Les Cahiers du GERAD ISSN: 0711–2440

Multilevel hybrid method for solving buffer sizing and inspection stations allocation problems

F. Mhada, M. Ouzineb,
R. Pellerin, I. El Hallaoui

G–2015–96

September 2015

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Multilevel hybrid method for solving buffer sizing and inspection stations allocation problems

Fatima Mhada [a]
Mohamed Ouzineb [b]
Robert Pellerin [c]
Issmail El Hallaoui [d]

[a] *University Mohammed V – ENSIAS, Rabat-Instituts, Rabat, Morocco*

[b] *Institut National de Statistique et d'Économie Appliquée, Rabat-Instituts, Rabat, Morocco*

[c] *CIRRELT & Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

[d] *GERAD & Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

fatima.zahra.mhada@gmail.com
ouzineb.insea@gmail.com
robert.pellerin@polymtl.ca
issmail.elhallaoui@polymtl.ca

September 2015

Les Cahiers du GERAD
G–2015–96

Abstract: This paper develops an efficient method to solve a typical combinatorial optimization problem that is frequently encountered when designing high levels of productivity and quality with a reasonable cost. Recent industrial applications have critical quality requirements that are not taken into account by the majority of methods proposed in the literature. In this paper, we explore the impact of adding these quality constraints on the system performance. We use a fast cost calculation model to solve the optimal buffer sizing problem in unreliable production lines simultaneously with the quality control problem constraints. The system studied in this contribution consists of n machines, n fixed-size buffers and m inspection stations in series. The objective is to minimize combined storage and shortage costs, and also to specify the optimal number and location of those inspection stations in the system. We combine a multilevel hybrid search method to identify search regions with promising locations of inspection stations and an exact method to optimize the assignment of buffer sizes for each location. Numerical results on test problems from previous research are reported. This approach reduces the solution time by more than 97% in some cases and allows a huge test problem with 30 machines to be solved.

Key Words: Inspection, production lines, quality, combinatorial optimization, meta-heuristics.

1 Introduction

Improving productivity with production control policy has been studied by several authors, and the major works in the literature are based on the assumption that all parts produced are conform, which is not realistic in industrial contexts. The current and standard quality analysis models tend to separately consider the problem of developing strategies for the quality preservation in production lines (by positioning inspection stations, for example) and the development of strategies to optimize the production problem (Kanban, CONWIP, or others). As has been well indicated [1, 2, 3, 4, 5, 6], both whole decisions are interdependent. We illustrate this interdependence through the example of the analysis of Kanban strategies introduced by Toyota in the sixties that have since become the paradigm of "lean manufacturing": they essentially advocate areas of limited storage between successive machines in a production line. The objective of these storage areas is to allow a certain degree of decoupling between machines to increase the productivity of the line (limitation of the effects of parts shortage for downstream machines or blocking machines that are upstream from one that is broken down.). These stocks should be limited because they are associated with frozen capital: the storage costs and extended transit times of parts in the workshop.

In an ideal "just in time" scenario, there would be no intermediate stock and finished parts would be pulled from the system as they are produced. However, hazards such as machinery breakdowns and lack of raw material or operators, will come to question this idealized vision, unable to ensure sufficient continuity of the workshop output. The sizes of the storage areas are directly related to the statistics of these hazards, as well as the estimated costs of the service losses that may result. So, if we choose to size the Kanban by completely ignoring the production quality dimension in the problem, we likely risk overestimating the security service level with storage areas which may contain important quantities of defective parts. These defective parts play a negative role for at least two reasons: on one hand, they correspond to misused production line time, since they decrease the efficiency of individual machines; they systematically undermine the "efficiency" of the intermediate storage and its ability to help increase the productivity of the line. On the other hand, as the investments associated with production line come from the same source, the costs associated with storage within the line are going to burden the budget associated with the improvement of the quality and vice versa. Authors in [7] present a survey of recent advances on the interface between quality and production system design. They provide evidence that production design impacts quality and quality impacts production design. Also, the location of the inspection station affects both the expected production cost per item and the production rate of the line.

The integration of quality aspects and maintenance in production policies for a single unreliable machine producing a singe product type has been receiving growing attention from researchers: the manufacturing system starts in an "in-control" state producing conforming parts and then switches to an "out-control" state and starts producing non conforming items. Due to the complexity of these models, different contributions with integrated models have been developed and can be classified as: i) joint production and quality problem ([8, 9, 10], ...); ii) joint production and maintenance problem ([11, 12, 13], ...); iii) joint production, quality and maintenance problem ([14, 15, 16], ...). They are many research works (see [17]) that focus on determining the optimal inspection station position in n serial production lines with or without: i) scrapping, ii) rework, iii) off-line repair, without considering the concept of buffer sizing. Considering serial production lines consisting of production and inspection machines that follow Bernoulli reliability and quality assumptions, Meerkov in [18] and [19] provided important insights into the nature of both 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 [20].

Like Kim and Gershwin [1, 2], we propose working with continuous production models (fluid models). However, in [1, 2], although the models are continuous, quality remains discrete and we see this as a potential source of contradictions in the modeling (unlike our model where quality is studied as a continuous flow) . In our case, we consider a tandem of machines where every machine has to satisfy a demand rate of good parts per time unit. We consider the problem of sizing the inventory level taking into account that the stock is a mixture of good and defective parts and there is generally elimination of defective parts along the line by positioning a number of inspection stations. A steady effort was made to develop decomposition methods

to reduce the analysis of the line to a series of an equivalent machines that can be isolated and sequentially analyzed [21].

Simulation is a real representation of the system studied. It can simulate any scenario involving any decision, but simulation is not an optimization method and the use of an exhaustive method to find the optimal solution is not practical when the systems are complex because the number of possibilities is huge (in our case, it equals $n^m * 100^n$) where n (resp. m) is the number of machines (resp. the number of inspection stations). Therefore, it is important to use optimization techniques on a simplified but realistic mathematical model. Modeling quality management issues combined with production management becomes very complex when considering all the factors that impact quality and production. Hence, there is a need to simplify the model by using some assumptions in a such a way that the resulting model can be solved optimally. We are motivated in this work to develop an optimization technique that would be able to find the optimal or near optimal solutions by performing a limited number of iterations. The same principle (but not similar methods) has been used in other areas such as: maintenance [22], pull control policies [23], buffer allocation [24]. These models are smaller and less complex than ours.

This paper efficiently solves a generalization of the model proposed in [21] where the question of finding the optimal number and positions of the inspection stations in the production line is addressed. We adapt and combine a partitioning technique, Tabu Search (TS) and Genetic Algorithm (GA) to identify search regions with promising locations of inspection stations and an exact method to optimize the assignment of buffer sizes for each given location. Hybridizing in this way (which we refer to as multilevel hybridizing) is very effective for high-dimensional combinatorial optimization problems. In fact, when the number of solutions is huge, as is the case in this paper, this step aims to locate promising search regions. This approach provides a balance between diversification and intensification. First, the selection of solutions by GA allows the identification of promising regions in the search space. One intended role of using GA is to facilitate the exploration by guiding the search to unvisited regions with good potential. This leads to a certain diversification in terms of subspaces to explore. Second, TS intends to search intensively around good solutions found in the past search. This leads to intensification by exploiting the neighborhood of each solution selected by GA.

