balance equations in a Markov chain can be obtained

$$\begin{aligned} P \cdot \Psi &= \Psi, \\ E \cdot \Psi &= 1, \end{aligned}$$

where $E = [1, 1, \ldots, 1]$, and the second equation is the normalization condition. Introduce Φ which is obtained by replacing the last row in P by E. Then Ψ can be solved by

$$\Psi = \Phi^{-1} \cdot \Psi. \tag{1}$$

Note that there exists a unique steady state solution since we consider an irreducible Markov chain with finite number of states. In addition, define $X_{j,\eta}$, $j = 1, \ldots, K$, $\eta = 0, 1, \ldots, N_j$, as the probability that buffer b_j has η parts. Then

$$X_{j,\eta} = \sum_{s_i \in \cup_l \{s_l | h_j = \eta\}} \psi_i.$$

Thus, the system performance can be derived from (1).

Theorem 1. *Under the assumption 1)-9), the system production rate of each product type can be calculated as:*

$$PR_k = \sum_{s_i \in \mathcal{V}} p_{2k} \psi_i \cdot \frac{1}{n}, \quad k = 1, \ldots, K, \tag{2}$$

where $n = 1$ for priority and cyclic policies, and n represents the number of buffers having the same highest occupancy simultaneously in WIP-based policy. In addition,

$$\mathcal{V} = \begin{cases} \cup_l \{s_l | h_k > 0, h_j = 0, \forall j < k\}, & \textit{for priority policy;} \\ \cup_l \{s_l | h_k \geq h_j, \forall j \neq k\}, & \textit{for WIP-based policy;} \\ \cup_l \{s_l = (h_1, \ldots, h_k, \ldots, h_K, u, k) | h_k > 0\}, & \textit{for cyclic policy.} \end{cases}$$

Proof: See Appendix B. ■

Other performance measures, such as WIP, probabilities of blockage and starvation, can be derived using ψ_i's as well. In limited scenarios, Theorem 1 can be represented by closed equations.

Corollary 1. *Under assumptions 1)-9) with* $K = 2$*,* $\alpha_1 = \alpha_2 = 0.5, N_1 = N_2 = 1, p_{11} = p_{12} := p_1$ *and* $p_{21} = p_{22} := p_2$*, the system performance under priority policy can be calculated as follows:*

$$PR_1 = PR_2 = \frac{p_1 p_2 [p_1(2 - 3p_2) + 2p_2]}{4p_1^2(p_2 - 1)^2 + 2p_1 p_2(3 - 4p_2) + 4p_2^2}. \quad (3)$$

Proof: See Appendix B. ■

4.3. Computation Efficiency

Theorem 1 provides an exact method to evaluate system performance. Clearly such a method is only suitable when the system is not too large. The computation efficiency of the method is illustrated in Tables 1-3, using parameters $p_{ij} = 0.9$, $\alpha_j = 1/K$ and $N_j = N$, $i = 1, 2$ and $j = 1, \ldots, K$. The results are obtained via MATLAB on AMD Opteron(tm) 6176 SE 2.30GHz, 32GB RAM and 64-bit system. The dash "-" in the tables represents that it is failed to compute within a reasonable time period. The colored cells indicate that longer than 1,000 seconds are needed to solve the case or it cannot be solved within a reasonable time. Therefore, for such large scale cases, a computation efficient method is needed, which will be discussed next.

5. Decomposition Method for Larger Systems

When the number of product types and capacity of the buffers increase, the number of states and thus the complexity increase dramatically. Therefore, the computation intensity limits the applicability of the Markov chain method, and an approximation method needs to be introduced.

The idea of the approximation method is explained as follows: From the point of view of each in-process buffer, there are an upstream operation and a downstream one. The machines are "up" when they process this type product, and are "down" when they are in failure mode or processing

Table 1: Computation time for priority policy (unit: second)

		N									
		1	2	3	4	5	6	7	8	9	10
K	2	0	0	0	0	0	0	0	0	0	0
	3	0	0	0	1	2	6	13	29	59	113
	4	0	1	6	43	226	1142	9507	18446	-	-
	5	1	9	225	3471	-	-	-	-	-	-
	6	1	164	2612	-	-	-	-	-	-	-
	7	5	3935	-	-	-	-	-	-	-	-
	8	35	-	-	-	-	-	-	-	-	-
	9	192	-	-	-	-	-	-	-	-	-
	10	1531	-	-	-	-	-	-	-	-	-

Table 2: Computation time for WIP-based policy (unit: second)

		N									
		1	2	3	4	5	6	7	8	9	10
K	2	0	0	0	0	0	0	0	0	0	0
	3	0	0	0	1	3	7	17	37	71	134
	4	0	1	11	70	323	1514	9256	28015	-	-
	5	1	21	393	6509	-	-	-	-	-	-
	6	3	376	32552	-	-	-	-	-	-	-
	7	20	7005	-	-	-	-	-	-	-	-
	8	142	-	-	-	-	-	-	-	-	-
	9	852	-	-	-	-	-	-	-	-	-
	10	5570	-	-	-	-	-	-	-	-	-

Table 3: Computation time for cyclic policy (unit: second)

		N									
		1	2	3	4	5	6	7	8	9	10
K	2	0	0	0	0	0	0	0	0	1	1
	3	0	0	1	5	17	49	125	295	645	1547
	4	0	4	73	677	8301	42178	-	-	-	-
	5	2	146	10819	-	-	-	-	-	-	-
	6	9	8205	-	-	-	-	-	-	-	-
	7	81	-	-	-	-	-	-	-	-	-
	8	962	-	-	-	-	-	-	-	-	-
	9	11190	-	-	-	-	-	-	-	-	-
	10	-	-	-	-	-	-	-	-	-	-

other product types. Thus, it can be viewed that there exists an equivalent serial line for each type of products. As shown in Figure 2, the system is decomposed into K dedicated serial lines. Each line has two machines, which have probabilities p'_{ij} to be up for type j product, and $1-p'_{ij}$ to be down, $i=1,2$ and $j=1,\ldots,K$. The capacity of the buffer is still N_j.

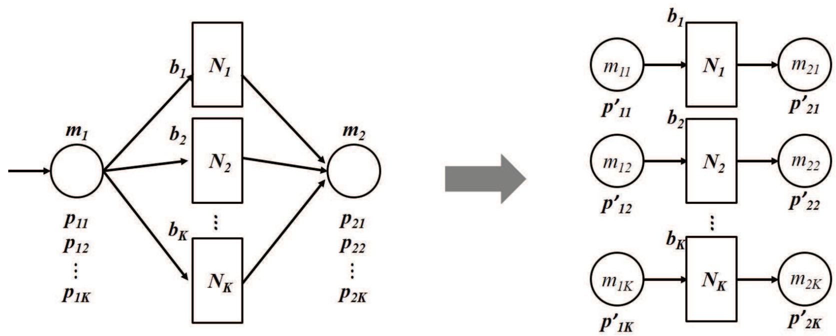

Figure 2: Decomposition of the systems

For each serial line, the performance can be evaluated using the results from Bernoulli two-machine lines with single product type (see [5]). Specifically, for the serial line with part type j, the line production rate, PR_j, can be calculated as:

$$PR_j = p'_{1j}[1 - Q(p'_{2j}, p'_{1j}, N_j)] = p'_{2j}[1 - Q(p'_{1j}, p'_{2j}, N_j)], \tag{4}$$

where

$$Q(p'_{1j}, p'_{2j}, N_j) = \begin{cases} \frac{(1-p'_{1j})(1-\phi)}{1-\frac{p'_{1j}}{p'_{2j}}\phi^{N_j}}, & \text{if } p'_{1j} \neq p'_{2j}, \\ \frac{1-p'}{N_j+1-p'}, & \text{if } p'_{1j} = p'_{2j} = p', \end{cases} \tag{5}$$

$$\phi = \frac{p'_{1j}(1-p'_{2j})}{p'_{2j}(1-p'_{1j})}. \tag{6}$$

Then the overall system performance is defined as

$$PR := \sum_{j=1}^{K} PR_j. \tag{7}$$

However, the above evaluation is dependent on acquiring parameters p'_{ij}, $i=1,2$, $j=1,\ldots,K$, of the decomposed machines. To obtain them, decompositions of machines m_1 and m_2 are described next.

