Mhada *et al. SpringerPlus (2016) 5:2045*
DOI 10.1186/s40064-016-3756-2

SpringerPlus

RESEARCH Open Access

Multilevel hybrid method for optimal buffer sizing and inspection stations positioning

Fatima Zahra Mhada[1], Mohamed Ouzineb[2], Robert Pellerin[3] and Issmail El Hallaoui[4*]

*Correspondence:
issmail.elhallaoui@gerad.ca
[4] GERAD and Polytechnique Montreal, CP 6079, succ. Centre-Ville, Montreal, QC, Canada
Full list of author information is available at the end of the article

Abstract

Designing competitive manufacturing systems with high levels of productivity and quality at a reasonable cost is a complex task. Decision makers must face numerous decision variables which involve multiple and iterative analysis of the estimated cost, quality and productivity of each design alternative. This paper adresses this issue by providing a fast algorithm for solving the buffer sizing and inspection positioning problem of large production lines by combining heuristic and exact algorithms. We develop a multilevel hybrid search method combining a genetic algorithm and tabu search to identify promising locations for the inspection stations and an exact method that optimizes rapidly (in polynomial time) the buffers' sizes for each location. Our method gives valuable insights into the problem, and its solution time is a small fraction of that required by the exact method on production lines with 10–30 machines.

Keywords: Production, Inspection, Quality, Combinatorial optimization, Metaheuristics

Background

The system we consider in this paper consists of n machines, n fixed-size buffers, and $m + 1$ inspection stations (one inspection station is at the end of the line and m are internal) in series. We explore the impact of quality constraints on the system performance and (numerical) complexity. The objective is to simultaneously minimize the combined storage and shortage costs, determine the optimal buffers' sizes, and specify the optimal number and positions of the m internal inspection stations. The resulting mathematical model is an intractable combinatorial nonlinear optimization model; it is difficult to find an exact solution in a reasonable time, especially when the production line is large. This paper develops an efficient evolutionary heuristic for this complex model. While the number of machines considered in the literature does not exceed 10 (to the best of our knowledge), this paper aims at solving larger production lines and a variable number of inspection stations.

In the recent past, several authors have investigated improving productivity with production control policies. Most studies assume that all the parts produced are conforming, which is unrealistic. Standard quality analysis models usually separate the problem of quality preservation in production lines (via the positioning of inspection stations, for example) and the optimization of production (via Kanban, CONWIP etc.). However,

these two problems are interdependent (see Kim and Gershwin 2005, 2008; Colledani and Tolio 2005, 2006, 2009, 2011). We illustrate this by considering the Kanban strategies introduced by Toyota in the 1960s that have since become the paradigm of "lean manufacturing". These strategies essentially advocate areas of limited storage between successive machines in a production line. The storage areas allow a degree of decoupling between machines to increase the productivity of the line. The idea is to limit the impact of shortages on downstream machines and to avoid blocking machines that are upstream from one that is broken down. These areas should be limited because they are associated with frozen capital: they introduce storage costs and extended transit times for parts in the workshop.

In an ideal "just in time" scenario, there would be no intermediate storage and finished parts would be pulled from the system as they are produced. In reality, machine breakdowns and shortages of raw material or operators make it impossible to ensure continuous workshop output. The sizes of the storage areas should depend on the likelihood of such events and the estimated costs of the associated service interruptions. If we decide the amount of storage without considering the quality dimension, we risk creating storage areas that contain important quantities of defective parts. These defective parts correspond to misused production-line time, and they undermine the effectiveness of the intermediate storage in terms of increasing productivity. Moreover, the costs associated with storage impact the budget available for improvements in quality and vice versa. In a similar manner, Inman et al. (2013) show that production design impacts quality and vice versa. Also, the location of an inspection station affects both the expected production cost per item and the production rate of the line. The quality control is obviously critical to any production system and has significant managerial impact. As a result, it is crucial to develop strategies to allocate inspection or quality control stations into a manufacturing processes that can help preventing all wastes resulting from unidentified defective items being processed (see Shetwan et al. 2011; Raviv 2013).

In a real manufacturing context, simulation can be used to determine the optimal buffers' sizes and the optimal positions of the inspection stations by considering all possible scenarios. However, this is not practical when the systems are complex: the number of scenarios is $\binom{n}{m} \times 100^n$ for a system with n machines and m inspection stations where each buffer of equal size is discretized to 100 levels. Note that the system is required to have an inspection station at the end of the line. Since the position of this station is known, it is not included in the optimization. An alternative to simulation is to develop a realistic model and a fast optimization technique to solve it. The real model is however very complex because many factors impact quality and production. So, we need elaborate some simplifying assumptions in such a way that the resulting model is still realistic and can be solved optimally.

Researchers have investigated the integration of quality aspects and production policies for a single unreliable machine producing a single product type. The system starts in an "in-control" state producing conforming items and then switches to an "out-of-control" state and starts producing nonconforming items. Integrated models can be classified as: (1) integrated production and quality management (e.g., Hajji et al. 2010; Kutzner and Kiesmüller 2013; Mhada et al. 2014b; Naebulharam and Zhang 2014; Matta and Simone 2016; 2) integrated production and maintenance management (e.g., Ben-Daya

2002; Dhouib et al. 2012; Rivera-Gómez et al. 2013; Jafari and Makis 2015; Nourelfath et al. 2016); and (3) integrated production, quality, and maintenance management (e.g., Radhoui et al. 2009, 2010; Njike et al. 2012; Zequeira et al. 2008; Rivera-Gmez et al. 2016; Bouslah et al. 2016). Some researchers (see Mandroli et al. 2006) focus on determining the optimal inspection-station position in n serial production lines with or without (1) scrapping, (2) reworking, and (3) offline repairs, without considering the buffer sizing.

A large part of published articles related to the inspection suppose the inspection reliable and without errors. They assumed that the produced items are subject to 100% inspection and there are no inspection errors. Some researchers have incorporated quality inspection errors caused, among others, by human failure in their models (see Hsu and Hsu 2013; Duffuaa and Khan 2002; Khan et al. 2014).

In the context of serial production lines with production and inspection machines that follow Bernoulli reliability and quality assumptions, Meerkov and Zhang (2010, 2011) provide important insight into the nature of production and quality bottlenecks. Such systems are encountered in automotive assembly and painting operations where the downtime is relatively short and the defects are a result of uncorrelated random events (Ju et al. 2013).

Mhada et al. (2014a) consider a situation where every machine has to satisfy a demand specified as good parts per time unit. Moreover, the inventory size takes into account that the stock is a mixture of good and defective parts, and defective parts are generally eliminated at inspection stations. Mhada et al. (2014a) improve the continuous models of Kim and Gershwin (2005, 2008). The former model is more fluid in the sense that quality is treated as a continuous flow, whereas in the Kim and Gershwin models quality is considered to be discrete. They considered a line with 4 machines.

Mhada et al. (2014a) develop decomposition methods to reduce the analysis of the line to a series on an equivalent machine that can be isolated and sequentially analyzed. They consider just one inspection station. Ouzineb et al. (2013, 2014) generalize the model proposed in Mhada et al. (2014a) by optimizing the number of internal inspection stations ($m \geq 1$) and their positioning, and develop an exact search method to solve it. The exact search method (ESM) is exhaustive: the search is guaranteed to generate all possible locations of inspection stations. For each location, the problem is reformulated as a network flow optimization problem that can be efficiently solved by a fast polynomial algorithm. This method is efficient for small instances, but it may need days for reasonably large instances. The solution time increases exponentially with the size of the problem, i.e. with n and m. For larger problems, we expect that the solution time may span over several months. We can have an idea on that from Table 4.

The present paper efficiently adapts a space partitioning technique that decomposes the search space into multiple levels. At each level, we use two well-known metaheuristics to explore some potential regions. A genetic algorithm (GA) is used to identify a population of promising locations of inspection stations. Tabu search (TS) then searches intensively around by exploring locations that are neighbors to some newborns in the population. The population evolves dynamically in a sense that worst locations are discarded and good ones are added during the search. We use a quick shortest path procedure (from ESM) to determine the optimal buffers' sizes and evaluate the objective function (the fitness) for each considered location. From time to time, GA replenishes

the population. GA facilitates thus the exploration by guiding the search to unvisited regions with good potential, and this gives a certain diversification in terms of the regions to explore. The new approach is hence a multilevel hybrid heuristic (MHH) that provides certain balance between diversification and intensification. This hybrid approach is effective when the number of solutions is huge: it finds optimal or near-optimal solutions in a fraction of the time required by ESM. We also discuss some important properties of the production line and its sensitivity to system parameters.

