Engineering Science and Technology, an International Journal 24 (2021) 299–309

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Full Length Article

Buffer capacity allocation in unreliable production lines: An adaptive large neighborhood search approach

Mehmet Ulaş Koyuncuoğlu [a], Leyla Demir [b,*]

[a] Pamukkale University, IT Department, Camlaralti District, Kinikli Campus, Universite Road, Denizli 20160, Turkey

[b] İzmir Bakırçay University, Faculty of Engineering and Architecture, Department of Industrial Engineering, Gazi Mustafa Kemal District, Kaynaklar Road, Seyrek/Menemen, Izmir 35665, Turkey

ARTICLE INFO

Article history:
Received 20 February 2020
Revised 1 July 2020
Accepted 18 August 2020
Available online 4 September 2020

Keywords:
Production lines
Buffer allocation problem
Throughput maximization
Adaptive large neighborhood search
Initialization procedures

ABSTRACT

The design of a production line directly affects the system performance which is usually measured by its throughput. The problem involving determination of the optimal capacity and location of the buffers in a production line is known as the buffer allocation problem (BAP). Due to the difficulties such as the NP-hard structure of the problem and not being able to be defined the throughput of the line in terms of the buffer capacities algebraically, meta-heuristic search algorithms are widely used to solve the BAP. In this study, an adaptive large neighborhood search (ALNS) algorithm is proposed to solve the BAP for throughput maximization in unreliable production lines. Different from the literature, for the first time, ALNS algorithm is employed to solve the problem of designing a production line. For this purpose, two different removal-insertion operator pairs are proposed and employed in an adaptive way by considering the nature of the problem. Moreover, a new initialization procedure based on the well-known storage bowl phenomenon concept is proposed to reduce the search effort. Performance of the proposed algorithm was tested on the existing benchmark instances. A computational study demonstrated the benefits of not only the adaptive mechanism embedded into the proposed algorithm but also the proposed initialization procedure.

1. Introduction

Production terminologically means generating output using inputs through the necessary processes according to a predefined schedule. Throughput is one of the key performance indicators for production lines. The throughput of a production line depends on the system characteristics such as processing times, buffer capacities, and machine reliabilities. If the machines on a production line fail, one or more machines will be starved or blocked. In order to reduce the cumulative negative effects of the status of starving/blocking along the line, it is necessary to allocate the buffers between the machines. Unlimited buffer could not be used due to both physical space and cost constraints. Also, buffering leads to high levels of work-in-processes (WIP) and in turn operating costs increase. Because of these conflicts, finding the optimum distribution of the buffers to obtain the maximum throughput is an important design problem in production systems and it is known as the buffer allocation problem (BAP).

The BAP is in the class of nonlinear integer knapsack problems [1,2]. As indicated by Chow [3] there is no closed form to calculate the throughput of the line in terms of buffer capacities. Because of that an evaluative algorithm which is used to calculate the throughput of the line, and an optimization algorithm used for searching the best buffer configuration are employed iteratively to solve the BAP. Moreover, the BAP has a stochastic nature due to different processing times and/or random machine failures in the line, and it is known as a NP-hard combinatorial optimization problem. In addition, when the problem size increases the number of feasible solutions also increases exponentially. Because of these features of the problem, meta-heuristic search algorithms are widely used to solve the BAP. These algorithms include genetic algorithm (GA) [4,5], simulated annealing (SA) [6–8], tabu search (TS) [9–11], degraded ceiling (DC) [7], artificial neural network (ANN) [12,13], ant colony optimization (ACO) [8,14] and particle swarm optimization (PSO) [15]. For more information on the solution methods and the formulations of the BAP the reader can refer to the recent review study presented by Weiss et al. [16].

* Corresponding author.
E-mail addresses: ulas@pau.edu.tr (M. U. Koyuncuoğlu), leyla.demir@bakircay.edu.tr (L. Demir).

Peer review under responsibility of Karabuk University.

https://doi.org/10.1016/j.jestch.2020.08.012
2215-0986/© 2020 Karabuk University. Publishing services by Elsevier B.V.

In recent years, some researchers hybridized the meta-heuristic algorithms to enhance the search efficiency. Xuemei et al. [17] solved the BAP and transfer line balancing problem simultaneously using a hybrid GA-PSO algorithm. Kose and Kilincci [18] hybridized non-dominated sorting GA (NSGA-II) and SA algorithms for solving the BAP with two different objectives: throughput maximization and total buffer capacity minimization. Similarly, Motlagh et al. [19] used NSGA-II and non-dominated ranked GA (NRGA) to solve the BAP with multiple objectives: throughput maximization, total buffer capacity minimization and total cost minimization. Dolgui et al. [20] examined multi-criteria optimization problem (throughput maximization, and minimization of capital and inventory costs) for series–parallel unreliable production lines. Zandieh et al. [15] compared the performance of the PSO and GA for solving the BAP with respect to preventive maintenance planning in unreliable production lines and it was stated that PSO was more successful than GA in terms of the solution time.

Besides the meta-heuristic search algorithms there were some problem specific heuristic algorithms proposed to solve the BAP. Li et al. [21] developed a fast algorithm based on a local search for throughput maximization. Shi and Gershwin [22] applied a segmentation approach to solve the BAP for profit maximization. Weiss et al. [23] proposed a rule-based search algorithm to solve the BAP for total buffer capacity minimization under the limited supply assumption. A gradient-based search algorithm was proposed by Liberopoulos [24]. The author investigated the problem under the installation buffer, echelon buffer and CONWIP policies. Xi et al. [25] proposed a decomposition-based method to solve the BAP for long series-parallel unbalanced production lines with the objective of minimizing WIP.

Lastly, in recent years, a few mathematical programming approaches were proposed to solve the BAP. Pedrielli et al. [26] proposed a mathematical programming approach combined with simulation to solve the BAP and task allocation problem simultaneously. Lopes et al. [27] developed an iterative decomposition procedure based on mathematical models of mixed-model asynchronous assembly lines to solve the BAP for throughput maximization.

Even if the recent studies deal with the BAP with multiple objectives and/or employ hybrid meta-heuristics, the problem still needs to be investigated stand-alone due to its difficult nature. Since the throughput improvement is still an important research area for both academy and industry, this study focuses on throughput maximization in production lines. Different from the previous studies in the BAP literature, an adaptive large neighborhood search (ALNS) algorithm is proposed to solve the problem. ALNS is the extension of the large neighborhood search (LNS) algorithm, proposed by Shaw [28], to solve the combinatorial optimization problems such as vehicle routing [29,39], multiple traveling repairman [40], order-batching [41], pollution-routing problems [42–45], and berth allocation-quay crane assignment [46]. It is well known from the literature, the ALNS algorithm is very successful in solving the VRPs. However, the studies on the performance analysis of the ALNS algorithm for other combinatorial optimization problems are very limited. To fill this gap in the literature, our aim in this study is to test the performance of the ALNS algorithm for a production line design problem, i.e. BAP. To the best of the knowledge of the authors, the ALNS algorithm has not been employed to solve the BAP before. As it can be seen from the BAP literature, neighborhood-based solution methods like TS yield very good results in solving the BAP (see [10,11]). Considering the great potential of solving combinatorial optimization problems successfully, and it's simple and neighborhood-based structure of the ALNS it is expected that ALNS may yield good-quality solutions for the BAP. In this study, to be able to employ the ALNS algorithm in solving the BAP new removal-insertion operators are proposed and applied in an adaptive way. Moreover, a new initialization procedure based on the well-known storage bowl phenomenon [47] is proposed to enhance the convergence speed of the proposed algorithm. This strategy is used to obtain the optimal buffer configurations for reliable production lines with variable processing times before. Different from the literature, this idea is employed as an initialization procedure for unreliable production lines for the first time.

