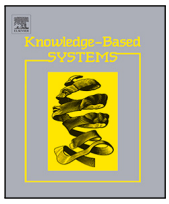
Knowledge-Based Systems 305 (2024) 112666

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



Path relinking strategies for the bi-objective double floor corridor allocation problem☆

Nicolás R. Uribe, Alberto Herrán, J. Manuel Colmenar *

Dept. of Computer Science and Statistics, Universidad Rey Juan Carlos, C/. Tulipán, s/n, Móstoles, 28933 Madrid, Spain

ARTICLE INFO

Dataset link: https://doi.org/10.5281/zenodo.13984255

Keywords:
Metaheuristics
Path relinking
Facility layout
Bi-objective

ABSTRACT

The bi-objective Double Floor Corridor Allocation Problem is an operational research problem with the goal of finding the best arrangement of facilities in a layout with two corridors located in two floors, in order to minimize the material handling costs and the corridor length. In this paper, we present a novel approach based on a combination of Path Relinking strategies. To this aim, we propose two greedy algorithms to produce an initial set of non-dominated solutions. In a first stage, we apply an Interior Path Relinking with the aim of improving this set and, in the second stage, apply an Exterior Path Relinking to reach solutions that are unreachable in the first stage. Our extensive experimental analysis shows that our method, after automatic parameter optimization, completely dominates the previous benchmarks, spending shorter computation times. In addition, we provide detailed results for the new instances, including standard metrics for multi-objective problems.

1. Introduction

Facility Layout Problems (FLP) are a family of operational research problems with the goal of finding the best arrangement of facilities in a given layout to minimize a certain objective function [1]. This family of problems has a wide range of applications, including manufacturing systems [2], delivery services [3], urban and office planning [4], and the design of file layouts for computer storage systems [5]. In addition, FLP have three main resolution approaches: exact methods, heuristic methods, and intelligence approaches [6]. In this paper, we focus on the second approach, through metaheuristics.

The initial research on FLP was conducted in [7], who focused on the Single Row Facility Layout Problem (SRFLP). Then, the Double Row Facility Layout Problem (DRFLP) was proposed in [8] which has two rows instead of one. Later, the Multiple Row Facility Layout Problem (MRFLP) was proposed in [9], where the layout has more than two rows. In addition to the number of rows, other variants consider additional features: spaces are allowed between facilities, the facilities have the same width, or the number of objectives is greater than one [10]. One of the most interesting variants of the family is the Corridor Allocation Problem (CAP), which was introduced in [11]. The CAP is a variant of the DRFLP, where there are no spaces between the facilities, and all rows are aligned to the leftmost point.

The bi-objective Corridor Allocation Problem (bCAP) has been introduced in the literature in [10]. As in CAP, this problem involves finding the optimal layout for several facilities to minimize material handling cost. In addition, there is a second objective that involves minimizing the length of the corridor. As stated in [10], these objectives are opposed. In this paper, the authors implemented a Permutation Genetic Algorithm (PGA), based on the algorithm proposed in [12]. They use a non-dominated set of solutions (*ND*) for this problem, and the crowding distance is also used to maintain a diverse set of non-dominated solutions. This paper proposes solutions for instances up to 80 facilities. The authors report the execution time to reach the best *ND* set for each instance together with the quality of the end points in each *ND*. The same authors improved their results in [13] by adding an insertion-based local search technique from [14] to their previous PGA algorithm.

The bi-objective Double Floor Corridor Allocation Problem (bDFCAP) is a bCAP variant with two different floors introduced in [15]. In this work the authors propose a Mixed Integer Linear Programming and also a Memetic Algorithm which hybridizes a Genetic Algorithm using a Variable Neighborhood Search algorithm (GAVNS). The authors compare their results with the bCAP in [13] for instances containing up to 15 facilities. Moreover, they executed their GAVNS for instances up

☆ This work has been partially supported by the Spanish Ministerio de Ciencia e Innovación (MCIN/AEI/10.13039/501100011033) and FEDER, UE, "ERDF A way of making Europe" under grant refs. PID2021-126605NB-I00 and RED2022-134480-T.

* Corresponding author.
E-mail addresses: nicolas.rodriguez@urjc.es (N.R. Uribe), alberto.herran@urjc.es (A. Herrán), josemanuel.colmenar@urjc.es (J.M. Colmenar).

https://doi.org/10.1016/j.knosys.2024.112666
Received 29 July 2024; Received in revised form 18 October 2024; Accepted 20 October 2024
Available online 30 October 2024

to size 30, reporting a set of non-dominated solutions bounded to ten solutions through the crowding distance.

