ISSN 2683-345X
journal homepage: http://ijiemjournal.uns.ac.rs/

International Journal of Industrial Engineering and Management

Volume 13 / No 4 / December 2022 / 239 - 249

Original research article

Optimized Buffer Allocation and Repair Strategies for Series Production Lines

N. Nahas[a,*], M. Nourelfath[b], M. Abouheaf[c]

[a] *Faculté d'Administration, Université de Moncton, Moncton (NB), Canada;*

[b] *Mechanical Engineering Department, Université Laval, Quebec (Qc), Canada;*

[c] *College of Technology, Architecture & Applied Engineering, Bowling Green State University, Bowling Green, USA*

ABSTRACT

This paper proposes an efficient approach for solving the buffer allocation problem with limited number of repair resources. In this problem, the objective is to maximize the throughput of a series production line through solving the buffer allocation optimization problem and electing the best repair polices. This is done using a simulation model to estimate the throughput combined with a self-adjusting search heuristic approach. The total count of repairmen is constrained to be strictly less than or equal to the count of repairable machines. Therefore, a self-adjusting search algorithm that is based on a nonlinear threshold accepting criterion is employed to reach the best production line configurations in terms of the buffer-levels and repair policies. The effectiveness of the combined simulation-search solution mechanism is tested using three different scenarios with 8 different repair policies. The numerical results showed that not only the proposed approach can find a near optimal solution in a limited and acceptable time, but also the overall production system performance can be improved by selecting the best repair policy.

ARTICLE INFO

Article history:

Received March 12, 2022
Revised October 14, 2022
Accepted October 25, 2022
Published online November 14, 2022

Keywords:
Buffer allocation;
Optimization;
Repair policy;
Heuristic simulation;
Non-linear Threshold algorithm

*Corresponding author:
Nabil Nahas
nabil.nahas@umoncton.ca

1. Introduction

The industrial entities continuously urge for innovative methodologies to survive the competition and maximize the throughput of production systems. Further, the recent advances in technology operations and market fluctuations imposed additional challenges to the operation of the production systems. To address the aforementioned challenges, the production operations need to be customized to control the costs, especially those related to equipment allocations, semi-finished products handling, and personnel circulations. This will definitely contribute to increasing the performance and effectiveness of the production systems. Therefore, reliable and flexible production methods are required to maintain a high-level of product quality, optimize and rationalize the use of equipment, and reduce the design and operating costs of the overall production process.

In order to satisfy the inevitable increasing demands, provide cheap and safe services, and at the same time guarantee continuity of service in the

Published by the University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia.

http://doi.org/10.24867/IJIEM-2022-4-316

event of failures, robust optimization techniques are recommended. These aim to increase the viability of the production processes (i.e., accounting for the reliability, availability, production rate, etc.), for example, by:

- increasing their level of redundancy,
- enhancing the reliability of system components,
- implementing corrective or preventive maintenance plans,
- introducing buffer stocks between machines.

In this paper, we are particularly interested in serial production lines (Figure 1) where a set of $\mathcal{N}$ machines are connected together and separated by intermediate buffer storages.

Referring to Figure 1, if machine $\mathcal{M}_i$ fails, machines $\mathcal{M}_{i\text{-}1}$, $\mathcal{M}_{i\text{-}2}$,..., *1* can continue to operate by unloading their production in the buffers located downstream of each of them. As for the machines $\mathcal{M}_{i+1}$, $\mathcal{M}_{i+2}$,..., $\mathcal{M}_{\mathrm{N}}$, they are fed by the buffers located upstream of them. If machine $\mathcal{M}_i$ is repaired before the upstream buffer is full or the downstream buffer is empty, then the line will not stop producing. Otherwise, machine $\mathcal{M}_i$ will be blocked if the upstream buffer is full and starved if the downstream buffer is empty. Therefore, the intermediate stocks play an important role in improving the performance of the production lines. However, these generate additional costs and the space allocated is generally limited. If the intermediate stocks are judiciously located and properly sized, then the failure of a machine does not necessarily cause the production line to stop.

Additionally, the number of available repairmen represents a decisive factor to ensure the robustness of the production line. In fact, an insufficient number of repairmen will cause longer down-times. Therefore, the goal is to optimize the distributions of the intermediate buffer-slots and select the number of repairmen in order to optimally operate the production line. However, there is no existing work dealing with such problem. The next section highlights this fact and lists other challenging aspects of the problem.

2. Related background

The modeling and efficacy of production lines can be analyzed and approximated using evaluative approaches such as decomposition and aggregation, Markov chains, and self-adjust simulation approaches as well. It is noted that, Markov chains are widely developed for such problems. However, these approaches are considered mainly for the short production lines (e.g. [7]). Other solutions employed Markov chains to find a set of linear equations that can be solved numerically. This is done using iterative techniques such as Gauss-Seidel (e.g. [11]), for example. Another class of the solution methods namely decomposition and aggregation, is employed when the state space is relatively large. Decomposition techniques [e.g. 8,29] rely on dividing the original production line into ($\mathcal{N}$ -1) virtual lines with only two machines. For each line *i*, the two virtual machines represent the aggregate behavior of the production line upstream and downstream of buffer B_i. Nonetheless, the variables of the stations such as average repair time, processing time, and the average time between failures are adjusted in an iterative manner. This will enable the material flow in the subsystems to properly reflect the flow in the original line. Thus, the variables of such modified stations are estimated using the DDX algorithm in [2]. The decomposition approaches exhibited robust and fast solution characteristics [5,6]. On other hand, aggregation methods count on replacing a two-machine-one-buffer sub-line by a single equivalent machine. Thus, for a production line with $\mathcal{N}$ machines and ($\mathcal{N}$ -1) buffers, $\mathcal{N}-1$ single aggregation steps will be applied [8]. The basic difference between these methods is related to the process of calculating the parameters of the aggregated stations [17,28]. All these approaches presume the availability of resources at all the times such as repairmen, for example.