In this context, there are three major contributions of this study. Different from the literature, for the first time, ALNS algorithm is employed to solve the problem of designing a production line other than vehicle routing problems which are solved by ALNS successfully. Second, two different removal-insertion operator pairs are introduced to the literature and employed in an adaptive way to be able to solve the problem effectively. Finally, a new initialization procedure is proposed and embedded into the ALNS algorithm to improve the search efficiency.

The remainder of this paper is organized as follows. The problem definition is given in the next section. The details of the proposed algorithm are explained in Section 3. The numerical results are provided and discussed in Section 4. Finally, the study is summarized, and some future research directions are given in Section 5.

2. Problem definition

In this study, the BAP is solved for unreliable production lines. In unreliable production lines, the machines are subject to breakdown. Fig. 1 shows a production line involving K-machines (i = 1, 2, …, K) in series and K-1 buffer areas (i = 2, …, K) which have finite capacities. In Fig. 1, the squares denote the machines ($M = M_1, M_2, \ldots, M_K$) and the circles denote the buffers ($B = B_2, B_3, \ldots, B_K$). The parts on a production line are processed sequentially on the first machine, then on the second machine and finally, on the Kth machine, so all parts are processed in the same order on all machines. The failure, repair, and the processing rates of the machines are depicted by β, r, and μ, respectively. In this study, the processing times of the machines are assumed to be deterministic which is one-time unit. The machines may fail randomly and the mean time between failures and the mean time to repairs are assumed to be geometrically distributed. Generally, in production lines, it is assumed that there is an unlimited buffer supply in front of the first machine and there is unlimited storage area after the last machine. Because of that, the first machine is never starved, and the last machine is never blocked. Moreover, the transfer times between the machines are assumed to be zero and it is assumed that there is no setup time.

In this study, the performance measure of such a production line, i.e. throughput of the line, is obtained by the decomposition method proposed by Gershwin [48]. The main idea of the decomposition method is to decompose the K-machine line into K-1 sub-lines and calculates the parameters of each sub-line so as to the throughput values are approximated to the same value. The values of the machine parameters in each sub-line are calculated by an iterative algorithm proposed by Dallery et al. [49] and called as DDX algorithm.

The BAP under the above assumptions is solved for throughput maximization. The mathematical model of the problem is given as follows:

max $Th(B)$

$$\sum_{i=2}^{K} B_i = N \qquad (1)$$

B_i nonnegative integer (i = 2, …, K)

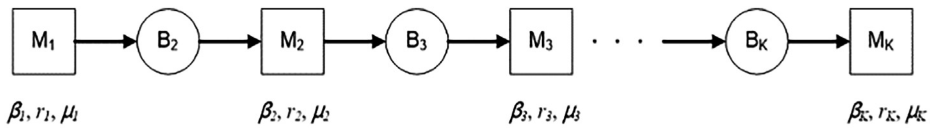

Fig. 1. *K*-machine production line.

In this formulation, B_i denotes the capacity of each buffer that should be nonnegative integers (i = 2, ..., K), $Th(B)$ denotes the throughput value obtained by the buffer configuration B, and N is the constant total buffer capacity. As it is stated by Demir et al. [50] the total number of possible buffer configurations for this problem can be calculated as follows:

$$\binom{N+K-2}{K-2} = \frac{(N+1)(N+2)\cdots(N+K-2)}{(K-2)!} \quad (2)$$

As it can be observed from Eq. (2), when N and K increase the total number of feasible solutions increases exponentially. For instance, if the production line involves ten machines and the total buffer capacity to be allocated is 50, then the total number of feasible solutions becomes 1,916,797,311. This small-sized example shows the computational difficulty of the problem. To overcome this difficulty, meta-heuristic approaches are widely used to solve the BAP. In this study, an ALNS algorithm, a neighborhood based meta-heuristic search algorithm, is proposed to solve the BAP for the first time in the literature. The following section provides the details of the proposed ALNS algorithm for solving the BAP in unreliable production lines under the above conditions.

3. Proposed ALNS algorithm

LNS is a meta-heuristic search method developed by Shaw [28] to solve the combinatorial optimization problems and thereafter it is successfully employed to solve many problems such as vehicle routing [29–39] and pollution-routing problems [42–45]. The main idea of the basic LNS is to improve the feasible initial solution by progressively employing alternative removal (destroy) and insertion (repair) operators. A removal operator removes a part of the current solution stochastically, while an insertion operator reinserts the partly removed solution to construct a new solution. The algorithm systematically applies these operators during the search process.

Many removal operators such as *random, worst* [30,34], *cluster* [40], *Shaw* [29], *request graph* [31,32], *time-based, neighbor graph* [31,32] and many insertion operators such as *greedy, regret* [30,37,38], *zone, noise function* have been employed for different problem types in the literature [35,39]. These removal and insertion operators can be implemented in different ways and/or various combinations depending on the type of the problem.

ALNS is an extension of the LNS algorithm which does not commit to one removal and insertion operator. Instead of that ALNS chooses operator pairs from a pool of heuristics at each iteration. There are many successful applications of ALNS for different problems in the literature [35,36,40,46,51]. In this study, we have adapted the ALNS algorithm for solving the BAP in production lines for the first time. In this context, new removal-insertion operator pairs are proposed and employed in an adaptive way to enhance the efficiency of the search process. Moreover, in addition to starting with a random solution, three different initialization procedures are proposed to reduce the search effort. Also, a new strategy is proposed to escape the local optimum. The details of the proposed solution procedure are given in the following subsections.

3.1. Proposed initialization procedures

Within the scope of this study, the effect of the initial solution on the convergence of the proposed algorithm has been investigated. For this purpose, in addition to a randomly generated solution (called as ALNS-Random in the remainder of this paper), the performances of three different initial solution procedures are tested. These procedures are the method proposed by Papadopoulos and Vidalis [52], the well-known storage bowl phenomenon developed by Hillier et al. [47], and the Smoothed Bowl Phenomenon method proposed by the authors.

The method proposed by Papadopoulos and Vidalis [52] suggests distributing more buffers in the middle of the line but also considers the bottleneck machines while distributing the buffers. For this purpose, the machines are ranked according to their mean effective service rates determined by Eq. (3), and then some predefined distribution rules are employed. The details of the method can be found in Papadopoulos and Vidalis [53]. In the remainder of this study, the proposed ALNS algorithm starting with this procedure is referred to as ALNS-PaVi. In this study, we employed this method to obtain a good initial solution.

$$\rho_i = \frac{\mu_i r_i}{\beta_i + r_i} \quad (3)$$

The main idea of the storage bowl phenomenon is to distribute more buffers in the middle of the line and less buffers towards the beginning and the end of the line to maximize the throughput of the line. Since the shape of distribution looks like an inverted bowl, this allocation is called also 'inverted bowl phenomenon'. In the remainder of this paper, the proposed algorithm starting with the solution generated by the storage bowl phenomenon is referred to as ALNS-BowlPh. It should be noted that this idea is used to obtain the optimal buffer configurations for reliable lines with variable processing times before. In this study, for the first time, we have adapted this strategy as an initialization procedure for unreliable production lines. We applied the storage bowl phenomenon according to the following formula:

$$B = \{B_2, 2B_2, 3B_2, \cdots, 3B_2, 2B_2, B_2\} \quad (4)$$

In this formula B_2 (the first buffer location) is calculated by dividing the total buffer capacity (N) to sum of all B_i's. It should be noted that the elements of the obtained buffer configuration having non-integer values are rounded up the next integer value. Then the total buffer capacity constraint is checked. If the obtained buffer configuration does not satisfy the total buffer capacity constraint the excess elements are reduced from the beginning and the end of the locations in the current buffer configuration. On the other hand, to be keep the total buffer capacity constant, if it is needed to add extra elements these values are added to the interior buffer locations in the current configuration.