In this paper, we extend a preliminary work where a multi-objective Path Relinking approach was proposed to tackle bDFCAP [16]. More precisely, we thoroughly explain the constructive method that produces an initial set of diverse non-dominated solutions and propose a different Path Relinking strategy based on two stages. In a first stage, we apply an Interior Path Relinking with the aim of improving this set, and, in a second stage, we apply an Exterior Path Relinking to reach solutions that are unreachable in the first stage. The results obtained with this proposal completely dominate the state-of-the-art solutions, spending shorter computation times. In addition, a new dataset of larger instances is provided as well as the detailed results obtained with our proposal. To our knowledge, this is the first work in which Interior and Exterior Path Relinking are combined and they are not hybridized with any other algorithm to solve a multi-objective optimization problem [17].

The following sections of this paper are outlined. Section 2 provides a problem description of the bDFCAP. Section 3 explains our proposal to solve this problem. Section 4 introduces automatic performance metrics to adjust algorithm parameters, compares the performance of our algorithm with the state of the art, and provides results for larger new instances. Finally, Section 5 presents the conclusions and potential future research.

2. Problem description

The bDFCAP considers a two-level floor: the lower and upper floors. The facilities are arranged along both sides of a corridor on each floor, ensuring that there is no space between consecutive facilities in a sequence. In addition, an elevator, located at the beginning of the sequences, enables material transfer between facilities situated on different floors. Fig. 1 illustrates the layout of the problem, showing that the rows are aligned to the leftmost point, where the elevator is placed. The purpose of the problem is to organize all the facilities within the layout to achieve minimal values for both the total Material Handling Cost (*MHC*) and the Corridor Length (*CL*). The *MHC* is calculated as the weighted aggregated sum of the distances from center to center between each pair of facilities in the layout, while *CL* is the measure of the longest row. Note that it is necessary to consider the route through the elevator when calculating the distances between the facilities on separate floors.

In a more formal manner, the problem involves a set F comprising n facilities, where $n = |F|$. Each facility $i \in F$ is associated with a length l_i, and there exists a flow cost per unit distance c_{ij} between every pair of facilities $i, j \in F$. The setting includes a layout with two floors (designated as floors 1 and 2) separated by a height h, as well as two rows on each floor (referred to as rows 1 and 2) separated by a width corridor w. The bDFCAP requires determining an allocation of facilities to floors $f : F \to \{1,2\}$ and rows $r : F \to \{1,2\}$, along with a vector $x \in \mathbb{R}^n$ representing the central positions of all the facilities in the layout measured from a fixed left origin where the elevator is located. The objective is to minimize both the total *MHC* and *CL*. Thus, a solution is given by the 3-tuple $\{f, r, x\}$ which locates every facility i in F at position x_i of floor f_i and row r_i. Table 1 summarizes the notation (sets, subscripts, parameters, and variables of the problem) used throughout the paper. Since a MILP model is provided in [15], we present an alternative mathematical formulation more suitable for our algorithmic proposal:

$$\min \ \mathcal{F}(f, r, x) = (\mathcal{F}_{MHC}, \mathcal{F}_{CL}) \tag{1a}$$

$$\text{s.t.} \ \mathcal{F}_{MHC} = \sum_{\substack{i,j \in F \\ i<j}} c_{ij} d_{ij} \tag{1b}$$

$$\mathcal{F}_{CL} = \max\{L_{11}, L_{12}, L_{21}, L_{22}\} \tag{1c}$$

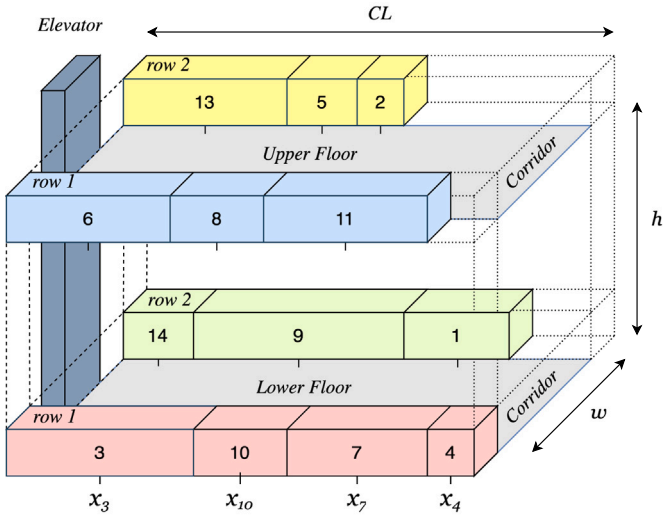


Fig. 1. Layout with two floors and one corridor per floor. Facility 3 is located in the first floor ($f_3 = 1$) first row ($r_3 = 1$). Similarly, facility 14 is located in the first floor, second row ($f_{14} = 1$, $r_{14} = 2$), facility 6 in ($f_6 = 2$, $r_6 = 1$), facility 13 in ($f_{13} = 2$, $r_{13} = 2$), and so on.