The simulation methods utilize heuristic structures to search for the near optimal solutions. The importance of simulation methods arises when there is no analytical procedure to study the investigated systems. Hence, these methods are considered reliable tools to assess the performance of the production lines. Further, an unreliable production line

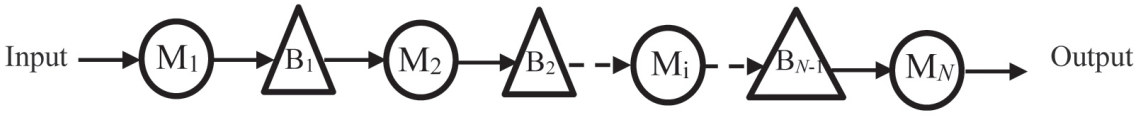

Figure 1. Serial production line

problem with intermediate buffers is analyzed using a discrete hybrid model (i.e., a combination of analytical and simulation procedures) in [13]. A simulation solution that is based on Tabu Search (TS) is developed for a manufacturing line to optimize the buffer allocation and storage size [19]. Another simulation mechanism advised a near-optimal buffer-placement solution for the serial production systems in [25]. A hybrid mechanism that combined together Genetic Algorithm (GA) and Simulated Annealing (SA) is considered to find a solution for the buffer allocation optimization problem in [16]. A discrete event based simulation model is used to calculate the average production rate. Overall, simulation-based approaches can be time-consuming and would need modified mechanisms to work faster [10]. Hence, meta-models that integrate the simulation model with artificial neural networks are developed to assess and solve production line problems [1]. More recently, in [30], the authors proposed a simulation-based metaheuristic approach to address the profit optimization problem of an automated manufacturing system with buffer units. In another study [32], the authors proposed a metamodeling solution to solve the buffer allocation problem. They applied a simulation-experiment design-optimization approach to get the metamodel.

Several methodologies employed search approaches to enhance the performance of production lines such as bottleneck [18], gradient [9], structural properties [26], and primal-dual search [16], for example. Other approaches relied on the Approximate Dynamic Programming (ADP) heuristic structures [4]. Over the last decade, meta-heuristics have been proposed to tackle the buffer allocation optimization problems such as GA [1], TS [3], Particle Swarm Optimization (PSO) [23], Variable Neighborhood Search (VNS) [31] and Non-Linear Threshold Algorithm (NLTA) [21],[22]. Additionally, hybrid search mechanisms that fuse together more than one heuristic are also considered such as a combination of SA and GA in [16] and TS and Nested Partition Algorithm (NPA) in [24].

As far as the authors know, there are no such exact or approximate solution approaches that are able to estimate the production rate of unreliable production lines with limited repair resources. Hence, this work develops and implements a simulation model for the continuous flow of production line and then employs it in a heuristic architecture to assess the production system's performance under a limited repairmen availability assumption (i.e., the total count of repairmen is strictly lower than the number of available stations or machines). The developed simulation model will be fused into an efficient optimization algorithm to search for a near-optimal optimization outcome. Therefore, this work provides two-fold contributions. On one hand, this represents the first attempt to solve the series production line problem to optimize both buffer allocation plans and repair strategies, simultaneously which is quite challenging. On the other hand, a combined simulation-optimization approach is proposed to solve the production line problem. As will be highlighted later, this solution approach will exhibit appealing characteristics such as having improved optimization quality and achieving fairly low implementation time.

The remaining sections are organized as follows. Section 3 lays out the mathematical foundation of the buffer allocation optimization problem. The simulation model of the production line and the search mechanism along with the solution methodology are detailed out in Section 4. The numerical validations are discussed in Section 5. Final observations and remakes are highlighted in Section 6.

3. Series production line optimization problem

This section introduces the mathematical layout of the Buffer Allocation Optimization Problem (BAOP) subject to a repair strategy. A series production line composed of $\mathcal{N}$ unreliable stations with ($\mathcal{N}$-1) intermediate buffers is considered (Figure 1). Further, each machine i $(i = 1,\dots,\mathcal{N})$ is characterized by a set that involves individual processing time T_i, failure rate λ_i, and repair rate μ_i. Once a machine failure occurs, a repairman will be assigned to handle it. If all on-duty repairmen are busy, then the failed machine will wait for the availability of a repairman. Furthermore, the repairman follows a repair policy RP, which extends from successfully completing a repair for one machine to the assignment on another failed machine. The intent of the BAOP is to optimally allocate the buffer-resources, find the necessary number of repairmen, and develop a repair policy that maximizes the production line's average throughput TH. Hence, the optimization problem can be stated as follows:

$$\text{Maximize } TH \quad (\mathcal{B}_1, \mathcal{B}_2, \dots, \mathcal{B}_{\mathcal{N}-1}, \mathcal{R}, RP) \tag{1}$$

$$\text{Subject to: } \sum_{i=1}^{\mathcal{N}-1} \mathcal{B}_i = \mathcal{K}, \tag{2}$$

$$1 \leq \mathcal{R} \leq \mathcal{N}, \tag{3}$$

$$\mathcal{B}_i \geq 0 \ for \ i = 1 \ to \ \mathcal{N}, \tag{4}$$

where $\mathcal{B}_i$ refers to a feasible buffer-slots allocation for buffer i, $\mathcal{N}$ determines the limitation on the total allowed capacity of repairmen $\mathcal{R}$, $\mathcal{K}$ refers to the total amount of buffer-slots that need to be distributed among the intermediate buffers, (4) indicates that each buffer i has to be a positive size value, and $TH(\mathcal{B}_1, \mathcal{B}_2,...,\mathcal{B}_{\mathcal{N}-1},\mathcal{R},RP)$ indicates the production line throughput and it is expressed a function in $\mathcal{B}_i,\forall i$ and the repair policy RP.

In the sequel, a meta-heuristic optimization approach that uses a discrete-event simulation model is introduced to provide a solution for the BAOP. This is done using a heuristic that employs a nonlinear threshold accepting function [20].

4. Solution methodology

The solution mechanism involves two phases, the first one estimates the throughput of the production line, while the second one uses a heuristic algorithm to optimize this throughput to optimally allocate the buffer resources and find the best repair policy. This generalized solution framework switches between the simulation of a production line and the optimization using a heuristic algorithm (i.e., Nonlinear Threshold Accepting Heuristic (NLTA)) as indicated by the flowchart in Figure (2). Particularly, the production line simulations are employed by the meta-heuristic to evaluate the quality of the generated solution.