In the third initialization procedure, the storage bowl phenomenon method is combined with the uniform distribution of the buffers. After generating a buffer configuration according to the storage bowl phenomenon, a second configuration is generated by distributing total buffers into all buffer areas equally. Then two configurations are combined according to the formula below (Eq. (5)). In this formula, 'BowlPh' represents the configuration generated by the storage bowl phenomenon method, 'Unif' represents the configuration generated by uniformly, and '*round*' represents the rounding operator. The main idea of employing this strategy is to obtain a better distribution of the buffers considering the advantages of both strategies. It should be noted that while applying the uniform operator if the total number of buffer capacity cannot be divided to K-1 exactly, the residual values are added to the middle buffer locations. Numerical results also showed that this initialization procedure yields quite good results as it is compared to the other initialization procedures (see Section 4).

$$\mathrm{ALNS-SBowlPh} = round\left(\frac{\mathrm{BowlPh}+\mathrm{Unif}}{2}\right) \tag{5}$$

The proposed ALNS algorithm starting with this initialization procedure is referred to as ALNS-SBowlPh in the remainder of this paper. It should be noted that when generating initial buffer configuration, if it is needed, the buffer values are rounded to integer values for all considered initialization procedures.

3.2. Removal and insertion heuristics

In this study, two new removal-insertion operator pairs are proposed to adapt the ALNS algorithm for solving the BAP. The first heuristic is based on the location and value of the buffers and the second one is based on the status of the machines.

3.2.1. Location-based removal and insertion heuristic

These operators are constructed based on the location and the value of the buffers. In this heuristic, first a buffer configuration is divided into sub locations called as *Left Edge Elements (LEE)*, *Right Edge Elements (REE)*, and *Centre Elements (CE)* using the following equations (Eqs. (6)–(8)):

$$L = round(\frac{K}{4})$$

$$B_{LEE} = [B_2, B_{L+1}] \tag{6}$$

$$R = abs(L - K)$$

$$B_{REE} = [B_{R+1}, B_K] \tag{7}$$

$$B_{CE} = [B_{L+2}, B_R] \tag{8}$$

In Eqs. (6)–(8), *round* stands for the rounding operator, and *abs* represents the absolute value operator. Then the removal and insertion operators are employed according to the rules depending on the location and the values of the buffers as given in Fig. 2. In Fig. 2, B denotes the current solution and B^{new} denotes the temporary new solution. This heuristic allows us to smooth the buffer configurations when the configurations have more wavy structure especially in the case of random initialization. For example, assume that we are dealing with the BAP in a 5-machine line and we have totally 31 buffers to be allocated (see Fig. 3). Assuming the current buffer configuration is $B = \{5, 9, 7, 10\}$, using the Eqs. (5)–(7), the elements of this configuration is determined as $B_{LEE} = B_2$, $B_{CE} = [B_3, B_4]$, and $B_{REE} = B_5$, respectively. As seen in Fig. 3, the maximum value of buffer is located at the last location, i.e. B_5 having the buffer capacity 10, and the minimum is at the first location, i.e. B_2 having the buffer capacity 5. After applying the Condition (1) (see Fig. 2) to the current buffer configuration the new buffer configuration is obtained as $B^{new} = \{7, 9, 7, 8\}$.

Condition 1:
if $(((\min(B) \in B_{LEE})$ OR $(\min(B) \in B_{REE}))$ AND $((\max(B) \in B_{LEE})$ OR $(\max(B) \in B_{REE})))$
$\min(B)' = \lfloor(\min(B) + \max(B))/2\rfloor$;
$\max(B)' = \lceil(\min(B) + \max(B))/2\rceil$;
$B^{new} = B$;
Condition 2:
if $((\min(B) \in B_{CE})$ AND $(\max(B) \in B_{CE}))$
$\min(B)' = \lfloor(\min(B) + \max(B))/2\rfloor$;
$\max(B)' = \lceil(\min(B) + \max(B))/2\rceil$;
$B^{new} = B$;
Condition 3:
if $((\min(B) \in B_{CE})$ AND $((\max(B) \in B_{LEE})$ OR $(\max(B) \in B_{REE})))$
$\min(B)' = \lceil(\min(B) + \max(B))/4\rceil$;
$\max(B)' = \lfloor 3(\min(B) + \max(B))/4\rfloor$;
$B^{new} = B$;
Condition 4:
if $((\max(B) \in B_{CE})$ AND $((\min(B) \in B_{LEE})$ OR $(\min(B) \in B_{REE})))$
$\min(B)' = \lceil(\min(B) + \max(B))/4\rceil$;
$\max(B)' = \lfloor 3(\min(B) + \max(B))/4\rfloor$;
$B^{new} = B$;

Fig. 2. The rules for the proposed location-based removal/insertion operators.

3.2.2. Machine status-based removal and insertion heuristic

This heuristic is constructed based on the status of the machines in the line. In a production line, if all buffers have infinite capacity, the maximum throughput is equal to the throughput value of the machine which has the lowest isolated efficiency in the line that is calculated by Eq. (3). Therefore, to maximize the throughput to the highest possible level, machine that have least mean effective service rate never should be starved. The main aim of this heuristic is to distribute the buffers among the machines to reduce the starving/blocking probabilities of the machines.

This heuristic is employed as follows. Two different buffer values (B_i, B_j) are randomly selected from the current buffer configuration (i, j; $i \neq j, j > i$) and they are summed as in Eq. (9). The total mean effective service rate (ρ) of the machines in front of these buffers are calculated by Eq. (10). Then, the new values of the buffers at these locations are calculated using Eqs. (11) and (12).

$$B_{SUM} = B_i + B_j \tag{9}$$

$$\rho_{SUM} = \rho_{(i)} + \rho_{(j)} \tag{10}$$

$$B_i' = \lfloor\frac{B_{SUM}\,\rho_{(i)}}{\rho_{SUM}}\rfloor \tag{11}$$

$$B_j' = \lceil\frac{B_{SUM}\,\rho_{(j)}}{\rho_{SUM}}\rceil \tag{12}$$

For example, again assume that we have a current buffer configuration $B = \{5, 9, 7, 10\}$ for a 5-machine line and there are 31 available total buffers (see Fig. 4). If the randomly selected buffer locations are B_2 and B_4, then to employ this heuristic the mean effective service rates of the second and fourth machines should be calculated. After employing the Eqs. (10)–(12) to the data in Table 1 (see Section 4), the new buffer configuration is obtained as $B^{new} = \{6, 9, 6, 10\}$.

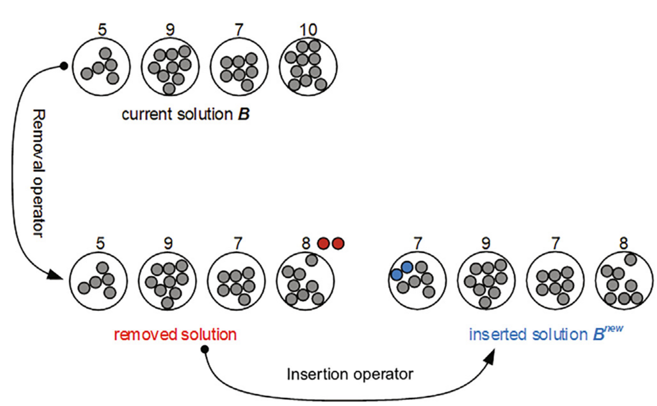

Fig. 3. The proposed location-based removal-insertion operators for solving the BAP: 5-machine line example.

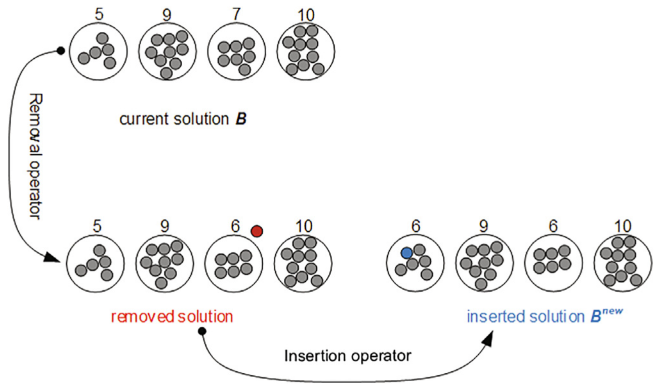

Fig. 4. The proposed machine status-based removal-insertion operators for solving the BAP: 5-machine line example.

