Journal of Intelligent Manufacturing (2020) 31:615–640
https://doi.org/10.1007/s10845-019-01468-x

A methodology for solving facility layout problem considering barriers: genetic algorithm coupled with A* search

Mariem Besbes[1,2] · Marc Zolghadri[1] · Roberta Costa Affonso[1] · Faouzi Masmoudi[2] · Mohamed Haddar[2]

Received: 6 April 2018 / Accepted: 15 February 2019 / Published online: 9 March 2019

Abstract
This work proposes a new methodology and mathematical formulation to address the facility layout problem. The goal is to minimise the total material handling cost subjected to production-derived constraints. This cost is a function of the distance that the products should cover within the facility. The first idea is to use the A* algorithm to identify the distances between workstations in a more realistic way. A* determines the shortest path within the facility that contains obstacles and transportation routes. The second idea is to combine a genetic algorithm and the A* algorithm with a homogenous methodology to improve the quality of the facility layouts. In an iterative way, the layout solution space is explored using the genetic algorithm. We study the impacts of the appropriate crossover and mutation operators and the values of the parameters used in this algorithm on the cost of the proposed arrangements. These operators and parameter values are fine-tuned using Monte Carlo simulations. The facility arrangements are all compared and discussed based on their material handling cost associated with the Euclidean distance, rectilinear distance, and A* algorithm. Finally, we present a set of conclusions regarding the suggested methodology and discuss our future research goals.

Keywords Manufacturing systems design · Facility layout problem · Genetic algorithm · A* search algorithm · Monte Carlo simulation

Introduction

The facility layout problem (FLP) concerns the determination of the most efficient arrangement of equipment in an area. A facility can be a machine tool, workstation, manufacturing cell, machine shop, department, warehouse, and so forth (Palomo-Romero et al. 2017). This problem has broad applications in fields such as layout of hospitals, airports, or manufacturing systems. From the manufacturing perspective, the efficiency of the facility layout is typically measured in terms of the material handling cost (MHC) (Liu and Meller 2007). The MHC is directly linked to the total distance the products and items should travel within the facility according to their manufacturing constraints, i.e., their routings. Various facility arrangements cause different total distances. The resolution of the FLP focuses on minimising the MHC because, according to Ahmadi et al. (2017), approximately 20–50% of the total operating expenses in manufacturing environments are attributed to the MHC.

There are two ways to formulate layout problems: discrete and continuous. The discrete representation of the layout is often linked to the traditional mathematical programming approach. This approach attempts to assign a set of workstations to a set of predetermined and distinct locations. The associated optimisation problem is then addressed as a quadratic assignment problem (Drira et al. 2007). Such discretisation of the space simplifies the solutions space. However, as numerous possible locations are ignored, the chance to obtain an optimal layout is lessened. The continuous

✉ Mariem Besbes
mariem.besbes@supmeca.fr

Marc Zolghadri
marc.zolghadri@supmeca.fr

Roberta Costa Affonso
roberta.costa@supmeca.fr

Faouzi Masmoudi
faouzi.masmoudi@enis.tn

Mohamed Haddar
mohamed.haddar@enis.tn

[1] Quartz-Supmeca, 93407 Saint-Ouen, France

[2] University of Sfax, LA2MP-ENIS, B.P. 1173, 3038 Sfax, Tunisia

formulation allows the workstations to be located anywhere within a planar site. The FLP models deal with workstations that could have all the same footprint or not.

Despite the significant literature in the FLP field, several improvements in the manufacturing context are still possible by (i) modelling the real characteristics of the studied cases in a more efficient way (more complex footprints of workstations, for instance); (ii) exploring new techniques to define the distance among equipment units, considering product routings, obstacles, and transportation routes; and (iii) using the knowledge of human experts for designing a layout, which can be seen for instance in Ahmadi et al. (2017).

Our main motivation is to explore a more relevant way to define layouts with a satisfactory MHC considering more realistic constraints. This research does not focus on finding the optimal solution; rather, it looks for a cost effective layout. This means that we search for a realistic distance determination among the workstations for products that should cross a workshop. Most of the FLP resolution approaches use the rectilinear or Euclidean distance between workstations (Gonçalves and Resende 2015; Hosseini-Nasab et al. 2017; Xie et al. 2018). These distances are easy to use but have shortcomings. The Euclidean distance can be defined as a straight line between the start and destination machines. It measures the distance between two workstations without considering the obstacles and machines in between (Gomathi et al. 2014). The rectilinear distance is computed by adding the vertical and horizontal distances between centroids of workstations. It is mainly used for problem modelling where the facility space is defined as a grid of surface units. The products are allowed to travel over the borders of these units and cross the obstacles (Friedrich et al. 2018). The obtained solutions based on these distances should be therefore modified afterwards to make them realistic.

To improve the quality of the model, we consider a facility area containing obstacles and transportation routes. The obstacles are the occupied spaces, such as walls and stairs. They cannot contain workstations and products cannot pass through them. The transportation areas refer to permanent transportation paths (i.e., aisles) where no facilities are allowed to be located. However, they can obviously be used for product movements. The facility is modelled as a grid of surface units. The start and destination points for a product in the whole facility area are the centres of these units. Every equipment covers a given number of surface units. The problem to solve is therefore to find out the configuration where the sum of all the product displacements from one item of equipment to the next one is small, smaller than any other suggested configurations. The retained configuration may not be the optimal solution. In this case, we use the A* algorithm for distance calculations.

To apply such ideas, we need to explore the whole solution space, which is composed of all possible arrangements of workstations within the facility area. This is time consuming and difficult for large scale FLPs. That is the reason why we rely on genetic algorithms (GA), which allow exploring the solution space without any guarantee of studying every possible facility arrangement. However, the application of these algorithms in other areas, including FLP studies (Sadrzadeh 2012; Palomo-Romero et al. 2017; Aiello et al. 2013), shows that very good solutions can be found if and only if suitable algorithm parameters are used. To obtain such a guarantee, we applied the Monte Carlo simulation principles to the entire methodology in order to find out the best set of parameters that provide good arrangements with low MHC.

Our simulations show that the selection of the GA operators and parameters generates two classes of facility arrangements. In the first class, the workstations are densely positioned in the facility area minimising drastically the MHC, while in the second class, the arrangements are porous, and the workstations occupy larger spaces.

The remainder of this paper is organised as follows. Section 2 describes the FLP and reviews the related literature. Section 3 contains the principles of the A* algorithm and GA. Section 4 provides a detailed description of the developed structure for the proposed approach. The mathematical model of the considered FLP is introduced and the customisation of the A* algorithm and GA to the FLP resolution is discussed. The identification of the parameter settings of the used GA via the Monte Carlo simulation and the evaluation of the facility arrangement performance is shown in Sect. 5. The whole methodology is applied on an illustrative presented in Sect. 6. Finally, the main conclusions of the paper and suggestions for further studies are presented.

Facility layout problem (FLP): related work

General statements

The facility layout design aims at efficiently positioning n facilities within a given area considering the architecture and structure of the manufacturing facility systems. Finding the most efficient physical layout is regarded as a key to improvement in plant productivity (Tarkesh et al. 2009) and has a significant impact on operational performance measured by lead times, throughput rate, and work in process (El-Baz 2004). This issue has been extensively studied during the past decades. The existing research works on the FLP fall into several categories such as equal- and unequal-sized FLPs, regular and irregular shapes (Azadeh et al. 2016), single- and multi-floor layouts (Park et al. 2011; Ahmadi and Jokar 2016), single- and multi-objective problems (Samarghandi et al. 2010; Jolai et al. 2012), and static and dynamic layout problems (Moslemipour and Lee 2012; Asl and Wong 2015). Kusiak and Heragu (1987), Singh and Sharma (2006), Drira et al. (2007), and Besbes et al. (2018) present an overview of the different schools of thought, trends, and the research niches of the area.

A facility-layout resolution process may involve two phases, the block and detailed layout phases. During the block layout phase, the relative location and size of each department are specified (Armour et al. 1964). Then, the detailed layout phase determines the exact position of workstations, aisle structures, flow paths, and the layout within each department (Meller and Gau 1996).

FLP formulation

It is possible to use a discrete or a continuous representation of an FLP. The discrete representation is the traditional FLP formulation and is the most popular one owing to its simplicity, as seen in Zhou et al. (2017), Ramkumar et al. (2008), Xiaoning and Weina (2011), among other authors. The quadratic assignment problem has been developed to model a discrete FLP. The main assumptions used are (i) unrestricted shop, (ii) equal-sized facilities, (iii) regular shape facilities, and (iv) predetermined locations. The quadratic assignment formulation simplifies the FLP mainly because of the first and fourth assumptions. The first assumption allows unfeasible layouts and the fourth assumption restricts the solution space. The continuous formulation and resolution approaches make it possible to allocate resources anywhere within the facility area (Drira et al. 2007). Table 1 presents a comparison of the main characteristics of the discrete and continuous approaches.

Different FLP resolution approaches

In almost all of the published papers, the layout efficiency is measured in terms of transportation costs and satisfaction of the adjacency requirement. These costs are related to one or more of the following parameters: distance, unit transportation cost, and estimated total flow to be transported from workstation *i* to workstation *j*. The workstations may also be defined by some adjacency requirements, which define the needed or desired nearness or remoteness for each pair of machines depending on the shared materials or personnel, or according to security, noise, and vibration reasons (Hoseini-Nasab et al. 2017).

To solve the FLP, various resolution methods have been developed. These approaches can be classified into exact, heuristic, and metaheuristic optimisation approaches. They all look to obtain either good solutions subject to certain constraints or global or local optimum solutions that meet one or more performance objectives. These approaches are applied to illustrative cases randomly generated or to real case studies.

Several research works use exact methods to seek optimal solutions for small-sized FLPs. The mainly used exact methods are the branch and bound, dynamic programming, and cutting plane techniques. Among the research papers that introduce exact algorithms, Solimanpur and Jafari (2008) suggested a branch and bound algorithm to solve a nonlinear mixed-integer programming model. Their objective is to minimise the total distance travelled by materials in a given area. Palubeckis (2012) proposed a branch and bound algorithm to solve the problem of the assignment of *n* facilities to *n* locations equally spaced along a straight line in order to minimise the MHC. To solve a dynamic FLP, Dunker et al. (2005) combined a GA to generate a set of candidate layouts with dynamic programming to evaluate the fitness function defined as the costs of material handling and rearrangements. The main weakness of these methods is that they cannot solve large scale FLPs because of the intractable and combinatorial nature of the problem (Ripon et al. 2013). They remain insufficient for realistically sized applications. Hence, heuristic techniques have been introduced to find near-optimal solutions for large size instances with reasonable computation time. These algorithms can be classified into construction, improvement, and hybrid algorithms. The construction algorithms build progressively a layout from scratch by placing a sequence of workstations until a completed layout is obtained. Some of the popular construction algorithms are the computerised relationship layout planning (CORELAP) (Lee and Moore 1967), automated layout design program (ALDEP) (Seehof and Evans 1967), and programming layout analysis and evaluation technique (PLANET) (Deisenroth and Apple 1972). The main drawback of these methods is that the final solution may be unsatisfactory in terms of quality as they generate only one solution. That is the reason why the improvement algorithms start with an initial

Table 1 Discrete versus continuous FLP formulation

Formulation types	Plant site	Equipment size	Distance
Discrete representation	Predetermined locations, i.e., fixed locations	Equal, with fixed or ignored dimensions	Parameters
Continuous representation	No predetermined locations, i.e., decision variables	Equal or unequal area	Variables: it depends on the location of the machines

solution and try then to improve it by swapping the location of facilities. The best-known examples of these methods are the computerised relative allocation of facilities technique (CRAFT) (Armour et al. 1964) and multi-floor plant layout evaluation (MULTIPLE) (Bozer et al. 1994). The hybrid algorithms utilise conjointly the principles of the construction and improvement algorithms. They generate the initial solution and attempt to improve it. BLOCPLAN is a hybrid algorithm (Donaghey and Pire 1990).

Because the FLP is known to be complex, in recent years, metaheuristics approaches have been developed to solve FLPs by GAs (Azadivar and Wang 2000; Shayan and Chittilappilly 2004; Wang et al. 2005; Mazinani et al. 2012; Vitayasak et al. 2016), simulated annealing (Tam 1992; Deb and Bhattacharyya 2003; Mckendall et al. 2006; Sahin and Turkbey 2009; Sahin 2011), tabu search (Chiang and Kouvelis 1996; Liang and Chao 2008; Samarghandi and Eshghi 2010; Bozorgi et al. 2015), ant colony optimisation (Solimanpur et al. 2004; Hani et al. 2007; Komarudin and Wong 2010), and particle swarm optimisation (Samarghandi et al. 2010; Jolai et al. 2012; Asl and Wong 2015; Wang et al. 2014). Interested readers are invited to refer to Kundu and Dan (2012) for an in-depth analysis of the different metaheuristics methods applied in FLPs.

Many hybrid algorithms have been proposed and used in the optimisation of manufacturing and design problems (Pholdee et al. 2017). For example, Moslemipour (2018) proposes a novel hybrid intelligent algorithm by combining the simulated annealing and clonal selection algorithms to solve uncertain dynamic FLPs. A GA combined with a strategy of partial solution deconstruction and reconstruction (PDR) is provided by Paes et al. (2017) to solve the unequal area FLP. Comprehensive comparisons of the efficiency of different optimisation algorithms have been presented by Karagöz and Yildiz (2017). However, Kiani and Yildiz (2016) suggest that it is not enough to present the best and worst results obtained by the used approach, but a comparative study must define whether the algorithms are significantly better than the others or not.