5.1. Decomposition of Machine m_1

First, the decomposition of machine m_1 is addressed. Let α'_j denote the probability that the first part to be processed by m_1 in the original system is type j, either being a newly arrived one or the one waiting for m_1's repair from breakdown. By conditioning α'_j on whether type j part is the first part during the previous cycle or not, and whether buffer b_j is full or not, and also assuming independence between buffer status and machine status, we can approximate α'_j as follows:

$$\alpha'_j = \alpha'_j X_{j,N_j}[1 - p'_{2j} + p'_{2j}(1 - p_{1j} + p_{1j}\alpha_j)] + \alpha'_j(1 - X_{j,N_j})[1 - p_{1j} + p_{1j}\alpha_j] \\ + \sum_{k=1,k\neq j}^{K} [\alpha'_k X_{k,N_k} p'_{2k} p_{1k}\alpha_j + \alpha'_k(1 - X_{k,N_k})p_{1k}\alpha_j], \tag{8}$$

where p'_{2j} represents the probability that machine m_2 processes type j part given that buffer b_j is not empty, and X_{j,N_j} is the probability that buffer b_j is full. Note that the first line in (8) conditions b_j being full in previous cycle, either m_2 is not processing, or m_1 is not processing, or m_1 is keeping processing type j part. The second line conditions the scenario that b_j is not full, but m_1 is either failed or keeping processing type j part. The last line conditions type k (other than j) part being processed in previous cycle.

Solving equation (8) we obtain

$$\alpha'_j = \frac{\alpha_j\Big(\sum_{k=1}^{K} \alpha'_k p_{1k}[1 - X_{k,N_k}(1 - p'_{2k})]\Big)}{p_{1j}[1 - X_{j,N_j}(1 - p'_{2j})]}. \tag{9}$$

As $\sum_{j=1}^{K} \alpha'_j = 1$, substituting α'_j from (9) into this summation implies that

$$\sum_{k=1}^{K} \alpha'_k p_{1k}[1 - X_{k,N_k}(1 - p'_{2k})] = \frac{1}{\sum_{i=1}^{K} \frac{\alpha_i}{p_{1i}[1 - X_{i,N_i}(1 - p'_{2i})]}}.$$

Substituting it back to (9), we have

$$\alpha'_j = \frac{\alpha_j}{p_{1j}\left[1 - X_{j,N_j}(1 - p'_{2j})\right] \sum_{k=1}^{K} \frac{\alpha_k}{p_{1k}\left[1 - X_{k,N_k}(1 - p'_{2k})\right]}}.$$

Then the parameter of machine m_1 in each decomposed line can be calculated as

$$p'_{1j} = \alpha'_j p_{1j}, \quad j = 1, \ldots, K. \tag{10}$$

Since X_{j,N_j} is unknown, we approximate it using the probability buffer b_j is full in the decomposed line, X'_{j,N_j}, which can be calculated as

$$X'_{j,N_j} = \begin{cases} \frac{(1-p'_{1j})(1-\phi)\phi^{N_j}}{(1-p'_{2j})\left(1-\frac{p'_{1j}}{p'_{2j}}\phi^{N_j}\right)}, & \text{if } p'_{1j} \neq p'_{2j}, \\ \frac{1}{(N_j+1-p')}, & \text{if } p'_{1j} = p'_{2j} = p'. \end{cases} \tag{11}$$

Then α'_j can be calculated as

$$\alpha'_j = \frac{\alpha_j}{p_{1j}\left[1 - X'_{j,N_j}(1-p'_{2j})\right] \sum_{k=1}^{K} \frac{\alpha_k}{p_{1k}\left[1-X'_{k,N_k}(1-p'_{2k})\right]}}, \tag{12}$$

However, still the value of α'_j is unable to be calculated directly, since it depends on p'_{2j} and X'_{j,N_j} (which also depends on p'_{1j} and p'_{2j}).

5.2. Decomposition of Machine m_2

5.2.1. Priority policy

Under this policy, machine m_2 selects a product type to process based on a pre-determined static priority. Such a policy can address several prevalent policies on the factory floor, such as longest processing time, shortest processing time, highest cost, and closed due date. In this paper, it is assumed that a product type with smaller number has a higher priority. Then, product type j will be processed at machine m_2 when buffer b_j is not empty, machine m_2 is up for type j, and the higher priority buffers, b_k, $k = 1, \ldots, j-1$, are all empty.

Thus, by taking into account of such a policy, p'_{2j}, which is used to define machine m_2 in decomposed lines for part type j, can be approximated as:

$$\begin{aligned} &p'_{21} = p_{21}, \text{ and} \\ &p'_{2j} = p_{2j} \prod_{k=1}^{j-1} X'_{k,0} \quad \text{for } j = 2, \ldots, K, \end{aligned} \tag{13}$$

where $X'_{k,0}$, $k = 1, \ldots, K$, is the probability that buffer b_k in line k is empty,

$$X'_{k,0} = Q(p'_{1k}, p'_{2k}, N_k). \tag{14}$$

As one can see from expressions (10) and (13), p'_{ij}, $i = 1, 2$, cannot be solved directly since it relies on X'_{k,N_k} and $X'_{k,0}$, which are dependent on

p'_{ij}. In order to solve p'_{ij}, an iteration algorithm is introduced. When the algorithm converges, the machine efficiencies for the decomposed production line can be obtained.

Let $p'_{ij}(n)$ represent the value of p'_{ij} at the n-th iteration. Then the following procedure is introduced:

Procedure 1. *Set* $\epsilon = 0.001$ *as convergence criterion.*

```
p'_1j(0) = α_j p_ij, p'_2j(0) = p_2j, j = 1,...,K.
While |p'_ij(n) − p'_ij(n − 1)| > ε, i = 1,2
   From (12): Calculate α'_j(n) using p'_ij(n − 1).
   From (10): Update p'_1j(n) using α'_j(n).
   From (14): Calculate X'_k,0(n) using p'_1j(n) and p'_2j(n − 1).
   From (13): Update p'_2j(n) using X'_k,0(n).
end
```

The convergence of Procedure 1 has been investigated numerically. A total of 1000 experiments have been carried out for each set of K and N with randomly and equiprobably selected parameters from the following sets:

$$\begin{aligned} & p_{ij} \in (0.7, 0.99), \quad i = 1, 2, j = 1, \ldots, K, \\ & N_j \in \{LBN_K, \ldots, 10\}, \quad j = 1, \ldots, K, \\ & \alpha_j \in (0.1, 1), \quad j = 1, \ldots, K, \quad s.t. \sum_{j=1}^{K} \alpha_j = 1, \end{aligned} \tag{15}$$

where LBN_K is the lower bound of buffer capacity, which is determined based on computation performance in Tables 1-3. In other words, LBN_K is the smallest buffer capacity that cannot be solved using exact Markov chain method for each row of K in Tables 1-3 (i.e., the smallest buffer capacity corresponding to the color portion in each row).

Among all the experiments, over 99.5% of cases the procedure converges, usually within 10 iterations. This leads to computation time within a fraction of second. Upon convergence, we obtain

$$p'_{ij} = \lim_{n \to \infty} p'_{ij}(n), \quad i = 1, 2, \quad j = 1, \ldots, K.$$

Remark 4. For the few (0.5%) non-convergent cases, through extensive numerical experiments, we observe that such scenarios typically occur with oscillating production rate in a very small range during iterations. After the production rate starts to oscillate, by selecting the average in last two iterations as an approximate, the accuracy is always within 5%. Thus, such an approximation can be used as production rate estimate in such cases. □

5.2.2. WIP-based policy

By checking the buffer occupancy, a product with the highest one will be processed by machine m_2 at each time slot. However, it is impossible to enumerate the occupancy at each time slot in steady state analysis. Thus, the probabilities of buffer occupancy in each decomposed line will be used for approximation. In addition, the independence of the buffers is assumed. Then, we approximate p'_{2j} as follows:

$$\begin{aligned} p'_{2j} &= \text{Prob}(h_j > h_k, \forall k \neq j) \cdot p_{2j} \\ &= \sum_{i=1}^{N_j} \left[\text{Prob}(h_j = i) \cdot \prod_{k=1, k\neq j}^{K} \text{Prob}(h_k < i) \right] \cdot p_{2j}, \end{aligned} \tag{16}$$

where $\text{Prob}(h_j = i)$ is denoted as $X'_{j,i}$, and $\text{Prob}(h_k < i)$ can be evaluated as

$$\text{Prob}(h_k < i) = \sum_{l=0}^{i-1} X'_{k,l}, \tag{17}$$

$$X'_{k,l} = \begin{cases} \frac{(1-p'_{1k})(1-\phi)\phi^l}{(1-p'_{2k})\left(1-\frac{p'_{1k}}{p'_{2k}}\phi^{N_j}\right)}, & \text{if } p'_{1k} \neq p'_{2k}, \\ \frac{1}{N_k+1-p'}, & \text{if } p'_{1k} = p'_{2k} = p'. \end{cases} \tag{18}$$

Note here to simplify the approximation formula, we ignore the scenario that multiple buffers have the same occupancy.