3.3. Adaptive mechanism

The adaptive mechanism aims to choose a removal-insertion heuristic in a way that accounts for their outcomes at previous iterations. The removal-insertion operator pair which improves the throughput rate is recorded at each iteration. In this way, the probability of re-selection of the same operator pair for the next iteration is enhanced. The selection is done using a roulette wheel method. In this manner, the proposed ALNS algorithm is employed to solve the BAP adaptively.

3.4. Increment-Decrement strategy

The Increment-Decrement (IncDec) strategy is proposed for escaping the local optima during the search process. If a local optimum (called *l-opt*) is reached, then the IncDec strategy is employed to obtain new candidate solutions. In this strategy, two buffer location pairs ((B_i, B_j), ($i \neq j$) and $i, j \in (2, \ldots, K)$) are selected in order and the amount of the first selected buffer capacity is increased by a certain amount while the other one is decreased by the same amount. The value of this amount is determined at the initialization phase of the algorithm and it is systematically reduced to search deeply as the search progresses. The IncDec value is set to 1 for the benchmark instance having 5-machine line and it is set to 2 for the problems involving 9 and 10 machines. This value is set to 1% of the total buffer capacity for the large-sized instances as proposed by Demir et al [11]. It should be noted that the IncDec strategy is implemented once after the initialization procedure in order to quickly obtain the optimum solution for small-sized problems. After evaluating all neighborhoods in this way, the buffer configuration producing the best throughput value is chosen as the new solution for the next iteration.

3.5. Acceptance criterion

In the proposed ALNS algorithm not only the improved solutions are accepted but also the worst solutions can be accepted on a criterion based on the SA algorithm. A solution producing a better result than the known best result during the search process is always accepted. On the other hand, the worst solutions are accepted with the probability ν (Eq. (13)) where T is the current temperature as in the SA algorithm.

$$\vartheta = e^{-10000\left(Th(B)-Th\left(B^{new}\right)\right)/T} \tag{13}$$

The current temperature is decreased during the course of the algorithm as cT, where $0 < c < 1$ is the cooling rate. c is a constant and it is set to $(K\text{-}1)/K$ in all experiments presented in this study. This approach allows to make a diversification in the search space.

3.6. Termination criterion

The termination criterion is based on the improvement in the objective function. If the best throughput value does not change for $5K$ consecutive iterations, then the algorithm is terminated. This number is determined by the preliminary tests.

The proposed ALNS algorithm is given in Algorithm 1. After generating an initial solution (B^{init}) by any of the initialization procedures, the throughput of the initial solution is obtained by the decomposition method. Following, a removal-insertion operator pair is employed to the initial buffer configuration. The throughput of each generated buffer configuration is calculated and then the configuration yielding the best throughput value is chosen as the current solution (B) for the next iteration. If a local optimum (called *l-opt*) is reached, then IncDec strategy is employed to obtain a new solution. Following, the new solution is evaluated and updated according to the acceptance criterion. This procedure continues until the termination criterion is satisfied.

4. Computational experiments

The benchmark instances in the BAP literature were used to evaluate the performance of the proposed ALNS algorithm. To be able to make a fair comparison, only the studies employed decomposition method as an evaluative tool were considered. As the CPU time was not provided in the previous studies and the computational experimental environment were not same, the CPU time comparison could not be carried out. Instead of this, the compar-

Table 1
5-machine line [53].

Machine	1	2	3	4	5
$1/r_i$	11	19	12	7	7
$1/\beta_i$	20	167	22	22	26

Table 2
Comparative results for the 5-machine line.

Method	Optimal buffer configuration				# of best solution	Avg. throughput	Avg. # of eval.
	1	**2**	**3**	**4**			
Gradient-based search [53]	5	11	8	7	*N.A.*	0.4914	*N.A.*
Gradient-based search [54]	7	10	10	4	*N.A.*	0.4943	*N.A.*
DTL-TS [10]	7	10	10	4	*N.A.*	0.4943	*N.A.*
ALNS-Random	7	10	10	4	10	0.4943	21.8
ALNS-PaVi	7	10	10	4	10	0.4943	14.5
ALNS-BowlPh	**7**	**10**	**10**	**4**	**10**	**0.4943**	**5.0**
ALNS-SBowlPh	7	10	10	4	10	0.4943	10.0

Algorithm 1: The proposed ALNS algorithm to solve the BAP

input : Removal/insertion operators ($o \in O$), initial temperature T_{init}, cooling rate c, IncDec value id
output: The best buffer configuration having the maximum throughput rate B^{best}

Generate an initial solution B^{init} using an initialization procedure
Set selection probability P_o for each removal/insertion pair, $o \in O$
Let $K \leftarrow T \leftarrow T_{init}$ be the temperature, i be the iteration as $i \leftarrow 1$, and cooling rate $c \leftarrow (K\text{-}1)/K$
Set id value considering the problem size
Let $B^{best} \leftarrow B \leftarrow B^{init}$
Set $2K \leftarrow$ *l-opt* allowed number of maximum iteration for local optimum
Set the maximum allowable consecutive iteration number as $5K$
repeat
    Select a removal/insertion operator pair, $o \in O$, with probability P_o
    Let B^{new} be the new solution obtained by applying operator pair o to B
    if $Th(B^{new}) > Th(B)$ **then**
        $B \leftarrow B^{new}$
        if $Th(B) > Th(B^{best})$ **then**
            $B^{best} \leftarrow B$
    else
        Let $\upsilon \leftarrow e^{-10000(Th(B)-Th(B^{new}))/T}$
        Generate a random number $\vartheta \in [0; 1]$
        if $\upsilon > \vartheta$ **then**
            $B \leftarrow B^{new}$
    if *l-opt* $\geq 2K$ **then**
    Let B^{new} be the new solution obtained by applying IncDec strategy to B
    if $Th(B^{new}) > Th(B)$ **then**
        $B \leftarrow B^{new}$
    Let $B^{best} \leftarrow B$
    $T \leftarrow cT$
    Update the acceptance probability of the worst solution
    Update the selection probability of each removal/insertion operator pair
    Update id value
    $i \leftarrow i + 1$
until the termination criterion is satisfied

ison was conducted on the average number of evaluated buffer configurations, i.e. how many throughput calculations were done during the search process.

The proposed algorithm was implemented in the MATLAB (R2016b). All experiments were conducted on an Intel (R) 2.40 GHz i5 processor with 4 GB of RAM. 10 replications were executed for each instance and the average throughput values were recorded as well as the average number of evaluated buffer configurations.

The presented tables in this section provide the information on the reliability parameters of the machines in the line, i.e. the mean time between failures (MTBF = $1/\beta_i$), and the mean time to repair (MTTR = $1/r_i$). All considered problems involve homogeneous production lines; i.e. it is assumed that all the machines in the line have same processing time which is one-time unit. The experimental analysis was carried out into 3 groups: small, medium and large-sized problems. The performance analysis of the proposed algorithm is presented in detail in the following subsections.

4.1. Small-sized problems

5-machine line: In the first example, the efficiency of the proposed algorithm was tested for the 5-machine production line where the specifics of the problem were given in Table 1. The lower bound for each buffer location was restricted by 4, and N was set to 31 for this problem. The results are presented in Table 2.

As can be noted from Table 2, the proposed algorithm yielded the optimum throughput value, i.e. 0.4943, with the buffer configuration {7, 10, 10, 4} as it was reported in the previous studies ([10;54]). It should be noted that the gradient-based search algorithm was employed in the studies of Ho et al. [53], who suggested the problem for the first time, and Gershwin and Schor [54], while tabu search was employed by Demir et al. [10].