Table 2 provides a synthesis of the related papers in the literature, where the gaps and overlaps are identified with a spotlight on the resolution approaches. Several observations can be made regarding these research works. First, it can be noted that there are two basic types of objectives adopted in the mathematical models for the FLP. Some of them aim to minimise a function related to the travel of parts and operators (MHC, travel distance, etc.) while the others aim to maximise the satisfaction of the proximity requests between two facilities that exchange a large number of parts. Second, it has been noticed that most of the published papers adopted some constraints. Two sets of constraints are introduced: no overlaps are allowed between facilities and the facilities should remain within the site boundaries (Saraswat et al. 2015). However, other sets of constraints such as aisle structure, existing restrictions, and complex geometrical constraints (e.g., fixed barriers, green land, walls, etc.) are rarely considered.

According to the literature, the existing detailed layouts generally use either the rectilinear distance (Ahmadi and Akbari Jokar 2016; Gonçalves and Resende 2015; Heragu and Kusiak 1991; Khalil 1973) or the Euclidean distance (Tompkins and Reed 1976; Van Camp et al. 1991) to evaluate their efficiency. Nevertheless, these methods cannot be applied in the presence of obstacles and barriers. The last column of Table 2 lists the different techniques used to solve the layout problem.

Gaps to fill in FLP resolution

As it can be seen from Table 2, the most frequently used objective function in the FLP is the minimisation of the MHC (71%) followed by the minimisation of the total travelled distance by the material. The table shows that the most commonly used distance determinations are the rectilinear (76%) and Euclidean (6%), which have the shortcomings indicated in Sect. 1. Therefore, the main motivation of this research is to improve the distance computation in order to obtain the most realistic MHC. This was necessary because the facility structure is defined through obstacles and transportation routes. This consideration led to the use of A*. The generation of various candidate layouts was performed by using a GA because we rely on the fact that the FLP does not require a global optimum but a sufficiently good solution, as noticed by Mitchell (1998). This choice was guided by some other facts provided by past research works. In fact, Mazinani et al. (2012) argue that a GA provides better solutions in FLPs than other metaheuristics methods, while other authors have proved the effectiveness of GAs in finding ‘good enough’ solutions to many problems (Palomo-Romero et al. 2017; Aiello et al. 2013; Datta et al. 2011; Hernández Gress et al. 2011). Sadrzadeh (2012) demonstrated that a GA is still an appropriate strategy for addressing problems in many different fields. The genetic-based approaches applied to FLPs make it possible to explore the complex search spaces efficiently while guaranteeing a sound population diversity, i.e., to explore almost every part of the solution space thanks to its crossing and mutation mechanisms. These techniques are encapsulated in a general approach, which is defined hereafter.

Overview of used techniques

Method of calculation of distance between facilities

A* is an optimisation algorithm mainly utilised for determining the shortest paths in a two-dimensional grid, which was

Table 2 Summary of papers related to the FLP

	References	Objective function				Constraints						Distance			Techniques or solver
		A	B	C	D	A	B	C	D	E	F	A	B	C	
Exact methods	Solimanpur et al. (2004)	✓				✓			✓			✓			Branch and bound
	Palubeckis (2012)		✓							✓		✓			Branch and bound
	Kouvelis and Kim (1992)	✓													Branch and bound
	Dunker et al. (2005)	✓		✓		✓			✓			✓			Dynamic programming
	Chraibi et al. (2016)		✓		✓	✓		✓		✓		✓			CPLEX
Heuristics methods	Lee and Moore (1967)				✓						✓				CORELAP
	Seehof and Evans (1967)				✓						✓				ALDEP
	Donaghey and pire (1990)		✓		✓					✓	✓	✓			BLOCPLAN
	Van Camp et al. (1991)		✓			✓			✓				✓		NLT algorithm
Meta-heuristics methods	Gonçalves and Resende (2015)		✓			✓			✓	✓		✓			GA
	Mazinani et al. (2012)		✓	✓		✓			✓	✓		✓			GA
	Sahin and Turkbey (2009)		✓		✓						✓	✓			Simulated annealing
	Sahin (2011)		✓		✓							✓			Simulated annealing
	Samarghandi et al. (2010)	✓	✓							✓		✓			Tabu search
	Liang and Chao (2008)		✓							✓		✓			Tabu search
	ASL et al. (2015)		✓	✓		✓			✓	✓	✓	✓			Particle swarm optimisation
	Guan and lin (2016)	✓	✓							✓		✓			Ant colony optimisation
% of total		29	71	18	35	41	0	6	35	53	29	76	6	0	

Notation conventions:

Objectives functions		Constraints		Distance	
A	Minimise the total distance travelled by the material	A	Overlap constraints	A	Rectilinear distance
B	Minimise MHC	B	Restricted area (obstacles, barriers, walls,…)	B	Euclidean distance
C	Minimise rearrangements cost	C	Aisle structure constraints	C	New method
D	Maximise the satisfaction of weighted adjacency	D	Floor space constraints: each facility should be within the workshop		
		E	Distance constraints: define the distance between all pairs of facilities		
		F	Relationship constraints: define the degree of proximity between facilities		

developed by Hart et al. (1972). The A* algorithm aims at finding an efficient, directed path among all possible paths to a destination. Among the different possible paths, it first examines the ones that lead more quickly to the goal and puts aside all the others. All these possibilities are stored, but not removed, as it is not possible to know in advance whether a path is the shortest path or not. If this road reaches a deadlock, this solution becomes inoperable. As described in Zhou et al. (2013), the implementation of the A* algorithm from source to destination includes the checking of many adjacent nodes, one after another. Using some internal indices, the A* algorithm is directed towards the destination while detecting the shortest possible path. These indices are (i) the total travel cost from the start node to destination, (ii) the cost of the path from the start node to any intermediate node lastly studied on the path, and finally (iii) the cost of the cheapest path from any intermediate node to the destination. Interested readers may refer to Appendix 3 for a more technical description of the algorithm or to Rafia (2010) and Saleh (2015).

Overview of genetic algorithms (GAs)

The Holland GAs (1975) are robust metaheuristics used to solve difficult optimisation problems (Sadrzadeh 2012). The GA solves problems based on the evolution mechanisms and nature of genes (Mazinani et al. 2012). It operates with a set of problem solutions named population. Each individual of the population is a chromosome and represents a possible solution. The fitness value of an individual (solution) represents its quality according to a given objective function; the higher the fitness value is, the more valuable the solution. Usually, the initial population is randomly generated, although a set of known individuals can be used to launch the evolution process. Then, the GA makes this population evolve iteratively. At each iteration, the evolution process works as follows. A subset of individuals is selected as the parents based on their high fitness value. The next generation is obtained thanks to crossover and mutation. The crossover operator allows the combination of two selected parents to produce a better offspring. The mutation operator is used to introduce a new genetic structure in the population by rearranging the structure of a chromosome. Owing to the randomness nature of both crossover and mutation, some children may violate the constraints that all individuals in a population should respect. Therefore, there would be feasible and infeasible individuals among these newly born children. The infeasible or low performance individuals are excluded from the rest of the process. This iterative evolution process may be applied again to these children if the quality of the solutions is not satisfactory. The iterations are stopped if the gap between the ‘i-th’ and ‘(i + 1)-th’ generations is below a specified threshold. The GAs are sensitive to the parameters (size of population, crossover probability, etc.) and the crossover and mutation operators.

In the following sections, the GA steps are detailed. A GA is implemented in two parts: the initial steps and the iterative steps.

Initial steps of the GA application

(1) **Chromosome encoding and representation.** Every individual should be encoded in a relevant way to make it usable for the whole solution determination. The encoding usually takes place after a mathematical modelling phase of the problem.

(2) **Generation of the initial population.** A set of initial solution individuals is either generated randomly or defined by the users as a feasible solution via a heuristic, for example. It is also possible to initiate the algorithm by including ‘good’ or ‘already known’ solutions (Grefenstette 1986; Sadrzadeh 2012; Wu et al. 2007). In fact, the GA needs a number of initial solutions to initiate the exploration of the solution space. The randomly generated solutions can be feasible and unfeasible solutions. On the contrary, the ‘solutions found heuristically’ represent a population of known and feasible solutions obtained thanks to ground experience, and they can be used as the initial solutions.

(3) **Initial evaluation of individuals.** A fitness function measures the quality of the solutions in the search space.

(4) **Filtering of individuals.** The evolutionary algorithms are unconstrained optimisation techniques. First, the individuals are randomly generated without considering constraints. Then, their fitness is evaluated to exclude the worst individuals. Various published papers review the different approaches for handling of constraints (Coello 1999). Among others, Ponsich et al. (2007) identify the following classes: elimination of infeasible individuals, penalisation of their objective function, dominance concepts, preservation of feasibility, repairing of infeasible individuals, and hybrid methods. According to Coello (2002), the majority of studies comparing these constraint-handling techniques are inconclusive. Thus, he suggests to adopt the penalty-based approaches owing to their implementation simplicity and efficiency. The key idea of this technique is to transform a constrained problem into an unconstrained one by introducing the constraints in the objective function via penalty terms, which assign a high penalty to the infeasible solutions that violate at least one constraint.

At the end of this step, if the target performance of the individuals is reached, the algorithm is stopped. Otherwise, a set of iterations will be performed.

Iterations of the GA

The GA seeks to find the best solution over the generations through the following steps that define the process of creating new generations.

(5) **Selection of parents.** The purpose is to create a mating pool consisting of individuals of the population to be combined to create the new generation. The mating pool is used by the crossover and mutation operators. Many selection operators exist to select the best chromosomes to be parents for reproduction. Table 9 in Appendix 4 presents the most cited selection operators with their advantages and shortcomings.

(6) **Crossover process.** This is the first step for producing new individuals (children) from selected parents. The crossover process is specified thanks to (i) the crossover probability and (ii) the crossover operator.

The crossover probability (P_c). It is defined to show the proportion of the population of parents that will be chosen for mating via the crossover operator. Generally, the most used rates are between 0.45 and 0.95 (Al-Zuheri et al. 2014). If P_c is 100%, then all offspring are obtained by crossover. If it is equal to 0%, the offspring represents an exact copy of the parents.

The crossover operator. The previous research works have defined various crossover operators: linear order crossover, uniform crossover, order crossover, etc. (Sastry et al. 2005; Eiben and Smith 2007; Dalle Mura and Dini 2017). The choice of a crossover operator is strongly linked to the kind of problem and the chromosome encoding. The majority of adopted crossover operators are applied to chromosomes that typically contain a gene sequence where the order is mandatory. In other crossover operators, the gene order is not considered. Among others, the N-point crossover techniques are used (see Appendix 5).

(7) **Mutation process.** Mutation helps to maintain diversity in the population by preventing population stagnation at a local optimal solution. The mutation process is specified thanks to (i) the mutation probability and (ii) the mutation operator.

The mutation probability. It defines how often parts of chromosome will be rearranged. If p_m is equal to zero for example, there is no mutation. By increasing this probability, more and more chromosomes will be mutated.

The mutation operator. The mutation operator is applied to two children generated by the crossover operation according to the probability p_m. The two main operators used for mutation are exchange and inversion (see Appendix 6).

(8) **Replacement and evaluation of the children fitness value.** The new solutions or chromosomes can be better or worse than their parents are. Therefore, a replacement strategy is applied here to define the new population by keeping the best individuals. Strategies such as replacement, crowding, and elitist can be used for this purpose (Triki et al. 2016). To do so, each child is evaluated using the fitness function (as in step (3)). Thus, to avoid any loss of the best solutions, the elitist strategy retains the best genetic information of the initial population along with the individuals obtained by the crossing and mutation operators to use them for the next generation.

(9) **Stopping criteria.** One or more criteria may be used to stop the iterations of the GA, such as maximum number of generations, fitness convergence, or computing time (Wu et al. 2007).

Overview of the proposed approach

The proposed approach of FLP resolution is structured in two phases, as presented in Table 3.

Initialisation. The main objective is to minimise the MHC. A set of initial facility arrangements is defined and the distance between workstations is computed. The cost of future facility layouts are compared to the cost of these initial arrangements. The facility area is defined by its global constraints regarding its dimensions and the pre-defined positions of some restricted elements. A restricted element is a limited space within the area that has its own local constraints. A forbidden element is a restricted element, such as walls, stairs, or other barriers, that does not allow any equipment assignment and through which parts cannot pass. A selective element, such as a transportation corridor for materials, parts, and human beings, cannot be assigned to facilities but can be used by product flows. All the workstations are characterised by a rectangular shape whose dimensions are known in advance. The equipment position is defined thanks to the coordinates of the centroid.

Loops of A* and GA application. The candidate layouts are generated by the GA. This algorithm is specified by its crossover and mutation operators and their parameters. The Monte Carlo simulation is used to determine their influence on the performance of the GA and to select the best set of operators and parameters. Finally, a penalty function is introduced in the GA to handle the violated constraints. The MHC of these layouts is computed.