The exact method of [25] has been shown to be efficient when dealing with small instances of buffer sizing and an inspection stations allocation problem. This approach is an Exhaustive Search Method (ESM) because the search is guaranteed to generate all possible locations of inspection stations. For each fixed location, the problem is reformulated as a network flow optimization problem that can be efficiently solved by a fast polynomial algorithm. Actually, the space that will be searched is finite but very large and the exhaustive search method may need hours for large instances. To evaluate the objective function (called fitness) for the fixed locations proposed by the hybrid method, we use the same shortest path procedure used in ESM. The new approach proposed in this paper finds optimal (in most cases) or near optimal solutions in a fraction of time compared to the exhaustive search method proposed in [25].

The knowledge resulting from this model, developing such optimization techniques, and business rules will enable the development of tools that can be combined with the simulation to evaluate and optimize more realistic and complex models without the simplifying assumptions (in our case we assume that the line is homogeneous, final demand is constant, etc.).

The remainder of the paper is organized as follows. In Section 2, the problem statement and some theoretical properties are discussed. A short description of the proposed approach is given in Section 3. A comparison between ESM [25] and this hybrid method, consolidated with some numerical results, will be presented in Section 4. Conclusions are drawn in Section 5.

2 Problem formulation and theoretical discussion

Figure 1 illustrates the production system studies. It consists of n machines separated by $n-1$ buffers and contains (m+1) inspection stations. Machines can be either up or down, starved or blocked. A machine M_i is starved if one of the upstream machines is down and all buffers between this machine and the machine M_i are empty. M_i is blocked if one of the downstream machines is down and all buffers between this machine

and the machine M_i are full. When a machine is neither starved nor blocked and is operational, it transfers parts from the upstream buffer to the downstream in a continuous way. We assume that the first machine can never be starved and an inspection station is located after the last machine to ensure the conformity of parts received by the customer.

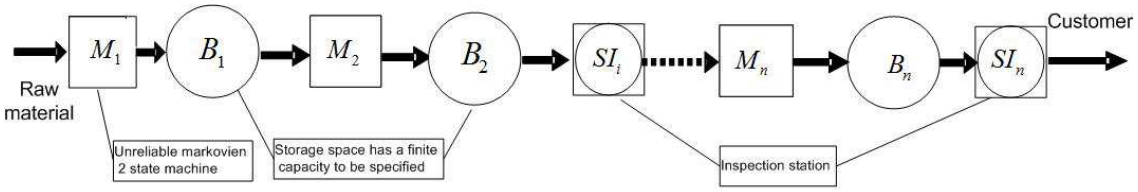

Figure 1: The production line

All machines $M_i, i = 1...n$ could be modeled as a continuous time Markov chain that produce a single part type with two different quality levels: conforming and non conforming with the same predefined ratio of non-conforming parts to conforming parts β. We consider that all the machines have the same maximum production rate k, failure rate p and repair rate r.

We denote d the demand rate for good parts from the last buffer, x_i the inventory level on the buffer i, $\tilde{d}_i$ the long term average number of parts pulled unit of time from the 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 the machine $M_{(i+1)}$ to ensure that all the parts being processed by $M_{(i+1)}$ are conforming. We suppose that m inspection stations are dispersed along the line (m is not known) i.e. $\sum_{i=1}^{n-1} \lambda_i = m$, $m \in \{0, 1, ..., (n-1)\}$ and $\lambda_n = 1$.

The optimization problem is the minimizing of the long term per unit time average global cost of storage, production shortages, and inspection. In other words, the cost to be minimized is:

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

under conditions:

$$\sum_{i=1}^{n-1} \lambda_i = m, m \in \{0, 1, ..., (n-1)\} \text{ and } \lambda_n = 1.$$

The average long term combined storage, shortage costs and inspection costs in (1) expressions can be written after development and calculation (see article [21]) as:

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

$$T^{(i)}(a,\lambda) = c_p \left(\frac{k\, \frac{(r(1-a_{i-1})+p)}{a_{i-1}}}{\sigma_i (k - \frac{\tilde{d}_i}{a_i})\, \frac{(r+p)}{a_{i-1}}} - \frac{k\,(1-a_i)}{\sigma_i (k - \frac{\tilde{d}_i}{a_i})} - \left[\frac{1}{\sigma_i} - \frac{(1-a_i)\, \frac{(r+p)}{a_{i-1}}}{\sigma_i^2 (k - \frac{\tilde{d}_i}{a_i})} \right] \right.$$
$$\left. ln \left[\frac{\frac{(r(1-a_{i-1})+p)}{a_{i-1}}\, \frac{\tilde{d}_i}{a_i}}{r\,(k - \frac{\tilde{d}_i}{a_i})} - \frac{\sigma_i\, \frac{(r(1-a_{i-1})+p)}{a_{i-1}}\, \frac{\tilde{d}_i}{a_i}}{\frac{(r+p)}{a_{i-1}}\, r\,(1-a_i)} \right] \right), \; i = 1, ..., n-1 \tag{3}$$

$$T_F(a,\lambda) = \frac{\rho_n\, c_p\, (\frac{k\,(1-\exp(-\mu_n(1-\rho_n)\, z_n(a_n^{des}))}{1-\rho_n} - \frac{(r+p)}{a_{n-1}} z_n(a_n^{des}) \exp(-\mu_n(1-\rho_n)\, z_n(a_n^{des}))}{\frac{(r+p)}{a_{n-1}} (1 - \rho_n\, \exp(-\mu_n(1-\rho_n)\, z_n(a_n^{des})))} \tag{4}$$

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

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

- $a_i, i = 1...n-1$ is the total parts wip availability coefficient at buffer x_i ; this coefficient is lower-bounded by $max\left[\left(\frac{r}{(r+p)}\right)^{i-1}, \frac{(r+p)\,\tilde{d}_i}{r\,k}\right]$ and upper-bounded by $min\left(\left[\frac{r+p}{r}\right]^{(n-i)} a_n^{des}, 1\right)$. (For more details see [26])
- $\tilde{d}_i, i = 1...n$ is the long term average number of parts pulled per unit time from stock x_i. The $\tilde{d}_i, i = 1...n$ expression depends on the selected position along the line of the m inspection stations. So if $e_j, j = 1...m$ the inspection station's position, i.e.