It was observed that all the proposed initialization procedures obtained the optimum solution for all ten runs but the proposed ALNS-BowlPh reached the optimum solution at the average of five evaluations which was the lowest value among the all initialization procedures. The reason was that the BowlPh initialization procedure yielded a configuration which was very close to the optimum solution.

The aim of solving this small-sized example is to verify that the proposed ALNS algorithm has capability to find the optimum solution. To be able to make more general comments on the performance of the proposed algorithm, the experimental study is extended to medium and large-sized instances.

4.2. Medium-sized problems

9-machine lines: In this example, a 9-machine line with different reliability parameters, i.e. total of nine cases, were considered (Table 3). This problem set was initially proposed by Shi and Men [9] and later used by Demir et al. [10]. The constant total buffer capacity was 160 for all cases. It should be noted that Shi and Men [9] proposed a hybrid method combining nested partitions with TS (NP/TS) for this problem while Demir et al. [10] employed another TS method called as DTL-TS.

As in the previous example, the proposed ALNS algorithm slightly yielded higher throughput values than other TS-based methods (see Table 3). The number of average evaluations required to reach the best solution for ALNS-SBowlPh was within the range

Table 3
Comparative results for 9-machine lines.

Case	Machine parameters		NP/TS [9]	DTL/TS [10]	ALNS-Random		ALNS-PaVi		ALNS-BowlPh		ALNS-SBowlPh	
	β_i	r_i	Throughput	Throughput	Avg. throughput	Avg. # of eval.	Avg. throughput	Avg. # of eval.	Avg. throughput	Avg. # of eval.	Avg. throughput	Avg. # of eval.
1	0.3	0.05	0.108143	0.108147	0.108240	310.0	0.108240	165.5	0.108240	143.6	**0.108240**	**129.9**
2	0.3	0.10	0.200250	0.200255	0.200357	248.8	0.200357	222.4	0.200357	179.2	**0.200357**	**138.5**
3	0.3	0.20	0.345491	0.345495	0.345580	240.6	0.345580	190.0	0.345580	145.6	**0.345580**	**124.8**
4	0.3	0.30	0.452074	0.452077	0.452151	260.3	0.452151	199.7	0.452151	150.5	**0.452151**	**130.2**
5	0.3	0.40	0.532002	0.532006	0.532091	308.0	0.532091	280.8	0.532091	182.6	**0.532091**	**151.5**
6	0.4	0.05	0.088777	0.088782	0.088857	270.8	0.088857	232.9	0.088857	142.6	**0.088857**	**132.7**
7	0.4	0.10	0.166232	0.166236	0.166322	350.5	0.166322	267.1	0.166322	151.4	**0.166322**	**143.4**
8	0.4	0.20	0.293041	0.293046	0.293199	351.5	0.293199	201.4	0.293199	152.2	**0.293199**	**139.7**
9	0.4	0.30	0.390814	0.390819	0.390881	245.1	0.390881	215.0	0.390881	188.3	**0.390881**	**147.9**

Table 4
10-machine line [7].

Machine	1	2	3	4	5	6	7	8	9	10
$1/r_i$	7	7	5	10	9	14	5	8	10	10
$1/\beta_i$	20	33	22	22	25	40	23	30	45	20

Table 5
Comparative results for the 10-machine line.

Method	# of best solution	Avg. throughput	Avg. # of eval.
DC [7]	*N.A.*	0.64135	*N.A.*
DTL-TS [10]	*N.A.*	0.64135	*N.A.*
ALNS-Random	8	0.64347	424
ALNS-PaVi	10	0.64348	396
ALNS-BowlPh	10	0.64348	376
ALNS-SBowlPh	**10**	**0.64348**	**371**

of 124.8 and 151.5, whereas it changed between 142.6 and 188.3, 165.5–280.8 and 240.6–351.5 for ALNS-BowlPh, ALNS-PaVi and ALNS-Random, respectively. Hence, it can be concluded that the ALNS-SBowlPh has the best performance in terms of the convergence speed. These experimental results showed the improving effect of the SBowlPh initialization procedure. It should be noted that the proposed ALNS algorithm obtained the best solution in 10 out of 10 replications for all initialization procedures, and the solution time was recorded as between 6.4 and 10.7 s.

10-machine line: Nahas et al. [7] proposed this example (Table 4). The maximum allowable total number of buffers was 270 for this problem. The degraded ceiling (DC) algorithm developed by Nahas et al. [7] and DTL-TS algorithm developed by Demir et al. [10] were employed for solving this problem before.

The best throughput value obtained by the proposed ALNS algorithm is 0.64348 with the buffer configuration {14, 20, 30, 53, 45, 27, 23, 25, 33}. The average number of evaluations required to reach the best solution are 424, 396, 376 and 371 for ALNS-Random, ALNS-PaVi, ALNS-BowlPh and LNS-SBowlPh, respectively (Table 5). As in the previous examples, the proposed initialization procedure SBowlPh is the best initialization procedure among the others to obtain the best solution in terms of fast convergence. When the performance of the proposed ALNS algorithm was evaluated in terms of the solution time it was observed that the CPU time required to obtain the best solution was between 10 and 14 s.

4.3. Large-sized problems

20 and 40 machine-lines: The efficiency of the proposed algorithm was tested for also large-sized problem instances. For this purpose, the production lines of the 20 and 40-machine lines with different reliability parameters were considered. The N value was set to 400 for 20-machine and 400, 800 and 1600 for 40-machine lines. These test cases were initially proposed by Li et al. [21]. The specifics of the 20-machine block were provided in Table 6. The 40-machine line was composed by duplicating same 20-machine blocks.

Comparative results were presented in Table 7. Li et al. [21] compared fast algorithm (FA), proposed by the authors, to other algorithms reported in the literature. As seen in Table 7, the proposed ALNS-SBowlPh algorithm yielded 2.14%, 1.61%, 3.88% and 3.50% better results on average than the FA for each case, respectively. Fig. 5 also shows that the proposed ALNS-SBowlPh algorithm is the best one among the other algorithms in terms of the solution quality. As it can be seen in both Table 7 and Fig. 5, the efficiency of the proposed ALNS-SBowlPh becomes more apparent for the large-sized instances. Moreover, Li et al. [21] stated that “pilot experiments show that if best throughput does not change for five consecutive iterations, it will not change anymore”. So, it can be inferred form this statement that the values reported by Li et al. [21] are the best values that can be achieved by FA. It

Table 6
20-machine line [21].

Machine	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
β_i	0.011	0.005	0.022	0.025	0.052	0.021	0.017	0.013	0.011	0.016
r_i	0.039	0.016	0.166	0.085	0.186	0.105	0.116	0.136	0.043	0.139
Machine	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
β_i	0.019	0.014	0.013	0.025	0.014	0.001	0.006	0.013	0.059	0.038
r_i	0.074	0.099	0.111	0.178	0.094	0.015	0.061	0.046	0.184	0.134

can be concluded that the proposed ALNS-SBowlPh is better than FA in terms of the solution quality. When the performance of the proposed ALNS-SBowlPh algorithm was evaluated in terms of the solution time it was observed that the CPU time required to obtain the best solution was between 65 and 81 s for the 20-machine line and it was recorded between 267 and 338 s, which was acceptable, for the 40-machine line. On the other hand, the solution time of the FA algorithm was recorded as few seconds. One reason of the differences in the solution time is the difference in the employed decomposition method. In FA algorithm the authors employed a three-machine-two-buffer decomposition method which reduced the number of throughput evaluations significantly. The other reason is that the proposed ALNS algorithm evaluates all the neighborhoods of the current solution when applying IncDec strategy. It should be noted that the BAP is a design problem and the design problems are in the class of long-term strategic problems, so they are generally solved once. Therefore, the solution quality is more important than the search time as it is indicated by Talbi [55]. As a result, this example shows that the proposed ALNS-SBowlPh algorithm has a great capability for solving the BAP in a reasonable computation time.