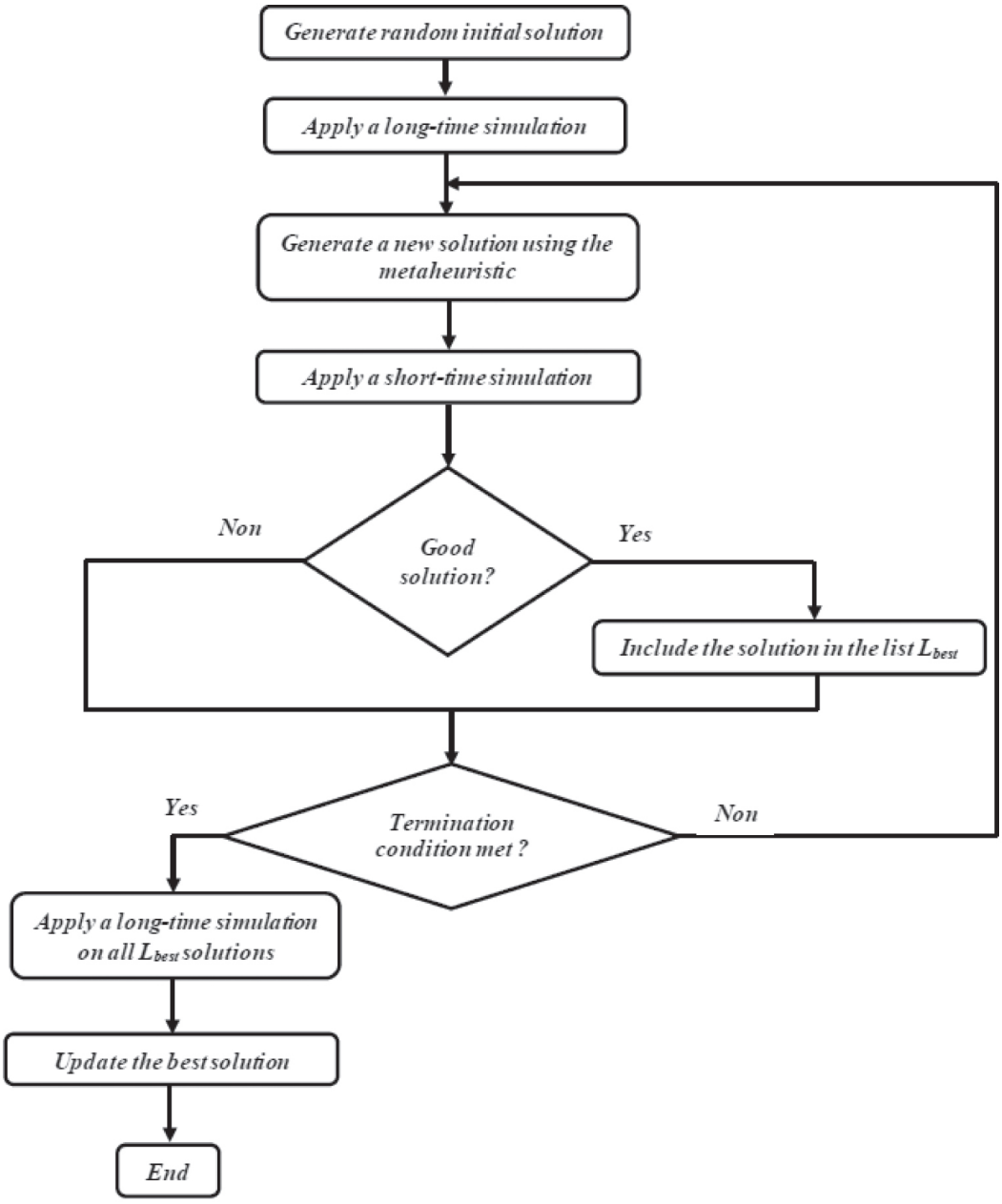

Figure 2. Layout of the proposed solution

The main challenge is the large computational time needed or consumed due to the generation of each solution using a long-horizon simulation. Therefore, a shorter simulation horizon is considered for the newly generated solutions to overcome this drawback. However, this degrades the accuracy of the solution since it relies on the number of successfully completed search episodes. To address this issue, a finite list of top performing solutions (i.e., L_{best}), that updates its content continuously, is considered during the optimization phase. Finally final longer horizon simulations will be done for the final solutions remaining in the list L_{best}. Then, they are evaluated against the main performance criterion to select the best solution.

4.1 Simulation model as throughput evaluation tool

The simulation concepts are employed to model and analyze complex systems which are not easily or can not be represented by analytical models. Further, these concepts can be used to observe the effects of adjusting the system parameters on the overall performance and can be employed as decision support tools in the design stages. The simulation of a system can be done in either continuous or discrete manner according to the variation form of the underlying states. Furthermore, the continuous flow simulation can be seen as a sequence of discrete infinitesimal segments [15]. An efficient and fast discrete-event continuous-flow simulation mechanism is adopted to estimate and approximate the throughput of a production line in [14]. This production line takes an unreliable serial form and it considers a limited number of repairmen. The numerical experimentation emphasized that the model is accurate and provides 3 or more times faster performance than, the conventional simulators. The proposed algorithm avoids extensive computational efforts related to the modeling of the individual units flow within the production system. In this paper, a continuous simulation model is considered and it is implemented using MATLAB software engine. Herein, a simulation model is briefly introduced and further details can be found in [14],[15].

4.1.1 Continuous flow model

The simulation model implemented in this work employs a production line of $\mathcal{N}$ un reliable stations and $(\mathcal{N}$ - 1) intermediate buffers. It computes the time elapsed between events for each machine and buffer. The simulation model considers some assumptions as follows

- The first and last machines avoid starving and blocking at any time, respectively.
- The machines operate at known processing times and the processing times may take different values.
- Any machine can only fail after a proper processing of one piece.
- The blocked or starved machines are not allowed to break down.
- The time-to-repair and time-to-failure are following well known distributions.