Similar ideas have been used to solve some high-dimensional combinatorial optimization problems in other contexts. In Ouzineb et al. (2011), the authors apply a GA to solve an optimal design model for assembly/disassembly manufacturing networks. The objective is to maximize the production rate subject to a total cost constraint, the machines are chosen from a list, and the buffers' sizes are within a predetermined range. Nahas et al. (2014) applies a GA–TS algorithm to optimize the nonhomogeneous redundancy of multistate series-parallel systems. In Ouzineb et al. (2010), the authors use space partitioning to solve two design optimization problems: the first is the expansion scheduling of multistate series-parallel systems, and the second is the redundancy allocation of binary-state series-parallel systems. We adapt space partitioning and GA–TS techniques to efficiently solve the design problem addressed in this paper.

The remainder of this paper is organized as follows. "Problem formulation and discussion" section presents the problem statement and discusses some theoretical properties, and "Methods" section describes our approach. "Numerical results" section presents numerical results for ESM (Ouzineb et al. 2013, 2014) and our method. "Conclusions and future work" section provides concluding remarks.

Problem formulation and discussion

Figure 1 illustrates the production system studied. It consists of n machines separated by n buffers with $m+1$ inspection stations. The machines can be either up or down, starved or blocked. Machine M_i is starved if one of the upstream machines is down and all the buffers between that machine and M_i are empty. M_i is blocked if one of the downstream machines is down and all the buffers between that machine and M_i are full. When an operating machine is neither starved nor blocked, it continuously transfers parts from the upstream buffer to the downstream machines. We assume that the first machine can never be starved, and an inspection station is located after the last machine to ensure the quality of the parts received by the customer. The machines $M_i, i = 1 \ldots n$ can be modeled as a continuous-time Markov chain that produces a single part type with two

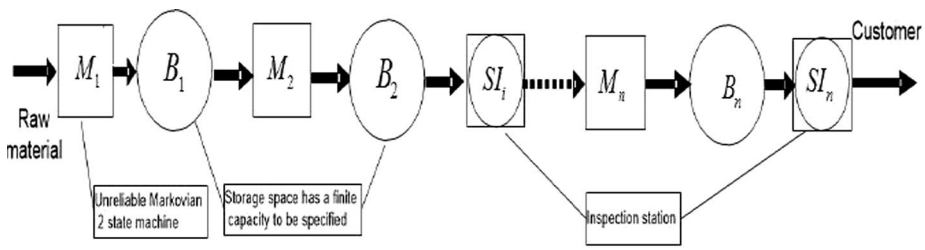

Fig. 1 Production line

quality levels, conforming and nonconforming, with a predefined ratio β of nonconforming parts to conforming parts. We assume that the machines have the same maximum production rate k, failure rate p, and repair rate r.

Let d be the demand rate for conforming parts, x_i the inventory level for buffer i, $\tilde{d}_i$ the long-term average number of parts pulled per unit of time from stock x_i, c_p the storage cost per time unit and per part, c_I the inspection cost per pulled part, and a^n_{des} the required availability rate of conforming finished parts.

A binary variable λ_i determines whether or not there is a station before machine $M_{(i+1)}$ to identify nonconforming parts. We assume that there are m inspection stations in the line (m is unknown), i.e., $\sum_{i=1}^{n-1} \lambda_i = m$, $m \in \{0, 1, \ldots, (n-1)\}$ and $\lambda_n = 1$. The problem is to minimize the long-term average global cost per unit time of storage, shortages, and inspection:

$$J_{T(\tilde{d}_i,\lambda_i)} = \lim_{T\to\infty} \frac{1}{T} \sum_{i=1}^{n} E\left[\int_0^T (c_p\, x_i(t) + c_I\, \tilde{d}_i\, \lambda_i) dt\right]. \tag{1}$$

The constraints are $\sum_{i=1}^{n-1} \lambda_i = m$, λ_i binary ($\lambda_n = 1$), and $m \in \{0, 1, \ldots, (n-1)\}$. The objective function (1) can be rewritten as (see Mhada et al. 2014a):

$$J(a, \lambda) = \sum_{i=1}^{n-1} T^{(i)}(a_i, a_{i-1}, \lambda) + T_F(a^n_{des}, a_{n-1}, \lambda) + c_I \sum_{i=1}^{n} \lambda_i\, \tilde{d}_i \tag{2}$$

with $a = (a_1, a_2, \ldots, a_{n-1})$ and $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{n-1})$. Here,

- $a_i, i = 1 \ldots n-1$ is the total work-in-progress availability coefficient at buffer x_i; this coefficient has a lower bound of $max\left[\left(\frac{r}{(r+p)}\right)^{i-1}, \frac{(r+p)\,\tilde{d}_i}{r\,k}\right]$ and an upper bound of $min\left(\left[\frac{r+p}{r}\right]^{(n-i)} a_n^{des}, 1\right)$. These bounds are proved in Ouzineb et al. (2014) (see Propositions 3.3 and 3.5 for more details). Note that $a_n = a_n^{des}$.
- $\tilde{d}_i, i = 1 \ldots n$ is the long-term average number of parts pulled per unit time from stock x_i. It depends on the positions of the inspection stations. Let $e_j, j = 1 \ldots m$ be the positions of the inspection stations, i.e.,

$$\lambda_i = \left\{ \begin{array}{ll} 1 & \text{if } i \in \{e_j,\, j = 1 \ldots m\} \\ 0 & \text{otherwise} \end{array} \right\}, \quad i = 1 \ldots n-1. \tag{3}$$

Then we have

$$\tilde{d}_i = \left\{ \begin{array}{lll} d(1+\beta)^n & \text{if} & 1 \le i \le e_1 \\ d(1+\beta)^{n-e_1} & \text{if} & e_1 < i \le e_2 \\ . & . & \\ . & . & \\ d(1+\beta)^{n-e_{m-1}} & \text{if} & e_{m-1} < i \le e_m \\ d(1+\beta)^{n-e_m} & \text{if} & e_m < i \le n \end{array} \right\}, \tag{4}$$

- and finally

$$T^{(i)}(a_i, a_{i-1}, \lambda) = c_p \left(\frac{k \frac{(r(1-a_{i-1})+p)}{a_{i-1}}}{\sigma_i (k - \frac{\tilde{d}_i}{a_i}) \frac{(r+p)}{a_{i-1}}} - \frac{k\,(1-a_i)}{\sigma_i (k - \frac{\tilde{d}_i}{a_i})} - \left[\frac{1}{\sigma_i} - \frac{(1-a_i)\frac{(r+p)}{a_{i-1}}}{\sigma_i^2 (k - \frac{\tilde{d}_i}{a_i})} \right] ln \left[\frac{\frac{(r(1-a_{i-1})+p)}{a_{i-1}} \frac{\tilde{d}_i}{a_i}}{r\,(k - \frac{\tilde{d}_i}{a_i})} - \frac{\sigma_i \frac{(r(1-a_{i-1})+p)}{a_{i-1}} \frac{\tilde{d}_i}{a_i}}{\frac{(r+p)}{a_{i-1}} r\,(1-a_i)} \right] \right), \quad i = 1, \ldots, n-1, \tag{5}$$

$$T_F(a^n_{des}, a_{n-1}, \lambda) = \frac{\rho_n c_p \left(\frac{k\,(1-\exp(-\mu_n(1-\rho_n) z_n(a_n^{des}))}{1-\rho_n} - \frac{(r+p)}{a_{n-1}} z_n(a_n^{des}) \exp(-\mu_n(1-\rho_n) z_n(a_n^{des}) \right)}{\frac{(r+p)}{a_{n-1}} (1 - \rho_n \exp(-\mu_n(1-\rho_n) z_n(a_n^{des})))} \tag{6}$$

with $\sigma_i = \frac{(\frac{(r+p)}{a_{i-1}}) \frac{\tilde{d}_i}{a_i} - k\,r}{(k - \frac{\tilde{d}_i}{a_i}) \frac{\tilde{d}_i}{a_i}}$, $\rho_n = \frac{r(k - \frac{\tilde{d}_n}{a_n^{des}})}{\frac{(r(1-a_{n-1})+p)}{a_{n-1}} \frac{\tilde{d}_n}{a_n^{des}}}$, $\mu_n = \frac{(r(1-a_{n-1})+p)}{a_{n-1}(k - \frac{\tilde{d}_n}{a_n^{des}})}$, and

$$z_n(a_n^{des}) = -\frac{\ln \left[\frac{1}{\rho_n} \left(1 - \frac{(1-\rho_n)}{(1-a_n^{des})(\frac{(r+p)}{(r(1-a_{n-1})+p)})} \right) \right]}{\mu_n(1-\rho_n)}.$$