Table 1
Summary of the notation used in this paper.

Sets and subscripts	Description
F	Set of facilities
i, j	Facilities
Parameters	**Description**
w	Corridor width
h	Height between two floors
n	Number of facilities in F
l_i	Length of facility i
c_{ij}	Flow cost per unit distance between facilities i and j
Variables	**Description**
f	Mapping assigning each facility in F to a floor in $\{1,2\}$
r	Mapping assigning each facility in F to a row in $\{1,2\}$
x_i	Distance between the left origin and the center of facility i
d_{ij}	Distance between the center of facilities i and j

$$|x_i - x_j| \geq (l_i + l_j)/2 \qquad i, j \in F, \ i < j, \ r_i = r_j \tag{1d}$$

$$d_{ij} = |x_i - x_j| \qquad i, j \in F, \ i < j, \ f_i = f_j, \ r_i = r_j \tag{1e}$$

$$d_{ij} = |x_i - x_j| + w \qquad i, j \in F, \ i < j, \ f_i = f_j, \ r_i \neq r_j \tag{1f}$$

$$d_{ij} = x_i + x_j + w + h \qquad i, j \in F, \ i < j, \ f_i \neq f_j \tag{1g}$$

$$L_{ab} = \sum_{\substack{i \in F \\ f_i = a \wedge r_i = b}} l_i \qquad a, b \in \{1,2\} \tag{1h}$$

Eqs. (1b) and (1c) express the *MHC* and *CL* objectives, respectively. Eq. (1d) prevents the overlap between two neighboring facilities in the same row. Eqs. (1e) to (1g) calculate the distance between two facilities in three distinct scenarios: (1e) when they are in the same row on the same floor, (1f) when they are in different rows on the same floor and (1g) when they are on different floors. Notice how, in this last case (1g), the distance between facilities i and j considers the distance x_i from the center of facility i to the left origin (where the elevator is located), half the width of the corridor $w/2$ to reach the elevator (symmetrically located between both rows), the height h that separates both floors,

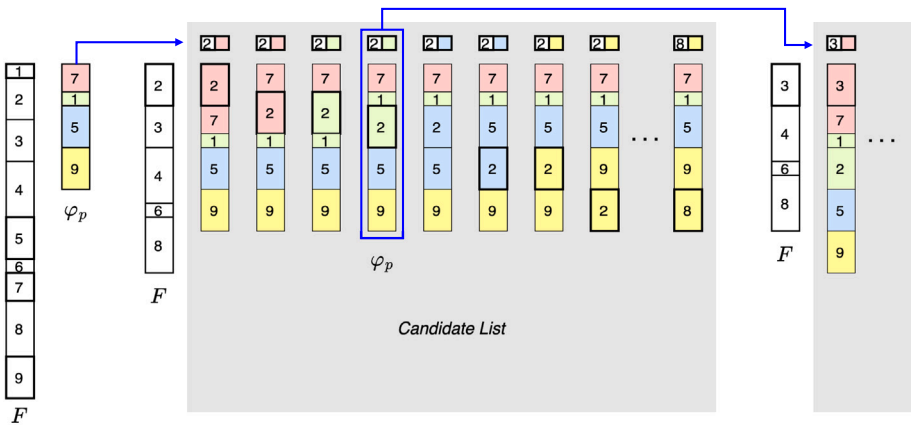


Fig. 2. Detailed selection of facilities in the constructive procedure.

other $w/2$ units to reach the row where facility j is located, and the distance x_j to reach facility j. Finally, Eq. (1h) determines the length of each row.

3. Path relinking

In this paper, we propose to address the bDFCAP problem by means of an ensemble of Path Relinking (PR) processes. Initially designed as a method to merge intensification and diversification strategies in Tabu Search [18], PR operates by creating a path between two solutions following different criteria. The aim of the method is to identify promising solutions in the generated path. In the case of multi-objective optimization, the promising solutions will be those that are not dominated by the current set of non-dominated solutions. PR has been hybridized with multiple algorithms to solve multi-objective problems [17]. However, in this work, we propose using PR as the unique algorithm in the optimization process, which is a novelty in the literature.

Before describing the proposed PR strategy, we next describe the constructive procedure that generates the initial set of non-dominated solutions.