$$\lambda_i = \left\{ \begin{array}{ll} 1 & \text{if } i = e_j,\ j = 1...m \\ 0 & \text{if not} \end{array} \right\} \tag{5}$$

then:

$$\tilde{d}_i = \left\{ \begin{array}{ll} d(1+\beta)^n & \text{if} \quad 1 \le i \le e_1 \\ d(1+\beta)^{n-e_1} & \text{if} \quad e_1 < i \le e_2 \\ . & . \\ . & . \\ d(1+\beta)^{n-e_{m-1}} & \text{if} \quad e_{m-1} < i \le e_m \\ d(1+\beta)^{n-e_m} & \text{if} \quad e_m < i \le n \end{array} \right\} \tag{6}$$

The problem can be formulated as follows: find the minimal average global cost system structure (a, λ, m) that satisfies both constraints. That is,

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

subject to

$$\sum_{i=1}^{n-1} \lambda_i = m \qquad \forall m, \quad 1 \le m \le n-1, \tag{8}$$

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

$$\lambda_n = 1 \ \text{ and } \ a_n = a_n^{des}, \tag{10}$$

$$\lambda_i \in \{0, 1\} \qquad \forall i, \quad 1 \le i \le n-1. \tag{11}$$

$$a_i \ge 0 \qquad \forall i, \quad 1 \le i \le n, \tag{12}$$

The next proposition discusses the sensitivity of the production line to c_I. A numerical example is given in Subsection 4.3 to illustrate this theoretical result.

Proposition 2.1 *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 costs is

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

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

$$\begin{aligned} J(a^*,\lambda^*) &= \sum_{i=1}^{n-1} T^{(i)}(a^*,\lambda^*) + T_F(a^*,\lambda^*) + c_I \sum_{i=1}^{n} \lambda_i^* \tilde{d}_i^* \\ &= Constant(c_I) + c_I \sum_{i=1}^{n} \lambda_i^* \tilde{d}_i^* \end{aligned} \tag{13}$$

where $(a^*,\lambda^*) = argmin(J(a,\lambda))$.

So, if $e_i, i = 1...m$ are the inspection station's positions , then we can write equation (13) as follows:

$$J(a^*,\lambda^*) = Constant(c_I) + c_I \left(\sum_{i=1}^{m} d(1+\beta)^{n-e_i} + d(1+\beta)^n \right) \tag{14}$$

This equation makes it possible to conclude the total cost as a function of c_I is a piecewise linear function, of which the slope depends only on m and the inspection station's position $e_i, i = 1...m$. □

The cost function is likely to be convex. Conjecturing on the convexity of the cost function may lead to reducing the number of iterations of the algorithm. For instance, the local minimum of a convex function is also a global minimum and there are many efficient specialized methods for optimizing convex functions. This reduction in the number of iterations will lead to a significant reduction in the overall solution time. Actually, Conjecture 2.2 emphasizes the fact that the cost function is convex on the position of the inspection station and on the number of the located inspection stations. This conjecture is likely to be true as empirically shown by the numerical results in Subsection 4.2.

Conjecture 2.2

- *The minimal total cost (storage and inspection costs) is a convex function of λ, i.e. the location of the internal inspection station.*
- *The minimal total cost (storage and inspection costs) is a convex function of the number of the internal inspection stations.*
- *If we divide the production line into two parts that are the same size (i.e. same number of machines) and we fix the number of stations in the first part, the minimal total cost (storage and inspection costs) is a convex function of the number of the internal inspection stations in the second part.*

3 Methodology of the solution

The objective is minimizing the average long term combined storage and shortage costs, while also specifying the optimal location of inspection stations that provide the specified final conditions set by the customer. The goal is to solve the optimization task given by Eqs. (7)–(12) for a larger test problem with 30 machines.

3.1 Exact method

This exhaustive search method (ESM) was first introduced in [25]. For a fixed λ, we obtain an objective function that is separable by variables a_i ($T^{(i)}(a,\lambda) = T^{(i)}(a_i, a_{i-1}, \lambda)$). The idea is to reformulate $min_a J(a,\lambda)$, for each fixed location λ, as a shortest path problem defined on a network (described below) and efficiently solve it by a standard shortest path algorithm to find an optimal assignment of buffer sizes.

Consider the connected network $G(E,V)$ consisting of a set of nodes E and a set of links V as depicted in Figure 2. Each column i corresponds to a set of buffer availability possibilities for machine i. Each node is connected to all nodes in the next column. Each arc is associated with a real number $c_{i,i+1}$ that corresponds to the shortage and inspection costs, such that: $c_{0,1} = 1$, $c_{i,i+1} = T^i(a_i, a_{i+1}, \lambda) + c_I \times \lambda_i \times \tilde{d}_i$, $i = 1...n-2$ and $c_{n-1,n} = T_F(a_{n-1}, a_n, \lambda) + c_I \times \lambda_n \times \tilde{d}_n$.

The algorithm is given by Algorithm 1. The exact method finds the optimal solutions for the 10 machine and 20 machine instances. However, the run time appears to be unreasonable for some instances: the exact

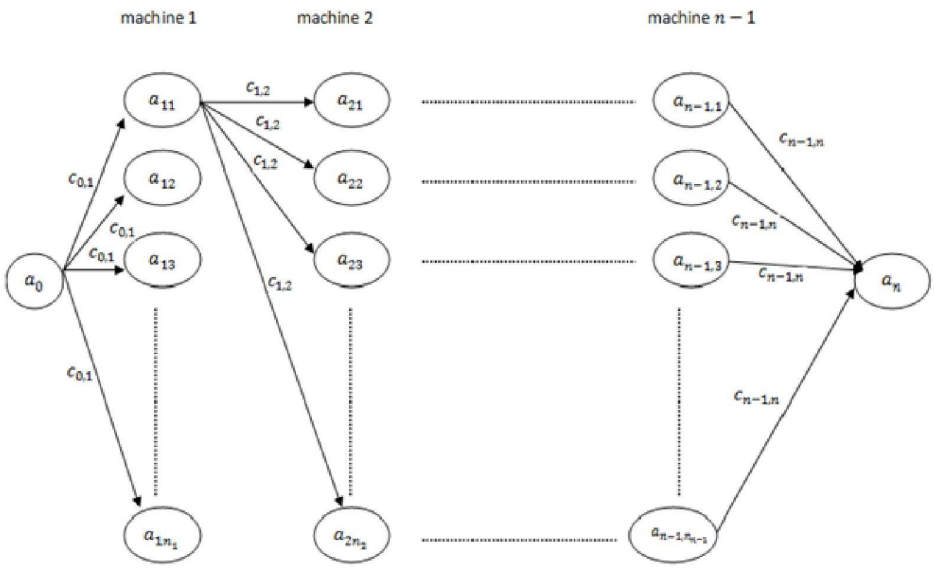

Figure 2: The network flow problem

method applied to the problem with 20 machines (for $7 \leq m \leq 13$) takes hours. Especially if we increase the number of machines, this method becomes drastically inefficient. For large instances (30 machines and a variable number of inspection stations), ESM was unable to solve them.