The problem can now be formulated as follows:

$$\text{minimize} \quad J(a, \lambda) \tag{7}$$

$$\text{subject to} \quad \sum_{i=1}^{n-1} \lambda_i = m \tag{8}$$

$$max \left[\left(\frac{r}{(r+p)} \right)^{i-1}, \frac{(r+p)\,\tilde{d}_i}{r\,k} \right] \leq a_i \leq min \left(\left[\frac{r+p}{r} \right]^{(n-i)} a_n^{des}, 1 \right), \quad \forall i,\ 1 \leq i \leq n-1 \tag{9}$$

$$\lambda_i \in \{0, 1\}, \quad \forall i,\ 1 \leq i \leq n-1 \tag{10}$$

$$m \in \{1, 2, \ldots, n-1\}. \tag{11}$$

We aim to find the optimal solution, i.e. the minimal average global cost system structure, (a, λ, m) that minimizes J and satisfies the constraint (8) on the number of inspection stations, constraints (9) on buffers sizes bounds, and integrality constraints (10)–(11) on admissible values of λ and m. The proposition below discusses the sensitivity of the production line to c_I. A numerical example is given in "Sensitivity analysis" section to illustrate this result.

Proposition 1 *The total cost as a function of c_I is a piecewise linear function. The slope of the line depends on the number of inspection stations m and their positions λ.*

Proof The minimum total storage and inspection cost is

$$J(a,\lambda) = \sum_{i=1}^{n-1} T^{(i)}(a_i, a_{i-1}, \lambda) + T_F(a^n_{des}, a_{n-1}, \lambda) + c_I \sum_{i=1}^{n} \lambda_i \tilde{d}_i.$$

The optimal total storage and inspection cost as a function of c_I is

$$\begin{aligned} J(a^*,\lambda^*) &= \sum_{i=1}^{n-1} T^{(i)}(a^*_i, a^*_{i-1}, \lambda^*) + T_F(a^n_{des}, a^*_{n-1}, \lambda^*) + c_I \sum_{i=1}^{n} \lambda^*_i \tilde{d}^*_i \\ &= Constant(c_I) + c_I \sum_{i=1}^{n} \lambda^*_i \tilde{d}^*_i \end{aligned} \tag{12}$$

where $(a^*,\lambda^*) = argmin(J(a,\lambda))$. Observe that $\sum_{i=1}^{n-1} T^{(i)}(a^*_i, a^*_{i-1}, \lambda^*) + T_F(a^n_{des}, a^*_{n-1}, \lambda^*)$ does not depend on the value c_I, so it is constant on c_I and referred to as $Constant(c_I)$. Thus, we can write Eq. (12) as follows:

$$J(a^*,\lambda^*) = Constant(c_I) + c_I \left(\sum_{i=1}^{m} d(1+\beta)^{n-e_i} + d(1+\beta)^n \right). \tag{13}$$

Hence, the total cost as a function of c_I is a piecewise linear function; its slope depends only on m and the positions of the inspection stations. □

The cost function J is likely to be convex as conjectured by Conjecture 1 and confirmed by the numerical results in "Characteristics of the production line" section. The convexity of the cost function is used to reduce the number of iterations. Actually, the local minimum of a convex function is also a global minimum, and there are many specialized methods and mathematical tools for optimizing convex functions.

Conjecture 1

- The minimal total cost J is a convex function of λ.
- The minimal total cost J is a convex function of the number of inspection stations m.
- If we divide the production line into two parts with the same number of machines and we fix the number of stations in the first part, the minimal total cost J is a convex function of the number of internal stations in the second part.

Methods

In Ouzineb et al. (2013, 2014), authors present an exact search methd (ESM) for the problem studied in this paper. For a given location of inspection stations, they show that the optimizing of the buffers' sizes with ESM can be quickly done in polynomial time. Actually, the bottleneck of ESM is the number of possible locations. This number could be reduced by using the property of convexity (Conjecture 1) but nevertheless, it would stay very large. The innovation in this paper is to smartly select potential locations using an evolutionary algorithm combined with tabu search. We summarize ESM in "Exact search method (ESM)" section and then present our new approach in "Multilevel hybrid heuristic (MHH)" section.

Exact search method (ESM)

For a fixed λ, Ouzineb et al. (2013, 2014) shows that the term $T^{(i)}(a_i, a_{i-1}, \lambda)$, and thus the objective function itself, is separable by the variables a_i. ESM reformulates then the problem defined by (7)–(11), for a given λ, as a shortest path problem defined on a network (described below). It uses a standard shortest path algorithm to find the optimal buffers' sizes. For this, we discretize the continuous variables a_i to take discrete values 1, 2, ..., 100%. For a fixed λ, the value of m is determined straightforwardly. So, the constraints (8), (10), and (11) are satisfied. They do not depend on a and consequently could be omitted.

Consider the connected network $G(N, A)$ consisting of a set of nodes N and a set of arcs A, as depicted in Fig. 2. With each machine we associate a set of nodes representing all the possible buffers' sizes, which range from 1 to 100%. The set N consists of $100 \times (n-1)$ nodes (representing the $n-1$ machines) plus two additional nodes N_0 and N_n that are adjacent to the nodes representing the first and last machines. The set A consists of $100^2 \times (n-2) + 200$ arcs weighted as follows:

- N_0 is connected to the nodes N_1^j, $j = 1, \ldots, 100$ representing machine 1. The weight of each arc (N_0, N_1^j) is null. N_n is connected to the nodes N_{n-1}^j, $j = 1, \ldots, 100$ representing machine $n-1$. The weight of each arc (N_{n-1}^j, N_n) is $c_{n-1,n}^j = T_F(a_n^{des}, j, \lambda) + c_I \times \tilde{d}_n, 1 \leq j \leq 100$.
- Each node N_i^j representing machine i is connected to each node N_{i+1}^k representing machine $i+1$ by an arc (N_i^j, N_{i+1}^k) that is weighted $c_{i,i+1}^{jk} = T^{(i)}(k, j, \lambda) + c_I \times \lambda_i \times \tilde{d}_i$, $i = 1, \ldots, n-2$ and $1 \leq j, k \leq 100$.

Nodes, and all incident arcs, corresponding to values that are not in the intervals imposed by constraints (9) are removed from the network. Algorithm 1 presents ESM.

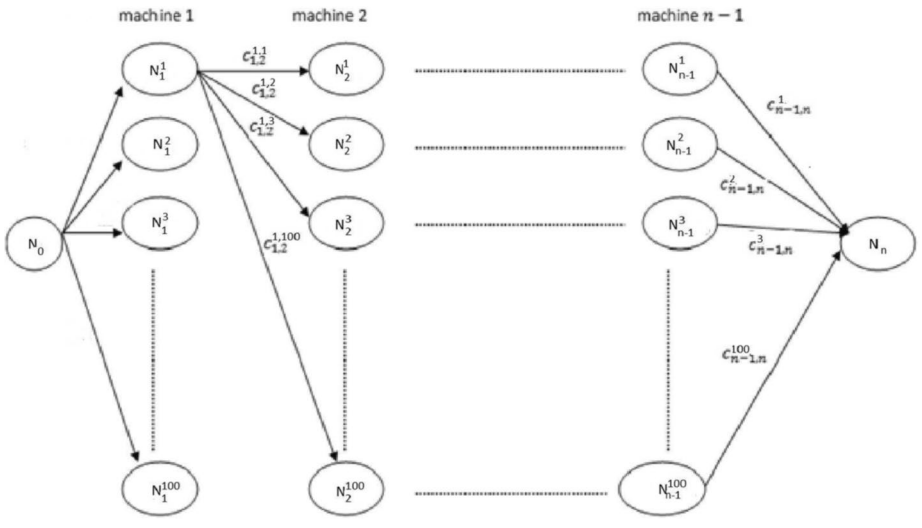

Fig. 2 Network flow problem

It finds optimal solutions but takes hours for instances with 20 machines. It is unable to solve some larger instances with 30 machines in one month!