Table 3 Different steps of our proposed approach

	General steps	Our approach
Initialisation	Define the objective of FLP	To minimise the MHC
	Define a path to go from machine i to machine j	Avoid the forbidden elements, allowing the selective elements
	Define the distance computation rules	A* algorithm
	1) Define the area	Global constraints
	by its global constraints	Area limits
	by its local constraints	Area forbidden and selective elements
	2) Define every facility	Facilities have a predetermined rectangular shape
	by its dimensions	Each facility is characterised by an accessibility plan (one unit element around each machine) that allows the operator to access the machine
	by its accessibility area	
	3) Define the flows of parts	A* algorithm
	Between every couple of facilities	
	Qualification of the flow (e.g., density)	
Iterative application of intertwined GA/A*	Generate candidate layouts	GA
	Evaluate the candidate layouts and choose the most appropriate one	Evaluation of MHC

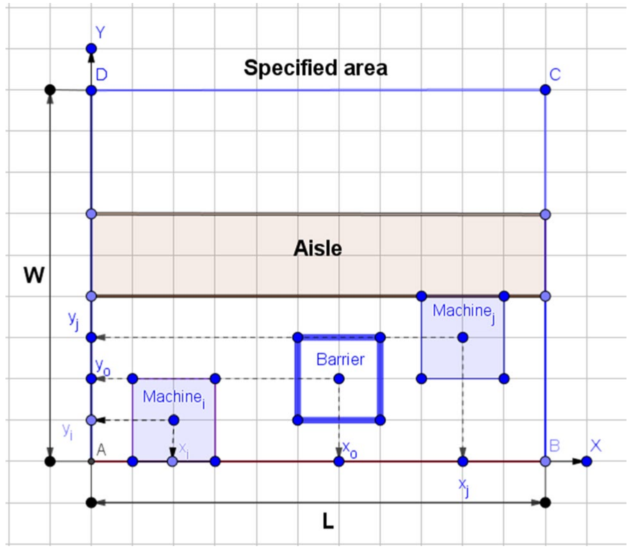


Fig. 1 Representation of the layout problem

The following sections define these two phases in detail.

Initialisation: FLP definition

The problem is to position *n* machines within a generic rectangular facility (see Fig. 1) with a fixed length (L) and width (W). The area is considered as a grid of uniform squares called surface units. An aisle, i.e., a selective element, allows transportation of goods and movement of operators. It is defined by the position of its lower and upper limits, (y_{lower}) and (y_{upper}). The area contains obstacles, i.e., the forbidden elements (the thick blue line in the figure). Around each forbidden element, an accessibility plan is defined (one unit element at all sides) to prevent the case of having a machine placed side by side with an obstacle. A machine *i* is defined by its horizontal (w_i) and vertical (l_i) dimensions and the coordinates of its centroid (x_i, y_i).

The following variables and symbols are used to model the FLP:

Parameters

N	Number of machines
P	Number of products
L	Fixed length of shop floor
W	Fixed width of shop floor
M	Number of obstacles
l_i	Length of machine i
w_i	Width of machine i
l_{oi}	Length of obstacle i
w_{oi}	Width of obstacle i
x_{oi}	The x coordinates of the geometric centre of obstacle i
y_{oi}	The y coordinates of the geometric centre of obstacle i
y_{lower}	Vertical coordinates of the lower side/wall of aisle
y_{upper}	Vertical dimension of the upper side/wall of aisle
f_{ij}	Number of trips between two machines
c_{ij}	Unit cost for transportation over a distance of one unit element from machine i to machine j

Decision variables

x_i	The x coordinates of the geometric centre of facility i
y_i	The y coordinates of the geometric centre of facility i

$$Z_{ij}^{x} \begin{cases} = 1 \text{ if facility i is strictly to the right of facility } j\left(x_i > x_j\right) \\ 0 \text{ otherwise} \end{cases}$$

$$Z_{ij}^{y} \begin{cases} = 1 \text{ if facility i is strictly above facility j}(y_i > y_j) \\ 0 \text{ otherwise} \end{cases}$$

$$Z_{iv}^{x}\begin{cases} = 1 \text{ if obstacle i is strictly to the right of facility v} \\ 0 \text{ otherwise}\end{cases}$$

$$Z_{iv}^{y}\begin{cases} = 1 \text{ if obstacle i is strictly above facility v} \\ 0 \text{ otherwise}\end{cases}$$

Fit function

MHC:

$$\mathrm{MHC} = \sum_{p=1}^{P}\sum_{i=1}^{N}\sum_{j=1}^{N} f_{ij}^{p} * c_{ij} * K_{ij} \tag{1}$$

Constraints:

$$\frac{l_i}{2} \le x_i \le L - \frac{l_i}{2} \quad \forall i = 1 \ldots N \tag{2}$$

$$\frac{w_i}{2} \le y_i \le W - \frac{w_i}{2} \quad \forall i = 1 \ldots N \tag{3}$$

$$\left(x_j - x_i\right) > Z_{ij}^{x}\left(\frac{l_i}{2} + \frac{l_j}{2}\right) \quad \forall i,j = 1 \ldots N \quad \text{with } i \ne j \tag{4}$$

$$\left(x_i - x_j\right) > (1 - Z_{ij}^{x})\left(\frac{l_i}{2} + \frac{l_j}{2}\right) \quad \forall i,j = 1 \ldots N \quad \text{with } i \ne j \tag{5}$$

$$\left(y_j - y_i\right) > Z_{ij}^{y}\left(\frac{w_i}{2} + \frac{w_j}{2}\right) \quad \forall i,j = 1 \ldots ..N \quad \text{with } i \ne j \tag{6}$$

$$\left(y_i - y_j\right) > (1 - Z_{ij}^{y})\left(\frac{w_i}{2} + \frac{w_j}{2}\right) \quad \forall i,j = 1 \ldots N \quad \text{with } i \ne j \tag{7}$$

$$\left(x_v - x_{oi}\right) > Z_{iv}^{x}\left(\frac{l_{oi}}{2} + \frac{l_v}{2}\right) \quad \forall i = 1 \ldots M, \quad \forall v = 1 \ldots N \tag{8}$$

$$\left(x_{oi} - x_v\right) > (1 - Z_{iv}^{x})\left(\frac{l_{oi}}{2} + \frac{l_v}{2}\right) \quad \forall i = 1 \ldots M, \quad \forall v = 1 \ldots N \tag{9}$$

$$\left(y_v - y_{oi}\right) > Z_{iv}^{y}\left(\frac{w_{oi}}{2} + \frac{w_v}{2}\right) \quad \forall i = 1 \ldots M, \quad \forall v = 1 \ldots N \tag{10}$$

$$\left(y_{oi} - y_v\right) > (1 - Z_{iv}^{y})\left(\frac{w_{oi}}{2} + \frac{w_v}{2}\right) \quad \forall i = 1 \ldots .M, \quad \forall v = 1 \ldots .N \tag{11}$$

$$\left(\left(y_i + \frac{w_i}{2}\right) - y_{lower}\right)\left(y_{upper} - \left(y_i - \frac{w_i}{2}\right)\right) < 0 \quad \forall i = 1 \ldots ..N \tag{12}$$

$$\left(\left(x_i + \frac{l_i}{2}\right) - x_{lower}\right)\left(x_{upper} - \left(x_i - \frac{w_i}{2}\right)\right) < 0 \quad \forall i = 1 \ldots N \tag{13}$$

$$Z_{ij}^{x} \in \{0,1\} \quad \forall 1 \le i,j \le N \tag{14}$$

$$Z_{ij}^{y} \in \{0,1\} \quad \forall 1 \le i,j \le N \tag{15}$$

$$Z_{iv}^{x} \in \{0,1\} \quad \forall 1 \le i \le N \quad \forall 1 \le v \le M \tag{16}$$

$$Z_{iv}^{y} \in \{0,1\} \quad \forall 1 \le i \le N \quad \forall 1 \le v \le M \tag{17}$$

$$\text{All other variables } >= 0 \tag{18}$$

Equation (1) minimises the cost of the material flow between machines. Constraint sets (2) and (3) ensure that the machines are assigned within the boundaries of the shop floor. Constraint sets (4)–(7) prevent overlapping of machines. Constraints (8)–(11) prevent overlapping between machines and obstacles. Constraints (12) and (13) ensure that no machine would be assigned in the aisle boundaries. Finally, constraints (14)–(18) define the domains for the different variables.

Iterative application of intertwined GA/A*

The very first step of the GA application is to design a relevant coding for the individuals. We use the equipment centroid to encode the position of each machine by the couple (x,y). Each couple (x,y) is a gene; *x* and *y* are the alleles of this gene. There are *n* machines to place in the facility defined by its limits, the points A, B, C, and D in Fig. 1. It is placed on a 2D orthonormal coordinate plan. The origin of the plan is positioned on the lower left corner of the area, point A in Fig. 1. Each chromosome is composed of $2 \times n$ genes, as shown in Fig. 2. The n alleles $\{ x_1 \ldots x_n \}$ form a vector. This vector represents the position of the machines in the horizontal direction and $\{ y_1 \ldots y_n \}$ defines their position in the vertical direction. The pseudocode of our algorithm is presented below.

Fig. 2 Chromosome associated to the location variables

Location variables	x_1	x_2	x_3	x_4	….	….	…	x_{n-2}	x_{n-1}	x_n
	y_1	y_2	y_3	y_4	….	….	…..	y_{n-2}	y_{n-1}	y_n

Algorithm 1: Pseudocode of our proposed approach

```
Define parameters of the GA using Monte Carlo simulation;
    • Population size
    • Crossover probability
    • Mutation probability
Generate the initial population randomly;
Evaluate the fitness of all the individuals using $A^*$ algorithm;
Repeat c times
    Perform selection of the best individuals for reproduction
    Perform crossover operator;
    Update the fitness of the modified individuals using $A^*$ algorithm;
    Perform mutation operator;
    Update the fitness of the modified individuals using $A^*$ algorithm;
    Generate the population for the next generation;
End;
```

The other specialisations of the used algorithms are defined hereafter:

Generation of the initial population	The initial-solution individuals are generated randomly
Evaluation of initial individuals	Evaluation of the MHC based on computation of the shortest distances between workstations (using A*), see Eq. (1)
Filtering of individuals	The penalty-based approach.
Selection of parents	Roulette wheel (Michalewicz 1994) and tournament (Goldberg and Deb 1991)
Crossover process	
Crossover probability (P_c)	A normal distribution with $\mu = 0.7$ and $\sigma = 0.1$ is used.
Crossover operator	One point crossover (Holland 1975), two point crossover (Starkweather et al. 1991), three point crossover, four point crossover
Mutation process	
Mutation probability	A normal distribution with $\mu = 0.18$ and $\sigma = 0.06$ is used.
Mutation operator	Exchange and inversion operator
Evaluation of the children fitness value	The elitist strategy
Stopping criteria	130 iterations (the fitness function cannot be further improved)

Tuning of GA parameters by Monte Carlo simulation

The performance of a GA relies on setting the values of the basic parameters, such as crossover probability, mutation probability, population size, and selection strategy. However, the local optimum solutions may survive throughout the algorithm if improper parameters are set. To solve this problem, several methods, such as the full factorial design and Taguchi method, are used to calibrate the parameters that influence the performance of a metaheuristic algorithm (Pourvaziri and Naderi 2014). We use the Monte Carlo simulation principle to identify the effects of the GA parameters on the quality of layout generation. Monte Carlo simulations are powerful tools to investigate randomised trials (Yang and Tian 2012). They produce results that are in good agreement with most of the randomised trials.

The experiments are generated to examine the effects of the population size (N), crossover probability (P_c), mutation probability (P_m), and behaviour of the operators (selection, crossover, and mutation). The parameters of the applied Monte Carlo simulation are the following (Fig. 3):

– Population size (number of layouts simultaneously considered in the experiment). It is randomly generated between 100 and 250 following a uniform probability distribution. This allows a sufficiently large population for every simulation. There is no preference of size between the limits.
– Crossover probability. We use a normal distribution with a mean of 0.7 and a standard deviation of 0.1. In this way, we guarantee that new individuals will be introduced, which may be better than the old ones (Angelova and Pencheva 2011).
– Mutation probability. It is generated according to a normal distribution with a mean of 0.18 and a standard deviation of 0.06. By choosing this distribution, it is guaranteed that the mutation will affect a few members of a population in any given generation, and small diversions in the genetic structures of the parents will be introduced.