3.1. Constructive procedure

In this work, we start by constructing a number of solutions using a greedy strategy for both, *MHC* and *CL*, objectives. Fig. 2 shows an example of this process that considers the instance S9H containing $n = 9$ facilities that can be located in 4 different rows in a layout similar to the one shown in Fig. 1, with four rows, two per floor. In addition, the background color represents the row in which each one is located. In order to favor diversification, the method starts by locating a randomly selected facility $i \in F$ at the beginning of each row and removing them from the set of available facilities F. Then, it sequentially adds a new facility to the partial solution φ_p until all the facilities are located. For this purpose, the method generates a list of candidate solutions by inserting each facility $i \in F$ into every position in the partial solution. Then, select the best solution from this candidate list in terms of the objective function used to measure the solution quality (*MHC* or *CL*). As seen in the figure, one of the solutions in the candidate list is selected as the new partial solution φ_p, removing the corresponding facility from F and generating a new candidate list. This selection process is repeated until F is empty.

This procedure is used to construct *maxCons* solutions using the *MHC* objective to make the greedy decision. Note that despite the greedy selection of facilities, the generated solutions are diverse since the first facility is chosen at random. Then, the whole constructive method is repeated to generate a different set of other *maxCons* solutions but using the *CL* objective as the greedy decision to incorporate new facilities into the partial solution at each iteration.

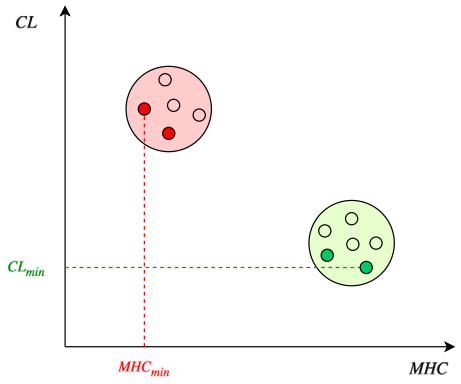


Fig. 3. Constructive procedure generates two sets of solutions that, theoretically, are situated in different areas of the objective functions space. Dark colored points represent non-dominated solutions.

As a result, this method will generate two different sets of solutions as shown in Fig. 3, where the set depicted in light red represents the *maxCons* solutions constructed using the *MHC* as greedy function, and the set depicted in light green represents the *maxCons* solutions constructed trying to minimize the *CL*. As stated in [15], the bDFCAP deals with opposing objectives, since reducing the *CL* implies to locate the facilities in different floors with the corresponding increase of the *MHC* due to a high value of h. Notice that h could include not only the vertical distance among the floors, but also the energy consumption of the elevator, increasing the actual cost associated with *MHC*.

In these scenarios, the question arises as to which of the generated solutions is better. In single-objective problems, it is elementary to determine the superiority of one solution over another. When dealing with minimization problems, the preferred solution is the one with the lowest objective function value, whereas in maximization problems, the one with the highest value is chosen. However, in multi-objective problems, the comparison involves multiple objective functions. Specifically, in this case, we are working with two functions that require minimization. A solution can dominate, be dominated by, or be non-dominated concerning another solution. To be more precise, a solution φ_1 dominates another solution φ_2 (represented as $\varphi_1 \prec \varphi_2$) if, for every objective function $\mathcal{F}_i$, φ_1 is superior or equal, and there is at least one

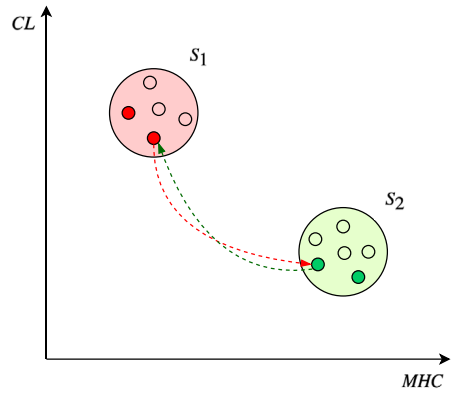


Fig. 4. Two hypothetical paths to explore by the IPR procedure between each solution $\varphi_1 \in S_1$ and $\varphi_2 \in S_2$ on both directions.

objective function where φ_1 excels. This concept is formally defined in Eq. (2).

$$\begin{aligned} &\varphi_1 \prec \varphi_2 \ \texttt{if} \\ &\forall i \in \{1,\dots,k\}, \mathcal{F}_i(\varphi_1) \leq \mathcal{F}_i(\varphi_2), \\ \wedge\ &\exists i \in \{1,\dots,k\}, \mathcal{F}_i(\varphi_1) < \mathcal{F}_i(\varphi_2). \end{aligned} \tag{2}$$

Given that our algorithm operates on multiple solutions simultaneously, it is essential to arrange them in an appropriate data structure. In this context, we will make use of a set referred to as *ND*, which is specifically designed to hold non-dominated solutions exclusively. In the example shown in Fig. 3, this set contains the solutions depicted in dark red and dark green colors.