Algorithm 1 Exact search method (ESM)

$J^* = \infty$.
for all λ **do**
  Evaluate the optimal cost function J, called also Fitness, and a for the current λ using Algorithm 2.
  if $J < J^*$ **then**
    $J^* = J$, $a^* = a$, $\lambda^* = \lambda$.
  end if
end for
return (J^*, a^*, λ^*).

Algorithm 2 Fitness evaluation

Given λ.
Construct the network G.
Find a shortest path in G (i.e. the buffers sizes vector a) and set J to its cost.
return (J, a).

Multilevel hybrid heuristic (MHH)

Our approach combines ideas from ESM and metaheuristic techniques. The choice of a search space and a neighborhood structure is a critical step in metaheuristic design (see Gendreau 2002). Our search space is the space of all possible line structures, i.e. the vectors (a, λ, m) that are feasible solutions to the system (8)–(11). As discussed above, the most determinant element of these vectors is λ because if we know λ, we can compute m in a straightforward manner and a in a polynomial time. Consequently, our search space S in this section is composed of all possible λ values. We reduce it by: (1) partitioning this space into a set of disjoint subspaces as explained in "Partitioning the search space" section; (2) applying a hybrid heuristic to selected subspaces in order to find potential solutions; we use Algorithm 2 above to evaluate the fitness of each selected λ (in a subspace) and to find the optimal buffers' sizes a. See "Hybrid heuristic" section for more details. The multilevel hybrid heuristic pseudocode is then given in "MHH pseudocode" section. This approach finds, as reported in the numerical results, near-optimal solutions in a fraction of the time required by ESM.

Partitioning the search space

We refer hereunder to λ as a solution vector or simply solution. Each solution vector is assigned an address, also called a *level*, as explained below.

Definition 1 We divide the solution vector λ into two equal parts and create its address r as follows: $r = Address(\lambda) = \sum_{l=1}^{\lfloor n/2 \rfloor} \lambda_l$. A search subspace of address r, denoted S_r, is defined to be the set of solutions that have the same address, i.e. equal to r.

The lower bound of r is 0 and its upper bound is m. It is clear that $(S_r)_{1 \leq r \leq m}$ is a partition of the space S, i.e., $S_{r_1} \bigcap S_{r_2} = \emptyset \; \forall \, r_1 \neq r_2$ and $S = \bigcup_{r=0}^{m} S_r$.

Example Consider a system with $n = 20$ machines and $m = 5$ inspection stations: 2 stations in the first part and 3 in the second part:

$$\boldsymbol{\lambda} = [(0, 0, 1, 0, 0, 0, 0, 1, 0, 0), (\mathbf{0}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{1})].$$

From the above definition, $address(\boldsymbol{\lambda}) = 2$. So, $\lambda \in S_2$.

Hybrid heuristic

The goal here is to locate promising regions in a given subspace S_r. Our hybrid optimization is based on a combination of a GA, which provides diversification, and TS, which provides intensification. The hybrid heuristic pseudocode is presented in Algorithm 3 and a flowchart is given in Fig. 3. N_{rep} and N_c are GA tuned by experimentation. The main steps are explained in the paragraphs that follow.

Algorithm 3 Hybrid heuristic algorithm

Given *level* r, N_{rep}, N_c
$J^* = \infty$, $k = 0$, $l = 0$.
Step 1. Generate a random population of N_s (chromosomes) solutions with the same address r.
Step 2. Select randomly from the population two parent solutions λ_1 and λ_2 that will "reproduce" using a 1-point crossover operator.
Step 3. Produce a new solution λ (child or offspring) from the selected pair of parent solutions *Parent* 1 and *Parent* 2.
Step 4. Find the best solution λ' in $V(\lambda)$, the effective neighborhood of the new child λ, using TS. Evaluate its value J' and its corresponding optimal a'.
if $J' < J^*$ **then**
  $J^* = J'$, $a^* = a'$, $\lambda^* = \lambda'$.
end if
Step 5. If λ' is better than the worst solution currently in the population, it replaces that solution; otherwise, it is discarded.
if $k < N_{rep}$ **then**
  $k = k + 1$ and go to Step 2.
end if
Step 6. Decide if the population needs to be replenished and then to start a new generation:
if $l < N_c$ **then**
  $k = 0$, $l = l + 1$, go to Step 1.
end if
return (J^*, a^*, λ^*).

In Step 3, we use a 1-point crossover operator, which, as illustrated in Table 1, creates a child string $\boldsymbol{\lambda}$ for a given pair of parent strings $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ by:

1. Copying the string elements belonging to the first part from $\boldsymbol{\lambda}_1$;
2. Copying the rest of the string elements from $\boldsymbol{\lambda}_2$.

Observe that the child has the same address as its parents.

In Step 4, we use TS, in which we iteratively move from the current solution λ to a new solution in its neighborhood *Neighborhood*($\boldsymbol{\lambda}$) until some stopping criterion has been satisfied. For a given λ, we define Neighborhood($\boldsymbol{\lambda}$) = {solutions (inspection stations configurations) obtained by applying a single move to $\boldsymbol{\lambda}$}. Moving to a new solution involves changing the positions of two inspection stations; the number of stations does not change. Note that we stay in the same level when we move to a neighbor solution. We use Algorithm 2 to evaluate the fitness of each new solution, and we use theses fitness values to compare different solutions.

TS uses a memory structure: a potential solution is declared *tabu* so that the algorithm does not visit it again; this prevents cycling. At each iteration, we select the best solution $\boldsymbol{\lambda}'$ in a subspace V($\boldsymbol{\lambda}$) $\subset$ Neighborhood($\boldsymbol{\lambda}$) as a tabu solution. The subspace V($\boldsymbol{\lambda}$), called the effective neighborhood, is obtained by eliminating the tabu solutions from Neighborhood($\boldsymbol{\lambda}$). The tabu solutions are stored in a short-term memory, called a tabu list, which contains the solutions that have been visited in the recent past. We use a variable-size tabu list (tabu tenure) because this is generally more efficient (Gendreau 2002; Ouzineb