As one can see, parameter p'_{2j} is dependent on the probability of buffer occupancy in other lines, i.e., relying on p'_{ik}'s, $i = 1, 2$, $k \neq j$. Thus, a procedure similar to Procedure 1 is introduced to estimate these parameters. Specifically, updating $\alpha'_j(n)$ and $p'_{1j}(n)$ are still the same as in Procedure 1, but (18) is used to calculate $X'_{k,l}(n)$ and (16) is used to update $p'_{2j}(n)$. Within thousands of experiments, all cases are convergent with less than a second computation effort.

5.2.3. Cyclic policy

As machine m_2 processes parts in the order of part types 1 to K, and then back to 1, as long as the buffer is not empty, we can approximate the probability of part type j being selected when buffer b_j is not empty (denoted as π_j) by ignoring the cycles that buffer b_k ($1 < k < j$) is not empty but machine m_2 is down for type k. Thus

$$\begin{aligned}
\pi_2 &= (1 - X_{2,0})\pi_1, \\
\pi_j &= (1 - X_{j,0})\Big[\prod_{k=2}^{j-1} X_{k,0}\pi_1 + \prod_{k=3}^{j-1} X_{k,0}\pi_2 + \ldots + X_{j-2,0}\pi_{j-2} + \pi_{j-1}\Big] \\
&= (1 - X_{j,0})\pi_1\Bigg\{\sum_{i=1}^{j-2}\Big[\pi_i \prod_{k=i+1}^{j-1} X_{k,0}\Big] + \pi_{j-1}\Bigg\}, \quad \text{for } j = 3, \ldots, K.
\end{aligned}$$

Using $\sum_{k=1}^{K} \pi_k = 1$, we can solve π_1 as

$$\pi_1 = \frac{1}{K - \sum_{k=2}^{K} X_{k,0}}. \tag{19}$$

Using $X'_{k,0}$ to replace $X_{k,0}$, parameter p'_{2j} can be approximated by:

$$p'_{2j} = \frac{p_{2j}}{K - \sum_{k=1,k\neq j}^{K} X'_{k,0}}. \tag{20}$$

Again, p'_{2j} depends on $X'_{k,0}$, which again relies on p'_{2j}. Thus another similar procedure is introduced to calculate p'_{2j}. Comparing with Procedure 1, calculation of $\alpha'_j(n)$, $p'_{1j}(n)$ and $X'_{k,0}(n)$ are still the same, but $p'_{2j}(n)$ is updated using (20). Again all cases converge within a fraction of second.

5.3. Accuracy of Decomposition Method

To evaluate the accuracy of the decomposition method, simulations are carried out. In each simulation experiment, 2,000 time slots are used for warm-up and the subsequent 10,000 time slots for data collection. For each set of K and N, 1,000 experiments are conducted by randomly and equiprobably selecting parameters from sets (15).

Let PR_j^{cal} and PR_j^{sim} denote the production rates obtained by decomposition method and simulation, respectively. Then, the accuracy of the decomposition methods is measured by

$$\text{Percentage error: } \delta_{PR_j} = \frac{PR_j^{cal} - PR_j^{sim}}{PR_j^{sim}} \times 100\%,$$
$$\text{Absolute error: } \Delta_{PR_j} = PR_j^{cal} - PR_j^{sim}. \tag{21}$$

By summarizing production rates for each part type, we obtain the overall production rate, i.e., $PR = \sum_{j=1}^{K} PR_j$. Then the accuracy of PR can be evaluated similarly.

The results are illustrated in Tables 4-6, where the rows of PR_1 represent the accuracy for part type 1, and the rest rows are for overall production rate. Results for other product types are similar to that of type 1.

Table 4: Average accuracy for priority policy

	$K=4, N_j \in [6,10]$		$K=5, N_j \in [4,10]$		$K=6, N_j \in [3,10]$		$k=7, N_j \in [2,10]$		$K=8, N_j \in [2,10]$		$K=9, N_j \in [2,10]$		$K=10, N_j \in [2,10]$	
	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$
PR_1	0.69	0.001	0.72	0.001	0.91	0.001	0.95	0.001	0.92	0.001	0.97	0.001	1.34	0.001
PR	0.40	0.003	0.39	0.003	0.52	0.004	0.55	0.004	0.51	0.004	0.49	0.004	0.96	0.008

Table 5: Average accuracy for WIP-based policy

	$K=4, N_j \in [6,10]$		$K=5, N_j \in [4,10]$		$K=6, N_j \in [3,10]$		$K=7, N_j \in [2,10]$		$K=8, N_j \in [2,10]$		$K=9, N_j \in [2,10]$		$K=10, N_j \in [1,10]$	
	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$
PR_1	3.09	0.006	2.88	0.005	2.65	0.004	2.36	0.003	2.24	0.002	2.33	0.002	2.33	0.002
PR	2.92	0.023	2.68	0.021	2.42	0.019	2.16	0.017	1.97	0.016	2.17	0.017	2.13	0.017

Table 6: Average accuracy for cyclic policy

	$K=4, N_j \in [5,10]$		$K=5, N_j \in [3,10]$		$K=6, N_j \in [2,10]$		$K=7, N_j \in [2,10]$		$K=8, N_j \in [2,10]$		$K=9, N_j \in [1,10]$		$K=10, N_j \in [1,10]$	
	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$	$\lvert\delta_{PR}\rvert(\%)$	$\lvert\Delta_{PR}\rvert$
PR_1	4.73	0.010	4.81	0.008	5.62	0.008	5.07	0.006	4.73	0.005	5.93	0.005	5.37	0.004
PR	4.73	0.038	4.82	0.039	5.58	0.045	5.07	0.041	4.77	0.039	5.88	0.047	5.47	0.044

Remark 5. For a few non-convergent cases under priority policy, by replacing α_j in (10) to calculate p'_{1j}, and using (13) to evaluate p'_{2j}, without iterations (i.e., stop after one iteration), we obtain a resulting overall production rate having 2.0% average and 10% maximum percent errors. □

Examining the accuracy results, we observe that:

- For priority policy, the average percent errors are less than 1.5% for all scenarios. The scenarios of larger errors are mainly due to larger K, which makes the values of individual production rate very small, so that even a very small absolute error can lead to a big percentage error.

- The average accuracy for WIP-based is usually around 3%. In case of cyclic policy, the results are similar, i.e., about 4-5%. The reason for larger discrepancies could be due to that the weighted sums of buffer occupancy are used in the derivation so that even small differences in estimation could lead to a larger discrepancy. However, usually the large percentage errors come with small absolute values.

In summary, the decomposition based iteration method can provide acceptable accuracy in estimation of system production rate.

5.4. Extension to Longer Lines

The above decomposition method provides a foundation or building block for analysis of large systems. Consider a multi-stage production line with more than two machines and dedicated buffers, as shown in Figure 3. Clearly, analyzing such a system with different scheduling policies is challenging, even for simulations since computation intensity will limit their applications. Using the two-machine decomposition method introduced in this study, we can analyze every pair of two machines and obtain an aggregated production rate. However, these two-machine pairs are not independent, i.e., the first and second machines in each pair could be starved and blocked, respectively. To solve this issue, the idea of backward and forward aggregation procedures introduced in [5] can be applied.

Specifically, the following approach is proposed: First we analyze backward. Starting from the last machine and using the two-machine analysis method, we can aggregate every pair of two machines with their intermediate buffers into a single backward machine. Then repeat this process backward until we aggregate the whole line into one backward machine. Next we go forward. Starting from the first machine and using the two-machine method again, we aggregate every pair into a single forward machine and continue until we obtain one forward machine for the whole line. This finishes the first iteration. Starting from the second iteration, we aggregate every pair of backward and forward machines into a new backward machine in the backward procedure, and then aggregate every pair of forward and backward machines into a new forward machine in the forward procedure. Repeat these iterations until convergence. Finally we will obtain the estimation of system production rate. The lower part of Figure 3 illustrates such an aggregation process.