3.2. Interior path relinking

Interior PR (IPR) creates a trajectory connecting two high-quality solutions by exploring new solutions while traversing the path, as described in [19]. The procedure consists of gradually incorporating the characteristics of a second high quality solution, referred to as the *guide* solution, into a first solution, known as the *initial* solution. Given that both solutions are of significant quality, the expected result is that exploring the path connecting both solutions will lead to uncovering novel and valuable regions of the search space.

Let us consider the initial set of solutions shown in Fig. 4, where S_1 (S_2) is the set of solutions constructed trying to minimize *MHC* (*CL*). Our IPR proposal generates a path between each solution $\varphi_1 \in S_1$ and each solution $\varphi_2 \in S_2$ in both directions (that is, using φ_1 as the *initial* solution and φ_2 as the *guide* one, and vice versa).

Since the aim of IPR is to perform moves that transform the *initial* solution into the *guide* one, and the given solutions may present different number of facilities on each row, we have divided the IPR process into two phases. The first phase will gradually transform the *initial* solution into an *intermediate* solution with the same number of facilities per row than the *guide* solution by using *insert* moves. The second phase will gradually transform the *intermediate* solution into the *guide* one by means of *exchange* moves.

Fig. 5 shows an example of the path followed by the first phase of our IPR proposal to reach an *intermediate* solution from the *initial* solution shown on the left side of the figure. As in Fig. 2, this example considers the instance S9H containing $n = 9$ facilities that can be located in two floors with 2 rows each. To simplify our discussion, we refer to the *initial* solution as φ and the *guide* solution as χ.

At this point, let us define a vector N^{φ} with the number of facilities in each row of φ, hence, in this example, $N^{\varphi} = \{4,3,1,1\}$ and $N^{\chi} = \{2,2,3,2\}$. Then, to match the number of facilities in each row of solutions φ and χ, $|N^{\varphi}_u| = |N^{\chi}_v|$ for all $u, v \in \{1,2,3,4\}$, all the candidate *insert* moves in each iteration are defined as those insertions of facilities from a row u with $N^{\varphi}_u > N^{\chi}_u$ to a row v with $N^{\varphi}_v < N^{\chi}_v$. In the example of Fig. 5, the facilities 6, 2, and 4 in row 1 of φ (not present in row 1 of χ) are candidate facilities to be inserted in rows 3 and 4, since $N^{\varphi}_3 < N^{\chi}_3$ and $N^{\varphi}_4 < N^{\chi}_4$, respectively. Specifically, observing the position of facilities 6, 2 and 4 in the *guide* solution χ, facility 6 can be inserted in row 3, and facilities 2 and 4 can be inserted in row 4. Moreover, we intend to benefit from placing a facility in the identical location as it appears in the *guide* solution (facility 6 is inserted at the beginning of row 3, and facilities 2 and 4 at the end or beginning of row 4, respectively). Similarly, facilities 1, 3 and 8 in row 2 of φ (not present in row 2 of χ) can be removed from row 2 but, in this case, only facilities 1 and 8 can be inserted in row 3, since the insertion of facility 3 in row 1 would lead to an increase on its size, when our objective in this first IPR phase is to reduce it to reach a layout with $|N^{\varphi}_1| = |N^{\chi}_1| = 2$.

Hence, starting from the *initial* solution, the first block of solutions with a gray background in Fig. 5 represents the five possible *insert* moves that can be applied to approach the *guide* solution during the first phase of our IPR proposal: insert facility 6 in row 3, insert facility 2 in row 4, insert facility 4 in row 4, insert facility 1 in row 3, and insert facility 8 in row 3. Note the move label on top of each solution.

Once all possible *insert* moves are identified at each iteration, our IPR proposal selects one of them to generate the next solution on the path to the *intermediate* solution. For this purpose, the procedure selects one of these moves using a greedy function, following a Greedy Randomized Adaptive Search Procedure (GRASP) methodology [20]. In this case, the greedy function of a candidate move $g(move)$ is the objective function of the resulting solution after the *insert* move, and the selected move is randomly chosen from a restricted candidate list built including all candidate moves with $g(move) \leq g_{max} - \alpha(g_{max} - g_{min})$, where α controls the balance between a purely random ($\alpha = 0$) or a purely greedy ($\alpha = 1$) selection. Note that the selected solution is surrounded by a blue line in Fig. 5. The first phase of the IPR procedure will continue until reaching an *intermediate* solution where all the rows have the same number of facilities as the *guide* solution.