Fig. 6 shows the performance of the initialization procedures used in the proposed ALNS algorithm. As it can be seen in Fig. 6, the proposed SBowlPh initialization procedure drastically reduces the number of throughput evaluations; i.e. search effort. It can be inferred from this experiment that starting with a 'good' initial solution significantly reduces the search effort as well as obtaining better solutions, as can be seen in Fig. 6. So, it can be stated that the proposed SBowlPh initialization procedure yields very good initial solutions for unreliable production lines.

4.4. Discussion

Table 8 summarizes the improvements achieved by the proposed ALNS algorithm over the best-known algorithm in the literature for all examined cases. As seen in Table 8, the proposed ALNS-SBowlPh algorithm yielded better results than other methods reported in the literature. Moreover, the efficiency of the proposed algorithm becomes more apparent as the problem size increases. It should be noted that the experiments indicate that the computational complexity of the proposed ALNS algorithm appears to be of the order of $O(K^2)$ in terms of average number of evaluations, where K is the number of the machines in the line.

Table 8 also shows the employed proportions of the proposed removal-insertion operator pairs for in case of each initialization procedure. As it can be seen in Table 8, the machine status-based removal-insertion heuristic is employed on average 76.5% of the cases where the removal-insertion operator is applied. Since the machines in the line are subject to breakdown, the proposed machine status-based removal-insertion heuristic is the dominant operator as it is expected. Moreover, the usage of this operator pair increases as the problem size increases as presented in Table 8.

Although the structure of the ALNS algorithm is very simple and the algorithm can yield very promising results for the BAP as it can be inferred from the experimental results in this section, there is always possibility to get stuck at the local optima because of the neighborhood-based structure of the algorithm. In the proposed ALNS algorithm this drawback is overcame by employing Inc-Dec strategy. This strategy is used not only to escape the local optima but also used to improve the solution quality. On the other hand, evaluating all neighborhoods while applying the Inc-Dec strategy increases the solution time of the algorithm.

The experimental results presented so far show that the proposed ALNS-SBowlPh algorithm is the best algorithm among the methods reported in the literature. As the differences between the throughput values obtained by the methods seemed to be

Table 7
Comparative results for 20 and 40-machine lines.

(K, N)	FA [21]		ALNS-Random			ALNS-PaVi			ALNS-BowlPh			ALNS-SBowlPh		
	Best throughput	Avg. # of eval.	Avg. throughput	Avg. # of eval.	Imp. over FA(%)[a]	Avg. throughput	Avg. # of eval.	Imp. over FA(%)[a]	Avg. throughput	Avg. # of eval.	Imp. over FA(%)[a]	Avg. throughput	Avg. # of eval.	Imp. over FA(%)[a]
(20, 400)	0.662264	100	0.666804	5490	0.69	0.671616	4462	1.41	0.673329	4334	1.67	**0.676440***	**3789**	**2.14**
(40, 400)	0.571864	N.A.	0.576316	6870	0.78	0.578814	5518	1.22	0.579128	5112	1.27	**0.581075***	**4357**	**1.61**
(40, 800)	0.651023	N.A.	0.664821	7543	2.12	0.672033	5790	3.23	0.675661	5627	3.78	**0.676265***	**5005**	**3.88**
(40, 1600)	0.707086	N.A.	0.719588	8569	1.77	0.727760	7363	2.92	0.730578	7046	3.32	**0.731847***	**6591**	**3.50**

* best solution.
[a] improvement over FA (%) = 100 × [Th (ALNS) − Th (FA)] / Th (FA).

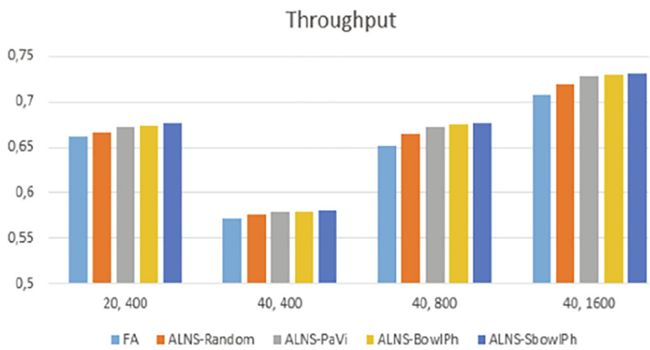

Fig. 5. Comparison of the FA and the proposed ALNS algorithm.

small, the Wilcoxon signed-rank test was employed to determine whether the difference between the throughput values were statistically significant. The following null hypothesis was tested at the 0.05 significance level:

H_0: f(Literature Best) -f(ALNS-SBowlPh) = 0

In Table 9, the Wilcoxon test having a p-value of 0.001 indicates that the differences between the throughput values are statistically significant. Considering all experiments presented in this section the proposed ALNS-SBowlPh algorithm yielded very promising results in solving the BAP. Moreover, it was observed that the proposed SBowlPh initialization procedure significantly reduced the throughput evaluations in obtaining the best solution. Overall, it can be stated that the proposed ALNS-SBowlPh algorithm is an effective algorithm for solving the BAP in unreliable production lines.

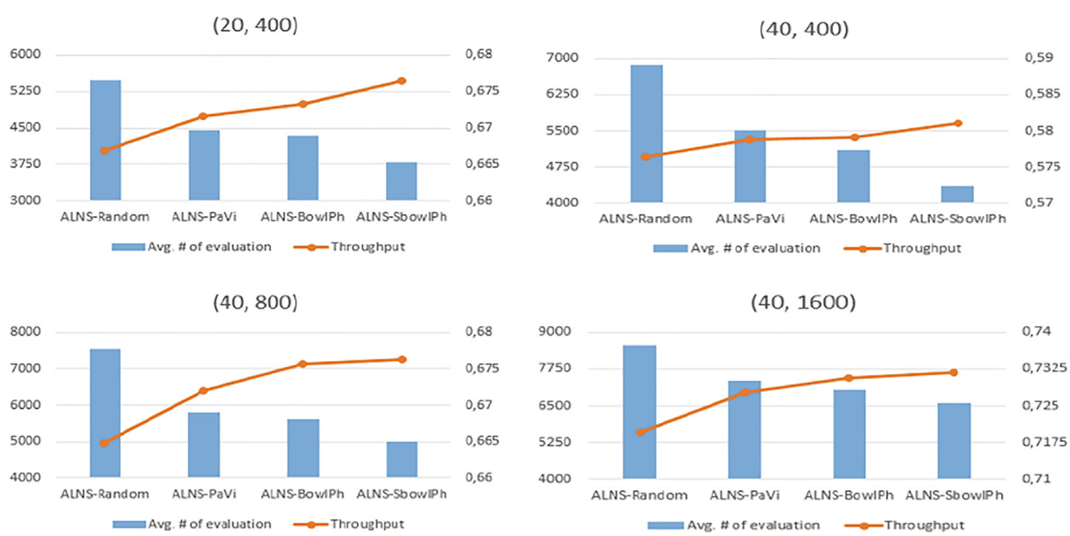

Fig. 6. Comparison of the initialization procedures in ALNS.

Table 8
Summary of experimental results.