In the simulation model, the production continues until it is interrupted by an event. The service is disturbed due to different reasons as follows, for example when all machines are active and their production rates vary, then the buffer levels can be adjusted accordingly. However, machine $\mathcal{M}_i$ starves if the associated buffer $\mathcal{B}_{i\text{-}1}$ is idle (i.e., empty). This case happens if machine $\mathcal{M}_i$ has a production rate larger than that of machine $\mathcal{M}_{i\text{-}1}$ and hence $\mathcal{M}_i$ is forced an equivalent production rate as that of $\mathcal{M}_{i\text{-}1}$. Similarly, $\mathcal{M}_i$ falls in a blocked-status if the associated buffer $\mathcal{B}_i$ is full. This case happens if machine $\mathcal{M}_i$ is producing at a rate that is larger than that of machine $\mathcal{M}_{i+1}$. Thus, this machine $\mathcal{M}_i$ is forced to work at a reduced production rate equal to that of $\mathcal{M}_{i+1}$. Therefore, a machine may function in one of three different possible states: producing at its nominal production rate, producing at a reduced rate if the machine is starved or blocked, and non-producing or idle state if the machine is down. It is noted that, the variations in the rates of production will propagate towards the end and the start machines, as well in the production line. This will appear in the form of successive starved and blocked machines. Such phenomena can be detailed as follows

1. Machine $\mathcal{M}_i$ becomes blocked when the associated buffer $\mathcal{B}_i$ is full. Thus, the production rate of $\mathcal{M}_i$ falls to $\mathcal{R}_{i+1}$. Assume a series of blocked machines (i.e., $\mathcal{M}_{i\text{-}1}$,$\mathcal{M}_{i\text{-}2}$,...,$\mathcal{M}_{i\text{-}k}$), then a procedure is followed recursively upstream so that

$$\mathcal{R}_{i-m} = \min\{\mathcal{R}_{i-m+1}, \mathcal{R}_{\mathcal{M}_i-m}\} \quad for \quad m = 1, 2, \dots, k, \tag{5}$$

untill a non-full buffer $\mathcal{B}_{i\text{-}k\text{-}l}$ is found.

2. If buffer $\mathcal{B}_i$ becomes empty such that the production rates are forced to follow $\mathcal{R}_{i+1}$=$\mathcal{R}_i$ and $\mathcal{M}_{i+1}$ becomes starved. Hence, for the possible sequence

of starved machines (i.e., $\mathcal{M}_{i+2}$,$\mathcal{M}_{i+3}$,...), the following procedure is considered repetitively up stream

$$\mathcal{R}_{i+m} = \min\{\mathcal{R}_{i+m-1}, \mathcal{R}_{\mathcal{M}_i+m}\} \quad for \quad m = 1, 2, \dots, k, \tag{6}$$

untill a non-full buffer $\mathcal{B}_{i+k}$ is attained.

3. If a machine $\mathcal{M}_i$ breaks down, there will be no production rate for machine i (i.e., $\mathcal{R}_i$=0). This indicates that all the sequences of starved (blocked) machines will have null-production rates downstream (upstream).

4. Upon repairing machine $\mathcal{M}_i$, its production rate is reinstated to the maximum value $\mathcal{R}_i$. During this situation, the machines ($\mathcal{M}_{i-1}$,$\mathcal{M}_{i-2}$,...) or ($\mathcal{M}_{i+2}$,$\mathcal{M}_{i+2}$,...) are forced to be idle, once $\mathcal{M}_i$ is repaired, those machines work at their nominal production capacities or adopt those production rates of the successor or predecessor machines, accordingly.

The times of possible next events for buffer i are determined and categorized as follows:

- If a machine $\mathcal{M}_i$ functions at a faster rate compared with $\mathcal{M}_{i+1}$. Thus, buffer $\mathcal{B}_i$ needs a time $T_{\mathcal{B}_i}$ to fill in as indicated below

$$T_{\mathcal{B}_i} = \frac{\mathcal{B}_i - N_i}{\mathcal{R}_i - \mathcal{R}_{i+1}}, \tag{7}$$

  where $\mathcal{B}_i$ indicates the rated capacity of buffer i, N_i refers to the count of items in buffer i, and $\mathcal{R}_i$ represents the processing rate (i.e., $1/T_i$) of machine i.

- If machine $\mathcal{M}_i$ is producing at a lower production rate compared with machine $\mathcal{M}_{i+1}$, then $\mathcal{B}_i$ will clear its stock of the items during the following time

$$T_{\mathcal{B}_i} = \frac{N_i}{\mathcal{R}_{i+1} - \mathcal{R}_i}. \tag{8}$$

- If both $\mathcal{M}_i$ and $\mathcal{M}_{i+1}$ share the same production rate values, then buffer i will maintain its current storage level. Accordingly, $T_{\mathcal{B}_i}$ is set to a very large number (i.e., $T_{\mathcal{B}_i}$=∞).

The possible times of next events at machine $\mathcal{M}_i$ are determined and classified as follows:

- Machine $\mathcal{M}_i$ is operating at a maximum production rate capacity of $\mathcal{R}_i$. The time until the machine fails is computed using

$$T_{\mathcal{M}_i} = \frac{N_{F_i}}{\mathcal{R}_i}, \tag{9}$$

  where N_{F_i} denotes the count of processed items by each station $\mathcal{M}_i$ just before the failure.

- Assuming availability of a repairman, the time needed to repair an idle machine $\mathcal{M}_i$ is given by:

$$T_{\mathcal{M}_i} = -\frac{ln(u)}{\mu_i}, \tag{10}$$

  where u is a stochastic number taking a value in (0,1) and μ_i is the average throughput performance for machine $\mathcal{M}_i$. The time-to-repair is assumed to be random and takes an exponential form.

- Assuming that no repairman is available, the repair time is considered a high value (i.e., $T_{\mathcal{M}_i}$=∞). Accordingly, the production rate of machine $\mathcal{M}_i$ is $\mathcal{R}_i$=0.

The model of the series production line with the aforementioned features can be simulated as follows:

Step 1. Initialization: Number of machines and buffers, initial and maximum capacities of each buffer i, initial production rates of each machine i, and allowed simulation interval t_{max}.

(a) Assign the rated production rate of each machine $\mathcal{M}_i$ to $\mathcal{R}_i$ such that $\mathcal{R}_i = \mathcal{R}_{\mathcal{M}_i}$; for $i = 1,2,\dots,\mathcal{N}$, at $t = 0$.

(b) Update the production rates of starved machines downstream the production line using (6).

(c) Adjust the production rates of blocked stations upstream the production line using (5).

(d) Calculate for each buffer and machine, the time estimation of next event.