Fig. 6 shows a bidimensional representation of the quality of the solution of the five different solutions generated by the *insert* move applied over the *initial* solution in the example depicted in Fig. 4. In this example, the solution represented by a circle with red (green) border would be the selected solution to continue the path if we use the *MHC* (*CL*) objective as a greedy function with a purely greedy selection. As will be shown later, we use both objectives alternatively.

The second phase of the IPR procedure iteratively applies *exchange* moves (interchanging the positions of two facilities i and j in the layout) from the *intermediate* solution to match the *guide* solution. Therefore, it analyzes all the facilities in φ to check if they are located at the same position in χ, otherwise an interchange is needed to match this facility in both solutions. Fig. 7 shows the second phase after the example in Fig. 5. As seen, only facilities 7, 6, 8 and 4 in the *intermediate* solution match their position in the *guide* solution in the first iteration of this phase. Hence, to reach the *guide* solution, we need to exchange the position of the other five facilities (2, 1, 3, 5 and 9) in the *intermediate* with their corresponding position in the *guide* solution. Fig. 7 shows in the first gray background block the five possible *exchange* moves together with its resulting solution in the first iteration of this second IPR phase. The procedure then selects one of these moves using the GRASP strategy defined before, and this process (identifying all possible *exchange* moves and selecting one) is repeated until it reaches the *guide* solution.

Algorithm 1 summarizes the pseudo-code of the `InteriorPR` procedure proposed in this study whose behavior has been described. The algorithm takes four input parameters, the *initial* solution φ, the *guide* solution χ, a parameter α that balances the GRASP selection of the next

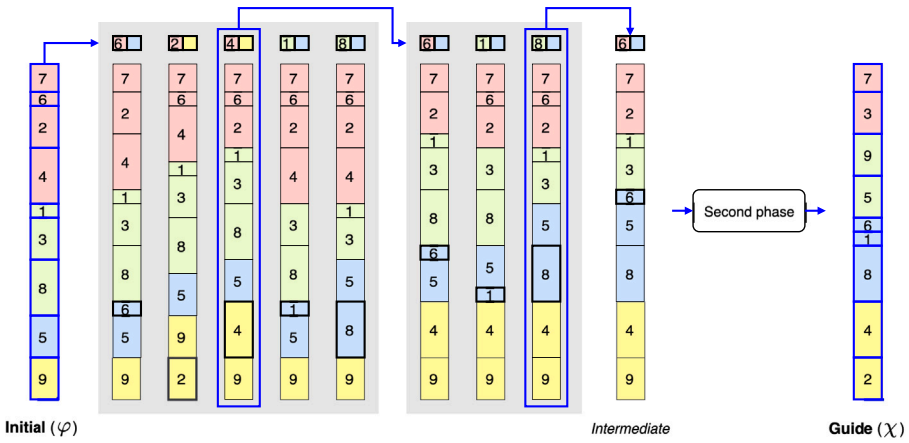


Fig. 5. First phase of the IPR procedure: Several *insert* moves are applied to match the number of facilities in each row of the *guide* solution.

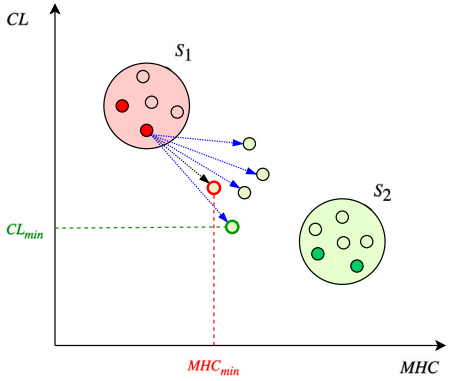


Fig. 6. Selected solution according to the greedy function used in each case: Circle with red solid line for *MHC*; and circle with green solid line for *CL*.

solution on the path, and the objective function *obj* (*MHC* or *CL*) used in this GRASP selection. The algorithm consists of a loop that iterates until the current solution becomes the *guide* solution. After applying the *insert* (step 3) or *exchange* (step 5) moves depending on whether we are in the first or second phase of the IPR procedure, respectively, line 6 selects (from S) the next solution in the path using a GRASP-based criterion according to the parameters *obj* and α. Then, step 7 updates the set of non-dominated solutions *ND* with the set of solutions S obtained in step 3 or 5. This function will assess whether any solution $\varphi \in S$ is dominated by any existing member of the set. If φ is not dominated, the function will evaluate all current solutions in the *ND* set, disregarding those dominated by φ. The loop ends when $\varphi = \chi$, returning the set of non-dominated solutions in step 8.