Case	Initialization procedure	Imp. over the best alg. (%)	The proportion of the removal-insertion heuristics	
			Location-based (%)	Machine status-based (%)
(5, 31)	Random	0.0	54	46
	PaVi	0.0	0	0
	BowlPh	0.0	0	0
	SBowlPh	0.0	0	0
(9, 160)	Random	0.05	49	51
	PaVi	0.05	27	73
	BowlPh	0.05	14	86
	SBowlPh	0.05	14	86
(10, 270)	Random	0.33	30	70
	PaVi	0.33	29	71
	BowlPh	0.33	27	73
	SBowlPh	0.33	21	79
(20, 400)	Random	0.82	23	77
	PaVi	1.33	17	83
	BowlPh	1.64	16	84
	SBowlPh	1.85	15	85
(40, 400)	Random	0.76	13	87
	PaVi	0.97	17	83
	BowlPh	1.32	18	82
	SBowlPh	1.59	19	81
(40, 800)	Random	1.99	10	90
	PaVi	2.18	18	82
	BowlPh	2.29	14	86
	SBowlPh	2.39	10	90
(40, 1600)	Random	1.81	10	90
	PaVi	3.09	19	81
	BowlPh	3.34	12	88
	SBowlPh	3.42	9	91

Table 9
Result of the Wilcoxon signed-rank test.

Null Hypothesis	Test	Sig. (p)	Decision
The median of differences between Literature Best and ALNS-SBowlPh equals 0	Related samples Wilcoxon Signed Rank Test	0.001	Reject the null hypothesis

5. Conclusion

The BAP is one of the major optimization problems in designing production lines. Solving this problem optimally is considerably difficult due its stochastic and nonlinear nature. In this study, an ALNS algorithm is proposed to solve the BAP for unreliable production lines which are widely encountered in manufacturing systems. It is well-known that ALNS algorithm is very successful for solving the vehicle routing problems. In this study different from the literature, for the first time, ALNS algorithm is employed to solve the problem of designing a production line. For this purpose, two different removal and insertion operator pairs are proposed. The selection of operators is done in an adaptive way. Moreover, a new initialization procedure based on the well-known storage bowl phenomenon is proposed and embedded into the proposed ALNS algorithm to improve the search efficiency. The BAP is solved for throughput maximization and benchmark instances are used to evaluate the performance of the proposed algorithm.

The experimental study proved that the proposed ALNS algorithm yields better results than other methods for all considered benchmark instances. Moreover, it was observed that the proposed initialization procedure, i.e. SBowlPh, significantly reduced the search effort. In addition, the efficiency of the proposed algorithm is more apparent especially for long lines, i.e. 20 and 40-machine lines. In summary, it can be stated that the proposed ALNS-SBowlPh algorithm yields very promising results for solving the BAP in unreliable production lines.

In this study, to prove the efficiency of the proposed ALNS algorithm hypothetical production lines were considered. However, the proposed algorithm has a great potential to solve the real-life buffer allocation problems. In this respect, as a future study, the performance of the proposed algorithm can be tested for other type of production systems, such as split and merge or assembly/disassembly lines which are encountered in many industries. Moreover, while the problem is considered from a throughput maximization perspective in this study, solving the BAP for total buffer capacity minimization could also be very valuable in the presence of budget constraints. Hence, our study can be extended to minimize the total buffer capacity using the proposed ALNS algorithm. Lastly, dealing with work-in-process (WIP) minimization in solving the BAP can be another future research direction. Since, the WIP costs are not desirable by the managers it could be very valuable to deal with the problem in this respect.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was done during the PhD study of the first author at the Graduate School of Natural and Applied Sciences, Pamukkale University. We would also like to thank the anonymous reviewers for their constructive comments.

References

[1] J.M.G. Smith, N. Chikhale, Buffer allocation for a class of nonlinear stochastic knapsack problems, Ann. Oper. Res. 58 (1995) 323–360, https://doi.org/10.1007/BF02038860.
[2] J.M.G. Smith, F.R.B. Cruz, The buffer allocation problem for general finite buffer queueing networks, IIE Trans. 37 (2005) 343–365, https://doi.org/10.1080/07408170590916986.
[3] W.M. Chow, Buffer capacity analysis for sequential production lines with variable process times, Int. J. Prod. Res. (1987).
[4] A. Dolgui, A. Eremeev, A. Kolokolov, V. Sigaev, A Genetic Algorithm for the Allocation of Buffer Storage Capacities in a Production Line with Unreliable Machines, J. Math. Model. Algorithms. 1 (2002) 89–104, https://doi.org/10.1023/A:1016560109076.
[5] A. Dolgui, A.V. Eremeev, V.S. Sigaev, HBBA: Hybrid algorithm for buffer allocation in tandem production lines, J. Intell. Manuf. 18 (2007) 411–420, https://doi.org/10.1007/s10845-007-0030-z.
[6] F. Altiparmak, B. Dengiz, A.A. Bulgak, Optimization of buffer sizes in assembly systems using intelligent techniques, in: Winter Simul. Conf. Proc., 2002. https://doi.org/10.1109/wsc.2002.1166373.
[7] N. Nahas, D. Ait-Kadi, M. Nourelfath, A new approach for buffer allocation in unreliable production lines, Int. J. Prod. Econ. 103 (2006) 873–881, https://doi.org/10.1016/j.ijpe.2006.02.011.
[8] N. Nahas, M. Nourelfath, D. Ait-Kadi, Selecting machines and buffers in unreliable series-parallel production lines, 2009. https://doi.org/10.1080/00207540701806883.
[9] L. Shi, S. Men, Optimal buffer allocation in production lines, IIE Trans. (2003), https://doi.org/10.1080/07408170304431.
[10] L. Demir, S. Tunali, A. Løkketangen, A tabu search approach for buffer allocation in production lines with unreliable machines, Eng. Optim. 43 (2011) 213–231, https://doi.org/10.1080/0305215X.2010.481022.
[11] L. Demir, S. Tunalı, D.T. Eliiyi, An adaptive tabu search approach for buffer allocation problem in unreliable non-homogenous production lines, Comput. Oper. Res. 39 (2012) 1477–1486, https://doi.org/10.1016/j.cor.2011.08.019.
[12] A.A. Bulgak, Analysis and design of split and merge unpaced assembly systems by metamodelling and stochastic search, Int. J. Prod. Res. 44 (2006) 4067–4080, https://doi.org/10.1080/00207540600564625.
[13] F. Altiparmak, B. Dengiz, A.A. Bulgak, Buffer allocation and performance modeling in asynchronous assembly system operations: An artificial neural network metamodeling approach, Appl. Soft Comput. J. 7 (2007) 946–956, https://doi.org/10.1016/j.asoc.2006.06.002.
[14] M. Nourelfath, N. Nahas, D. Ait-Kadi, Optimal design of series production lines with unreliable machines and finite buffers, J. Qual. Maint. Eng. (2005), https://doi.org/10.1108/13552510510601348.
[15] M. Zandieh, M.N. Joreir-Ahmadi, A. Fadaei-Rafsanjani, Buffer allocation problem and preventive maintenance planning in non-homogenous unreliable production lines, Int. J. Adv. Manuf. Technol. (2017), https://doi.org/10.1007/s00170-016-9744-4.
[16] S. Weiss, J.A. Schwarz, R. Stolletz, The buffer allocation problem in production lines: Formulations, solution methods, and instances, IISE Trans. (2019), https://doi.org/10.1080/24725854.2018.1442031.
[17] L. Xuemei, S. Huan, Z. Rui, J. Yongqi, L. Aiping, Collaborative Optimization of Transfer Line Balancing and Buffer Allocation Based on Polychromatic Set, Procedia CIRP. 63 (2017) 213–218, https://doi.org/10.1016/j.procir.2017.02.045.
[18] S. Yelkenci Kose, O. Kilincci, A multi-objective hybrid evolutionary approach for buffer allocation in open serial production lines, J. Intell. Manuf. (2018), https://doi.org/10.1007/s10845-018-1435-6.
[19] M.M. Motlagh, P. Azimi, M. Amiri, G. Madraki, An efficient simulation optimization methodology to solve a multi-objective problem in unreliable unbalanced production lines, Expert Syst. Appl. 138 (2019), https://doi.org/10.1016/j.eswa.2019.112836.
[20] A.B. Dolgui, A.V. Eremeev, V.S. Sigaev, Analysis of a multicriterial buffer capacity optimization problem for a production line, Autom. Remote Control. 78 (2017) 1276–1289, https://doi.org/10.1134/S0005117917070098.
[21] L. Li, Y. Qian, Y.M. Yang, K. Du, A fast algorithm for buffer allocation problem, Int. J. Prod. Res. 7543 (2015), https://doi.org/10.1080/00207543.2015.1092612.
[22] C. Shi, S.B. Gershwin, A segmentation approach for solving buffer allocation problems in large production systems, Int. J. Prod. Res. 7543 (2014) 1–21, https://doi.org/10.1080/00207543.2014.991842.
[23] S. Weiss, A. Matta, R. Stolletz, Optimization of buffer allocations in flow lines with limited supply, IISE Trans. 50 (2018) 191–202, https://doi.org/10.1080/24725854.2017.1328751.
[24] G. Liberopoulos, Comparison of optimal buffer allocation in flow lines under installation buffer, echelon buffer, and CONWIP policies, Springer, US (2019), https://doi.org/10.1007/s10696-019-09341-y.
[25] S. Xi, Q. Chen, J. MacGregor Smith, N. Mao, A. Yu, H. Zhang, A new method for solving buffer allocation problem in large unbalanced production lines, Int. J. Prod. Res. (2019) 1–22, https://doi.org/10.1080/00207543.2019.1685709.
[26] G. Pedrielli, A. Matta, A. Alfieri, M. Zhang, Design and control of manufacturing systems: a discrete event optimisation methodology, Int. J. Prod. Res. 56 (2018) 543–564, https://doi.org/10.1080/00207543.2017.1412532.
[27] T.C. Lopes, C.G.S. Sikora, A.S. Michels, L. Magatão, An iterative decomposition for asynchronous mixed-model assembly lines: combining balancing,