Algorithm 1 Optimal buffer sizing and inspection stations location algorithm [25]

$Z^* = \infty$.
for all λ **do**
  Evaluate the cost function Z and a for λ by Algorithm 2.
  if $Z < Z^*$ **then**
    $Z^* = Z$, $a^* = a$, $\lambda^* = \lambda$.
  end if
end for
return (Z^*, a^*, λ^*).

Algorithm 2 Fitness evaluation

Given a λ:
- Construct the network G.
- Find a shortest path in G (the buffer size vector a) and set Z to its cost.
return (Z, a).

3.2 Multilevel Hybrid Heuristic

To solve very large instances of the combinatorial optimization problem formulated in (7)–(12), it is important to have an effective and fast procedure to locate the extra m inspection stations (without going through all the combinations). The main idea of our approach consists of combining ideas from the exact method and meta-heuristic techniques. In fact, the search space S, which is very large, is composed of all possible λ_i values considered during the search. A particularity of the proposed approach is that it proceeds in two steps to deal with this huge number of solutions: (i) partitioning the search space into a set of disjoint subspaces; (ii) applying meta-heuristic techniques (a hybrid heuristic) to subspaces to select potential solutions. Algorithm 2 presented in Section 3.1 is applied for each selected λ (in a subspace) to evaluate its fitness and select the

optimal combination of buffer sizes a_i. This approach finds near optimal solutions for the same instances in a fraction of time compared to ESM. Details are given in the following subsections.

3.2.1 Partitioning the search space

We present some basic concepts of the partitioning used in the proposed approach. Here-forth, a solution vector refers to λ. Each solution vector is assigned an address, also called *level*, as explained below.

Definition 3.1 *We divide the solution vector into two equal parts and we create an address for each solution as follows: the address of λ is defined by: $Address(\lambda) = \sum_{l=1}^{n/2} \lambda_l$.*

Example

Let us consider a system of $n = 20$ machines with $m = 5$ inspection stations: 2 inspection stations located in the first part and 3 inspection stations located in the second part:

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

From the above definition, the address of λ is simply obtained by counting the number of 1 in the first part. That is, $address(\lambda) = 2$.

Definition 3.2 *A search subspace of address r, denoted by S_r is defined as the set of solutions which have the same address, equal to r.*

Following the above definitions, the lower bound of r is 0 and its upper bound is m. It is obvious that $(S_r)_{1 \leq r \leq m}$ is a partition of the search space S. This means that:

1. $S_{r_1} \bigcap S_{r_2} = \emptyset \ \ \forall\ r_1 \neq r_2$, and
2. $S = \bigcup_{r=0}^{m} S_r$.

3.2.2 Hybrid heuristic

When the number of combinations in λ is huge in a certain S_r, the objective of this step is to locate promising regions in S_r. The proposed hybrid optimization approach is based on a combination of Genetic Algorithm (GA) and Tabu Search (TS). The approach tends to provide a balance between diversification and intensification. First, the selection of λ by some GA-operators allows the identification of the search space. One intended role of the used GA is to facilitate the exploration by guiding the search to unvisited regions. Second, the intended role of TS is to search carefully and intensively around good solutions found in the past search. The hybrid heuristic is presented in Algorithm 3. N_{rep} and N_c are parameters tuned by the experimentation.

In Step 3, we use the so-called one-point crossover operator, which, as illustrated in Table 1, creates the child string O for the given pair of parent strings λ_1 and λ_2 by:

1. Copying string elements belonging to the first fragment from λ_1;
2. Copying the rest of the string elements from λ_2.

Table 1: An example illustrating the 1-point crossover procedure

Parent string λ_1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	1
Parent string λ_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**

Algorithm 3 Hybrid Heuristic Algorithm

Given *level* r, N_{rep}, N_c
$Z^* = \infty$, $k = 0$, $l = 0$.
Step 1. Generate random population of N_s chromosomes (solutions of the same address).
Step 2. Select randomly from the populations two solutions of S_r named *parent*1 and *parent*2 that will "reproduce" to create a new solution.
Step 3. Produce a new solution O (child or offspring) from a selected pair of parent solutions.
Step 4. Find the best solution (Z, a, λ) in the neighborhood of the new child O $(V(O))$ using the TS algorithm.
if $Z < Z^*$ **then**
$Z^* = Z$, $a^* = a$, $\lambda^* = \lambda$.
end if
Step 5. If the new solution obtained by TS is better than the worst solution currently in the population, it replaces it, or else it is discarded.
if $k < N_{rep}$ **then**
$k = k + 1$ and go to Step 2.
end if
Step 6. Decide if the population is replenished and start a new genetic.
if $l < N_c$ **then**
$k = 0$, $l = l + 1$ and go to Step 1.
end if
return (Z^*, a^*, λ^*).

Tabu search, used in Step 4, is a meta-heuristic that consists of an iterative method, where at each iteration we move from a current solution to a new solution in a neighborhood, until some stopping criterion has been satisfied. Our neighborhood structure is defined as follows: at each iteration of TS, the local transformations (or moves) that can be applied to the current solution λ, define a set of neighboring solutions for a given subspace as: Neighborhood(λ) = {inspection stations located in a production line obtained by applying a single move to λ}. The move applied to λ consists of changing two positions of the inspection stations. It follows that the number of inspection stations located in the production line does not change after a local transformation of λ. Each new solution obtained by TS is decoded to obtain its fitness value. These fitness values, which are a measure of quality, are used to compare different solutions. This fitness is evaluated according to the objective function $J(a, \lambda)$ optimized with an exact method (Algorithm 2).

TS enhances the local search performance by using memory structures: once a potential solution has been determined, it is marked as *tabu*, so that the algorithm does not visit that possibility repeatedly. That is, tabus are used to prevent cycling when moving away from local optima through non-improving moves. At each iteration, the best solution λ' in a subspace V(λ) $\subset$ Neighborhood(λ) is selected and considered as a tabu solution for some next iterations. The subspace V(λ), called the effective neighborhood, is generated by eliminating the tabu solutions from Neighborhood (λ). 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. The size of the tabu list (tabu tenure) used in this paper is dynamic, as it is usually found that using a variable size tabu list is more efficient [27, 28]. Finally, the stopping criterion is specified in terms of a maximum number of local iterations ($mnli$) without improving the best-known solution.