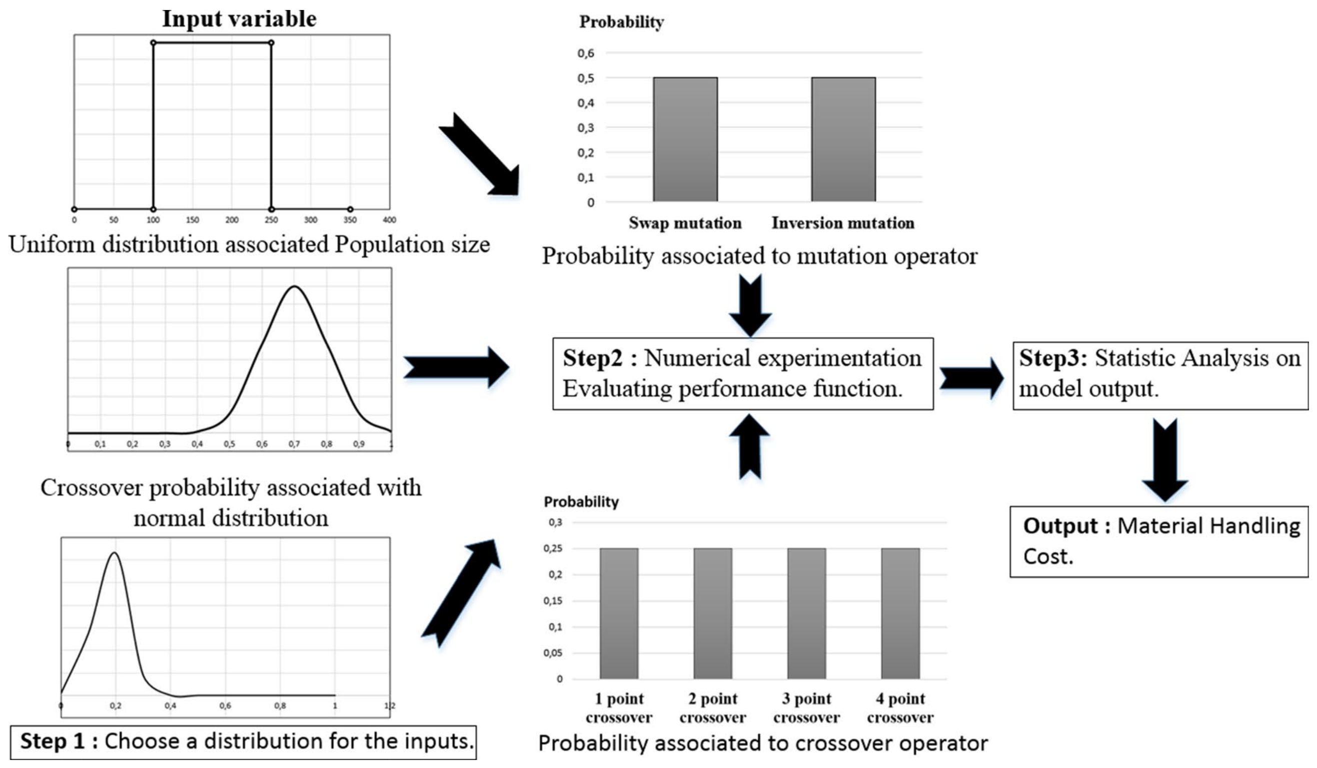

Input variable
Uniform distribution associated Population size
Crossover probability associated with normal distribution
Step 1 : Choose a distribution for the inputs.
Probability
Swap mutation
Inversion mutation
Probability associated to mutation operator
Step2 : Numerical experimentation Evaluating performance function.
Step3: Statistic Analysis on model output.
Output : Material Handling Cost.
1 point crossover
2 point crossover
3 point crossover
4 point crossover
Probability associated to crossover operator

Fig. 3 Monte Carlo simulation

Table 4 Input data of obstacles

	Coordinates (x,y)	Width	Length
Obstacle 1	(12,3)	2	4
Obstacle 2	(7,18)	4	2

A difference between the roulette wheel and tournament selection methods is revealed. The two simulation results are provided in Appendixs 1 and 2 to study the impact of selection operators on the output of our proposed approach. The crossover and mutation operators also affect the performance of the GA. We choose to make a comparison between {one point, two points, three points, four points} crossover operators and also between the exchange and inversion mutation operators. Therefore, in each iteration, one of these crossover and mutation operators is chosen randomly and applied to the selected parents. Each of these operators has an equal probability of being chosen.

Simulations with these operators and parameters are conducted, giving a total of 100 runs for the Monte Carlo simulation and 130 iterations for the GA.

Table 5 Quantity of material flow from facility i to facility j

Facilities	1	2	3	4	5	6	7	8
1		1	2				2	
2			4	3	6			2
3				2		3	1	
4					5	2		2
5								4
6							4	
7								1
8								

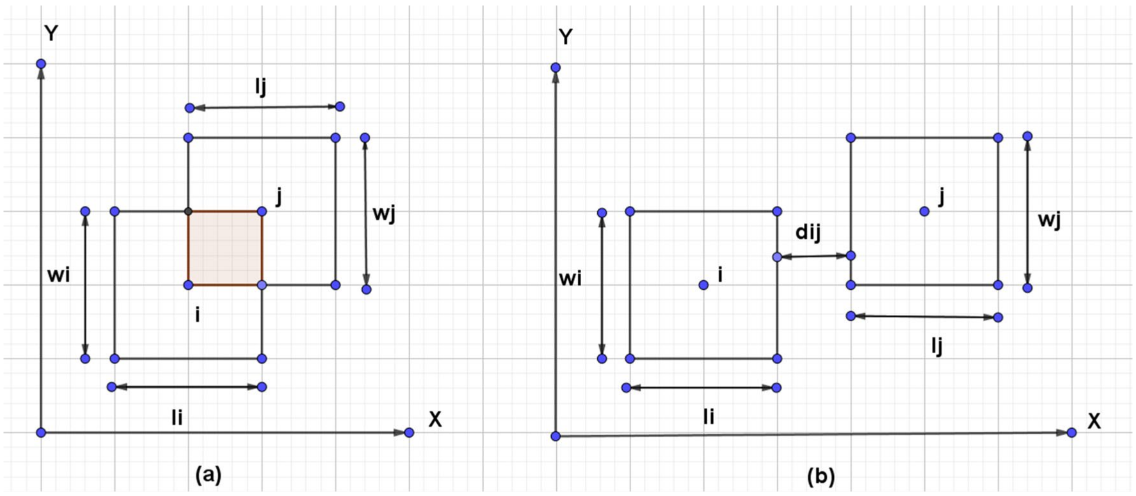


Fig. 4 Illustrative example of the FLP: machines are overlapped (**a**) and machines are not overlapped (**b**)

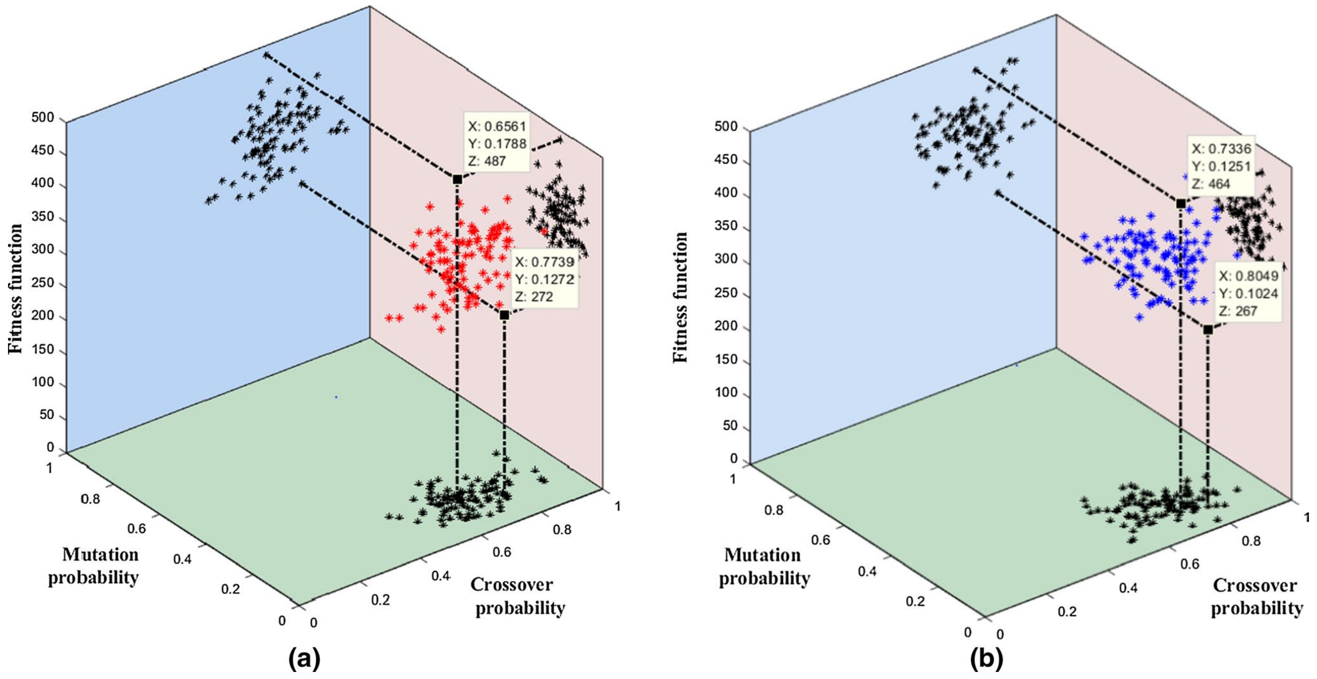


Fig. 5 Best fitness function value versus experiences of the Monte Carlo simulation: roulette operator (**a**) and tournament operator (**b**)

Experiments

To evaluate the efficiency of the proposed approach, we use an illustrative case inspired by an industrial case we studied in (Besbes et al. 2017). The proposed algorithm has been applied to a layout of eight facilities that must be arranged in a plant floor with 30 * 20 square surface units. An aisle with the same length as the plant configuration layout and two different vertical dimensions ($y_{lower} = 9$ and $y_{upper} = 12$) was considered. The input data corresponding to the obstacles are presented in Table 4.

The quantity of material flow between machines is presented in Table 5. This matrix is extracted from Appendix 7. For each couple of facilities, the unit cost to move one product per unit distance from facility i to facility j is fixed equal to 1 monetary unit. The resolution method of the model was programmed and implemented in Matlab and run on an

Intel(R)Core™ i5 3360 M CPU@2.8 GHZ processor with 8 GB RAM. The Monte Carlo simulations are used afterwards to make a sensitivity analysis of the used GAs against its various parameters.

The parameter values for the proposed method and the obtained MHC for each obtained configuration are described in Appendixs 1 and 2.

In the first set of tests, the roulette (wheel selection) operator is used to select parents in order to create better offspring. The second set of tests is done with the tournament operator.

Roulette operator

The best and worst solutions, in terms of the fit function obtained over 100 Monte Carlo simulations by roulette are illustrated in Fig. 5a. This Figure shows the best and worst MHC under different Monte Carlo simulations and the corresponding crossover and mutation probabilities. The best solution (minimum cost) obtained by the roulette operator is 272 and the worst one is equal to 487 (Appendix 1). Figure 12a in Appendix 8 demonstrates the convergence to the best function value of the layout problem. The corresponding layouts are represented in Fig. 6a, b. The machines are not overlapped. Each machine is characterised by its centroid (x_i, y_i), a predetermined length l_i, and width w_i. The obtained results demonstrate that between each couple of machines there is a distance $d_{ij} \geq 1$, as shown in Fig. 4b, and there is not a common surface between two machines as illustrated in Fig. 4a.

In the same Fig. (6a, b), we report the shortest paths found by A*, represented by coloured broken lines. The corresponding heat map of distances is illustrated in Fig. 7, in the top maps. These figures show the contrast between the best and the worst solutions obtained by the roulette operator.

Tournament operator

Applying the tournament operator, the cost of the best and worst solutions found by the GA combined with the A* algorithm over the 100 Monte Carlo simulations are illustrated in Fig. 5b. The best-fit function value of the FLP is equal to 267 and the worst one is equal to 464. Figure 12b in Appendix 8 presents the convergence to the best function value of the layout problem. The configurations associated with the best and the worst solutions are given in Fig. 6c and d. All the machines respect all the constraints. The shortest path values using the A* algorithm are reported in Appendix 9, and the corresponding heat map of distances is illustrated in Fig. 7.

Discussion of the obtained results

Regarding all the experimentations performed, the combination of GA parameters that produces the best results is the following:

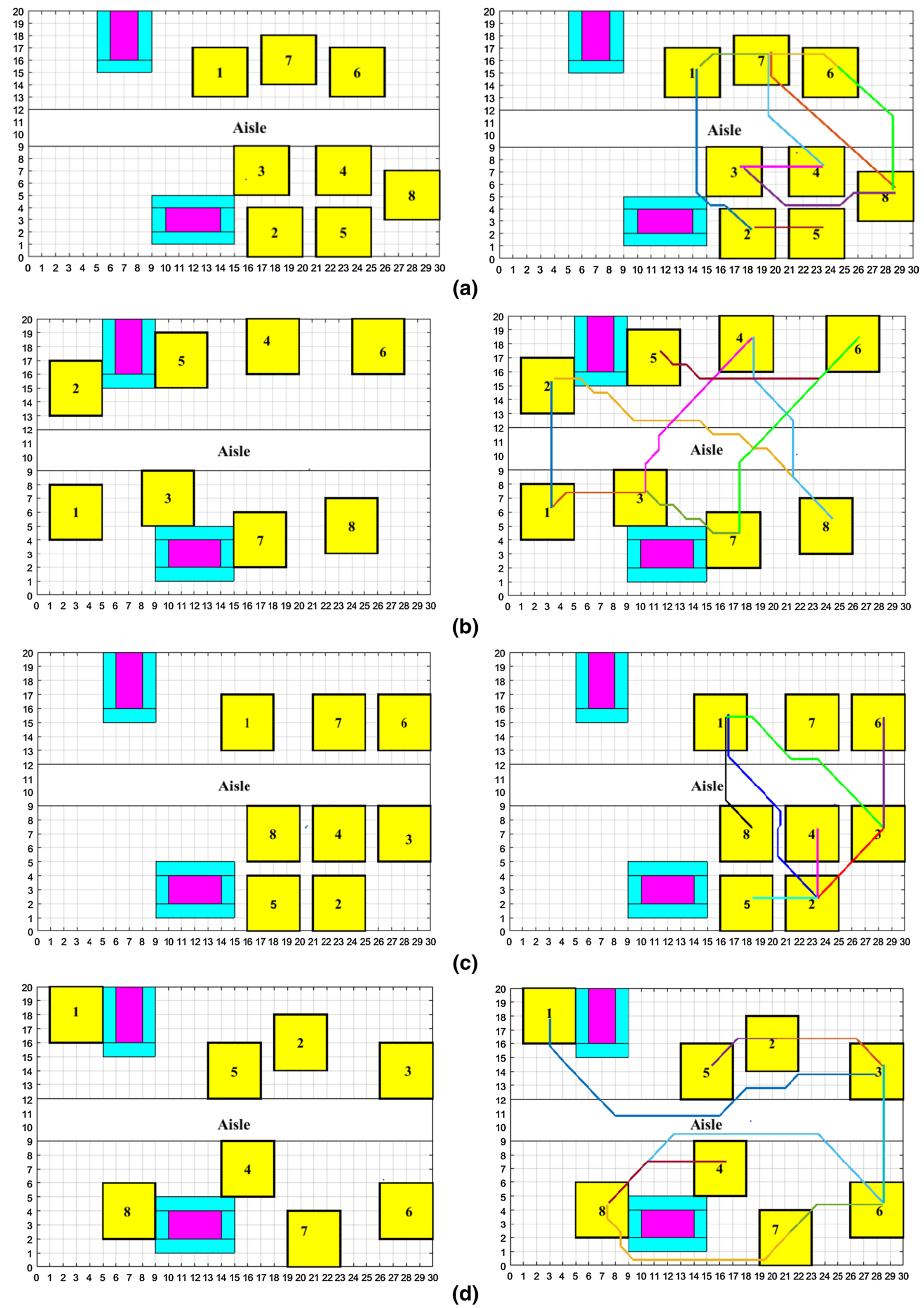
– Population size is 194
– Crossover probability is 0.8049
– Mutation probability is 0.1024
– Selection operator is tournament