We hypothesize that such a procedure is convergent and will lead to accurate estimation. The challenging part is to retain the monotonic property

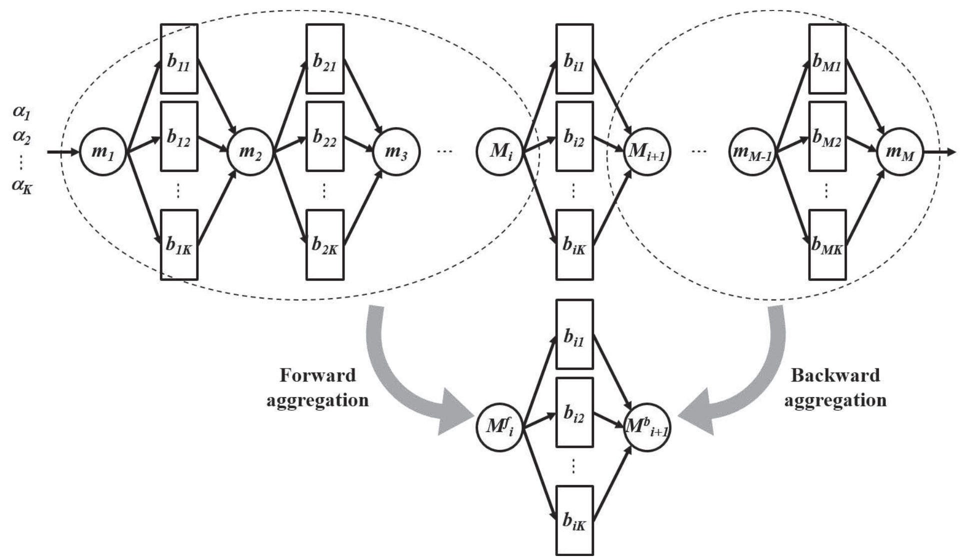

Figure 3: Example of the multiple machine lines (six machines with both of dedicated and non-dedicated buffers)

during aggregations, which is critical in ensuring convergence. A detailed study is planned in future work.

6. Discussions

6.1. Conservation of Flow

The conservation of flow, i.e., PR of machine m_1 equals to PR of machine m_2, holds when $N_j < \infty$, $\forall j$. This can be explained either through the Markov chain model with limiting behavior or using the decomposed lines where conservation of flow holds. Mathematically, it can be proved that

Proposition 1. *Under assumptions 1)-9), the product ratio is conserved regardless of policy. That is,*

$$\frac{\alpha_j}{\alpha_k} = \frac{PR_j}{PR_k}, \quad \forall j, k. \tag{22}$$

Proof: See Appendix B. ■

Note that although the proof is based on the decomposition method, the results are validated in general, since in steady state the number of parts produced by each machine will be the same during a long time period. In addition, such a property is also justified by extensive numerical studies and simulations (same for subsequent propositions). However, the conservation of flow will not hold anymore when $N_j = \infty$ since steady state may not exist.

6.2. Asymptotic Properties

When the capacity of every buffer becomes infinite, the following property holds.

Proposition 2. *Under assumptions 1)-9), for any* $j \in \{1, \ldots, K\}$*, if* $p_{1j} = p_1$*,* $p_{2j} = p_2$*,* $\alpha_j = 1/K$*, and* $N_j = \infty$*, then the following holds:*

$$PR = \min(p_1, p_2). \tag{23}$$

In addition, for WIP-based and cyclic policies

$$PR_j = \frac{PR}{K}. \tag{24}$$

Proof: See Appendix B. ■

The rationale of equation (23) is that no machine can produce more than the worst machine. Note that under WIP-based and cyclic policies, all part types have the same production rate. However, under priority policy, machine m_2 will process more higher priority parts but less lower ones. For example, when $p_1 = 0.9$, $p_2 = 0.7$ and $K = 5$, type 1 has the highest priority so that PR_1 is larger than 0.14 (which is one fifth of overall production rate, i.e., $\frac{\min(0.9,0.7)}{5}$). On the other hand, type 5 has the lowest priority, thus PR_5 is close to zero since the type 5 part is least processed.

6.3. Reversibility

Although reversibility holds for serial lines making single product type (see [5]), it is not true in general for systems under assumptions 1)-9). For example, consider a production line with parameters shown in Table 7. Under priority, WIP-based, and cyclic policies, using the Markov chain method, the production rates are calculated. By reversing the line, the correspond production rates can be evaluated, which are different with the original ones (see Table 7 for details). As one can see, reversibility does not hold.

A possible explanation is that due to scheduling policies, m_1 and m_2 load parts using different rules. When the line is reversed, only the machines

Table 7: Production rates in original and reversed lines

	System	PR		
	$(K=2, \alpha_1=0.7, \alpha_2=0.3, N_1=1, N_2=5)$	Priority	WIP	Cyclic
Original line	$p_{11}=p_{12}=0.5, p_{21}=0.9, p_{22}=0.3$	0.4739	0.4119	0.4505
Reversed line	$p_{11}=0.9, p_{12}=0.3, p_{21}=p_{22}=0.5$	0.4299	0.3957	0.3978

are switched but not the loading policies. Therefore, only under certain conditions (e.g., no difference in selecting the part types), reversibility may still hold. One of the conditions is introduced below.

Proposition 3. *Under assumptions 1)-9), for WIP-based and cyclic policies, if* $\forall j \in \{1, \ldots, K\}$, $p_{1j} = p_1$, $p_{2j} = p_2$, $\alpha_j = \frac{1}{K}$, *then* $PR^{original} = PR^{reverse}$.

Proof: See Appendix B. ■

It turns out that in this case, the loading policies are equivalent in both machines m_1 and m_2 due to identical machines and ratio for each product type.

6.4. Monotonicity with respect to Buffer Capacity

Through extensive numerical experiments, we study the monotonicity with respect to buffer capacity.

Numerical Fact 1. *Under assumptions 1)-9), both the production rate of each product type,* PR_j, $1 \leq j \leq K$, *and the overall production rate,* PR, *are monotonically increasing in buffer capacity* N_j, $j = 1, \ldots, K$.

Intuitively, larger buffers reduce the possibilities of blockage and starvation, which lead to higher production rate. Due to conservation of flow, the increase of buffer capacity in N_j will not only lead to increase of PR_j, but production rates of other part types, PR_k, $k \neq j$. Consider the example illustrated in Figure 4, where $K = 3$, $p_{1j} = 0.7$, $p_{21} = 0.7$, $p_{22} = 0.8$, $p_{23} = 0.9$, $\alpha_j = 1/3$, $N_j = 1$, $\forall j \in \{1, 2, 3\}$. The buffer capacity of part type j increases while others retain at 1. Using the Markov chain method, we evaluate the corresponding production rates. Clearly, in all cases PR increases no matter which scheduling policy is used. Similar to the single product case, the growth rate of PR is decreasing.

As one can see, each buffer's capacity increase will have different impact on production rate increase. To further investigate this, we observe:

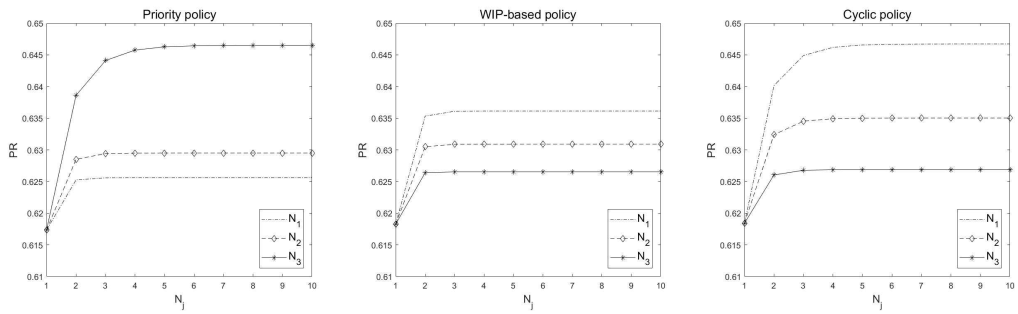

Figure 4: Monotonicity with respect to buffer capacity

Numerical Fact 2. *Under assumptions 1)-9), the production rate is most sensitive to the buffer capacity with the highest blockage probability. In other words, increasing one unit in* N_j*, which has the largest* BL_j*, will lead to the largest improvement in throughput comparing with increasing one unit in other buffers. That is, if*

$$BL_j \geq BL_k, \quad \forall k \neq j, \quad j, k \in \{1, \ldots, K\},$$

then we have

$$PR(N_j + 1) \geq PR(N_k + 1).$$

Since blockage will prevent entrance of all product types, reducing it will lead to an increase of production rate. Thus, it is effective to add one additional buffer capacity into the type with the largest blocking probability. For small systems ($K = 3$), we verify that increasing one unit on the buffer having the largest blocking probability will result in the highest production rate (calculated using Markov chain method) in priority, WIP-based, and cyclic policies, respectively. The percentages Numerical Fact 2 holds are shown in Table 8. For large systems ($K = 5$), as illustrated in Table 8, these percentages reduce just slightly under WIP-based and cyclic policies but more under priority policy (evaluated using decomposition method). The experiments are carried out 1,000 times by randomly selecting parameters from sets (15) except $N_j \in [1, 5]$.