3.2.3 Multilevel Hybrid Algorithm

Multilevel Hybrid Heuristic (MHH) is an iterative procedure, where at each iteration we move from a current level to a new level. Choosing a search space and a neighborhood structure is by far the most critical step in the design of any meta-heuristic. It is at this step that one must make the best use of the understanding and knowledge he/she has of the problem at hand [27]. In the problem studied in this paper, the search space is the space of all possible line structures considered (visited solutions) during the search. After dividing the search space into a set of disjoint subspaces, this approach applies hybrid heuristic presented in Section 3.2.2 to each selected subspace. The algorithm can be presented as follows:

Algorithm 4 Multilevel Hybrid Heuristic

$Z^* = \infty$.
Step 1. The initial solution is chosen such that there is no inspection station ($\lambda_i = 0\ \forall i \leq n-1$).
Step 2. The address r is randomly incremented and the best solution (Z, a, λ) in S_r is calculated by Algorithm 3.
if $Z < Z^*$ **then**
$Z^* = Z$, $a^* = a$, $\lambda^* = \lambda$.
end if
Step 3. The termination criterion is checked. It is defined as the maximum time (max_t) without finding an improvement in the best known solution. If it is reached, the search is concluded and the best obtained solution is recommended. Otherwise, return to step 2.
return (Z, a).

4 Numerical results

To analyze the solution method introduced in this paper, three design optimization problems (benchmarks) are used: a sample with 10 machines, a larger test problem with 20 machines and a huge one with 30 machines. The algorithms are implemented in C^{++}. The numerical tests were completed on an Intel Core i7 at 2.8 GHz with 8 GB of RAM running Linux. Table 2 shows the different parameters used for the numerical results:

Table 2: System parameters

	n	p	r	k	β	d	c_p	c_I	a_n^{des}	$max_t(sec)$	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

4.1 Comparing MHH to ESM

To compare MHH to ESM, we re-implemented ESM in the same conditions (computer, programming language, operating systems, etc.) as the MHH algorithm. On the one hand, this allows us to determine the optimal solutions, for small to medium size instances, to compare. On the other hand, it allows the observation of how ESM behaves on the third problem instance, which is an important benchmark that has been newly proposed in this paper. The percent that one solution improves upon another is defined in terms of objective value and CPU time as follows:

$$\text{MPTI} = 100 \quad \times \quad \frac{\text{(ESM Time - MHH Time)}}{\text{ESM Time}}\%. \tag{15}$$

For 10 machines, MHH and ESM find the same optimal solutions in comparable times. The comparison results are given in Tables 3 and 4 for 20 and 30 machines with $m = 1...10$. The first column indicates the number m of inspection stations. Note that there is no column for ESM cost because it coincides with MHH cost. MHH finds optimal solutions for all instances. The columns MHH/Best tell the time in which the best solution is found while the column MHH/Final tells the time MHH finishes according to the stopping criteria. ESM takes hours with the problem with 20 machines. Comparisons show that MHH outperforms ESM in terms of the execution time. In fact, MHH finds optimal solutions in a fraction of the MHH time on the 20 machine problem. For the problem with 30 machines, our method finds high-quality solutions in a reasonable time. ESM cannot solve instances with $m \geq 7$ in a reasonable times. For example, for $m = 7$, after one month, the algorithm cannot converge to a global solution. So, we cannot expect to obtain an optimal solution by ESM in a reasonable times except for the problem with 10 machines. For this reason, we need very fast heuristics to provide very good results in acceptable times.

Table 3: The comparison results for 20 machines

	Cost	Running time (s)			
m	MHH	ESM	$MHH/Best$	$MHH/Final$	$MPTI$
1	9.2315	15.3	34.3	177.6	-91.39
2	6.9004	105.7	77.4	440.5	-76.01
3	6.2806	421.7	422.0	1011.9	-58.33
4	6.2998	1601.2	591.7	1191.7	25.58
5	6.4972	4743.3	571.2	879.4	81.46
6	6.7430	11108.7	631.4	992.0	91.07
7	7.0159	20613.0	635.2	1108.9	94.62
8	7.3165	29790.2	960.7	1560.7	94.76
9	7.6498	35764.9	476.5	1076.5	96.99
10	8.0147	35767.6	109.0	709.0	98.02

Table 4: The comparison results for 30 machines

	Cost	Running time (s)			
m	MHH	ESM	$MHH/Best$	$MHH/Final$	$MPTI$
1	43.3517	31.2	109.6	829.3	-96.24
2	24.5994	389.4	1186.7	1554.6	-74.95
3	21.1825	3524.2	3779.7	3926.9	-10.25
4	19.5903	18769.2	4470.1	5671.9	69.78
5	19.1385	115432.0	6584.2	8065.0	93.01
6	19.3456	421630.0	10034.2	13634.2	96.77
7	19.5901	–	10713.9	14313.9	–
8	19.8583	–	12300.6	15900.6	–
9	20.1527	–	10549.2	14149.2	–
10	20.4792	–	11128.0	14728.0	–

Table 5 shows the results for 20 machines by MHH and ESM while limiting the running time to one hour. Both methods find optimal solutions in one hour for $m \leq 5$. MHH finds better solutions for $m \geq 6$.

Table 5: Results by MHH and ESM for 20 machines during 1 hour

m	MHH	ESM
1	09.2315	09.2315
2	06.9004	06.9004
3	06.2806	06.2806
4	06.2998	06.2998
5	06.4972	06.4972
6	06.7430	07.0839
7	07.0159	07.4549
8	07.3165	07.8745
9	07.6498	08.4677
10	07.7947	08.8909

4.2 Characteristics of the production line

After demonstrating how quickly and efficiency the hybrid method can be, we will show some results of interest from the homogenous case with 10 and 20 machines. Figure 3 displays the optimal cost as a function of the location of the internal inspection station λ_i, $i = 1, 2, ..., 9$. The optimal number of inspection stations is $m = 1$, and in this case the minimal total cost (storage and inspection costs) is a convex function of the location of the internal inspection station. The optimal position of the station is after the fourth machine i.e. $\lambda_4 = 1$.

Figure 4 displays the optimal cost as a function of a number of internal inspection stations $m = i$, $i = 1, 2, ..., 19$. We notice that the minimal total cost (storage and inspection costs) is a convex function of

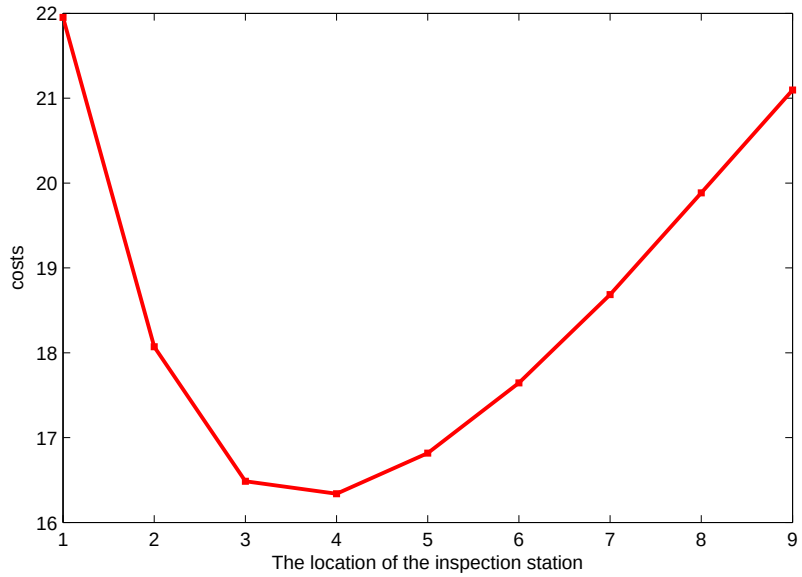

Figure 3: The optimal cost as a function of λ_i, $i = 1, 2, ..., 9$: The 10 machines case