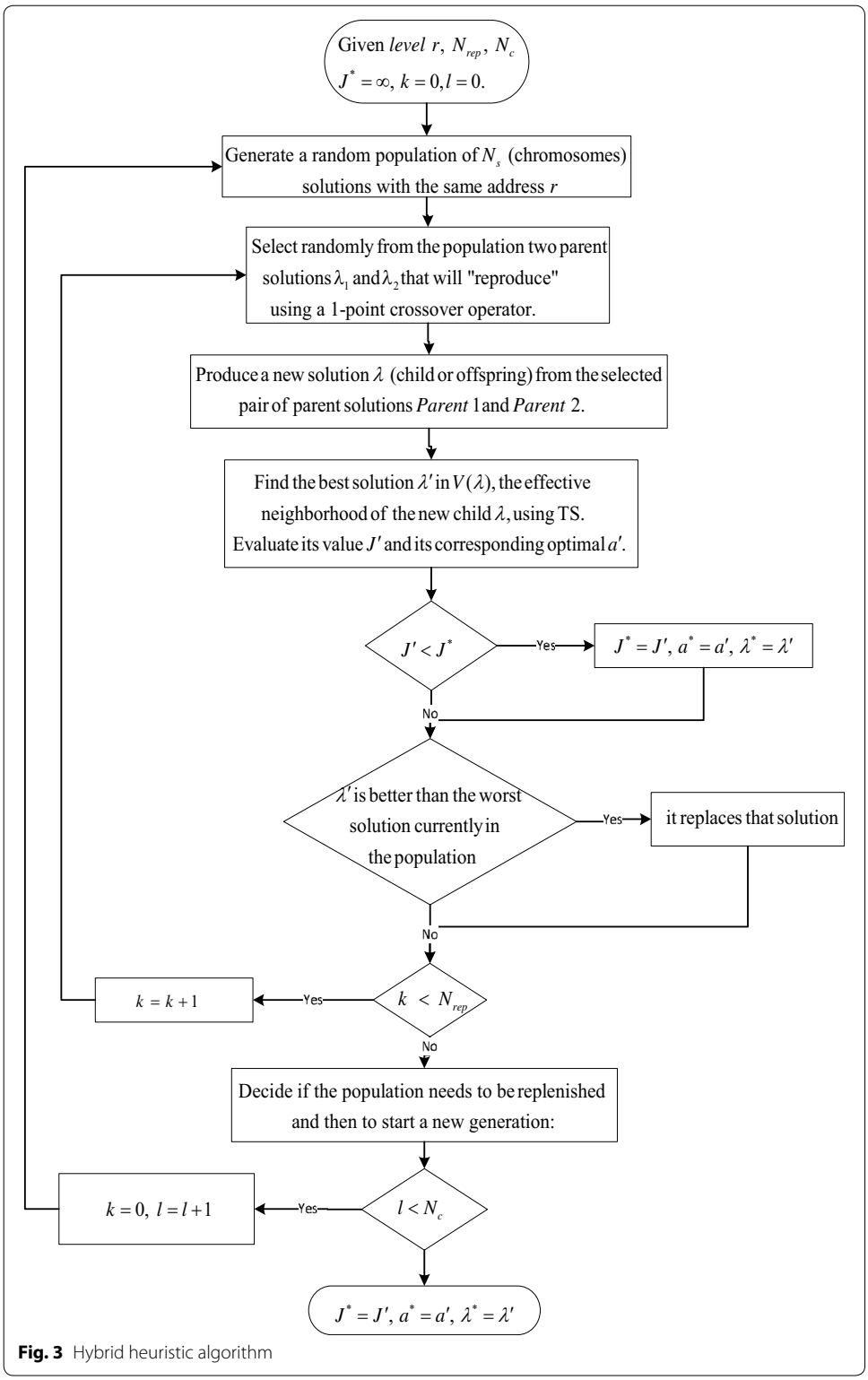

Fig. 3 Hybrid heuristic algorithm

et al. 2008). Finally, our stopping criterion is defined by the maximum number of local iterations (*mnli*), i.e. the number of visited neighbors that do not improve the best-known solution. The value of *mnli* is determined by the experimentation.

Table 1 Example of 1-point crossover operator

Parent string $\boldsymbol{\lambda}_1$	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	1
Parent string $\boldsymbol{\lambda}_2$	*0*	*0*	*0*	*0*	*1*	*0*	*1*	*0*	*0*	*0*	*0*	*1*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*1*
Child string	0	0	0	1	0	0	1	0	0	0	*0*	*1*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*1*

Italic values indictate the chromosome of Parent $\boldsymbol{\lambda}_2$

MHH pseudocode

In MHH, at each iteration we move from the current level to a new level. We divide the search space into a set of disjoint subspaces (levels) using the partitioning technique of "Partitioning the search space" section and then apply the heuristic presented in "Hybrid heuristic" section to each selected subspace. Algorithm 4 presents the proposed pseudo code and Fig. 4 the corresponding flowchart.

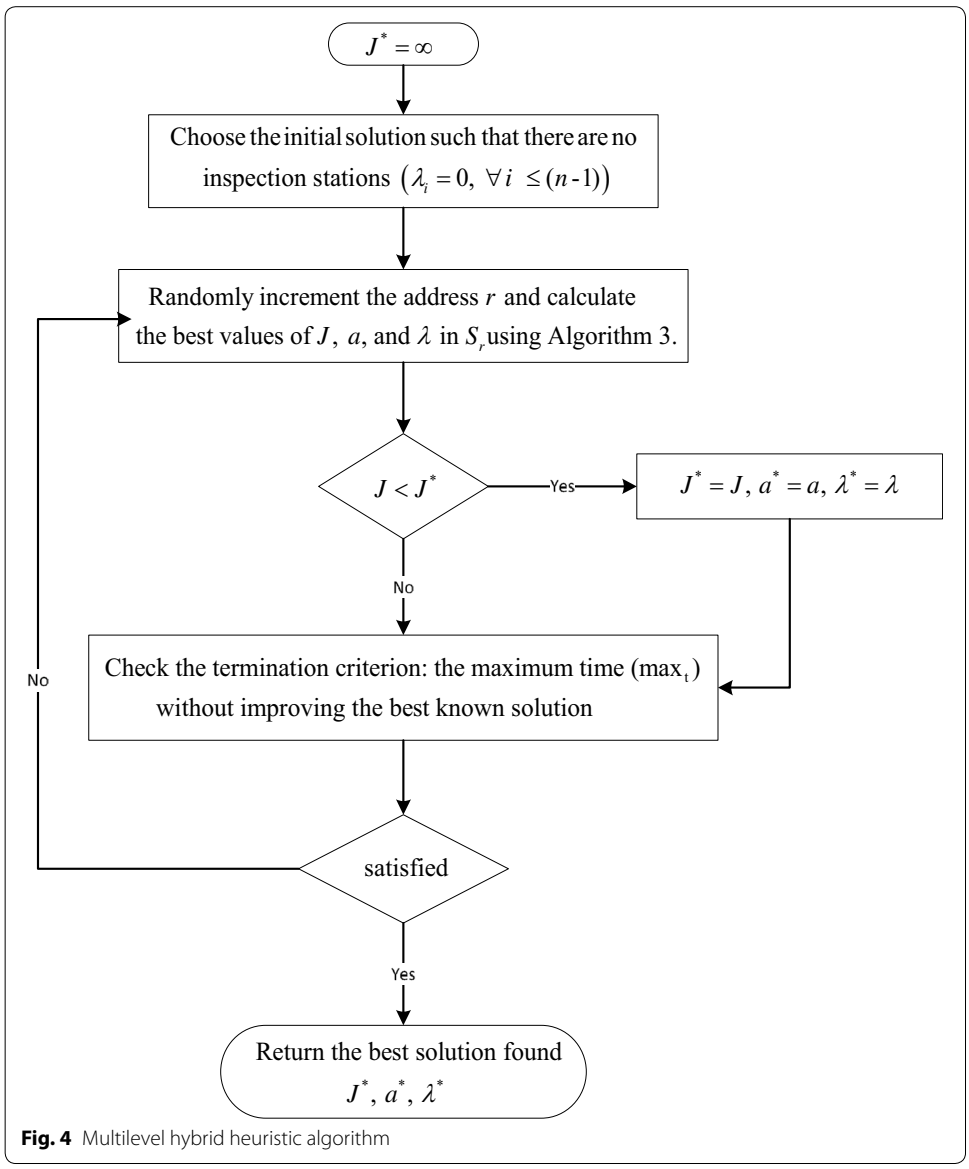

Fig. 4 Multilevel hybrid heuristic algorithm

Algorithm 4 Multilevel hybrid heuristic

$J^* = \infty$.
Step 1. Choose the initial solution such that there are no inspection stations ($\lambda_i = 0\ \forall i \leq n-1$).
Step 2. Randomly increment the address r and calculate the best values of $J, a,$ and λ in S_r using Algorithm 3.
if $J < J^*$ **then**
$J^* = J$, $a^* = a$, $\lambda^* = \lambda$.
end if
Step 3. Check the termination criterion: the maximum time (max_t) without improving the best known solution. If it is satisfied, return the best solution found. Otherwise, go to step 2.
return (J, a, λ).

Numerical results

We use four optimization problems (benchmarks) to test our algorithms: I10 has 10 machines, I20 has 20 machines, I30 has 30 machines and I40 has 40 machines. To get an insight into the algorithms behavior, we fixed the value of m in the model (7)–(11). The optimal objective value can be obtained equivalently by varying m between $1, \ldots, n-1$, sloving the resulting models and take the minimum possible. We implemented MHH and ESM in C^{++} under the same conditions (computer architecture, operating system, etc). The tests were performed on an Intel Core i7 at 2.8 GHz with 8 GB of RAM, running Linux. Table 2 shows the parameters we used.