Algorithm 1: INTERIORPR $(\varphi, \chi, \alpha, obj)$

```
1 while φ ≠ χ do
2     if ∃u : |N_u^φ| ≠ |N_u^χ| then
3         S ← InsertMoves(φ)
4     else
5         S ← ExchangeMoves(φ)
6     φ ← NextSolution(S, α, obj)
7     ND ← Update(ND, S)
8 return ND
```

3.3. Exterior path relinking

Exterior PR (EPR) is a PR variant that has been successfully used in different works [21–23]. However, EPR was never applied to a problem of the FLP family, which is one the main contributions of this paper.

EPR involves creating a path starting from the *guide* solution with the aim of moving beyond both the *initial* and the *guide* solutions. While in the preceding section the search space was confined to the region between the *initial* and *guide* solutions, in this approach the exploration extends beyond these boundaries. Therefore, two decisions have to be made. On the one hand, how to measure the distance between the current solution and both the *initial* and the *guide* ones. On the other hand, which criteria will determine the stop of the process. Regarding the distance, we have opted for the Kendall–Tau distance [24], which measures how many inversions exist between two permutations, considering that an inversion is a pair of elements that are in a different order in one permutation with respect to the other, as it is described in [25]. Finally, we have established a number of maximum iterations as a parameter of the procedure.

Algorithm 2 shows the pseudo-code of the `ExteriorPR` procedure proposed in this work. The algorithm receives four input parameters, the *initial* solution φ, the *guide* solution χ, a parameter β that controls the balance between a random or greedy selection of the next solution on the path, and the number of iterations of the algorithm *maxIters*. The algorithm starts by initializing the set of non-dominated solutions ND to an empty set in step 1, and the current solution in the path χ' to χ. Then it uses the Kendall–Tau distance in step 3 to calculate the distance D_φ from the current solution χ' to φ. Notice that we set $D_\chi = 0$ in step 4 since $\chi' = \chi$ in the first iteration. The algorithm then iterates *maxIters* times through steps 5 to 15 trying to explore different regions of the search space than the one explored by the IPR procedure. In each iteration, the loop starts obtaining the solutions from the generated neighborhoods applying *insert* (N_{ins}) and *exchange* (N_{exc}) moves to the current solution χ' in steps 6 and 7, respectively, and updating the *ND* set with the solutions in these neighborhoods ($N_{ins} \cup N_{exc}$) in step 8. Then, in step 9 it gets all the solutions in *ND* with a Kendall–Tau distance greater than D_φ and D_χ, storing the result in set S in order to get the next solution for the path. If there are solutions in S, the algorithm gets the new solution on the path χ' from S through a GRASP-based selection controlled by the parameter β and the greedy function $\mathcal{F}_{MHC}$ and updates the Kendall–Tau distances D_φ and D_χ keeping iterating (steps 11 to 13). In this selection process $\mathcal{F}_{CL}$ was also tested in a preliminary experimentation, but the results were very unsuccessful. If S is empty, the iteration process is stopped before reaching the maximum number of iterations. Finally, the algorithm ends up returning *ND* in step 16.

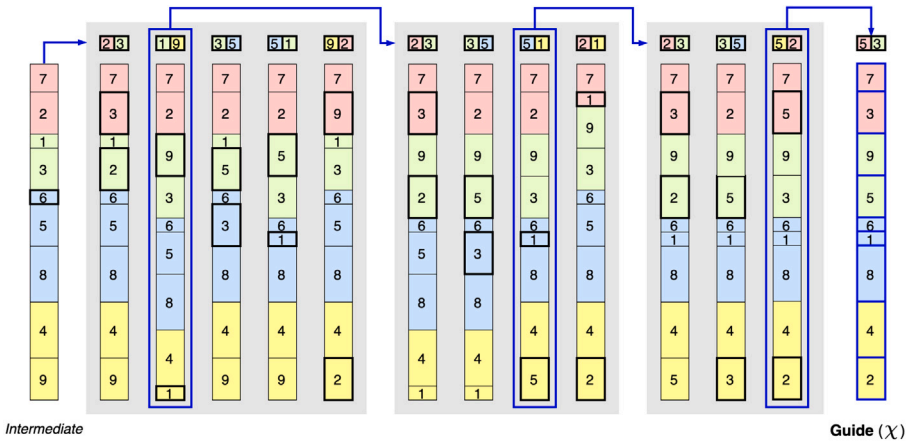


Fig. 7. Second phase of the IPR procedure: Several *exchange* moves are applied to match the *guide* solution.