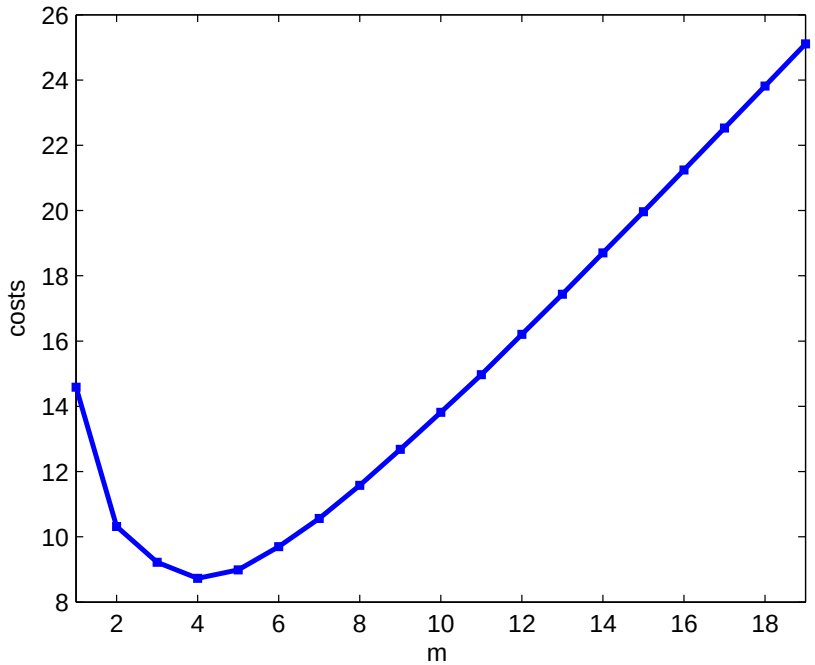

Figure 4: The optimal cost as a function of $m = 1, 2, ..., 19$: The 20-machine case

a number of internal inspection stations m. The optimal number of inspection stations is $m = 3$, and the optimal positions of these stations are: $\lambda_2 = 1$, $\lambda_7 = 1$, $\lambda_{18} = 1$, otherwise $\lambda_i = 0$. The results obtained by MHH and ESM are identical.

Figure 5 displays the best cost found by MHH as a function of the number of the internal inspection stations $m = i$, $i = 1, 2, ..., 29$. We notice that the minimal total cost is a convex function of the number of the internal inspection stations m and the optimal number of inspection stations is $m = 5$. The optimal

positions of these stations are: $\lambda_1 = 1$, $\lambda_3 = 1$, $\lambda_7 = 1$, $\lambda_{15} = 1$ and $\lambda_{29} = 1$. These results confirm Conjecture 2.2.

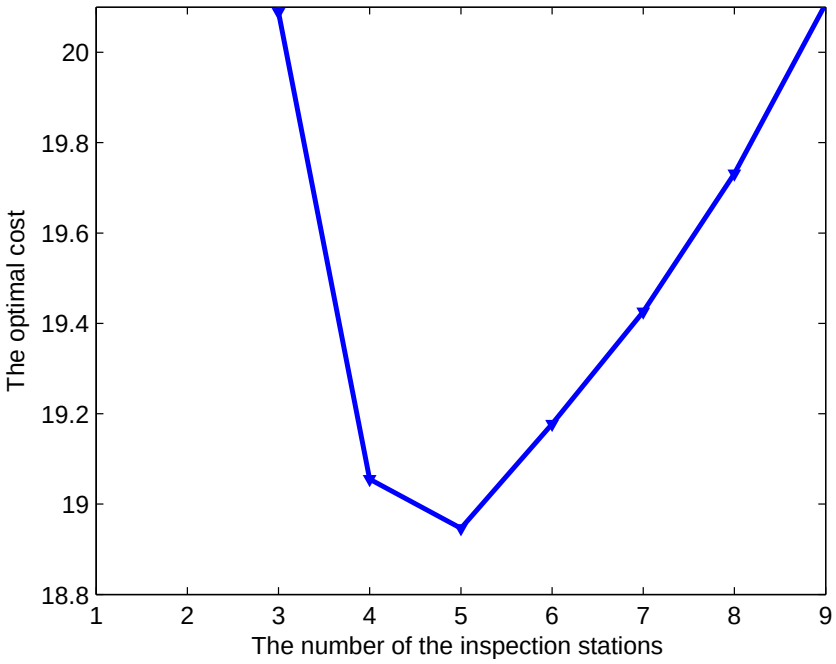

Figure 5: The optimal cost as a function of the number of inspection stations, m: The 30-machine case.

4.3 Sensitivity analysis

In this subsection, we study how the optimal total cost and the optimal number of inspection stations reacts to the variation of some parameters of the model. In our case, the parameters will be β and c_I. Table 6 defines the set of parameters we are using for this study.

Table 6: System parameters

	n	p	r	k	d	c_p	a_n^{des}
20 machines	20	0,2	0,9	9	1	0.1	0.95

4.3.1 The inspection cost c_I

We begin by studying the effect of changing the inspection cost per time unit and per part c_I (for a fixed $\beta = 0,1$). Figure 6 (the top one) shows that for values of c_I close to the storage cost per time unit and per part c_p, the optimal number of inspection stations is quite high (bottom figure), while for a very high value of c_I, the optimal number of required inspection stations decreases until it reaches zero (for a very high total cost). This makes sense since the system will encourage storage when it is more profitable or help eliminate non-conforming parts by imposing more inspection stations otherwise.

Also, when the inspection cost per time unit and per part c_I increases, the locations of the inspection stations tend to be closer to the end of the line and when one of those stations is placed after the $n-1$ machine then any increase in c_I will make this station unnecessary, meaning that the optimal number of stations will be reduced by 1 (Table 7).