Moreover, for priority policy, the low priority products typically have more blocking comparing to the higher priority parts. Therefore, selecting the most blocked buffer is almost equivalent to choosing the lowest priority buffer. Specifically, we obtain

Table 8: Cases Numerical Fact 2 holds

	Method	Priority	WIP-based	Cyclic
$K = 3$	Markov chain	93.9%	96.3%	92.5%
$K = 5$	Decomposition	75.4%	96.1%	92.2%

Numerical Fact 3. *Under assumptions 1)-9) and priority policy, the production rate is more sensitive to the buffer capacity for low priority product. Particularly, the buffer capacity of the least prioritized type is most critical to system performance.*

Consider the four cases in Figures 5 and 6 using Markov chain and decomposition methods, respectively. The buffer capacity for part type j increases while others retain at 1 and 5. As one can see, increasing the least prioritized part's buffer (N_2 in Figure 5 and N_5 in Figure 6) has more significant results in production rate improvement. Note that $\alpha_j = 1/K$ in all cases.

6.5. Policy Comparison

The priority policy is typically introduced due to specific reasons, such as due date, cost, and time constraint. Still there are questions when WIP-based and cyclic policies should be used and how much difference in system production rate they may exhibit. To answer them, the following observations are obtained through extensive numerical experiments.

Numerical Fact 4. *Under assumptions 1)-9), WIP-based policy is favorable when buffers for all product types are large. On the other hand, cyclic policy is superior when the buffer capacities for different product types have large variations. However, the differences in production rate between these two policies are quite small in all cases.*

When buffers for all product types are large enough, WIP-based policy can focus on processing part types more close to generating blockage while cyclic policy still circulates between product types without considering blockage, which may lead to smaller production rate. For example, WIP-based policy is superior to cyclic policy (calculated using Markov chain method) for 91.4% of 1,000 experiments where $K = 2$, $N_j \in [4, 10]$, $j = 1, 2$. However, WIP-based policy may tend to produce only products with big buffers when the buffers are unbalanced, which lead to blocking in small buffers so that less production rate is obtained. As a result, WIP-based policy is superior

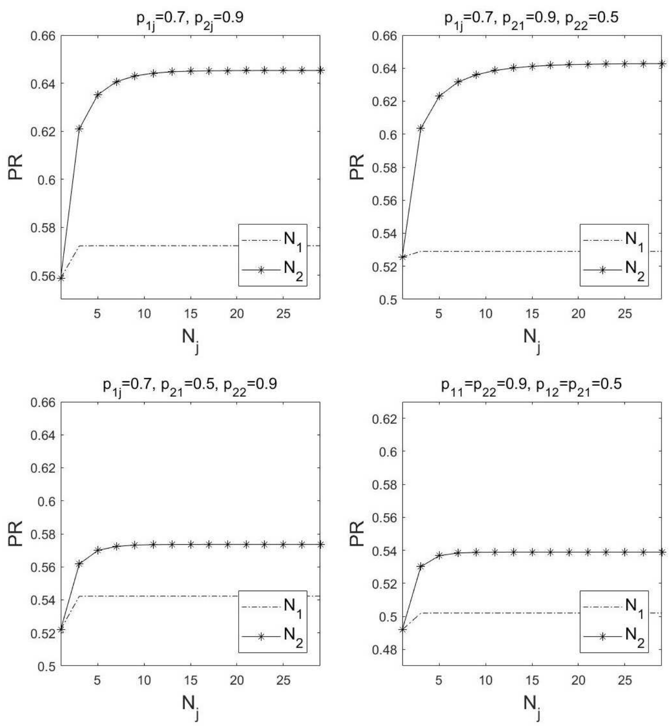

Figure 5: Production rate with respect to buffer capacity under priority policy ($K = 2$, Markov chain model)

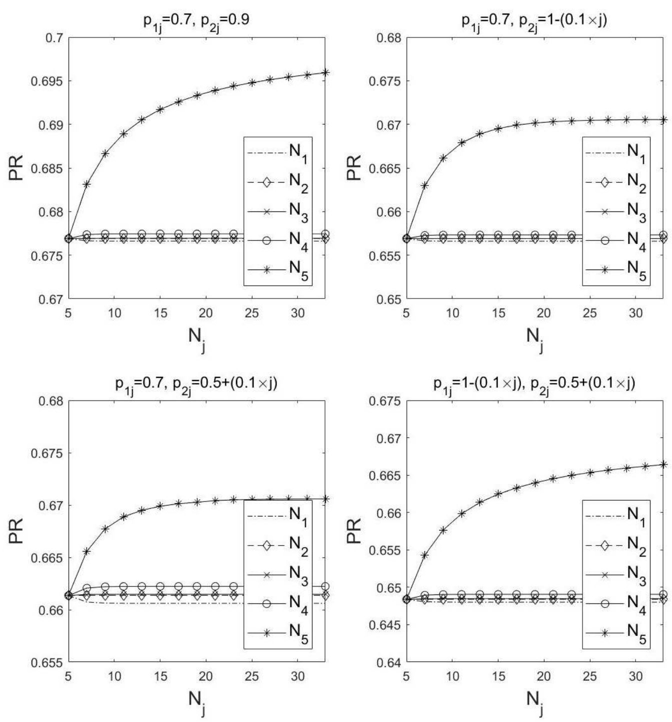

Figure 6: Production rate with respect to buffer capacity under priority policy ($K = 5$, decomposition model)

to cyclic policy only for 26.8% of 1,000 experiments when $K = 5$, $N_1 \in [4, 7]$, $N_j \in [1, 3]$, $j = 2, \ldots, 5$ (evaluated by simulations). Again parameters in these experiments are randomly selected from sets (15).

Nevertheless, the differences in production rates under these two policies are small. The average differences are less than 1.0% for all the experiments we conducted, where $K = 2, \ldots, 10$, $N_j \in [1, 10]$, $j = 1, \ldots, K$. This implies one can use either policy (depending on other factors that may impact scheduling policy selection), or use WIP-based policy when all buffers are large and cyclic policy in other scenarios.

7. Conclusions

In this paper, analytical methods are developed to evaluate the performance of two-machine Bernoulli lines with dedicated finite buffers under three prevalent scheduling policies: priority, WIP-based and cyclic polices. For each policy, exact solution is derived for small scale systems. Particularly, when buffer capacities are smaller than 6 and there exist up to 5 product types, production rate can be calculated using Markov chain method. For larger systems, in order to overcome computation intensity, approximation algorithms through decomposition and iteration procedures are introduced, which leads to acceptable accuracy, i.e., average percentage errors less than 1.5% for priority policy, around 3% and 5% for WIP-based and cyclic policies, respectively. In addition, system properties and the impact of buffers are discussed. It is shown that asymptotic and monotonic properties hold as in serial lines, and conservation of flow is still kept, but reversibility does not hold anymore. Finally, by comparing the scheduling policies, we observe that WIP-based policy is more suitable when all buffers are large, while cyclic policy is favored when the buffer capacities differ significantly.

Future work can be directed as follows:

- Extending study from two-machine lines to longer lines. Particularly, as discussed in Subsection 5.4, developing a convergent aggregation method is of key importance.
- Generalizing the model from Bernoulli machines to other reliability models, such as geometric, exponential or general reliability ones.
- Investigating other scheduling policies which are widely used in industry, such as processing time based (longest or shortest) policy.

- Developing production control, buffer design, and continuous improvement (e.g., bottleneck analysis) methods with respect to machine parameters and buffer capacity under different policies.

- Incorporating sequence dependent and independent setup and changeover times during product type switch in systems operations.

- Validating and applying the work on the factory floor.

Disclaimer

Certain commercial software products or services may have been identified in this paper. These products or services were used only for demonstration purposes. This use does not imply approval or endorsement by NIST, nor does it imply that these products are necessarily the best for the purpose.