As shown in Fig. 8, the best MHCs found by the approach with the Monte Carlo simulations, for both the tournament and roulette operators, follow a normal distribution. These variations can be represented by the mean μ and the standard deviation σ, as reported in the following table.

Roulette wheel selection method	Tournament selection method
$\mu_r = 359.61$	$\mu_t = 374.39$
$\sigma_r = 40.1147$	$\sigma_t = 36.3092$
$cv_r = \frac{\sigma_r}{\mu_r} = 11.15\%$	$cv_t = \frac{\sigma_t}{\mu_t} = 9.70\%$

It might be observed that $\mu_r < \mu_r$; however, $\sigma_r > \sigma_t$. This means that by focusing only on the mean value, the tournament operator gives in general better solutions than the roulette wheel selection method. However, the density of MHCs is lower than that provided by the roulette operator. By comparing the standard deviations of these two operators, it might be concluded that the tournament gives a distribution less flattened. However, to compare these two distributions appropriately, we use the coefficient of variations. In this case, it can be concluded that these two distributions remain quite similar owing to the very low difference between their coefficients of variations. The final conclusion about the use of these two operators is that the roulette operator tends to suggest more effective configurations with less MHC. However, it is necessary to apply both operators to be sure that the obtained configuration reduces as much as possible the MHC (the best configurations obtained for the roulette and tournament operators are shown in Fig. 6a, c).

These results show the practicability and applicability of the suggested method. Further, they demonstrate that the proposed method is efficient and useful for the placement of a set of rectangular machines with defined area requirements, which have to be located, without overlapping, in a given organisation satisfying a set of constraints. The main advantage of the proposed approach is its ability to explore a large space of solutions, keeping the practicability of the design and considering more realistic distances between the facilities.


Aisle
(a)
(b)
(c)
(d)

◂**Fig. 6** Monte Carlo simulations and product flow obtained by A* algorithm: best (**a**) and worst (**b**) layouts found via roulette, best (**c**) and worst (**d**) layouts found via tournament in the Monte Carlo simulations

Finally, as presented in Table 6, the proposed approach gives a better solution than a GA integrated with Euclidean or rectilinear distances. By using the roulette wheel selection method, the MHC is decreased by 8.93% compared to that with Euclidean distance, and by 33% compared to that with rectilinear distance. As for the tournament selection method, the MHC is decreased by 13.15% compared to that with Euclidean distance and by 36.7% compared to the one with rectilinear distance.

Finally, once the best configuration is obtained, the approach determines the best theoretical routes for products from one workstation to another (see in Fig. 6 the configurations with an extract of the routes to obtain a clearer visibility). Hence, it can be presented as a good basis for the development of an advanced support tool to help engineers and designers in determining the most effective layout through a realistic approach. The convergence of the used algorithms is based on genetic parameters, and it is significantly influenced by the population size and the probabilities of crossover and mutation. These solutions are characterised by different values of the fit function. Nevertheless, all of the solutions (the facility arrangements) satisfy all of the constraints and path requirements. Observing the best and worst solutions obtained in both cases, we notice that, in the first one, most equipment units share the same accessibility plan to allow the transportation of materials and operators. In fact, all the facilities are concentrated in a determined location. Consequently, the remaining space in the workshop can be employed for other usages (storage, for example) or new value-added equipment. On the contrary, in the worst solutions, the workstations are distributed and scattered across the plant floor. Each machine has its own accessibility plan. This aspect could be attractive for the decision makers when negative factors (e.g., bad smells and noise) that have not been modelled in the FLP formulation exist within a physical spatial environment, and thus, it is necessary to avoid certain locations for particular facilities. However, if this is not the case, a significant amount of space, which can be exploited for other activities, will be lost.

The best class of configurations may need adjustment by the decision makers if the company needs to include qualitative considerations in the design.

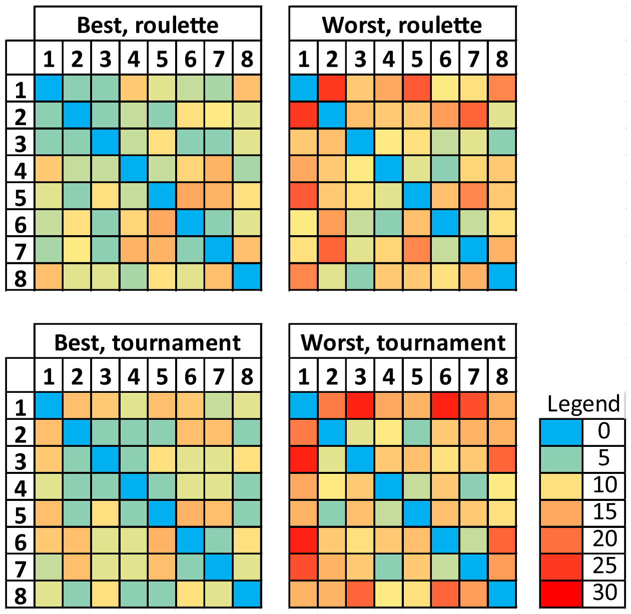


Fig. 7 Heat map of distances obtained by roulette and tournament

Conclusions and future scope of work

This work proposes a new approach to solve the FLP. The main idea is to arrange facilities in a planar site while taking into account various kinds of geometric facility requirements. Some issues, such as site boundaries, overlap elimination, and aisle and obstacle consideration, were mathematically formulated. The objective is to minimise the MHC.

GAs are adopted in the literature to solve large facility design problems. A GA combined with A* allows to explore the solution space. The A* algorithm is used as a new method to determine the shortest path between two facilities. The GA involves chromosome encoding and generation of initial population, fitness function definition, and choice of the selection strategy and the crossover and mutation operators, while defining the handling constraints and stopping criteria.

The proposed algorithm parameters are calibrated using Monte Carlo simulations. The whole approach is applied on an illustrative case and the obtained results show the practicability and applicability of the proposed model. The random variation of the parameters provides one solution for each simulation. Our simulations produce two classes of arrangements. In the first class, the different workstations are concentrated in a determined location. In the second class, the machines are scattered across the workshop. The discussions about the results show that expert knowledge is required to adapt the final solution to the un-modelled usage constraints, i.e., those ones difficult to model through a mathematical model. For future studies, the method should evaluate solutions in terms of flow density on plant aisles in order to make it possible, for example, the identification of potential critical cross points. In addition, some rules can be incorporated to obtain diverse layout configurations, and

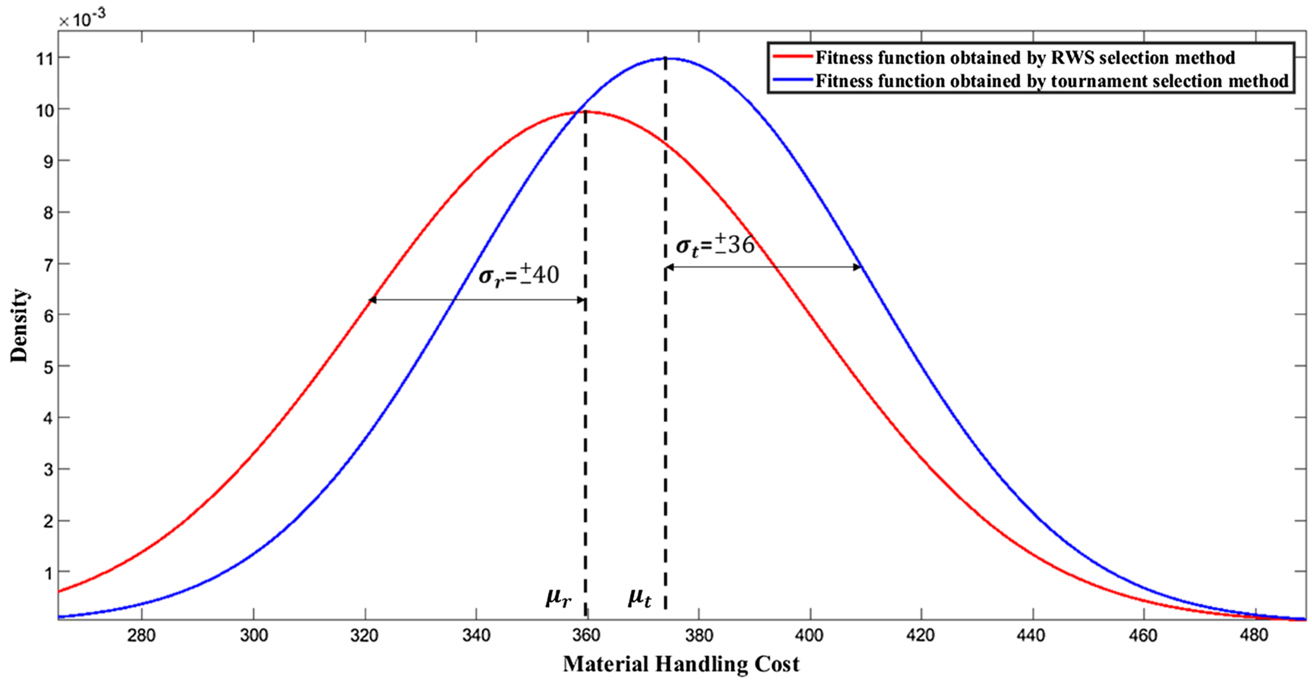


Fig. 8 Probability associated with normal distribution of the fitness function

Table 6 Comparison of total cost associated with the Euclidean and rectilinear distances and A* algorithm

	Euclidean distance	Rectilinear distance	A^* Algorithm
Best roulette	296.3148	362	272
Best tournament	302.1296	365	267

the most suitable one could be chosen among them. For example, we should avert bypassing and backtracking and eliminate the overlap of all kinds of flows in order to reduce the risk of collisions. The shape of the existing plant floor is another element that can be considered. In fact, a real area can have regular shapes or it could be irregular. In a future research work, the authors aim to integrate these constraints as well as some parameters such as the machines' types, their supply of parts and delivery of products/components and the potential reworking in the proposed method and to validate its effectiveness through the application to real cases.

Appendix 1

See Table 7.