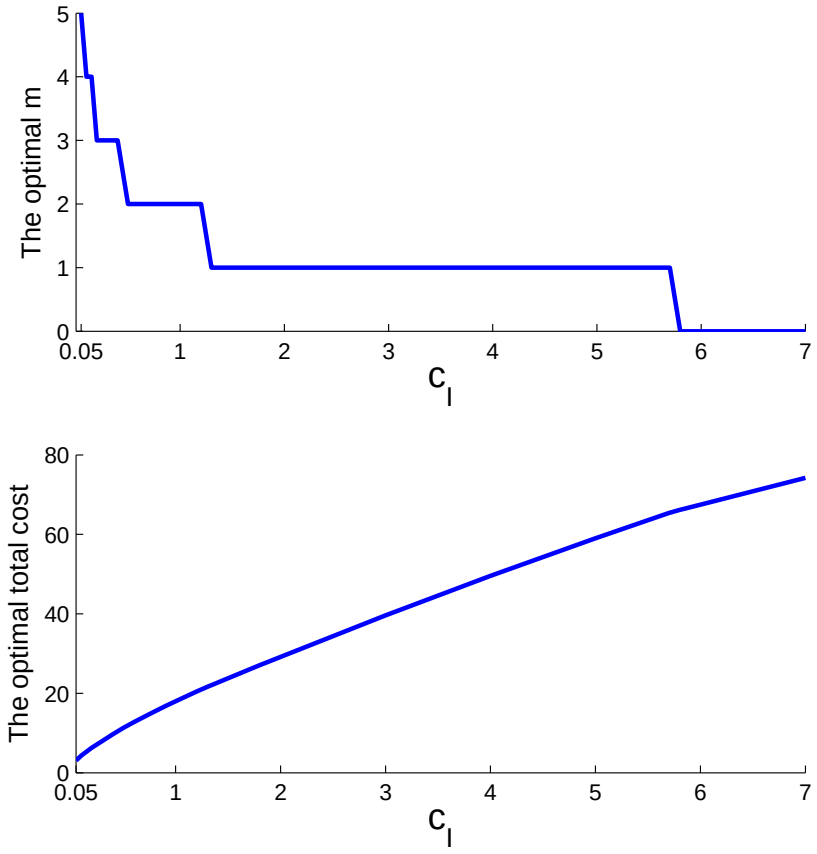

Figure 6: The optimal number of inspection stations and the optimal total cost as a function of c_I

Table 7: The optimal location of inspection stations

c_I	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
The optimal m	3	3	2	2	2	2	1	1	1	1	1	1	1	1
The optimal positions	2; 7; 18	2; 7; 19	3; 11	3; 13	3; 14	4; 19	5	6	7	8	9	10	11	11

Figure 7 displays the case $m = 1$ (top figure) and $m = 0$ (bottom figure). In the first case and for each inspection station position, the total cost as a function of c_I is linear. Whereas in the second case, and because there is no inspection station, the total cost as a function of c_I is a linear function.

4.3.2 The fraction of non conforming to conforming parts β

Figure 8 and Table 8 show the behavior of the line as a function of β. When β increases, the quantity of non-conforming parts increases, unlike the quantity of good parts that will decrease. This means that if we do not eliminate the nonconformity from the system, then the machines will work harder and the stocks will fill faster than if we do and the line will not be able to meet the final demand. Unlike the case in which c_I varies, the cost does not vary linearly (Figure 7) but exponentially (Figure 9) .

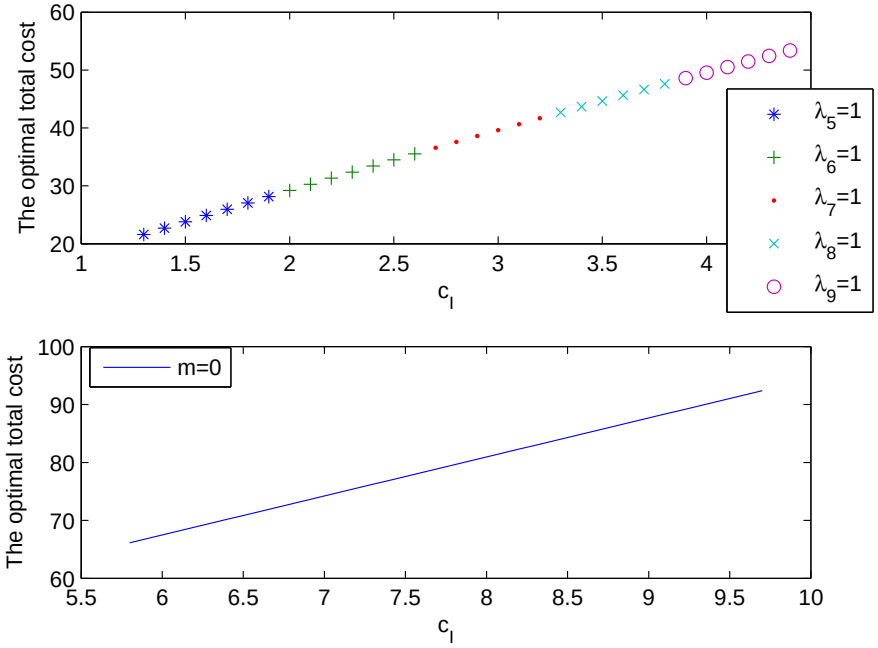

Figure 7: The optimal total cost as a function of c_I (Two cases: $m^* = 1$ and $m^* = 0$)

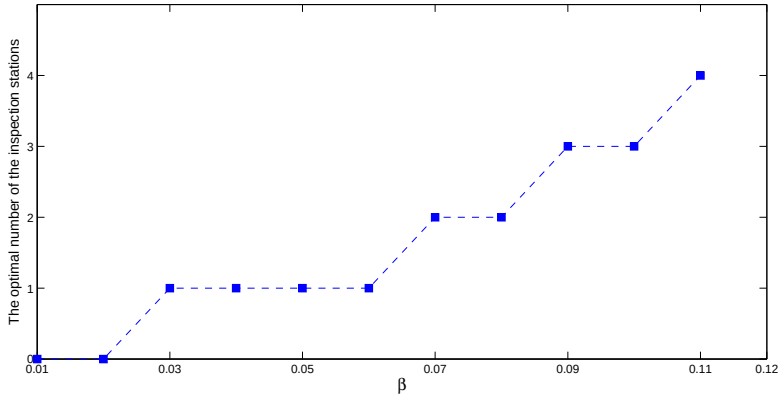

Figure 8: The optimal number of inspection stations as a function of β

Table 8: The optimal inspection stations location

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

5 Conclusions

This paper proposes an efficient hybrid approach based on ideas from an exact method and meta-heuristics to solve the buffer sizing problem in unreliable homogeneous production lines with several inspection stations. This is a very hard mixed integer nonlinear program. The proposed approach combines the genetic algorithm and tabu search to identify profitable configurations (locations of the inspection stations). For these locations,