Appendix A: Derivations of State Transition Probabilities

7.1. Priority and WIP-based Policies

To calculate the transition probability P_{s_1,s_2} from state s_1 to s_2, consider the following scenarios:

- All buffers are empty, i.e., $s_1 = (0, 0, \ldots, 0, u)$. Then state s_1 either stays (machine m_1 is down for part type u) or transits to the states with one u type part (m_1 is up for type u) in buffer b_u, and type k part to be processed by m_1 in next cycle, $k = 1, \ldots, K$. All other transitions will not occur. Thus, the transition probabilities are:

  - $s_2 = (0, \ldots, 1, \ldots, 0, k)$, $k \in \{1, \ldots, K\}$,

  $$P_{s_1,s_2} = p_{1u}\alpha_k.$$

  - $s_2 = s_1 = (0, 0, \ldots, 0, u)$,

  $$P_{s_1,s_2} = 1 - p_{1u}.$$

- Machine m_1 will process type u with an empty buffer, but not all other buffers are empty. State $s_1 = (h_1, \ldots, 0, \ldots, h_K, u)$ with at least one $h_j > 0, j \in \{1, \ldots, K\}, j \neq u$. Let v denote the product type to be processed by machine m_2. Since there could be multiple buffers having the same highest occupancy in WIP-based policy, let set V consist of part types of all these buffers and n represent the number of buffers having the highest occupancy simultaneously.

$$\begin{aligned} &V = \cup_k \{k | h_k \geq h_j, \forall j \in \{1, \ldots, K\}, j \neq k\}, \\ &n = \dim(V). \end{aligned}$$

To simplify the notation, we assume $n = 1$ and $V = \{v\}$ for priority policy. It follows that the transition probabilities are:

  - $s_2 = (h_1, \ldots, 1 \ldots, h_v - 1, \ldots, h_K, k)$, $k \in \{1, \ldots, K\}$, $v \in V$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} p_{2v} \alpha_k,$$

  - $s_2 = (h_1, \ldots, 1, \ldots, h_K, k)$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} \alpha_k \sum_{v \in V} (1 - p_{2v}),$$

  - $s_2 = (h_1, \ldots, 0, \ldots, h_v - 1, \ldots, h_K, u)$,

$$P_{s_1,s_2} = \frac{1}{n} (1 - p_{1u}) p_{2v},$$

  - $s_2 = s_1 = (h_1, \ldots, 0, \ldots, h_K, u)$,

$$P_{s_1,s_2} = \frac{1}{n} (1 - p_{1u}) \sum_{v \in V} (1 - p_{2v}).$$

- The buffer whose product type to be processed at m_1 is full. That is, $s_1 = (h_1, \ldots, N_u, \ldots, h_K, u)$. Then the transition probabilities are:

  - State is unchanged, i.e., $s_2 = s_1$,

$$P_{s_1,s_2} = \begin{cases} \frac{1}{n} \sum_{v \in V} (1 - p_{2v}) + \frac{1}{n} p_{1u} p_{2u} \alpha_u, \\ \qquad\qquad \text{if } u \in V, \\ \frac{1}{n} \sum_{v \in V} (1 - p_{2v}), \text{ if } u \notin V. \end{cases}$$

- State is changed, machine m_2 will process the same part type:
  - $s_2 = (h_1, \ldots, N_u, \ldots, h_K, k)$, $k \neq u$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} p_{2u} \alpha_k,$$

  - $s_2 = (h_1, \ldots, N_u - 1, \ldots, h_K, u)$,

$$P_{s_1,s_2} = \frac{1}{n}(1 - p_{1u}) p_{2u}.$$

- State is changed, machine m_2 will process another part type v, $v \neq u$, $v \in V$, then $s_2 = (h_1, \ldots, N_u, \ldots, h_v - 1, \ldots, h_K, u)$,

$$P_{s_1,s_2} = \frac{1}{n} p_{2v}.$$

- The buffer whose product type to be processed at m_1 is neither full nor empty. That is, $s_1 = (h_1, \ldots, h_u, \ldots, h_K, u)$ and $0 < h_u < N_u$. Then transition probabilities are:
  - State is unchanged, i.e., $s_2 = s_1$,

$$P_{s_1,s_2} = \begin{cases} \frac{1}{n}(1 - p_{1u}) \sum_{v \in V}(1 - p_{2v}) + \frac{1}{n} p_{1u} p_{2u} \alpha_u, & \text{if } u \in V, \\ \frac{1}{n}(1 - p_{1u}) \sum_{v \in V}(1 - p_{2v}), & \text{if } u \notin V. \end{cases}$$

  - State is changed, machine m_2 will process the same part type, $u = v$,
    - $s_2 = (h_1, \ldots, h_u, \ldots, h_K, k)$, $k \neq u$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} p_{2u} \alpha_k,$$

    - $s_2 = (h_1, \ldots, h_u - 1, \ldots, h_K, u)$,

$$P_{s_1,s_2} = \frac{1}{n}(1 - p_{1u}) p_{2u}.$$

  - State is changed, machine m_2 will process another part type, $v \neq u$, then
    - $s_2 = (h_1, \ldots, h_u + 1, \ldots, h_v - 1, \ldots, h_K, k)$, $k \in \{1, \ldots, K\}$, $v \in V$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} p_{2v} \alpha_k,$$

* $s_2 = (h_1, \ldots, h_v - 1, \ldots, h_K, u)$, $v \in V$,

$$P_{s_1,s_2} = \frac{1}{n}(1 - p_{1u})p_{2v}.$$

- State is changed, machine m_2 cannot work, then $s_2 = (h_1, \ldots, h_u + 1, \ldots, h_K, k)$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = \frac{1}{n} p_{1u} \alpha_k \sum_{v \in V} (1 - p_{2v}).$$

7.2. Cyclic policy

The transition probabilities are addressed in the following scenarios.

- All buffers are empty, i.e., $s_1 = (0, 0, \ldots, 0, u, 0)$, $1 \le u \le K$. Then next cycle m_2 will prepare to process part type u which is just processed by machine m_1 and sent to buffer b_u during this cycle. Thus, the transition probabilities are
  - $s_2 = (0, \ldots, 1, \ldots, 0, k, u)$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = p_{1u} \alpha_k,$$

  - $s_2 = s_1 = (0, 0, \ldots, u, 0)$,

$$P_{s_1,s_2} = 1 - p_{1u}.$$

- The other transition probabilities can be derived from using the similar methods for priority and WIP-based policies by adding v and $v+1$ into the last element of s_1 and s_2, respectively. For example, in the last case (i.e., $0 < h_u < N_u$), $s_1 = (h_1, \ldots, h_u, \ldots, h_K, u, v)$, $0 < h_u < N_u$, the transition probabilities are as follows:
  - Buffer status is unchanged, i.e., $s_2 = (h_1, \ldots, h_u, \ldots, h_K, u, v+1)$,

$$P_{s_1,s_2} = \begin{cases} (1 - p_{1u})(1 - p_{2v}) + p_{1u} p_{2u} \alpha_u, & \text{if } v = u, \\ (1 - p_{1u})(1 - p_{2v}), & \text{if } v \ne v. \end{cases}$$

  - Buffer status is changed, machine m_2 will process the same part type $v = u$, then

- $s_2 = (h_1, \ldots, h_u, \ldots, h_K, k, v+1)$, $k \neq u$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = p_{1u} p_{2u} \alpha_k,$$

- $s_2 = (h_1, \ldots, h_u - 1, \ldots, h_K, u, v+1)$,

$$P_{s_1,s_2} = (1 - p_{1u}) p_{2u}.$$

– Buffer status is changed, machine m_2 will process another part type $v \neq u$, then

- $s_2 = (h_1, \ldots, h_u{+}1, \ldots, h_v{-}1, \ldots, h_K, k, v{+}1)$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = p_{1u} p_{2v} \alpha_k,$$

- $s_2 = (h_1, \ldots, h_v - 1, \ldots, h_K, u, v+1)$,

$$P_{s_1,s_2} = (1 - p_{1u}) p_{2v}.$$

– Buffer status is changed, machine m_2 cannot work, then $s_2 = (h_1, \ldots, h_u + 1, \ldots, h_K, k, v+1)$, $k \in \{1, \ldots, K\}$,

$$P_{s_1,s_2} = p_{1u}(1 - p_{2v}) \alpha_k.$$

Appendix B: Proofs

Proof of Theorem 1: The production rate of part type k is evaluated by enumerating the scenarios that m_2 is ready to process a type k part and its buffer is not empty. ■

Proof of Corollary 1: By state definition (h_1, h_2, u), we obtain

$$s_1 = (0,0,1),\ \ s_2 = (0,0,2),\ \ s_3 = (1,0,1),\ \ s_4 = (1,0,2),$$
$$s_5 = (0,1,1),\ \ s_6 = (0,1,2),\ \ s_7 = (1,1,1),\ \ s_8 = (1,1,2).$$