Table 7 Experiment layout and sample experimental data (RWS method)

	GAs parameters			Fitness function		GAs parameters			Fitness function
	P_c	P_m	N			P_c	P_m	N	
1	0.6731	0.1787	137	355	51	0.6406	0.0855	181	434
2	0.6469	0.2175	114	373	52	0.6125	0.1053	122	350
3	0.7707	0.1097	196	337	53	0.8265	0.1817	168	427
4	0.6315	0.2550	211	347	54	0.8698	0.2099	120	335
5	0.7667	0.2006	138	417	55	0.5722	0.2538	194	384
6	0.6161	0.0888	249	348	56	0.7802	0.1378	109	408
7	0.7413	0.1770	241	331	57	0.6975	0.2073	185	379
8	0.6921	0.1209	178	390	58	0.8889	0.2693	147	333
9	0.6160	0.1961	145	367	59	0.6195	0.1266	177	347
10	0.6609	0.1640	170	351	60	0.6561	0.1788	119	487
11	0.6108	0.1895	192	387	61	0.6812	0.2027	239	344
12	0.6531	0.0832	193	346	62	0.7433	0.1098	179	298
13	0.6665	0.1825	138	448	63	0.6579	0.1362	235	304
14	0.6682	0.2073	158	296	64	0.6462	0.2030	112	286
15	0.6995	0.1554	117	305	65	0.6956	0.1549	123	407
16	0.6970	0.1474	162	394	66	0.6270	0.0931	143	413
17	0.6513	0.1951	138	350	67	0.7854	0.1761	113	322
18	0.6518	0.2836	111	382	68	0.7487	0.2641	215	335
19	0.7871	0.2016	162	375	69	0.9012	0.1360	175	342
20	0.4858	0.2010	164	302	70	0.6392	0.1055	228	327
21	0.7307	0.1464	249	372	71	0.6603	0.2933	161	346
22	0.7615	0.1760	102	380	72	0.7345	0.1390	191	381
23	0.6698	0.1985	199	370	73	0.6441	0.2745	182	427
24	0.6858	0.1657	169	299	74	0.7170	0.0723	240	338
25	0.5868	0.1167	185	345	75	0.6680	0.2056	138	313
26	0.7015	0.1565	230	296	76	0.6971	0.2156	222	405
27	0.7502	0.1754	198	356	77	0.5651	0.1244	122	288
28	0.7388	0.2246	136	384	78	0.5523	0.1891	109	379
29	0.6005	0.1582	217	379	79	0.7809	0.1246	151	404
30	0.8269	0.2241	154	376	80	0.7197	0.1979	247	401
31	0.6336	0.3123	121	380	81	0.7739	0.1272	228	272
32	0.5985	0.2110	215	377	82	0.7662	0.1468	193	393
33	0.6326	0.1552	150	403	83	0.8010	0.2082	193	366
34	0.6323	0.2275	219	403	84	0.4652	0.2205	155	301
35	0.6811	0.0691	207	389	85	0.6343	0.1889	170	365
36	0.7668	0.1715	149	392	86	0.6066	0.1874	216	399
37	0.6727	0.2378	117	330	87	0.8866	0.3108	133	333
38	0.6543	0.2118	177	324	88	0.5107	0.1540	118	323
39	0.9020	0.1221	223	378	89	0.9176	0.1821	156	328
40	0.7052	0.1786	159	365	90	0..7768	0.1377	211	357
41	0.6380	0.2044	112	337	91	0.7216	0.1317	117	334
42	0.6038	0.1968	141	283	92	0.6688	0.1853	224	387
43	0.5457	0.1797	128	311	93	0.7288	0.0878	134	421
44	0.8004	0.2045	147	382	94	0.5528	0.0981	208	377
45	0.7146	0.1995	226	378	95	0.8222	0.1660	235	368
46	0.6605	0.1175	206	324	96	0.5854	0.1436	195	318
47	0.7681	0.2347	194	365	97	0.6212	0.1597	179	350
48	0.6029	0.1542	203	418	98	0.6106	0.2381	156	344
49	0.6871	0.2959	123	355	99	0.6545	0.1488	195	343
50	0.6117	0.2264	246	309	100	0.7767	0.1203	215	377

Appendix 2

See Table 8.

Table 8 Parameter sets of the proposed GAs (tournament selection)

	GAs parameters			Fitness function		GAs parameters			Fitness function
	P_c	P_m	N			P_c	P_m	N	
1	0.6966	0.2349	180	395	51	0.8147	0.2041	150	372
2	0.7216	0.1599	142	349	52	0.8461	0.1549	242	391
3	0.7717	0.1466	185	361	53	0.7141	0.3117	161	379
4	0.7561	0.1814	222	382	54	0.7741	0.1290	242	374
5	0.7603	0.1049	153	318	55	0.5567	0.2967	202	369
6	0.7116	0.1495	224	313	56	0.6747	0.2085	113	383
7	0.7215	0.1377	205	410	57	0.7265	0.1764	105	349
8	0.6400	0.1229	247	331	58	0.5721	0.1554	245	370
9	0.7662	0.1104	236	404	59	0.7383	0.1694	236	332
10	0.6128	0.1442	234	371	60	0.8049	0.1024	194	267
11	0.7860	0.1447	215	331	61	0.7120	0.0483	121	399
12	0.7770	0.1990	180	352	62	0.8113	0.1873	220	426
13	0.5859	0.2447	117	408	63	0.7656	0.1603	217	349
14	0.7564	0.2159	249	387	64	0.7596	0.1404	144	386
15	0.8451	0.2407	178	458	65	0.7872	0.1447	223	366
16	0.7175	0.2954	195	373	66	0.7086	0.2010	164	383
17	0.7118	0.1893	171	355	67	0.8634	0.1415	176	415
18	0.6515	0.2598	122	394	68	0.6886	0.3135	158	403
19	0.7009	0.0958	168	374	69	0.8220	0.2168	191	383
20	0.6675	0.2297	197	379	70	0.8449	0.0810	141	341
21	0.6155	0.2330	117	337	71	0.8180	0.1416	115	361
22	0.8026	0.1764	164	312	72	0.8106	0.1471	206	372
23	0.8245	0.0931	208	446	73	0.6309	0.1233	217	329
24	0.6960	0.1978	223	396	74	0.6884	0.2705	239	412
25	0.9176	0..1251	118	353	75	0.7510	0.2135	198	326
26	0.7420	0.2016	204	343	76	0.7263	0.1409	128	378
27	0.5460	0.2624	197	381	77	0.7371	0.1975	250	365
28	0.5751	0.1197	235	363	78	0.7267	0.2404	181	420
29	0.7458	0.1935	192	371	79	0.7306	0.1662	197	408
30	0.6736	0.2215	113	405	80	0.6094	0.1825	218	427
31	0.7673	0.2348	179	355	81	0.6040	0.1330	235	314
32	0.6929	0.1367	178	392	82	0.6508	0.2387	159	376
33	0.7981	0.1832	224	337	83	0.7143	0.2587	148	377
34	0.7336	0.1251	106	464	84	0.6121	0.1886	169	375
35	0.6389	0.2396	249	382	85	0.7768	0.1153	238	428
36	0.6779	0.1970	207	367	86	0.7216	0.2509	152	373
37	0.7765	0.1220	242	355	87	0.6064	0.2308	194	408
38	0.7805	0.0315	129	343	88	0.5596	0.1976	170	408
39	0.6031	0.2256	210	376	89	0.7079	0.1887	228	308
40	0.7655	0.1378	203	331	90	0.6214	0.1689	189	415
41	0.8636	0.1545	180	455	91	0.7934	0.1898	161	359
42	0.6682	0.1450	130	372	92	0.6956	0.2041	129	409
43	0.7372	0.1709	174	401	93	0.6470	0.1543	154	410
44	0.8453	0.1721	192	287	94	0.6762	0.2560	136	347

Table 8 (continued)

	GAs parameters			Fitness function		GAs parameters			Fitness function
	P_c	P_m	N			P_c	P_m	N	
45	0.5369	0.0806	108	429	95	0.5477	0.0832	154	398
46	0.8952	0.2277	106	353	96	0.5731	0.3236	229	397
47	0.6258	0.1762	153	334	97	0.6639	0.1853	241	402
48	0.7197	0.1134	134	404	98	0.6339	0.1442	133	391
49	0.5650	0.2573	241	399	99	0.6758	0.2627	128	345
50	0.6801	0.1359	174	328	100	0.7554	0.1336	199	348

Appendix 3

The A* algorithm seeks a new measure of distance (see Fig. 9). It starts by placing the starting node in the open list called current node. The open list contains the list of nodes to be verified. It consists of nodes that have been visited but not explored yet. After inspecting its entire adjacent squares, the starting node is removed from the open list and placed in another list named closed list. The closed list contains all the nodes that, at one time or another, have been considered as part of the solution path. Before switching to the closed list, a node must first pass through the open list. Before being judged as good, it must first be studied by looking at all surrounding squares of the starting point while ignoring squares with obstacles. These studied nodes are also added to the open list. For each of these squares, the current node is saved as its ‘parent square’. The parent square is required when we want to trace the path. To determine if a node is susceptible to be part of the solution path, it is necessary to quantify its quality. To do this, we calculate the distance between the point studied and the last point that was judged as good. We also calculate the distance between the point studied and the point of destination. The sum of these two distances gives the quality of the studied node. This operation will be carried on until the target node is reached.

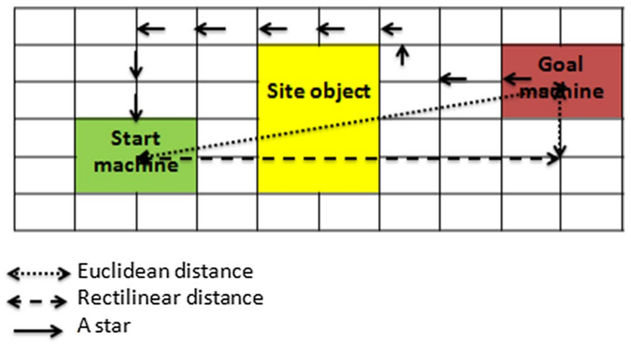


Fig. 9 Different shortest paths between two facilities: rectilinear, Euclidean, A* Algorithm

Definition of different elements of pseudo code

node start	It represents the start position.
node goal	It represents the destination.
X	It represents the node under consideration.
Open list	Open list stores all successive paths that remain to be explored.
Closed list	Closed list stores all paths that have been explored.
g(x)	g(x) is the cost of the path from the initial state to node x.
k(x,y)	It is the cost of the path from x to y, x is the successor of y.
h(x)	h(x) is the heuristic estimate or the cost or a path from node x to a goal.
f(x)	f(x) estimates the lowest total cost of any solution path going through node x.

Pseudo code*

```
Open list     { node_start}
Closed list
g(node start)    0
f(node start)    h(node start)
Put all obstacles on the Closed list
while          the Open list < >
|              Find the node x with the lowest f on the Open list
|              f(x) = g(x) + h(x)
|              Insert x in Closed list
|              if        x is node goal we have found the solution;
|              then      Break
|              else      Generate each state y that come after node current
|              |         for each y of x do
|              |         |          g(y) = g(x) + k(x,y)
|              |         |          if  y     { Open list    Closed list} or g(y) > g(x) + k(x,y)
|              |         |          |  g(y)    g(x) + k(x,y)
|              |         |          |  f(x)    g(x) + h(x)
|              |         |          |  Parent (y)    x
|              |         |          |  Insert "node successor" in Open list
|              |         |           End  %if
|              |         end  % for
|              end    % if
end    % while
```

Appendix 4

See Table 9.

Roulette wheel selection. A chromosome is selected from the mating pool with a probability proportional to its fitness value. A chromosome with a high fitness value has a higher chance of being selected as a parent. Thus, the main idea is to prefer better solutions to worse ones. Consider a circle divided into *n* slices, where *n* is the number of individuals in the population. Each individual is assigned a slice of the roulette wheel, which is proportional to its fitness value. A pointer is chosen on the wheel circumference and the wheel is rotated in a repetitive way. The portion of the wheel that is in front of the fixed point is chosen as the parent.

Tournament selection. N individuals are randomly taken up from the population in a tournament selection. A copy of the best individual (based on fitness values) is kept in the mating pool as a parent. The number of individuals competing in each tournament is referred to as the tournament size.

Appendix 5

N point crossover. Almost all of the N point crossover processes use the same strategy. Hereafter, we define the single point crossover. A split line is randomly determined between 1 and (n – 1) genes (see Fig. 10), where *n* is the number of genes. Each parent is divided into two blocks of alleles. The portion before the split line is exchanged between the parents to create two new solutions, as illustrated in Fig. 10. This crossover operation is analogue to the binary crossover operation of GAs.

See Fig. 10.

Table 9 Advantages and disadvantages of different selection strategies

Advantages	Disadvantages
Roulette wheel selection	
Easy to implement It gives a chance to all individuals to be selected Diversity in the population is preserved. Even poor performance individuals are may be chosen	At the beginning of the search, a fit individual will introduce a bias that may cause a premature convergence and a loss of diversity For instance, if an initial population includes one or two very fit individuals and the rest of the population is not good, these fit individuals will quickly dominate the whole population and prevent the population from exploring other potentially better individuals (Razali and Gerghty 2011)
Tournament selection	
Efficient Simple to implement No requirement for fitness scaling or sorting	Some individuals may never be selected for a tournament while other individuals participate in each tournament. This leads to a loss of diversity (Razali and Gerghty 2011)
Random selection	
Easy to implement The diversity is preserved, all the individuals have an equal probability of selection	If individuals in a population have the same probability to be selected, it will be very difficult for the population to improve because the selection probabilities for fit and unfit individuals are very similar. Therefore, random selection is, on average, slightly more disruptive than the roulette wheel selection in terms of disruption of genetic codes (Sivanandam and Deepa 2008)

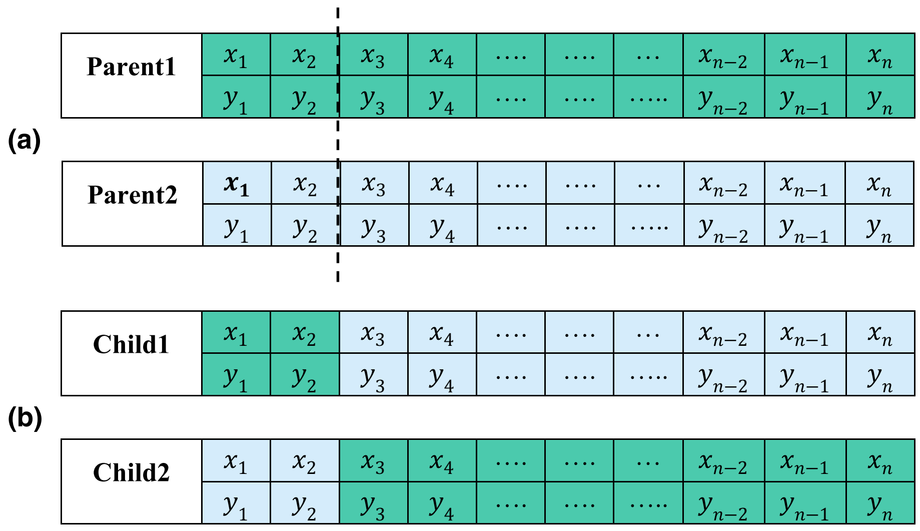


Fig. 10 **a** Before single point crossover, **b** after single point crossover

Appendix 6

Several types of mutation exchange and inversion operators are used at this step to permute two or more workstations. The exchange mutation performs a swap of two randomly selected genes. For instance, in Fig. 11, the two genes (x2, y2) and (x4, y4) are swapped. The inversion operator consists in selecting two genes randomly and inverting the position in the chromosome between these two blocks as shown in Fig. 12.