Step 2. Event-Time Allocation: Take note of the most recent time $\mathcal{T}$=t. Consider all buffers or machines and elect the closest imminent event to happen. Hence, adjust the clock accordingly such that

$$\min_{\forall \mathcal{M}_i, \mathcal{B}_i} \{T_{\mathcal{M}_i}, T_{\mathcal{B}_i}\} \tag{11}$$

Terminate the simulation process if $t > t_{max}$ and assign $t = t_{max}$. Finally, update the states of the buffers and machines.

Step 3. Event Routine: The following steps explains the event routine

(a) Detect the starved and/or blocked machine sequences which will be influenced by the event. Hence, update the list of failing machines, cumulative production rates, buffers will be influenced in those sequence, and the pending capacity of each buffer.

(b) Modify the production rates of the impacted machines.

(c) Evaluate the following event for each impacted component using ((7))-((10)).

(d) Go to **Step 2**.

4.1.2 Determining the number of replications

Herein, upon executing large number of long-horizon simulations, the number of replications is required to adequately reflect the average throughput performance μ (i.e., the output of the simulation model). Let $\mathcal{P}$ be a single statistic value that refer to the production line throughput obtained after one replication. A set of independent and identically distributed throughput values, following n independent experiments, is defined by the tuple ($\mathcal{P}_1$, $\mathcal{P}_2$, ..., $\mathcal{P}_n$). Hence the average value $\bar{\mathcal{P}} = (\mathcal{P}_1 + \cdots + \mathcal{P}_n)/n$ is considered approximately normally distributed if the number of the population or experiments n is large enough. A 100 (1-δ) % confidence range with n-1 degrees of freedom (i.e., with δ significance level), of the average performance μ is expressed as

$$\bar{\mathcal{P}} - t_{n-1,\delta/2} \times \frac{V_n}{\sqrt{n}} \leq \mu \leq \bar{\mathcal{P}} + t_{n-1,\delta/2} \times \frac{V_n}{\sqrt{n}}, \tag{12}$$

where $t_{n\text{-}1,\delta/2}$ refers to the standard student t-distribution quantile and V_n represents the standard deviation of the samples $\mathcal{P}_1$,...,$\mathcal{P}_n$.

In order to determine the adequate number of replications n, two associated parameters are needed, (1) the significance level δ and (2) the required precision d_{req} or equivalently the required length of the Confidence Interval. After n replications, the precision d is calculated as follows [12]:

$$d = 100 \times \frac{t_{n-1,\delta/2} \times \frac{V_n}{\sqrt{n}}}{\bar{\mathcal{P}}}. \tag{13}$$

The procedure used to determine the number of replications n is illustrated by Figure (3). Firstly, the user assigns an initial number of replications n, the level of significance δ, and the needed precision d_{req}. Then while running experiments or simulations, the precision variable d is calculated using (13) and compared with the desired precision d_{req}. If the obtained precision is greater, one more simulation (i.e., n=n+1) is allowed and the precision parameter is evaluated accordingly by taking into consideration the result of the recent simulation. This procedure is reinstated until the desired precision value d_{req} is obtained. Herein, the initialization is done as follows n=3, d_{req}=10% and δ=10%.

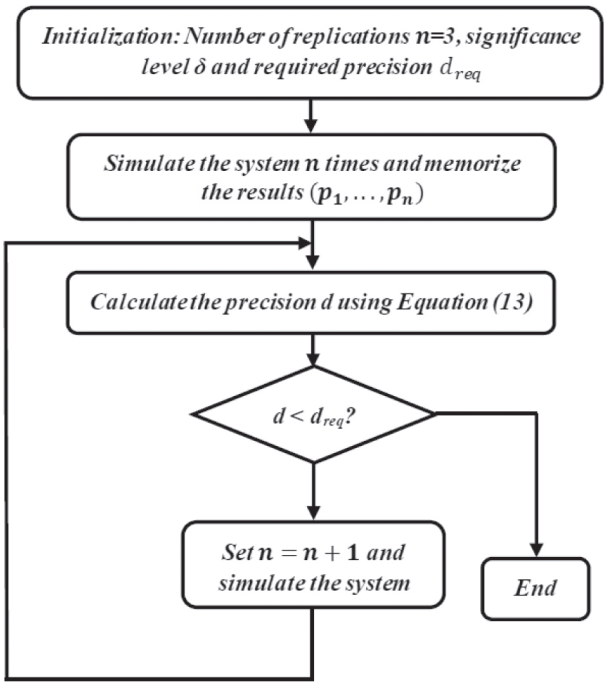

Figure 3. Determination of the number of replications

4.2 NLTA - heuristic solution

In the sequel, a metaheuristic approach that utilizes an accepting function with a nonlinear threshold criterion, namely NLTA is considered to maximize the throughput performance of the series production line [20]. This algorithm compares two feasible solutions, one is the current solution which is denoted by $\mathcal{S}$ and the other one is the neighboring solution which is referred to as $\mathcal{S}'$. The NLTA algorithm compares the two solutions and accepts the current one if it provides a better performance value (i.e., $\mathcal{Z}(\mathcal{S}') < \mathcal{Z}(\mathcal{S})$). This is done using a utility or cost function which will work as a performance index. Additionally, to improve the quality of the heuristic, it is advisable to use exploration criterion which prevents the search mechanism from tripping into a local maximum/minimum or simply exploit for the best solution all the time. Hence, a nonlinear structure which is inspired by the magnitude of the transfer function of a low-pass filter is considered as an accepting criterion. This fucntion passes the low frequency content of the signal and attenuate the high frequency content of the signal. Thus, the accepting function is expressed as follows:

$$\mathcal{H}(\vartheta) = \frac{1}{\sqrt{1+(\vartheta/\vartheta_0)^2}}, \tag{14}$$

where ϑ and ϑ0 denote the angular frequencies of the signal and cutoff pattern, respectively. These

values can be expressed in terms of the respective frequencies in Hz (i.e., f,f_0) such that $\vartheta=2\pi f$ and $\vartheta 0=2\pi f_0$.

In the solution, $\mathcal{H}(\vartheta) \leq 1$ and $\mathcal{H}(\vartheta)=0$ when $\vartheta\rightarrow\infty$. In addition to the above accepting rule, the solution will be accepted if $\mathcal{Z}(\mathcal{S}')\times\mathcal{H}(\vartheta)\leq\mathcal{Z}(\mathcal{S})$. This will help to better explore the solutions spaces.