Then the transition probability matrix P can be derived (see next page). By solving equation (1), the steady-state probability ψ_m for state s_m, $m = 1, \ldots, 8$, can be obtained. From Theorem 1,

$$PR_1 = PR_2 = p_2(\psi_3 + \psi_4 + \psi_7 + \psi_8) = p_2(\psi_5 + \psi_6).$$

$$P = \begin{bmatrix} 1-p_1 & 0 & \frac{p_1}{2} & \frac{p_1}{2} & 0 & 0 & 0 & 0 \\ 0 & 1-p_1 & 0 & 0 & \frac{p_1}{2} & \frac{p_1}{2} & 0 & 0 \\ (1-p_1)p_2 & 0 & (1-p_2)+\frac{p_1p_2}{2} & \frac{p_1p_2}{2} & 0 & 0 & 0 & 0 \\ 0 & (1-p_1)p_2 & 0 & (1-p_1)(1-p_2) & \frac{p_1p_2}{2} & \frac{p_1p_2}{2} & \frac{p_1(1-p_2)}{2} & \frac{p_1(1-p_2)}{2} \\ (1-p_1)p_2 & 0 & \frac{p_1p_2}{2} & \frac{p_1p_2}{2} & (1-p_1)(1-p_2) & 0 & \frac{p_1(1-p_2)}{2} & \frac{p_1(1-p_2)}{2} \\ 0 & (1-p_1)p_2 & 0 & 0 & \frac{p_1p_2}{2} & (1-p_2)+\frac{p_1p_2}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & (1-p_1)p_2 & 0 & (1-p_2)+\frac{p_1p_2}{2} & \frac{p_1p_2}{2} \\ 0 & 0 & 0 & 0 & 0 & p_2 & 0 & 1-p_2 \end{bmatrix}.$$

Expression (3) can be obtained. ■

Proof of Proposition 1: Define $PR_j^{m_i}$ be the production rate of type j part at machine m_i, $i = 1, 2$, $j = 1, \ldots, K$.

$$\begin{aligned} PR_j^{m_1} &= \alpha'_j p_{1j}[(1 - X_{j,N_j}) + X_{j,N_j} p'_{2j}] \\ &= \frac{\alpha_j p_{1j}[(1 - X_{j,N_j}) + X_{j,N_j} p'_{2j}]}{p_{1j}[1 - X_{j,N_j}(1 - p'_{2j})] \sum_{k=1}^{K} \frac{\alpha_k}{p_{1k}[1 - X_{k,N_k}(1 - p'_{2k})]}} \\ &= \frac{\alpha_j}{\sum_{k=1}^{K} \frac{\alpha_k}{p_{1k}[1 - X_{k,N_k}(1 - p'_{2k})]}}. \end{aligned}$$

Thus,

$$\frac{PR_j^{m_1}}{PR_k^{m_1}} = \frac{\alpha_j}{\alpha_k}.$$

Let use $I_j(t)$ to denote the amount of type j parts entered into buffer b_j during time 0 to t, which can be rewritten as

$$I_j(t) = PR_j^{m_1} \times t.$$

Then type j parts produced by m_2 during 0 to t, $PR_j^{m_2}(t)$, and production rate of type j part at m_2, $PR_j^{m_2}$, can be evaluated as

$$\begin{aligned} PR_j^{m_2}(t) &= I_j(t) + h_j(0) - h_j(t) = PR_j^{m_1} \cdot t + h_j(0) - h_j(t), \\ PR_j^{m_2} &= \lim_{t \to \infty} \frac{PR_j^{m_2}(t)}{t} = PR_j^{m_1} + \lim_{t \to \infty} \frac{h_j(0) - h_j(t)}{t} = PR_j^{m_1}, \end{aligned}$$

where $h_j(0)$ and $h_j(t)$ are the occupancy in buffer b_j at time 0 and t, respectively, and $0 \leq h_j(t) \leq N_j$. Then, for steady state, we obtain

$$\frac{PR_j}{PR_k} = \frac{PR_j^{m_2}}{PR_k^{m_2}} = \frac{PR_j^{m_1}}{PR_k^{m_1}} = \frac{\alpha_j}{\alpha_k}.$$

■

Proof of Proposition 2: Since all product types have identical machines (p_{ij}) and ratio (α_j), from (9) we have $\alpha' = \alpha_j$. Thus

$$p'_{1j} = p_{1j}\alpha_j = \frac{p_{1j}}{K}.$$

In addition, from (16), for WIP-based policy, all buffers have equal probability to have the highest occupancy. Thus

$$p'_{2j} = \frac{1}{K} \cdot p_{2j}.$$

For cyclic policy, from (20), all buffers have the same empty probability. Thus

$$p'_{2j} = \frac{p_{2j}}{K - (K-1)X'_{j,0}} = p'_{2l}, \quad l \neq j.$$

In both cases, we obtain K identical serial lines. In each line we have

$$PR_j = \min\left(\frac{p_{1j}}{K}, \frac{p_{2j}}{K}\right) = \frac{1}{K}\min(p_1, p_2).$$

Thus

$$PR = \sum_{j=1}^{K} PR_j = \sum_{j=1}^{K} \frac{1}{K}\min(p_1, p_2) = \min(p_1, p_2).$$

It also follows that

$$PR_j = \frac{PR}{K}.$$

Under priority policy, from (13), if $\frac{p_1}{K} \geq p_2$, then we have

$$p'_{11} > p'_{21},$$

which leads to $\phi > 0$ and $Q(p'_{11}, p'_{21}, N_1) = 0$ when $N_1 \to \infty$. Thus,

$$PR = PR_1 = \min\left(\frac{p_1}{K}, p_2\right) = p_2 = \min(p_1, p_2).$$

If $\frac{p_1}{K} < p_2 \prod_{k=1}^{K-1} X'_{k,0}$, which implies $p_1 < p_2$, then

$$p'_{1j} < p'_{2j}, \quad \forall j.$$

Again it follows that

$$PR_j = \min\left(\frac{p_1}{K}, p'_{2j}\right) = \frac{p_1}{K},$$
$$PR = \sum_{j=1}^{K} PR_j = p_1 = \min(p_1, p_2).$$

If $\frac{p_1}{K} \leq p_2 \prod_{k=1}^{l-1} X'_{k,0}$ and $\frac{p_1}{K} > p_2 \prod_{k=1}^{l} X'_{k,0}$, i.e., in the first l lines

$$PR_j = \frac{p_1}{K}, \quad j = 1, \ldots, l,$$
$$X'_{1,0} = \frac{p_2 - \frac{p_1}{K}}{p_2},$$
$$X'_{2,0} = \frac{p_2 - \frac{p_1}{K} - \frac{p_1}{K}}{p_2 - \frac{p_1}{K}} = \frac{p_2 - \frac{2p_1}{K}}{p_2 - \frac{p_1}{K}},$$
$$X'_{3,0} = \frac{p_2 - \frac{2p_1}{K} - \frac{p_1}{K}}{p_2 - \frac{2p_1}{K}} = \frac{p_2 - \frac{3p_1}{K}}{p_2 - \frac{2p_1}{K}},$$
$$X'_{k,0} = \frac{p_2 - \frac{kp_1}{K}}{p_2 - \frac{(k-1)p_1}{K}}, \quad k = 1, \ldots, l,$$

and for line $l+1$,

$$PR_{l+1} = p_2 \cdot \frac{p_2 - \frac{p_1}{K}}{p_2} \cdot \frac{p_2 - \frac{2p_1}{K}}{p_2 - \frac{p_1}{K}} \cdot \ldots \cdot \frac{p_2 - \frac{l \cdot p_1}{K}}{p_2 - \frac{(l-1)p_1}{K}}$$
$$= p_2 - \frac{l \cdot p_1}{K}.$$

and no production is made in the next $K - l - 1$ lines due to $X'_{l+1,0} = 0$. In addition, this also implies $p_1 > p_2$. Thus, the overall production rate is

$$PR = l \cdot \frac{p_1}{K} + p_2 - \frac{l \cdot p_1}{K} = p_2 = \min(p_1, p_2).$$

■

Proof of Proposition 3: Under WIP-based and cyclic policies, from the proof of Proposition 2, we have

$$p'_{1j} = \frac{p_{1j}}{K}, \quad j = 1, \dots, K,$$
$$p'_{2j} = \begin{cases} \frac{1}{K} \cdot p_{2j}, & \text{for WIP-based policy,} \\ \frac{p_{2j}}{K-(K-1)X'_{j,0}} = p'_{2l}, & \text{for cyclic policy}, l \neq j. \end{cases}$$

Thus, K identical serial lines are obtained. As reversibility holds for each line,

$$PR_j^{original} = PR_j^{reverse}, \quad j = 1, \dots, K,$$

the overall production rate also exhibits such a property:

$$PR^{original} = \sum_{j=1}^{K} PR_j^{original} = \sum_{j=1}^{K} PR_j^{reverse} = PR^{reverse}.$$

■

[1] Viswanadham N, Narahari Y. Performance modeling of automated manufacturing systems. Englewood Cliffs, NJ: Prentice Hall; 1992.