See Figs. 11, 12.

Appendix 7

Pre-treatment phase In this study, we try to determine the most efficient arrangement of equipment in an area of such company that produce four products {A-E-C-H} and five semi-final products {G-D-B-F-I}. The assembly of G and D gives the product A and the product E comes from the assembly of B, F and I. Data of the products of our problem are provided in Table 10.

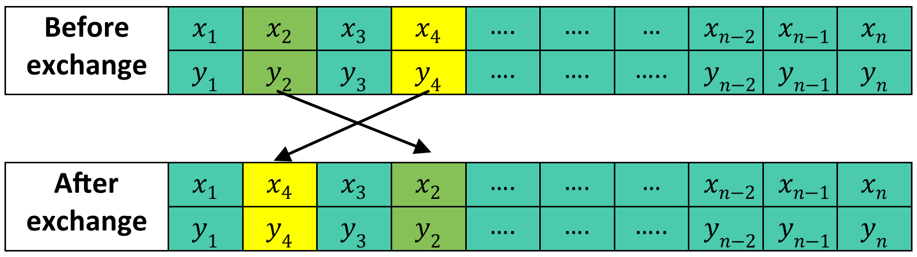


Fig. 11 Exchange operator

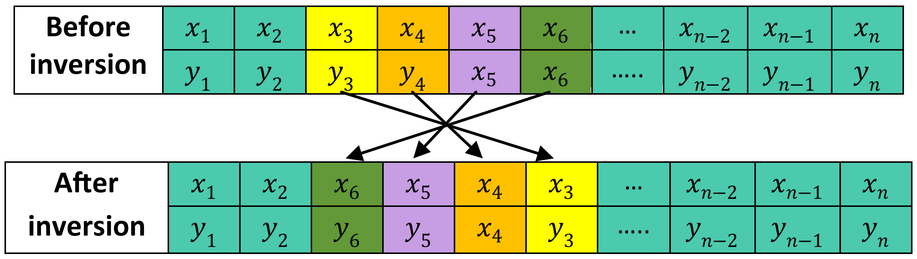


Fig. 12 Inversion operator

Table 10 Data of the products of our problem

Product type	Production route	Quantity to produce
A	1-2-8	1
B	1-3-4-8-5	2
C	5-2-3-6	3
D	2-3-7-8	1
E	1-7-6	2
F	4-6-7	2
G	2-8	1

Appendix 8

See Fig. 13.

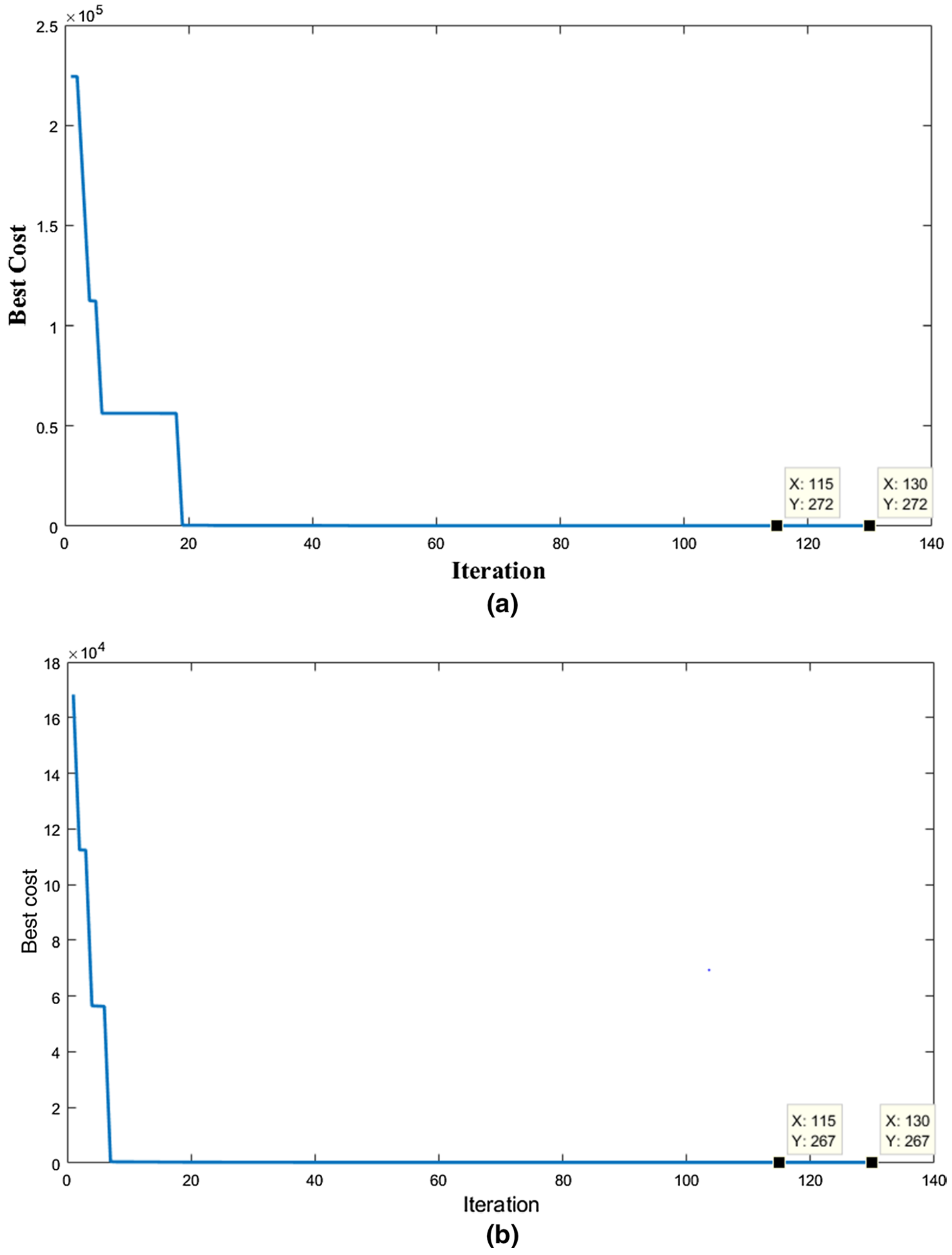


Fig. 13 Best cost versus number of iterations, roulette operator (**a**) and tournament operator (**b**)

Appendix 9

SeeTable 11.

Table 11 Obtained distance among equipment units according to the roulette and tournament operators

	Best, roulette									Worst, roulette							
	1	2	3	4	5	6	7	8		1	2	3	4	5	6	7	8
1	0	14	8	9	13	10	5	14	1	0	9	7	15	11	23	14	21
2	14	0	5	5	5	13	14	10	2	9	0	8	15	8	23	14	21
3	8	5	0	6	6	8	9	11	3	7	8	0	11	10	16	7	14
4	9	5	6	0	5	8	9	5	4	15	15	11	0	7	8	14	13
5	13	5	6	5	0	13	14	5	5	11	8	10	7	0	15	13	13
6	10	13	8	8	13	0	5	10	6	23	23	16	8	15	0	14	13
7	5	14	9	9	14	5	0	11	7	14	14	7	14	13	14	0	7
8	14	10	11	5	5	10	11	0	8	21	21	14	13	13	13	7	0
	Best, tournament									Worst, tournament							
	1	2	3	4	5	6	7	8		1	2	3	4	5	6	7	8
1	0	13	12	8	13	12	7	8	1	0	19	27	15	14	27	23	14
2	13	0	5	5	5	13	13	5	2	19	0	8	9	5	12	14	14
3	12	5	0	5	10	8	8	10	3	27	8	0	12	13	10	12	21
4	8	5	5	0	5	8	8	5	4	15	9	12	0	7	12	5	9
5	13	5	10	5	0	14	13	5	5	14	5	13	7	0	13	12	10
6	12	13	8	8	14	0	5	10	6	27	12	10	12	13	0	7	21
7	7	13	8	8	13	5	0	8	7	23	14	12	5	12	7	0	16
8	8	5	10	5	5	10	8	0	8	14	14	21	9	10	21	16	0

References

Ahmadi, A., & Jokar, M. R. A. (2016). An efficient multiple-stage mathematical programming method for advanced single and multi-floor facility layout problems. *Journal of applied Mathematical Modelling, 40*(9–10), 5605–5620.

Ahmadi, A., Pishvaee, M. S., & Jokar, M. R. A. (2017). A survey on multi-floor facility layout problems. *Journal of Computers and industrial engineering, 107*(2017), 158–170.

Aiello, G., Scalia, G. L., & Enea, M. (2013). A non dominated ranking multi objective genetic algorithm and electre method for unequal area facility layout problems. *Expert Systems with Applications, 40*(12), 4812–4819.

Al-Zuheri, A., Luong, L., & Xing, K. (2014). Developing a multiobjective genetic optimisation approach for an operational design of a manual mixed-model assembly line with walking workers. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-014-0934-3.

Angelova, M., & Pencheva, T. (2011). Tuning genetic algorithm parameters to improve convergence time. *International Journal of Chemical Engineering*. https://doi.org/10.1155/2011/646917.

Armour, G. C., Buffa, E. S., & Vollmann, T. E. (1964). Allocating facilities with CRAFT. *Harvard Business Review, 42,* 136–158.

Asl, A. D., & Wong, K. Y. (2015). Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-015-1053-5.

Azadeh, A., Moghaddam, M., Nazari, T., & Sheikhalishahi, M. (2016). Optimization of facility layout design with ambiguity by an efficient fuzzy multivariate approach. *The International Journal of Advanced Manufacturing Technology, 84*(1), 565–579.

Azadivar, F., & Wang, J. (2000). Facility layout optimization using simulation and genetic algorithms. *International Journal of Production Research, 38,* 4369–4383.

Besbes, M., Costa Affonso, R., Zolghadri, M., Masmoudi, F. & Haddar, M. (2017). Multi-criteria decision making for the selection of a performant manual workshop layout: a case study. In *The 20th World Congress of the International Federation of Automatic Control, (IFAC2017)* 9th–14th July, 2017, Toulouse, France.

Besbes, M., Costa Affonso, R., Zolghadri, M., Masmoudi, F. & Haddar, M. (2018). A survey of different design rules-based techniques for facility layout problems. In *Proceedings of the Tools and Methods of Competitive Engineering Conference.*

Bozer, Y. A., Meller, R. D., & Erlebacher, S. J. (1994). An improvement type layout algorithm for single and multiple-floor facilities. *Management Science, 40*(7), 918–932.

Bozorgi, N., Abedzadeh, M., & Zeinali, M. (2015). Tabu search heuristic for efficiency of dynamic facility layout problem. *International Journal of Advanced Manufacturing Technology, 77*(1–4), 689–703.

Chiang, W. C., & Kouvelis, P. (1996). An improved tabu search heuristic for solving facility layout design problems. *International Journal of Production Research, 34,* 2565–2585.

Chraibi, A., Kharraja, S., Osman, I. H., & Elbeqqali, O. (2016). A particle swarm algorithm for solving the multi-objective operating theater layout problem. *IFAC-Papers Online, 49*(12), 1169–1174.

Coello, C.A.C. (1999). A survey of constraint handling techniques used with evolutionary algorithms. Lania-RI-99-04, Laboratorio Nacional de de Informtica Avanzada, 1–33.

Coello, C. A. C. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics, 16,* 193–203.

Dalle Mura, M., & Dini, G. (2017). A multi-objective software tool for manual assembly line balancing using a genetic algorithm. *CIRP Journal of Manufacturing Science and Technology, 19,* 72–83.

Datta, D., Amaral, A. R., & Figueira, J. R. (2011). Single row facility layout problem using a permutation-based genetic algorithm. *European Journal of Operational Research, 213*(2), 388–394.

Deb, S. K., & Bhattacharyya, B. (2003). Manufacturing facility layout design based on simulated annealing. In *Proceedings of the National Conference on Advances in Manufacturing Systems, India*, pp. 117–122.

Deisenroth, M.P., & Apple, J.M. (1972). A computerized plant layout analysis and evaluation technique (PLANET).Tech. Papers 1962, Annual AIIE Conference and Commission, Norcross, GA, pp. 75–87.

Donaghey, C. E., & Pire, V. F. (1990). *Solving the facility layout problem with BLOCPLAN" Technical Report*. Houston: Industrial Engineering Department, University of Houston.

Drira, A., Pierrev, H., & Hajri-Gabouj, S. (2007). Facility layout problems: A survey. *Annual Reviews in Control, 31*(2), 255–267.

Dunker, T., Radonsb, G., & Westka¨mpera, E. (2005). Combining evolutionary computation and dynamic programming for solving a dynamic facility layout problem. *European Journal of Operational Research, 165*(1), 55–69.