4.2.1 Solution algorithm

The procedure of the NLTA heuristic solution [20], as in other cases such as SA, can introduce undesired bad solutions in a different manner. Initially, the angular frequency is set to a positive high value $\vartheta>0$ and this value will decrease by a fixed rate of $\Delta\vartheta$ during each search episode. This implies that, the nonlinear accepting function $\mathcal{H}(\vartheta)$ (14) takes an increasing pattern and consequently it will make it progressively hard to accept and introduce a worse solution outcome. There are cases where a new solution outcome $\mathcal{S}'$ that is of lower quality when compared with the neighboring solution $\mathcal{S}$ (i.e., the first accepting rule is violated $\mathcal{Z}(\mathcal{S}')>\mathcal{Z}(\mathcal{S})$) will be accepted. This happens when the value of the fraction $\mathcal{H}'=(\mathcal{Z}(\mathcal{S}))/(\mathcal{Z}(\mathcal{S}'))$ satisfies the inequality $\mathcal{H}(\vartheta) \leq \mathcal{H}'$ (i.e., the second accepting criterion). Finally, the search parameters ϑ_0 and $\Delta\vartheta$ are employed to control the convergence of the search mechanism. The following algorithm summarizes the steps of the heuristic solution [20]

- Step 1 Set the initial values of the search frequency and cutoff frequency ϑ and ϑ_0. Indicate the decrement rate $\Delta\vartheta$.
- Step 2 Define the desired number of search steps N_{iter}.
- Step 3 Collect a random feasible solution $\mathcal{S}_0$ and then assign $\mathcal{S}=\mathcal{S}_0$.
- Step 4 Loop when the total of iterations N_{iter} is not reached.
  (a) Find a new random and feasible solution $\mathcal{S}'$ from the solution space.
  (b) Compute the fraction $\mathcal{H}' = \mathcal{Z}(\mathcal{S}') / \mathcal{Z}(\mathcal{S})$.
  (c) Find $\mathcal{H}(\vartheta) = 1/\sqrt{1+(\vartheta/\vartheta_0)^2}$.
  (d) Check if $(\mathcal{H}' \geq 1)$ or $(\mathcal{H}' \geq \mathcal{H}(\vartheta))$, then accept that solution (i.e., $\mathcal{S}=\mathcal{S}'$).
  (e) Decrease the frequency of the accepting function (i.e., $\vartheta=\vartheta-\Delta\vartheta$).
- Step 5 Terminate the loop upon reaching N_{iter}.

4.2.2 Solution structure layout

Each solution is represented by two vectors $\mathcal{S}_1$ and $\mathcal{S}_2$. Each entry in vector $\mathcal{S}_1$ (i) $(i=1,\ldots,\mathcal{N}-1)$ denotes the size of buffer i and vector $\mathcal{S}_2$ (i) $(i=1,\ldots,\mathcal{N})$ stores the repair policy. Therefore, each entry in $\mathcal{S}_2$ (i) reflects the priority level of repairing each Machine $\mathcal{M}_i$. It is convenient to assume that, based on the production process, the highest and lowest repair priorities are assigned to machines '1' and '$\mathcal{N}$', respectively. Assume that a production line has 4 machines, where the repair policy follows $\mathcal{S}_2=(3,2,1,4)$. This implies that machine 3 has the highest repair priority ($\mathcal{S}_2$ $(3)=1$) followed by machine 2 with repair priority 2, machine 1 with repair priority 3, and finally machine 4 with repair priority 4. Accordingly, when machines 1 and 2 fail and acquire a repairman service, for example. Then, machine 2 will be repaired before machine 1. Finally, in order to generate an initial solution (i.e., initial solution vectors $\mathcal{S}_1$ and $\mathcal{S}_2$), the total value $\mathcal{K}$ is randomly distributed to ($\mathcal{N}$-1) buffers to get an initial solution vector $\mathcal{S}_1$. Then, the repair priorities are assigned randomly to obtain an initial solution vector $\mathcal{S}_2$. It is worth to note that, any two machines can not be given the same repair priority.

4.2.3 Solution generation

The next solution is generated by adjusting one entry of the current solution vectors (i.e., $\mathcal{S}_1$ or $\mathcal{S}_2$). Hence, one move is done during each step towards the near optimal solution. This procedure can be generally implemented as follows; (1) select two buffers randomly, where a random value $\mathcal{Q}$ is subtracted from the first buffer and added to the second one. (2) two machines are randomly selected and their repair priorities are swapped. The solution mechanism can be organized as follows

Pick q, a uniformly random distributed value picked from [0,1].

If $q > 0.5$ (apply a modification on the sizes of buffers)

- Choose randomly any two buffers i and r.
- Generate a random value $\mathcal{Q}$ from the interval $[0,\mathcal{B}_i]$.
- Set $\mathcal{B}_i=\mathcal{B}_i-\mathcal{Q}$ and $\mathcal{B}_r=\mathcal{B}_r+\mathcal{Q}$.

If $q \leq 0.5$ (apply a change in the repair policy)

- Select randomly any two machines i and r.
- Swap the repair priorities of the two machines.

Terminate the search process.

5. Illustrative simulation examples

All solution procedures and algorithms have been implemented using MATLAB software. The size of the list containing the best solutions (i.e., L_{best}) is defined to be 50. The same tuning procedure considered in [20] has been adopted to tune the NLTA search parameters. The total number of search iterations N_{iter} is set to 20,000. The remaining NLTA parameters ϑ, ϑ_0, and $\Delta\vartheta$ have been set to 30,40, and $\vartheta/N_{iter}=30/20000=0.0015$, respectively. In short-time runs, the simulation is stopped when the number of produced units reaches 4,000 and the replications number is defined to be 3. The mechanism shown in Figure (3) has been applied to determine the replications number to be used in long-horizon simulations. The obtained results showed that, at least 100 replications are needed to reach a precision d less than 10%, for a significance level $\delta=10\%$. The stopping criterion for long-horizon simulations corresponds to a maximum of 20,000 produced units.