[2] Buzacott JA, Shantikumar JG. Stochastic models of manufacturing systems. Englewood Cliffs, NJ: Prentice Hall; 1993.

[3] Papadopoulos HT, Browne J, Heavey C. Queueing theory in manufacturing systems analysis and design. London: Chapman & Hall; 1996.

[4] Gershwin SB. Manufacturing systems engineering. Englewood Cliffs, NJ: Prentice Hall; 1994.

[5] Li J, Meerkov SM. Production systems engineering. New York, NY: Springer; 2009.

[6] Dallery Y, Gershwin SB. Manufacturing flow line systems: a review of models and analytical results. Queueing Systems 1992;12:394.

[7] Papadopoulos HT, Heavey C. Queueing theory in manufacturing systems analysis and design: a classification of models for production and transfer lines. European Journal of Operational Research 1996;92:127.

[8] Li J, Blumenfeld DE, Huang N, Alden JA. Throughput analysis of production systems: recent advances and future topics. International Journal of Production Research 2009;47:3823-3851.

[9] Buzacott JA, Yao DD. Flexible manufacturing systems: a review of analytical models. Management Science 1986;32:890-905.

[10] Sethi AK, Sethi SP. Flexibility in manufacturing: a survey. International Journal of Flexible Manufacturing Systems 1990;2:289-328.

[11] Tempelmeier H, Kuhn H. Flexible manufacturing systems: decision support for design and operation. New Nork, NY: John Wiley & Sons; 1993.

[12] Zhou M, Venkatesh K. Modeling, simulation and control of flexible manufacturing systems: a Petri net approach. Singapore: World Scientific Publishing; 1999.

[13] Beach R, Muhlemann AP, Price DHR, Paterson A, Sharp JA. A review of manufacturing flexibility. European Journal of Operational Research 2000;122:41-57.

[14] Shi D, Daniels RL. A survey of manufacturing flexibility: implications for e-business flexibility. IBM Systems Journal 2003;42:414-427.

[15] Nemec JE. Diffusion and decomposition approximations of stochastic models of multiclass processing networks. PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA; 1998.

[16] Syrowicz D. Decomposition analysis of a deterministic, multiple-part-type, multiple-failure-mode production line. Master Thesis, Massachusetts Institute of Technology, Cambridge, MA; 1999.

[17] Baynat B, Dallery Y, Di Mascolo M, Frein Y. A multi-class approximation technique for the analysis of kanban-like control systems. International Journal of Production Research 2001;39:307-328.

[18] Buitenhek R, Baynat B, Dallery Y. Production capacity of flexible manufacturing systems with fixed production ratios. International Journal of Flexible Manufacturing Systems 2002;14:203-225.

[19] Colledani M, Matta A, Tolio T. Performance evaluation of production lines with finite buffer capacity producing two different products. OR Spectrum 2005;27:243-263.

[20] Colledani M, Gandola F, Matta A, Tolio T. Performance evaluation of linear and non-linear multi-product multi-stage lines with unreliable machines and finite homogeneous buffers. IIE Transactions 2008;40:612-626.

[21] Alavian P, Denno P, Meerkov SM. Multi-job production systems: definition, problems, and product-mix performance portrait of serial lines. International Journal of Production Research. 2017;55:7276-7301.

[22] Krieg GN, Kuhn,H. A decomposition method for multi-product kanban systems with setup times and lost sales. IIE Transactions 2002;34:613-625.

[23] Krieg GN, Kuhn H. Analysis of multi-product kanban systems with state-dependent setups and lost sales. Annnals of Operations Research 2004;125:141-166.

[24] Dasci A, Karakul M. Performance evaluation of a single-stage two-product manufacturing system operating under pull-type control. Computer & Operations Research 2008;35:2861-2876.

[25] Feng W, Zheng L, Li J. Scheduling policies in multi-product manufacturing systems with sequence-dependent setup times and finite buffers, International Journal Production Research 2012;50:7479-7492.

[26] Feng W, Zheng L, Li J. The robustness of scheduling policies in multi-product manufacturing systems with sequence-dependent setup times and finite buffers. Computer & Industrial Engineering 2012;63:1145-1153.

[27] Kang N, Zheng L, Li J. Multi-product manufacturing systems with arbitrary processing times. International Journal of Production Research 2015;53:983-1001.

[28] Ryan SM, Vorasayan J. Allocating work in process in a multipleproduct CONWIP system with lost sales. International Journal of Production Research 2005;43:223-246.

[29] Satyam K, Krishnamurthy A. Performance evaluation of a multiproduct system under CONWIP control. IIE Transactions 2008;40:252-264.

[30] Jang YJ. Mathematical modeling and analysis of flexible production lines. PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA; 2007.

[31] Li J, Huang N. Modeling and analysis of a multiple product manufacturing system with split and merge. International Journal of Production Research 43 (2005) 4049-4066.

[32] Zhao C, Li J. Analysis of multi-product manufacturing systems with homogeneous exponential machines. IEEE Transactions on Automation Science and Engineering 11 (2014) 828-838.

[33] Zhao C, Li J, Huang N. Efficient algorithms for analysis and improvement of flexible manufacturing systems. IEEE Transactions on Automation Science and Engineering 13 (2016) 105-121.

[34] Zhao C, Li J. Analysis and improvement of multiproduct Bernoulli serial lines: theory and application. IEEE Transactions on Systems, Man, and Cybernetics: Systems 45 (2015) 1218-1230.

[35] Zhao C, Li J, Huang N, Horst J. Flexible lines with setups: analysis, improvement, and application. IEEE Robotics and Automation Letters 2 (2017) 120-127.

[36] Zhao C, Li J. Analysis and improvement of multi-product assembly systems: an application study at a furniture manufacturing plant. International Journal of Production Research 52 (2014) 6399-6413.

[37] Zhao C, Kang N, Li J, Horst J. Production control in partially flexible production-inventory systems: theory and application. IEEE Transactions on Automatic Control 63 (2017) 477-491.

[38] Kang N, Zhao C, Li J, Zheng L. A sub-optimal control policy in a two-product door manufacturing line with Geometric reliability machines. IEEE Robotics and Automation Letters 2 (2017) 157-164.

[39] Lim J-T, Meerkov SM, Top F. Homogeneous, asymptotically reliable serial production lines: theory and a case study. IEEE Transactions on Automatic Control 1990;25:524-534.

[40] Lim J-T, Meerkov SM. On asymptotically reliable closed serial production lines. Control Engineering Practice 1993;1:147-152.

[41] Kuo C-T, Lim J-T, Meerkov SM. Bottlenecks in serial production lines: a system-theoretic approach. Mathematical Problems in Engineering 1996;2:233-276.

[42] Chiang S-Y, Kuo C-T, Lim J-T, Meerkov SM. Improvability of assembly systems II: improvability indicators and case study. Mathematical Problems in Engineering 2000;6:359-393.

[43] Li J, Meerkov SM. Customer demand satisfaction in production systems: a due-time performance approach. IEEE Transactions on Robotics and Automation 2001;17:472-482.

[44] Wang C, Li J. Approximate analysis of reentrant lines with Bernoulli reliability model. IEEE Transactions on Automation Science and Engineering 2010;7:708715.

[45] Arinez J, Biller S, Meerkov SM, Zhang L. Quality/quantity improvement in an automotive paint shop: a case study. IEEE Transactions on Automation Science and Engineering 2010;7:755-761.

[46] Jia Z, Zhang L, Arinez J, Xiao G. Finite production run-based serial lines with Bernoulli machines: Performance analysis, bottleneck, and case study. IEEE Transactions on Automation Science and Engineering 2016;13:134-148.

[47] Feng Y, Zhong X, Li J, Fan W. Analysis of closed loop production lines with Bernoulli reliability machines: theory and application. IISE Transactions 2018;50:143-160.