Comparing MHH to ESM

ESM can find the optimal solutions for I10 and I20. We compare the MHH solutions to the optimal solutions to evaluate the performance of MHH. ESM is unable to solve I30 and I40, which has been newly proposed in this paper, for large m. The reduction percentage in terms of CPU time (RPCPU) is defined as follows:

$$\text{RPCPU} = 100 \times \frac{(\text{ESM Time} - \text{MHH Time})}{\text{ESM Time}}\%. \tag{14}$$

For I10, MHH and ESM find the same optimal solutions in comparable times. Tables 3 and 4 give the results for I20 and I30, with $m = 1, \ldots, 10$. The first column indicates the number m of inspection stations. The MHH cost is given in the second column; if the solution is optimal the entry is in italics. MHH finds optimal solutions for all the instances solved by ESM. The column MHH/Best gives the time to find the best solution, and the column MHH gives the total computational time. ESM takes hours on I20 for $m \geq 5$. MHH outperforms ESM in terms of the computational time; for I20, it finds

Table 2 Parameters

	n	p	r	k	β	d	c_p	c_l	a_n^{des}	$max_t(s)$	N_c	N_{rep}	$mnli$
10 machines	10	0.2	0.9	4	0.1	1	1	2	0.95	10	2	3	5
20 machines	20	0.2	0.9	9	0.1	1	0.1	0.2	0.95	600	2	5	5
30 machines	30	0.2	0.9	22	0.1	1	0.1	0.2	0.95	3600	2	5	5
40 machines	40	0.2	0.9	9	0.05	1	0.1	0.2	0.95	3600	2	2	5

Table 3 Results for I20

m	MHH cost	GA cost	Running time (s)				
			ESM	*MHH/Best*	*MHH*	*RPCPU*	GA
1	*8.3125*	9.23149	15.3	34.3	177.6	−91.39	457.0
2	*6.2645*	6.90041	105.7	77.4	440.5	−76.01	983.0
3	*5.9802*	6.28064	421.7	422.0	1011.9	−58.33	1524.4
4	*6.0663*	6.32419	1601.2	591.7	1191.7	25.58	1364.0
5	*6.2772*	6.51936	4743.3	571.2	879.4	81.46	1402.9
6	*6.5230*	6.76087	11,108.7	631.4	992.0	91.07	1649.9
7	*6.7959*	7.02709	20,613.0	635.2	1108.9	94.62	1710.7
8	*7.0965*	7.37334	29,790.2	960.7	1560.7	94.76	1255.8
9	*7.4298*	7.67294	35,764.9	476.5	1076.5	96.99	1176.9
10	*7.7947*	8.05445	35,767.6	109.0	709.0	98.02	977.7

Italic values indictate optimal costs

Table 4 Results for I30

m	MHH cost	Running time (s)			
		ESM	*MHH/Best*	*MHH*	*RPCPU*
1	*40.7297*	31.2	109.6	829.3	−96.24
2	*22.9847*	389.4	1186.7	1554.6	−74.95
3	*20.0887*	3524.2	3779.7	3926.9	−10.25
4	*19.0494*	18,769.2	4470.1	5671.9	69.78
5	*18.9133*	115,432.0	6584.2	8065.0	93.01
6	*19.1256*	421,630.0	10,034.2	13,634.2	96.77
7	19.4257	–	10,713.9	14,313.9	–
8	19.7315	–	12,300.6	15,900.6	–
9	20.1073	–	10,549.2	14,149.2	–
10	20.3278	–	11,128.0	14,728.0	–

Italic values indictate optimal costs

optimal solutions in a fraction of ESM time. We also report results for a "pure" genetic algorithm (GA) which is obtained by modifying Algorithm 3 as follows:

- Remove Step 4.
- Replace Step 5 by: If the new solution is better than the worst solution currently in the population, it replaces that solution; otherwise, it is discarded.

GA is coded in the same technology and runs on the same machine as MHH and ESM. MHH results are signifcantly better in terms of quality and CPU time than GA. For larger instances, we omit GA and present results of MHH only.

For I30, MHH finds "optimal" solutions in a reasonable time. ESM cannot solve the instances with $m \geq 7$. For example, for $m = 7$, ESM did not converge after a month of running time. We believe that the MHH solution is optimal or near-optimal for $m = 7$ and beyond, but we cannot confirm this. Large problems require fast heuristics such as MHH that provide good results in an acceptable time.

Table 5 presents results for I40 which consists of very large instances. In this table, we report the cost values for both MHH and ESM because they are slightly different for

Table 5 Results for I40

m	ESM cost	MHH cost	Running time (s)		
			ESM	*MHH/Best*	*MHH*
1	*25.2938*	*25.2938*	109,147	194.46	387.83
2	*16.0119*	16.1078	1105,72	1008.23	1013.94
3	*13.3200*	13.3208	13,200,38	2336.34	2623.08
4		12.5685		3932.31	5827.07
5		12.3450		6092.55	6920.45
6		12.5043		5266.51	11,511.62
7		12.6973		10,604.86	16,425.48
8		12.8781		18,929.32	23,346.94
9		13.1145		12,666.15	25,374.23
10		13.3623		15,501.52	31,699.22

Italic values indictate optimal costs

$m = 2$ and $m = 3$. We did not compute *RPCPU* for the same reason. For I40, MHH finds good solutions in a reasonable time while ESM takes too much time for $m = 3$ and cannot solve the instances with $m \geq 4$; ESM did not converge after many hours of running time. The MHH solution is optimal or near-optimal for $m = 3$ and below. We think that the optimal solution is obtained when $m = 5$ despite the fact that we cannot solve it by the exact method.

Table 6 shows the results for I20 when the computational time is limited to one hour. Both methods find optimal solutions for $m \leq 4$; MHH finds better solutions for $m \geq 5$.

Characteristics of the production line

Figure 5 shows the optimal cost as a function of (the number of internal inspection stations) $m = 1, 2, \ldots, 19$. The cost is a convex function of m. The optimal number of internal inspection stations is $m = 3$, and the optimal positions are $\lambda_2 = 1$, $\lambda_7 = 1$, and $\lambda_{18} = 1$. The results obtained by MHH and ESM are identical.

Figure 6 shows the best cost found by MHH as a function of $m = 1, 2, \ldots, 19$. The cost is a convex function of m. The optimal value for m is $m = 5$, and the optimal positions are $\lambda_1 = 1$, $\lambda_3 = 1$, $\lambda_7 = 1$, $\lambda_{15} = 1$, and $\lambda_{29} = 1$. These results support Conjecture

Table 6 MHH and ESM results for I20 with computational time limited to one hour

m	*MHH cost*	*ESM cost*
1	*8.3125*	*8.3125*
2	*6.2645*	*6.2645*
3	*5.9802*	*5.9802*
4	*6.0663*	*6.0663*
5	*6.2772*	6.4972
6	*6.5230*	7.0839
7	*6.7959*	7.4549
8	*7.0965*	7.8745
9	*7.4298*	8.4677
10	*7.7947*	8.8909

Italic values indictate optimal costs

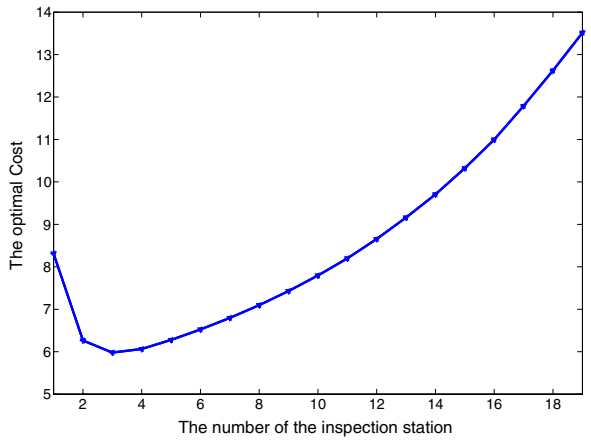

Fig. 5 Optimal cost as a function of $m = 1, 2, \ldots, 19$ for I20

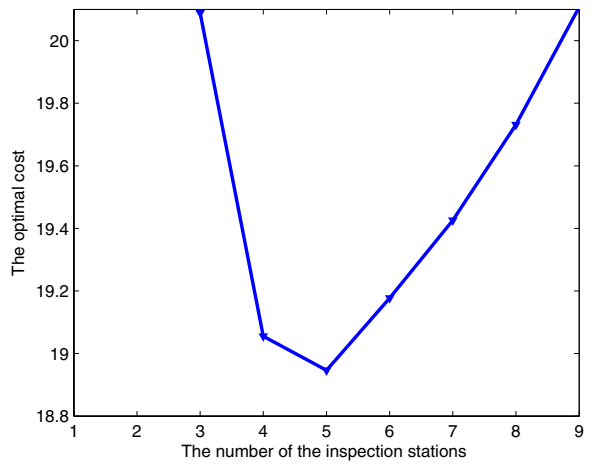

Fig. 6 Optimal cost as a function of the number of inspection stations for I30

1. Convexity can play an important role in reducing the solution time. Using convexity, we can conjecture that even I30 is solved optimally by ESM and MHH because the cost starts increasing for $m \geq 6$.

Figure 7 shows the best cost found by MHH as a function of $m = 1, 2, \ldots, 10$ for I40. The cost is a convex function of m. The optimal value for m is $m = 5$.

Sensitivity analysis

We now study the sensitivity of the optimal cost and number of inspection stations to the parameters of the model. We study particularily β and c_I because they directly affect quality. Table 7 gives the parameters used for this study; we investigate I20.