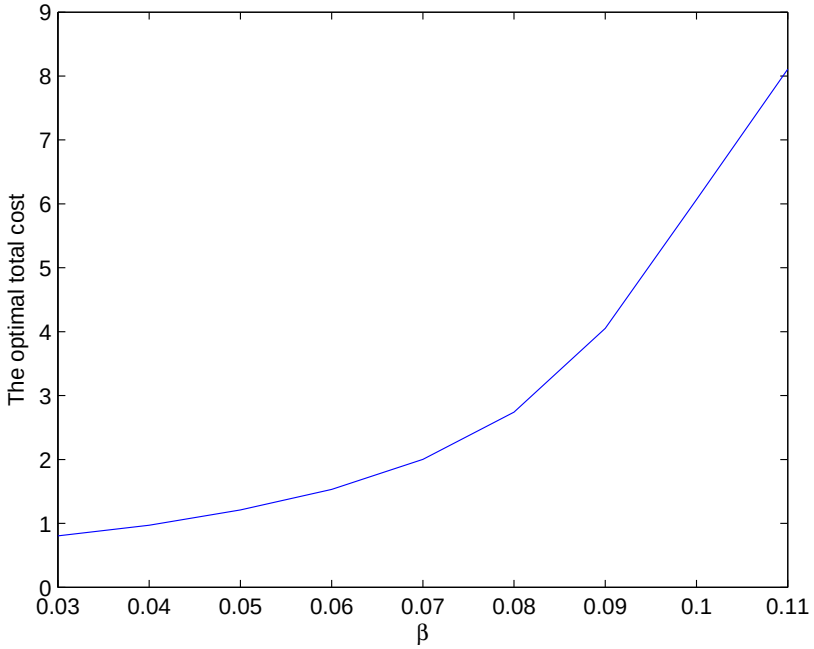

Figure 9: The optimal total cost as a function of β

we use an exact approach to decide the optimal sizes of buffers to use. Our final goal is to find an optimal or near optimal design. This hybrid approach provides a balance between diversification and intensification that is demonstrated to work well on such homogeneous production lines with up to 30 machines. When compared to ESM, MHH results are superior in terms of CPU time for the same quality of solutions. Future research will focus on developing a more realistic analytic model for non homogeneous production lines and solving it by combining simulation and optimization methods.

References

[1] J. Kim, S. Gershwin, Integrated quality and quantity modeling of a production line, OR Spectrum 27 (2005) 287–314.

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

[3] M. Colledani, T. Tolio, Impact of statistical process control (spc) on the performance of production systems-part 2 (large systems), Zakynthos Island, Greece, 2005.

[4] M. Colledani, T. Tolio, Impact of quality control on production system performance, Annals of the CIRP 55(1) (2006) 453–456.

[5] M. Colledani, T. Tolio, Performance evaluation of production systems monitored by statistical process control and off-line inspections, International Journal of Production Economics 120(2) (2009) 348–367.

[6] M. Colledani, T. Tolio, Integrated analysis of quality and production logistics performance in manufacturing lines, International Journal of Production Research 49(2) (2011) 485 –518.

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

[8] A. Hajji, F. Mhada, A. Gharbi, R. Pellerin, R. Malhamé, Integrated product specifications and productivity decision making in unreliable manufacturing systems, International Journal of Production Economics 129(1) (2010) 32–42.

[9] S.C. Kutzner, G.P. Kiesmüller, Optimal control of an inventory-production system with state-dependent random yield, European Journal of Operational Research 227(3) (2013) 444–452.

[10] F. Mhada, R. Malhamé, R. Pellerin, A stochastic hybrid state model for optimizing hedging policies in manufacturing systems with randomly occurring defects, Discrete Event Dynamic Systems 24 (2014) 69–98.

[11] M. Ben-Daya, The economic production lot-sizing problem with imperfect production processes and imperfect maintenance, International Journal of Production Economics 76 (2002) 257–264.

[12] H. Rivera-Gómez, A. Gharbi, J.P. Kenné, Joint production and major maintenance planning policy of a manufacturing system with deteriorating quality, International Journal of Production Economics 146(2) (2013) 575–587.

[13] K. Dhouib, A. Gharbi, M. B. Aziza, Joint optimal production control/preventive maintenance policy for imperfect process manufacturing cell, International Journal of Production Economics 137(1) (2012) 126–136.

[14] M. Radhoui, N. Rezg, A. Chelbi, Integrated model of preventive maintenance, quality control and buffer sizing for unreliable and imperfect production systems, International Journal of Production Research 47(2) (2009) 389–402.

[15] A. Njike, R. Pellerin, J.P. Kenné, Simultaneous control of maintenance and production rates of a manufacturing system with defective products, Journal of Intelligent Manufacturing 23(2) (2012) 323–332.

[16] R.I. Zequeira, J.E. Valdes, C. Berenguer, Optimal buffer inventory and opportunistic preventive maintenance under random production capacity availability, International Journal of Production Economics 111(2) (2008) 686–696.

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

[18] S. Meerkov, L. Zhang, Product quality inspection in bernoulli lines: analysis, bottlenecks, and design, International Journal of Production Research 48(16) (2010) 4745–4766.

[19] S. Meerkov, L. Zhang, Bernoulli production lines with quality-quantity coupling machines: Monotonicity properties and bottlenecksn, Annals of Operations Research 182(1) (2011) 119–131.

[20] F. Ju, J. Li, G. Xiao, J. Arinez, Quality flow model in automotive paint shops, International Journal of Production Research 51(21) (2013) 6470–6483.

[21] F. Mhada, R. Malhamé, R. Pellerin, Joint assignment of buffer sizes and inspection points in unreliable transfer lines with scrapping of defective parts, Production and Manufacturing Research 1 (2014) 79–101.

[22] O. Roux, M.A. Jamali, D.A. Kadi, E. Chatelet, Development of simulation and optimization platform to analyse maintenance policies performances for manufacturing systems, International Journal of Computer Integrated Manufacturing 21(4) (2008) 407–414.

[23] D. Koulouriotis, A. Xanthopoulos, V. Tourassis, Simulation optimization of pull control policies for serial manufacturing lines and assembly manufacturing systems with genetic algorithms, International Journal of Production Research 48(10) (2010) 2887–2912.

[24] S. Jeong, H. Jung, Optimal buffer allocation in flexible manufacturing systems using genetic algorithm and simulation, Journal of Advanced Mechanical Design, Systems, and Manufacturing 6(7) (2012) 1071–1080.

[25] M. Ouzineb, F. Mhada, I. El Hallaoui, R. Pellerin, An exact method for solving the bffer sizing and inspection stations allocations problem, IESM'2013, Rabat, Morocco.

[26] M. Ouzineb, F. Mhada, R. Pellerin, I. El Hallaoui, Optimal planning of buffer sizes and inspection station positions, Les Cahiers du GERAD, G–2014–53, HEC Montréal.

[27] M. Gendreau, Recent Advances in Tabu Search, in Essays and Surveys in Metaheuristics, C.C. Ribeiro and P.Hansen (eds.), Kluwer Academic Publishers, 2002.

[28] M. Ouzineb, M. Nourelfath, M. Gendreau, Tabu search for the redundancy allocation problem of homogenous series parallel multi-state systems, Reliability Engineering and System Safety 93(8) (2008) 1257–1272.