5.1 Experiment 1

In this section, a simulation example proposed by [15] is employed to validate the effectiveness of the developed solution. It utilizes a series production line system composed of $(\mathcal{N}=10)$ unreliable stations and $(\mathcal{N}-1=9)$ buffers. The total amount of buffer slots to be distributed among $(\mathcal{N}-1)$ buffers is $\mathcal{K}=90$. The data of the machines in the production line are highlighted in (1). In this experiment, 8 different repair policies are applied as follows:

- Policy 1. Shortest repair time (i.e., min $1/\mu_i$).
- Policy 2. Longest repair time (i.e., max $1/\mu_i$).
- Policy 3. Shortest up-time (i.e., min $1/\lambda_i$).
- Policy 4. Longest up-time (i.e., max $1/\lambda_i$).
- Policy 5. Smallest count of parts-to-failure (i.e., min $1/(T_i \times \lambda_i)$).
- Policy 6. Longest count of parts-to-failure (i.e., max $1/(T_i \times \lambda_i)$).
- Policy 7. Smallest efficiency in isolation (i.e., min $\mu_i/(\mu_i+\lambda_i)$).
- Policy 8. Longest efficiency in isolation (i.e., min $\mu_i/(\mu_i+\lambda_i)$).

In the proposed NLTA, the neighbor solution is selected by modifying only the buffers allocations (the repair policy is supposed to be fixed) as follows:

- Choose randomly any two buffers i and r.
- Generate random value Q from the interval $[0,\mathcal{B}_i]$.
- Set $\mathcal{B}_i=\mathcal{B}_i-Q$ and $\mathcal{B}_r=\mathcal{B}_r+Q$.

Table 1 exhibits the best solution outcomes for 8 repair policies. As can be concluded from this table, the change in the policy will not result in any significant variations in the results, except the case when there is only one repairman. In this case, repair policies 4 and 8 provide the best production system performances. For most cases, 4 repairmen are needed to maximize the production line throughput.

5.2 Experiment 2

In this experiment, the machines failure rates are increased as illustrated in Table 2. Hence, the best NLTA optimization results for 8 repair policies are listed in Table 3. These outcomes reveal that, the minimum number of repairmen needed to maximize the average throughput of the production line is 5. When only one repairman is available, the repair policies 2 and 8 provide the highest throughput.

Table 1. Experiment 1: Production line (10 machines and 9 buffers)

Policy	Number of repairmen			
	1	2	3	4
1	3.9938	5.3673	5.7976	5.9104
2	4.1068	5.3768	5.8145	5.8947
3	4.0872	5.3971	5.8261	5.9411
4	4.2003	5.3777	5.8038	5.9138
5	3.8119	5.3463	5.7997	5.8081
6	3.9055	5.3517	5.8098	5.9174
7	4.121	5.4313	5.7967	5.9248
8	4.2361	5.4116	5.8259	5.8802

Table 2. Experiment 2: Production line (10 machines and 9 buffers)

	Machine									
Parameter	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_5$	$\mathcal{M}_6$	$\mathcal{M}_7$	$\mathcal{M}_8$	$\mathcal{M}_9$	$\mathcal{M}_{10}$
μ_i	0.65	0.61	1.02	1.31	1.29	1.45	0.9	0.91	0.95	0.8
λ_i	0.4	0.24	0.64	0.8	0.48	0.56	0.32	0.48	0.72	0.88
Processing rate $1/T_i$	20	18	16	14	10	11	15	17	19	21

Table 3. Experiment 2: Best results for various repair policies

Policy	**Number of repairmen**				
	1	**2**	**3**	**4**	**5**
1	3.1722	4.577	5.0709	5.1928	5.2243
2	3.3724	4.5972	5.0402	5.1875	5.2294
3	3.2869	4.6597	5.0466	5.1926	5.221
4	3.3052	4.6195	5.046	5.1976	5.1986
5	3.0515	4.6119	5.022	5.1846	5.2286
6	3.0187	4.6198	5.0552	5.213	5.2263
7	3.3156	4.6444	5.0244	5.2106	5.2333
8	3.4002	4.6814	5.0287	5.2192	5.2036

5.3 Experiment 3

The objective of this experiment is to optimize the repair policy and the buffer allocation when the number of repairmen is limited to 1. The NLTA is applied using the neighboring solution generation presented in Section (4.2.3). The best result obtained by the NLTA is 3.527 with RP=(4,5,3,2,9,10,7,1,6) and $\mathcal{B}$=(4,1,12,6,1,59,8,8).

6. Conclusion

This paper considers a serial production line consisting of $\mathcal{N}$ unreliable workstations associated with $\mathcal{N}$-1 buffers. An integrated model is developped to solve the buffer allocation and repair policies optimization problem. While the majority of existing works consider that the only parameters to find are the buffer sizes, the proposed model aims to optimize simultaneously buffer sizes, number of repairmen and repair policies. A combined local search mechanism and a continuous flow simulation model have been adopted to allocate the total amount of buffer-slots to the buffers and elect the associated repair strategies. The search mechanism employed a heuristic that is based on a nonlinear threshold accepting function. The numerical simulations are conducted using three scenarios with 8 different repair policies to test the effectiveness of the proposed approach. The validation results showed that:

- the proposed approach can be employed to find a near optimal solution in a limited and acceptable time with reasonable computational and implementation efforts;
- the overall production system performance can be improved by selecting the best repair policy.
- the change in the policy will not result in any significant variations in the results if the number of repairmen is high.

Future work will concern the improvement of this model by integrating maintenance aspects in the optimal design of serial production systems.

Funding

The authors would like to thank the Canadian Natural Science and Engineering Research Council for supporting the project through the Discovery Development Grants program (DDG-2021-00017).

References

[1] Can B. and Heavey, C. "A comparison of genetic programming and artificial neural networks in metamodeling of discrete-event simulation models", Computers, Operations Research, Vol. 39, No. 2, pp. 424-436, 2012.

[2] Dallery, Y., David, R. and XIE, X.L., "An efficient algorithm

for analysis of transfer lines with unreliable equipments and finite buffers", IIE transactions, Vol. 20, No. 3, pp. 280-283, 1988.

[3] Demir, L., Diamantidis, A., Eliiyi, D. T., O'Kelly, M. E. J., Papadopoulos, C. T., Tsadiras, A. K., and Tunali, S., "A comparison of three search algorithms for solving the buffer allocation problem in reliable production lines", IFAC Conference on Manufacturing Modelling, Management, and Control, Vol. 1626-1631, St. Petersburg, Russia, 2013.