Eiben, A. E., & Smith, J. E. (2007). *Introduction to evolutionary computing* (2nd ed.). Berlin: Springer.

El-Baz, M. A. (2004). A genetic algorithm for facility layout problems of different manufacturing environments. *Computers & Industrial Engineering, 47*(2–3), 33–46.

Friedrich, C., Klausnitzer, A., & Lasch, R. (2018). Integrated slicing tree approach for solving the facility layout problem with input and output locations based on contour distance. *European Journal of Operational Research, 270*(3), 837–851.

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlin (Ed.), *Foundations of genetic algorithms* (Vol. 1, pp. 69–93). San Mateo: Morgan Kaufmann.

Gomathi, V. V., Karthikeyan, S., & Sohar, O. (2014). Performance analysis of distance measures for computer tomography image segmentation. *International Journal of Computer Technology and Applications*, *5*(2), 400–405.

Gonçalves, J. F., & Resende, M. G. (2015). A biased random-key genetic algorithm for the unequal area facility layout problem. *European Journal of Operational Research, 246*(1), 86–107.

Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, 16,* 122–128.

Guan, J., & Lin, G. (2016). Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem. *European Journal of Operational Research, 248,* 899–909.

Hani, Y., Amodeo, L., Yalaoui, F., & Chen, H. (2007). Ant colony optimization for solving an industrial layout problem. *European Journal of Operational Research, 183,* 633–642.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1972). Correction to 'A formal basis for the heuristic determination of minimum cost paths'. *SIGART Newslett, 37,* 9–28.

Heragu, S. S., & Kusiak, A. (1991). Efficient models for the facility layout problem. *European Journal of Operational Research, 53*(1), 1–13.

Hernández Gress, E. S., Mora-Vargas, J., Herrera del Canto, L. E., & Diaz-Santillan, E. (2011). A genetic algorithm for optimal unequal-area block layout design. *International Journal of Production Research, 49*(8), 2183–2195.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. Ann Arbor: Michigan Press.

Hosseini-Nasab, H., Fereidouni, S., Seyyed, M. T. F. G., & Fakhrzad, M. B. (2017). Classification of facility layout problems: A review study. *The International Journal of Advanced Manufacturing Technology*. https://doi.org/10.1007/s00170-017-0895-8.

Jolai, F., Tavakkoli- Moghaddam, R., & Taghipour, M. (2012). A multi-objective particle swarm optimisation algorithm for unequal sized dynamic facility layout problem with pickup/drop-off locations. *International Journal of Production Research, 50*(15), 4279–4293.

Karagöz, S., & Yıldız, A. R. (2017). A comparison of recent metaheuristic algorithms for crashworthiness optimisation of vehicle thin-walled tubes considering sheet metal forming effects. *International Journal of Vehicle Design, 73*(1–3), 179–188.

Khalil, T. M. (1973). Facilities relative allocation technique (FRAT). *International Journal of Production Research, 11*(2), 183–194.

Kiani, M., & Yildiz, A. R. (2016). A comparative study of non-traditional methods for vehicle crashworthiness and NVH optimization. *Archives of Computational Methods in Engineering, 23*(4), 723–734.

Komarudin, K., & Wong, Y. (2010). Applying ant system for solving unequal area facility layout problems. *European Journal of Operational Research*, *202*, 730–746.

Kouvelis, P., & Kim, M. W. (1992). Unidirectional loop network layout problem in automated manufacturing systems. *Operations Research, 40,* 533–550.

Kundu, A., & Dan, P. K. (2012). Metaheuristic in facility layout problems: current trend and future direction. *International Journal of Industrial and Systems Engineering, 10*(2), 238–253.

Kusiak, A., & Heragu, S. S. (1987). The facility layout problem. *European Journal of Operational Research, 29*(3), 229–251.

Lee, R., & Moore, J. M. (1967). CORELAP-computerized relationship layout planning. *Journal of Industrial Engineering, 18,* 195–200.

Liang, L. Y., & Chao, W. C. (2008). The strategies of Tabu search technique for facility layout optimization. *Automation in Construction, 17,* 657–669.

Liu, Q., & Meller, R. D. (2007). A sequence-pair representation and MIP-model-based heuristic for the facility layout problem with rectangular departments. *IIE Transactions, 39,* 377–394.

Mazinani, M., Abedzadeh, M., & Mohebali, N. (2012). Dynamic facility layout problem based on flexible bay structure and solving by genetic algorithm. *The International Journal of Advanced Manufacturing Technology, 65*(5–8), 929–943.

Mckendall, A. R., Shang, J., & Kuppusamy, S. (2006). Simulated annealing heuristics for the dynamic facility layout problem. *Computers & Operations Research, 33*(8), 2431–2444.

Meller, R. D., & Gau, K.-Y. (1996). The facility layout problem: recent and emerging trends and perspectives. *Journal of Manufacturing Systems, 15,* 66–351.

Michalewicz, Z. (1994). *Genetic algorithms + data structures = evolution programs* (2nd ed.). Berlin: Springer.

Mitchell, M. (1998). *An introduction to genetic algorithms*. Cambridge: MIT Press.

Moslemipour, G. (2018). A hybrid CS-SA intelligent approach to solve uncertain dynamic facility layout problems considering dependency of demands. *Journal of Industrial Engineering International, 14*(2), 429–442.

Moslemipour, G., & Lee, T. (2012). Intelligent design of a dynamic machine layout in uncertain environment of flexible manufacturing systems. *Journal of Intelligent Manufacturing, 23,* 1849–1860.

Paes, F. G., Pessoa, A. A., & Vidal, T. (2017). A hybrid genetic algorithm with decomposition phases for the unequal area facility layout problem. *European Journal of Operational Research, 256*(3), 742–756.

Palomo-Romero, J. M., Salas-Morera, L., & García-Hernández, L. (2017). An island model genetic algorithm for unequal area facility layout problems. *Expert Systems with Applications, 68,* 151–162.

Palubeckis, G. (2012). A branch-and-bound algorithm for the single-row equidistant facility layout problem. *OR Spectrum, 34,* 1–21.

Park, K., Koo, J., Shin, D., Lee, C. J., & Yoon, E. S. (2011). Optimal multi-floor plant layout with consideration of safety distance based on mathematical programming and modified consequence analysis. *Korean Journal of Chemical Engineering, 28*(4), 1009–1018.

Pholdee, N., Bureerat, S., & Yıldız, A. R. (2017). Hybrid real-code population-based incremental learning and differential evolution for many-objective optimisation of an automotive floor-frame. *International Journal of Vehicle Design, 73*(1–3), 20–53.

Ponsich, A., Azzaro-Pantel, C., Domenech, S., & Pibouleau, L.(2007). Constraint handling strategies in Genetic algorithms: application to optimal batch plant design. *Chemical Engineering and Processing*, in press, https://doi.org/10.1016/j.cep.2007.01.020.

Pourvaziri, H., & Naderi, B. (2014). A hybrid multi-population genetic algorithm for the dynamic facility layout problem. *Applied Soft Computing, 24,* 457–469.

Rafia, I. (2010). A* Algorithm for Multicore Graphics Processors. M.S. thesis, Department of Computer Science and Engineering Division of Computer Engineering, Chalmers University of Technology, Gteborg, sweden.

Ramkumar, A. S., Ponnambalam, S. G., Jawahar, N., & Suresh, R. K. (2008). Iterated fast local search algorithm for solving quadratic assignment problems. *Robotics and Computer-Integrated Manufacturing, 24,* 392–401.

Razali, N.M., & Gerghty, J. (2011). In Proceedings of World Congress Engineering vol. II. London:WCE.

Ripon, K. S. H. N., Glette, K., Khan, K. N., Hovin, M., & Torresen, J. (2013). Adaptive variable neighborhood search for solving multi-objective facility layout problems with unequal area facilites. *Swarm and Evolutionary Computation, 8,* 1–12.

Sadrzadeh, A. (2012). A genetic algorithm with the heuristic procedure to solve the multi-line layout problem. *Computer and Industrial Engineering, 62*(4), 1055–1064.

Sahin, R. (2011). A simulated annealing algorithm for solving the bi-objective facility layout problem. *Journal Expert Systems with Applications, 38*(4), 4460–4465.

Sahin, R. A., & Turkbey, O. (2009). Simulated annealing algorithm to find approximate Pareto optimal solutions for the multi, objective facility layout problem. *International Journal of Manufacturing Technology and Management, 41,* 1003–1018.

Saleh, A. A. (2015). Analysis of Dijkstra's and A* Algorithm to Find the Shortest Path. (Doctoral dissertation, Universiti Tun Hussein Onn Malaysia).

Samarghandi, H., & Eshghi, K. (2010). An efficient tabu algorithm for the single row facility layout problem. *European Journal of Operational Research, 205,* 98–105.

Samarghandi, H., Taabayan, P., & Jahantigh, F. F. (2010). A particle swarm optimization for the single row facility layout problem. *Computers & Industrial Engineering, 58,* 529–534.

Saraswat, A., Venkatadri, U., & Castillo, I. (2015). A framework for multi-objective facility layout design. *Computers & Industrial Engineering, 90,* 167–176.

Sastry, K., Goldberg, D., & Kendall, G. (2005). Genetic algorithms. In E. Burke & G. Kendall (Eds.), *Search methodologies: Introductory tutorials in optimization and decision support techniques* (pp. 97–125). New York: Springer.

Seehof, J. M., & Evans, W. O. (1967). Automated layout design program. *The Journal of Industrial Engineering, 18,* 690–695.

Shayan, E., & Chittilappilly, A. (2004). Genetic algorithm for facilities layout problems based on slicing tree structure. *International Journal of Production Research, 42,* 4055–4067.

Singh, S. P., & Sharma, R. R. K. (2006). A review of different approaches to the facility layout problems. *The International Journal of Advanced Manufacturing Technology, 30,* 425–433.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to genetic algorithms*. Berlin: Springer.

Solimanpur, M., & Jafari, A. (2008). Optimal solution for the two-dimensional facility layout problem using a branch-and-bound algorithm. *Computers & Industrial Engineering, 55*(3), 606–619.

Solimanpur, M., Vrat, P., & Shankar, R. (2004). Ant Colony optimization algorithm to the inter-cell layout problem in cellular manufacturing. *European Journal of Operational Research, 157,* 592–606.

Starkweather, T., Mcdaniel, S., Whitley, D., Mathias, K., Whitley, D., & Dept, M. E. (1991). A comparison of genetic sequencing operators. In *Proceedings of the fourth International Conference on Genetic Algorithms, Morgan Kaufmann*, pp. 69–76.

Tam, K. Y. (1992). A simulated annealing algorithm for allocating space to manufacturing cells. *International Journal of Production Research, 30,* 63–87.

Tarkesh, H., Atighehchian, A., & Nookabadi, A. S. (2009). Facility layout design using virtual multi-agent system. *Journal of Intelligent Manufacturing, 20,* 347–357.

Tompkins, J. A., & Reed, R., Jr. (1976). An applied model for the facilities design problem. *International Journal of Production Research, 14*(5), 583–595.

Triki, H., Mellouli, A., Hachicha, W., & Masmoudi, F. (2016). A hybrid genetic algorithm approach for solving an extension of assembly line balancing problem. *International Journal of Computer Integrated Manufacturing, 29,* 19–504.

Van Camp, D. J., Carter, M. W., & Vannelli, A. (1991). A nonlinear optimization approach for solving facility layout problems. *European Journal of Operational Research, 57,* 174–189.

Vitayasak, S., Pongcharoen, P., & Hicks, C. (2016). A tool for solving stochastic dynamic facility layout problems with stochastic demand using either a genetic algorithm or modified backtracking search algorithm. *International Journal of Production Economics, 190,* 146–157.

Wang, M.-J., Hu, M. H., & Ku, M.-Y. (2005). A solution to the unequal area facilities layout problem by genetic algorithm. *Computers in Industry, 56,* 207–220.

Wang, S., Zuo, X., & Zhao, X. (f). (2014). Solving dynamic double-row layout problem via an improved simulated annealing algorithm. In *2014 IEEE Congress on (CEC)*, pp. 1299–1304.

Wu, X., Chu, C. H., Wang, Y., & Yue, D. (2007). Genetic algorithms for integrating cell formation with machine layout and scheduling. *Computers & Industrial Engineering, 53,* 277–289.

Xiaoning, Z., & Weina, Y. (2011). Research on layout problem of multi-layer logistics facility based on simulated annealing algorithm. In *The Fourth International Conference on Intelligent Computation Technology and Automation IEEE*, pp. 892–894.

Xie, Y., Zhou, S., Xiao, Y., Kulturel-Konak, S., & Konak, A. (2018). A β-accurate linearization method of Euclidean distance for the facility layout problem with heterogeneous distance metrics. *European Journal of Operational Research, 265,* 26–38.

Yang, W., & Tian, C. (2012). Monte-Carlo simulation of information system project performance. *Systems Engineering Procedia, 3,* 340–345.

Zhou, W., Han, B., Li, D., Zheng, B. (2013). Improved Reversely A star Path Search Algorithm based on the Comparison in Valuation of Shared Neighbor Nodes. In *Fourth International Conference on Intelligent Control and Information Processing (ICICIP) June 9–11, Beijing, China.*

Zhou, J. P. E. D., Love, K. L., Teo, H., & Luo, H. (2017). An exact penalty function method for optimising QAP formulation in facility layout problem. *International Journal of Production Research, 55*(10), 2913–2929.