Inspection cost c_I

We begin by varying c_I, the inspection cost per time unit and per part (for a fixed $\beta = 0.1$). The upper plot in Fig. 8 shows that for values of c_I close to c_p (the storage cost per time unit and per part), the optimal number of stations is high, while for high values

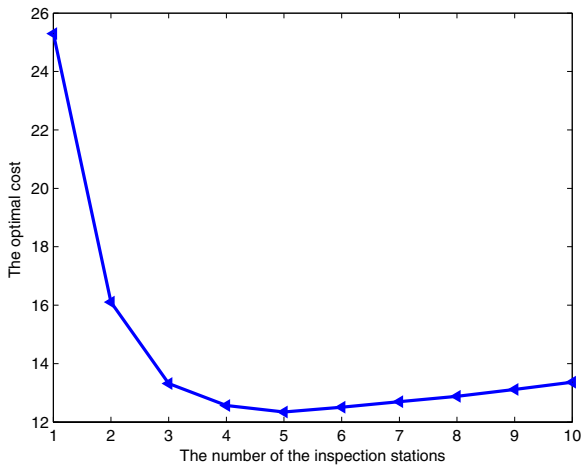

Fig. 7 Optimal cost as a function of the number of inspection stations for I40

Table 7 Parameters

	n	p	r	k	d	c_p	a_n^{des}
20 machines	20	0.2	0.9	9	1	0.1	0.95

of c_I, the optimal number decreases until it reaches zero. This is because the system encourages storage when it is profitable to do so, and it reduces nonconforming parts by imposing more inspection stations when that option is better.

When c_I increases, by Proposition 1, $\left(\sum_{i=1}^{m} d(1+\beta)^{n-e_i}\right)$ must decrease (m or $(n-e_i)$ decreases). The stations will thus be closer to the end of the line. When a station is placed after machine $n-1$, any increase in c_I will make it unnecessary, so the optimal number of stations will be reduced by 1 (see Table 8).

Figure 9 shows the cases $m=1$ (upper plot) and $m=0$ (lower plot). For $m=1$ and for each station position, the total cost as a function of c_I is linear. For $m=0$, the total cost as a function of c_I is linear.

Fraction β of nonconforming to conforming parts

Figure 10 and Table 9 show the behavior of the line as a function of β. As β increases, the quantity of nonconforming parts increases. If we do not eliminate the nonconformity from the system, the machines will work harder and the stocks will fill faster, and the line will not be able to meet the final demand. Here the cost varies exponentially (Fig. 11).

Conclusions and future work

We proposed an efficient hybrid approach, based on ideas from an exact method and metaheuristics, for the buffer sizing problem in unreliable homogeneous production lines with several inspection stations. This is a difficult mixed integer nonlinear program. Our approach combines a genetic algorithm and tabu search to identify profitable configurations (locations of inspection stations). For these locations, we use an exact approach to decide the optimal buffers' sizes. Our final goal is to find an optimal

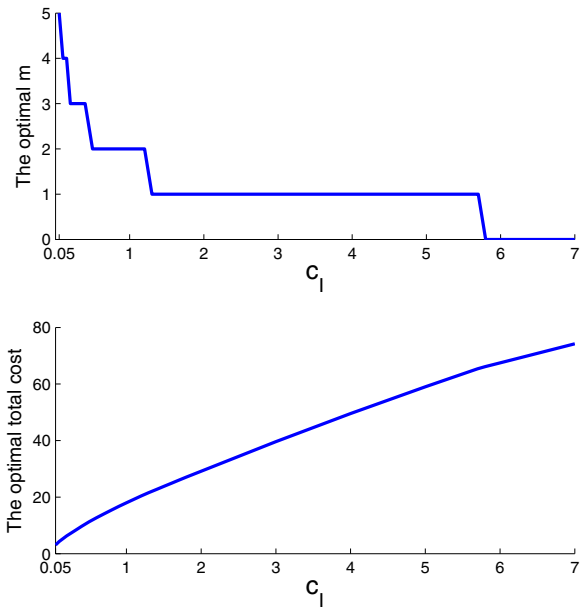

Fig. 8 Optimal number of inspection stations and cost as a function of c_l

Table 8 Optimal location of inspection stations

c_l	0.2	0.3	0.5	0.6	0.8	1.2	1.3	2	2.7	3.3	3.9	4.5	5.2	5.7
Optimal m	3	3	2	2	2	2	1	1	1	1	1	1	1	1
Optimal positions	2; 7; 18	2; 7; 19	3; 11	3; 13	3; 14	4; 19	5	6	7	8	9	10	11	11

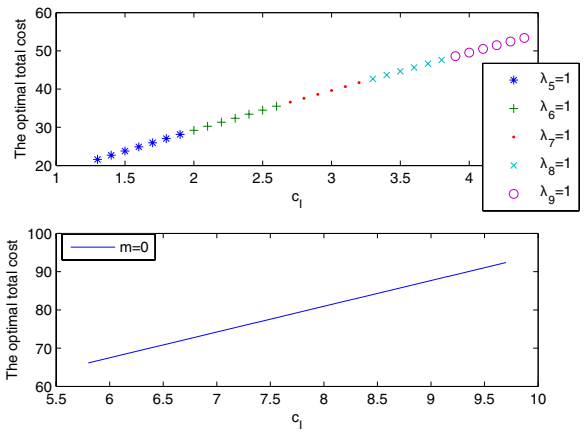

Fig. 9 Optimal total cost as a function of c_l ($m^* = 1$and $m^* = 0$)

or near-optimal design as rapidly as possible. This hybrid approach provides a balance between diversification and intensification and works well on homogeneous production lines with up to 40 machines. MHH is significantly faster than ESM and produces solutions that are equally good.

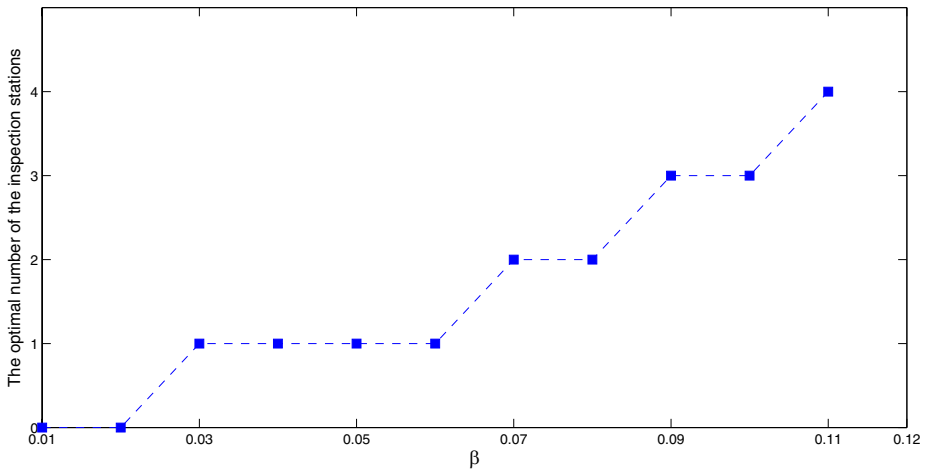

Fig. 10 Optimal number of inspection stations as a function of β

Table 9 Optimal station locations

β	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1	0.11
Optimal m	1	1	1	1	2	2	3	3	4
Optimal positions	18	18	18	16	8; 18	6; 18	4; 10; 19	2; 7; 18	2; 4; 8; 18

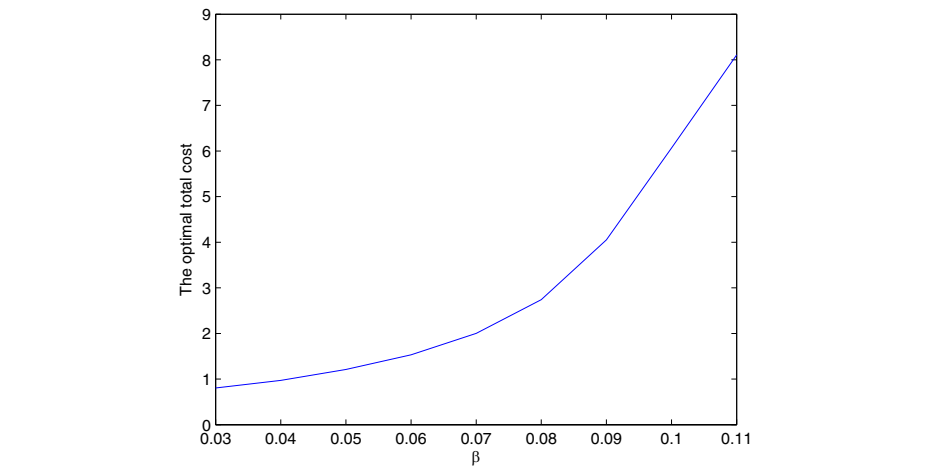

Fig. 11 Optimal total cost as a function of β

Future research should focus on developing an optimization method for more realistic nonhomogeneous production lines and solving it by combining simulation and optimization. The model developed here could be used as an approximation model for more complex real life design problems. Fast optimization techniques, like the one proposed

in this paper, would help in rapidly selecting potential scenarios/configurations for a more realistic simulation, and hence reducing the solution time without losing solution applicability in real life context.