[4] Diamantidis, A. C. and Papadopoulos, C. T., "A dynamic programming algorithm for the Buffer Allocation Problem in homogeneous asymptotically reliable serial production lines", Mathematical Problems in Engineering, Vol. 2004, No. 3, pp. 209-223, 2004.

[5] Diamantidis A.C., Papadopoulos C.T. and Heavey C, "Approximate analysis of serial flow lines with multiple parallel-machine stations", IIE Transactions, Vol. 39, No. 4, pp. 361-375, 2007.

[6] Diamantidis A., Lee J.-H., Papadopoulos C.T., Li J. and Heavey C., "Performance evaluation of flow lines with non-identical and unreliable parallel machines and finite buffers", International Journal of Production Research, 2020, DOI: 10.1080/00207543.2019.1636322.

[7] Gershwin S.B. and Schick I.C., "Modeling and analysis of three-stage transfer lines with unreliable machine and finite buffers", Operations Research, Vol. 31, pp. 354-380, 1983.

[8] Gershwin, S.B., "An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking", Operational Research, Vol. 35, No. 2, pp. 291-305, 1987.

[9] Gershwin, S.B. and Schor, J.E., "Efficient Algorithms for Buffer Space Allocation", Annals of Operations Research, Vol. 93, No. 1-4, pp. 117-144, 2000.

[10] Helber S., Katja S. and Raik S., "Setting Inventory Levels of CONWIP Flow Lines via Linear Programming", BuR - Business Research, Vol. 4, No. 1, pp. 1-18, 2011.

[11] Hillier, M.C., "Characterizing the optimal allocation of storage space in production line systems with variable processing times", IIE Transactions, Vol. 32, No. 1, pp. 1-8, 2000.

[12] Hoad K., Robinson S. and Davies R., "Automated selection of the number of replications for a discrete-event simulation", The Journal of the Operational Research Society, Vol. 61, No. 11, pp. 1632-1644, 2010.

[13] Kouikoglou V.S. and Phillis Y.A., "An exact discrete-event model and control policies for production lines with buffers", IEEE Transactions on Automatic Control, Vol. 36, No. 5, pp. 515-527, 1991.

[14] Kouikoglou V.S., "Discrete Event Modeling and Optimization of Unreliable Production Lines with Random Rates", IEEE Transactions on Robotics and Automation, Vol. 10, No. 2, pp. 153-159, 1994.

[15] Kouikoglou, V. S., and Phillis, Y. A., "Hybrid Simulation Models of Production Networks", Kluwer Academic/ Plenum Publishers, New York, 2001.

[16] Kose, S. Y. and Kilincci, O., "Hybrid approach for buffer allocation in open serial production lines", Computers & Operations Research, Vol. 60, pp. 67-78, 2015.

[17] Li, J. and Meerkov, S.M., "Production Systems Engineering", Springer, New York, NY, USA, 2009.

[18] Li J., "Continuous improvement at Toyota manufacturing plant: Applications of production systems engineering methods", International Journal of Production Research, Vol. 51, No. 23-24, pp. 7235-7249, 2013.

[19] Lutz C M, Davis K R. and Sun M., "Determining buffer location and size in production lines using tabu search", European Journal of Operational Research, Vol. 106, pp. 301-316, 1998.

[20] Nahas, N. and Nourelfath, M., "Non-linear threshold accepting meta-heuristic for combinatorial optimization problems", International Journal of Metaheuristics, Vol. 3, No. 4, pp. 265-290, 2014.

[21] Nahas, N. and Nourelfath, M., "Joint optimization of maintenance, buffers and machines in manufacturing lines", Engineering Optimization, Vol. 50, No. 1, pp. 37-54, 2018.

[22] Nahas, N. "Buffer allocation, equipment selection and line balancing optimisation in unreliable production lines", European Journal of Industrial Engineering, Vol. 14, No. 2, pp. 217-246, 2020.

[23] Narasimhamu, K. L., Reddy, V. V., and Rao, C. S. P., "Optimization of Buffer Allocation in Manufacturing System Using Particle Swarm Optimization", International Review on Modelling and Simulations, Vol. 8, No. 2, pp. 212, 2015.

[24] Shi, L. and Men, S., "Optimal buffer allocation in production lines", IIE Transactions, Vol. 35, No. 1, pp. 1-10, 2003.

[25] Vergara H.A and Kim D.S., "A new method for the placement of buffers in serial production lines", International Journal of Production Research, Vol. 47, No. 16, pp. 4437-4456, 2009.

[26] Weiss, S., Matta, A., and Stolletz, R., "Optimization of buffer allocations in flow lines with limited supply", IISE Transactions, Vol. 50, No. 3, pp. 191-202, 2018.

[27] Whitley D, Kauth J., "GENITOR: a different genetic algorithm", Technical Report CS-88-101, Colorado State University, 1988.

[28] Yan, F.-Y., Wang, J.-Q., li, Y. and Cui, P.-H., "An Improved Aggregation Method for Performance Analysis of Bernoulli Serial Production Lines", IEEE Transactions on Automation Science and Engineering, vol. 18, No. 1, pp. 114-121, 2020.

[29] Koyuncuoğlu M.U. and Demir L., "Buffer capacity allocation in unreliable production lines: An adaptive large neighborhood search approach", Engineering Science and Technology, vol. 24, No. 2, pp. 299-309, 2021.

[30] Herps K., Dang Q.-V, Martagan T. and Adan I., "A simulation-based approach to design an automated high-mix low-volume manufacturing system", Journal of Manufacturing Systems, vol 64, pp 1-18, 2022.

[31] Demir L. and Koyuncuoğlu M.U., "The impact of the optimal buffer configuration on production line efficiency: A VNS-based solution approach", Expert Systems with Applications, vol. 172, 2021.

[32] Vasquez J.O., Gonzalez S.H., Vasquez J.I.H, Fernandez V.F. and Cancino de la Fuente C.I., "Buffer allocation problem in a shoe manufacturing line: A metamodeling approch", Revista Facultad de Ingeniera, Universidad de Antioquia, No 103, pp. 175-185, 2022