sequencing, and buffer allocation, Int. J. Prod. Res. 58 (2020) 615–630, https://doi.org/10.1080/00207543.2019.1598597.
[28] P. Shaw, A new local search algorithm providing high quality solutions to vehicle routing problems, Glasgow, Scotland, 1997.
[29] P. Shaw, Using constraint programming and local search methods to solve vehicle routing problems, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 1998. https://doi.org/10.1007/3-540-49481-2_30.
[30] S. Ropke, D. Pisinger, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, Transp. Sci. (2006), https://doi.org/10.1287/trsc.1050.0135.
[31] G. Mattos Ribeiro, G. Laporte, An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem, Comput. Oper. Res. 39 (2012) 728–735, https://doi.org/10.1016/j.cor.2011.05.005.
[32] S. Ropke, D. Pisinger, A unified heuristic for a large class of Vehicle Routing Problems with Backhauls, Eur. J. Oper. Res. (2006), https://doi.org/10.1016/j.ejor.2004.09.004.
[33] D. Pisinger, S. Ropke, A general heuristic for vehicle routing problems, 34 (2007) 2403–2435. https://doi.org/10.1016/j.cor.2005.09.012.
[34] A.S. Pepin, G. Desaulniers, A. Hertz, D. Huisman, A comparison of five heuristics for the multiple depot vehicle scheduling problem, J. Sched. 12 (2009) 17–30, https://doi.org/10.1007/s10951-008-0072-x.
[35] S. Akpinar, Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem, Expert Syst. Appl. 61 (2016) 28–38, https://doi.org/10.1016/j.eswa.2016.05.023.
[36] S. Chen, R. Chen, G. Wang, J. Gao, A. Kumar, An adaptive large neighborhood search heuristic for dynamic vehicle routing problems R, Comput. Electr. Eng. 67 (2018) 596–607, https://doi.org/10.1016/j.compeleceng.2018.02.049.
[37] M. Alinaghian, N. Shokouhi, Multi-depot multi-compartment vehicle routing problem, solved by a hybrid adaptive large neighborhood search ☆, Omega. 76 (2018) 85–99, https://doi.org/10.1016/j.omega.2017.05.002.
[38] J.Y. Potvin, J.M. Rousseau, A parallel route building algorithm for the vehicle routing and scheduling problem with time windows, Eur. J. Oper. Res. (1993), https://doi.org/10.1016/0377-2217(93)90221-8.
[39] N. Azi, M. Gendreau, J.Y. Potvin, An adaptive large neighborhood search for a vehicle routing problem with multiple routes, Comput. Oper. Res. 41 (2014) 167–173, https://doi.org/10.1016/j.cor.2013.08.016.
[40] M.G. Avci, M. Avci, An adaptive large neighborhood search approach for multiple traveling repairman problem with profits, Comput. Oper. Res. 111 (2019) 367–385, https://doi.org/10.1016/j.cor.2019.07.012.
[41] I. Žulj, S. Kramer, M. Schneider, A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem, Eur. J. Oper. Res. 264 (2018) 653–664, https://doi.org/10.1016/j.ejor.2017.06.056.
[42] E. Demir, T. Bektaş, G. Laporte, An adaptive large neighborhood search heuristic for the Pollution-Routing Problem, Eur. J. Oper. Res. (2012), https://doi.org/10.1016/j.ejor.2012.06.044.
[43] T. Bektaş, G. Laporte, The pollution-routing problem, Transp. Res. Part B Methodol. (2011), https://doi.org/10.1016/j.trb.2011.02.004.
[44] E. Demir, T. Bektas, G. Laporte, The bi-objective Pollution-Routing Problem, Eur. J. Oper. Res. 232 (2014) 464–478, https://doi.org/10.1016/j.ejor.2013.08.002.
[45] A. Franceschetti, E. Demir, D. Honhon, T. Van Woensel, G. Laporte, M. Stobbe, A metaheuristic for the time-dependent pollution-routing problem, 259 (2017) 972–991. https://doi.org/10.1016/j.ejor.2016.11.026.
[46] Ç. Iris, D. Pacino, S. Ropke, Improved formulations and an Adaptive Large Neighborhood Search heuristic for the integrated berth allocation and quay crane assignment problem, Transp. Res. Part E Logist. Transp. Rev. 105 (2017) 123–147, https://doi.org/10.1016/j.tre.2017.06.013.
[47] F.S. Hillier, K.C. So, R.W. Boling, Toward characterizing the optimal allocation of storage space in production line systems with variable processing times, 39 (1993) 126–133.
[48] S.B. Gershwin, Efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking, Oper. Res. (1987), https://doi.org/10.1287/opre.35.2.291.
[49] Y. Dallery, R. David, X.L. Xie, An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers, IIE Trans. (Institute, Ind. Eng. 20 (1988) 281–283, https://doi.org/10.1080/07408178808966181.
[50] L. Demir, S. Tunali, D.T. Eliiyi, The state of the art on buffer allocation problem: A comprehensive survey, J. Intell. Manuf. (2014), https://doi.org/10.1007/s10845-012-0687-9.
[51] V. Ghilas, E. Demir, T. Van Woensel, An adaptive large neighborhood search heuristic for the Pickup and Delivery Problem with Time Windows and Scheduled Lines, Comput. Oper. Res. 72 (2016) 12–30, https://doi.org/10.1016/j.cor.2016.01.018.
[52] H.T. Papadopoulos, M.I. Vidalis, A heuristic algorithm for the buffer allocation in unreliable unbalanced production lines, Comput. Ind. Eng. 41 (2001) 261–277, https://doi.org/10.1016/S0360-8352(01)00051-1.
[53] Y.C. Ho, M.A. Eyler, T.T. Chien, A gradient technique for general buffer storage design in a production line, Int. J. Prod. Res. (1979), https://doi.org/10.1080/00207547908919637.
[54] S.B. Gershwin, J.E. Schor, Efficient algorithms for buffer space allocation, Ann. Oper. Res. (2000).
[55] E.G. Talbi, Metaheuristics: From Design to Implementation, 2009. https://doi.org/10.1002/9780470496916.