Authors' contributions

FM developed the theoretical study, designed the numerical study and participated in the writing of the paper. MO coded the algorithms, carried out the numerical testing and participated in the writing of the paper. RP participated in the paper coordination and helped to draft the manuscript. IEH participated in the design of the paper, coordinated the development works and finalized the writing. All authors read and approved the final manuscript.

Author details

[1] GERAD and University Mohammed V-Rabat/ENSIAS, B.P. 713, Rabat-Instituts, Madinat Al Irfane, Rabat, Morocco. [2] GERAD and Institut National de Statistique et d'Economie Appliquée, B.P. 6217, Rabat-Instituts, Madinat Al Irfane, Rabat, Morocco. [3] CIRRELT and Polytechnique Montreal, CP 6079, succ. Centre-Ville, Montreal, QC, Canada. [4] GERAD and Polytechnique Montreal, CP 6079, succ. Centre-Ville, Montreal, QC, Canada.

Competing interests

The authors declare that they have no competing interests.

Received: 4 June 2016 Accepted: 26 November 2016
Published online: 30 November 2016

References

Ben-Daya M (2002) The economic production lot-sizing problem with imperfect production processes and imperfect maintenance. Int J Prod Econ 76:257–264

Bouslah B, Gharbi A, Pellerin R (2016) Integrated production, sampling quality control and maintenance of deteriorating production systems with AOQL constraint. Omega 61:110–126

Colledani M, Tolio T (2005) Impact of statistical process control (SPC) on the performance of production systems—part 2 (large systems). In: 5th international conference on analysis of manufacturing systems-production management, Zakynthos Island, Greece

Colledani M, Tolio T (2006) Impact of quality control on production system performance. Ann CIRP 55(1):453–456

Colledani M, Tolio T (2009) Performance evaluation of production systems monitored by statistical process control and off-line inspections. Int J Prod Econ 120(2):348–367

Colledani M, Tolio T (2011) Integrated analysis of quality and production logistics performance in manufacturing lines. Int J Prod Res 49(2):485–518

Dhouib K, Gharbi A, Aziza MB (2012) Joint optimal production control/preventive maintenance policy for imperfect process manufacturing cell. Int J Prod Econ 137(1):126–136

Duffuaa SO, Khan M (2002) An optimal repeat inspection plan with several classifications. J Oper Res Soc 53(9):1016–1026

Gendreau M (2002) Recent advances in tabu search. In: Ribeiro C, Hansen P (eds) Essays and surveys in metaheuristics. Kluwer Academic Publishers, Dordrecht, pp 369–377

Hajji A, Mhada F, Gharbi A, Pellerin R, Malhamé R (2010) Integrated product specifications and productivity decision making in unreliable manufacturing systems. Int J Prod Econ 129(1):32–42

Hsu J-T, Hsu L-F (2013) An EOQ model with imperfect quality items, inspection errors, shortage backordering, and sales returns. Int J Prod Econ 143(1):162–170

Inman R, Blumenfeld D, Huang N, Li J (2013) Survey of recent advances on the interface between production system design and quality. IIE Trans 45(6):557–574

Jafari L, Makis V (2015) Joint optimal lot sizing and preventive maintenance policy for a production facility subject to condition monitoring. Int J Prod Econ 169:156–168

Ju F, Li J, Xiao G, Arinez J (2013) Quality flow model in automotive paint shops. Int J Prod Res 51(21):6470–6483

Khan M, Jaber MY, Ahmad A-R (2014) An integrated supply chain model with errors in quality inspection and learning in production. Omega 42(1):16–24

Kim J, Gershwin S (2005) Integrated quality and quantity modeling of a production line. OR Spectr 27:287–314

Kim J, Gershwin S (2008) Analysis of long flow lines with quality and operational failures. IIE Trans 40:284–296

Kutzner SC, Kiesmüller GP (2013) Optimal control of an inventory-production system with state-dependent random yield. Eur J Oper Res 227(3):444–452

Mandroli S, Shrivastava A, Ding Y (2006) A survey of inspection strategy and sensor distribution studies in discrete-part manufacturing processes. IIE Trans 38(4):309–328

Matta A, Simone F (2016) Analysis of two-machine lines with finite buffer, operation-dependent and time-dependent failure modes. Int J Prod Res 54(6):1850–1862

Meerkov S, Zhang L (2010) Product quality inspection in Bernoulli lines: analysis, bottlenecks, and design. Int J Prod Res 48(16):4745–4766

Meerkov S, Zhang L (2011) Bernoulli production lines with quality-quantity coupling machines: monotonicity properties and bottlenecks. Ann Oper Res 182(1):119–131

Mhada F, Malhamé R, Pellerin R (2014a) Joint assignment of buffer sizes and inspection points in unreliable transfer lines with scrapping of defective parts. Prod Manuf Res 1:79–101

Mhada F, Malhamé R, Pellerin R (2014b) A stochastic hybrid state model for optimizing hedging policies in manufacturing systems with randomly occurring defects. Discrete Event Dyn Syst 24:69–98

Naebulharam N, Zhang L (2014) Bernoulli serial lines with deteriorating product quality: performance evaluation and system-theoretic properties. Int J Prod Res 52(5):1479–1494
Nahas N, Nourelfath M, Gendreau M (2014) Selecting machines and buffers in unreliable assembly/disassembly manufacturing networks. Int J Prod Econ 154:113–126
Njike A, Pellerin R, Kenne J (2012) Simultaneous control of maintenance and production rates of a manufacturing system with defective products. J Intell Manuf 23(2):323–332
Nourelfath M, Nahas N, Ben-Daya M (2016) Integrated preventive maintenance and production decisions for imperfect processes. Reliab Eng Syst Saf 148:21–31
Ouzineb M, Mhada F, El Hallaoui I, Pellerin R (2013) An exact method for solving the buffer sizing and inspection stations allocations problem. IESM'2013, Rabat, Morocco
Ouzineb M, Mhada F, Pellerin R, El Hallaoui I (2014) Optimal planning of buffer sizes and inspection station positions. Cahiers du GERAD, (G-2014-53)
Ouzineb M, Nourelfath M, Gendreau M (2008) Tabu search for the redundancy allocation problem of homogenous series parallel multi-state systems. Reliab Eng Syst Saf 93(8):1257–1272
Ouzineb M, Nourelfath M, Gendreau M (2010) An efficient heuristic for reliability design optimization problems. Comput Oper Res 37(2):223–235
Ouzineb M, Nourelfath M, Gendreau M (2011) A heuristic method for non-homogeneous redundancy optimization of series-parallel multi-state systems. J Heuristics 17(1):1–22
Radhoui M, Rezg N, Chelbi A (2009) Integrated model of preventive maintenance, quality control and buffer sizing for unreliable and imperfect production systems. Int J Prod Res 47(2):389–402
Radhoui M, Rezg N, Chelbi A (2010) Joint quality control and preventive maintenance strategy for imperfect production processes. J Intell Manuf 21(2):205–212
Raviv T (2013) An efficient algorithm for maximizing the expected profit from a serial production line with inspection stations and rework. OR Spectr 35(3):609–638
Rivera-Gmez H, Gharbi A, Kenné JP, Montao-Arango O, Hernandez-Gress E (2016) Production control problem integrating overhaul and subcontracting strategies for a quality deteriorating manufacturing system. Int J Prod Econ 171:134–150
Rivera-Gómez H, Gharbi A, Kenné JP (2013) Joint production and major maintenance planning policy of a manufacturing system with deteriorating quality. Int J Prod Econ 146(2):575–587
Shetwan AG, Vitanov VI, Tjahjono B (2011) Allocation of quality control stations in multistage manufacturing systems. Comput Ind Eng 60(4):473–484
Zequeira RI, Valdes JE, Berenguer C (2008) Optimal buffer inventory and opportunistic preventive maintenance under random production capacity availability. Int J Prod Econ 111(2):686–696