Algorithm 2: EXTERIOR PR ($\varphi, \chi, \beta, maxIters$)

1 $ND \leftarrow \emptyset$
2 $\chi' \leftarrow \chi$
3 $D_\varphi \leftarrow$ `KendallTauDistance`(φ, χ')
4 $D_\chi \leftarrow 0$
5 **for** $i = 1$ **to** $maxIters$ **do**
6 $\quad N_{ins} \leftarrow$ `InsertNeighborhood`(χ')
7 $\quad N_{exc} \leftarrow$ `ExchangeNeighborhood`(χ')
8 $\quad ND \leftarrow$ `Update`$(ND, N_{ins} \cup N_{exc})$
9 $\quad S \leftarrow$ `FilteredSolutions`(ND, D_φ, D_χ)
10 $\quad$ **if** $S \neq \emptyset$ **then**
11 $\quad\quad \chi' \leftarrow$ `NextSolution`$(S, \beta, \mathcal{F}_{MHC})$
12 $\quad\quad D_\varphi \leftarrow$ `KendallTauDistance`(φ, χ')
13 $\quad\quad D_\chi \leftarrow$ `KendallTauDistance`(χ, χ')
14 $\quad$ **else**
15 $\quad\quad$ **break**
16 **return** ND

Fig. 8 graphically shows the behavior of the EPR procedure. The left part of the figure shows solutions φ, χ and χ', denoting with a red area the theoretical forbidden space of solutions, since the distance with both φ and χ will decrease in that area. The green areas denoted as N_{ins} and N_{exc} represent the theoretical space of solutions generated by both *insert* and *exchange* moves, and may overlap with the red area. The green dot in the figure shows a candidate χ' solution that is separated from both solutions φ and χ. Once the solution is updated, the new forbidden area is formed by the distances from χ' to φ and from χ' to χ, as shown in the right part of the figure.

3.4. Final algorithmic proposal

Once the constructive procedure and the PR methods have been described, we now show how they are combined to form the final algorithm proposed in this work, which is purely based on PR. We name this proposal Combined Bi-objective Path Relinking (CBPR).

Algorithm 3 depicts the pseudo-code of the CBPR. The algorithm takes four input parameters: the number of iterations for the greedy construction phase *maxCons*; the number of iterations to run the `ExteriorPR` procedure, *maxIters*; and the parameters α and β controlling the balance of the random/greedy selection of the next solution in the path for the `InteriorPR` and `ExteriorPR` procedures, respectively.

The algorithm starts in step 1 initializing the set of non-dominated solutions *ND* to an empty set. Then, steps 2 and 3 generate *maxCons* solutions each, according to the procedure explained in Section 3.1, storing these initial solutions in sets S_1 and S_2, respectively, using both objective functions. Next, steps 4 to 10 connect each solution in S_1 to each of the solutions in S_2 (and the reverse path) using the `InteriorPR` procedure (see Section 3.2) for both objective functions, updating the *ND* set with all the solutions found along these paths in step 10. Notice how $\alpha = 1$ during this phase, leading to the selection of the best possible solution (according to the corresponding objective) in each iteration (pure *greedy* behavior). At this point, an initial set of non-dominated solutions *ND* is created. The algorithm enters now into a loop (steps 12 to 25) trying to improve the quality of *ND*. For this purpose, step 22 updates this set with all the solutions found in the paths generated by the `InteriorPR` and `ExteriorPR` procedures applied to every pair of solutions in *ND*, and the process continues while any of the solutions generated in these paths modifies *ND*. Notice how, unlike in the first phase of the algorithm (steps 6 to 9), the `InteriorPR` (`ExteriorPR`) procedure does not state $\alpha = 1$ ($\beta = 1$), since it would lead the algorithm to explore the same paths for those pairs of solutions in the *ND* set which have not changed from one iteration to another. If the set of non-dominated solutions does not change, the final loop ends. Otherwise, the set is updated and the loop continues (steps 23 to 25). Finally, the algorithm returns the set *ND* of non-dominated solutions in step 26.

As stated before, the CBPR algorithm does not match with any previous algorithm from the literature, since PR is usually a complement to other algorithms like local search [23], GRASP [22], Scatter Search [18] or Tabu Search [26]. However, as we will show in the experimental results, it is able to obtain quality solutions in competitive execution times.

4. Computational results

This section presents the experimental findings of our proposed method. In particular, while previous research examined only 22 instances [15], we have expanded the data set to include 10 additional and larger instances, resulting in a total of 32 instances analyzed. We believe these instances will be valuable for future scientific investigations.

We implemented our algorithms using Java 17 and conducted the experiments on a Windows 11 laptop equipped with an Intel i7 1065G7 processor operating at 1.3 GHz, along with 16 GB of RAM. All the code, instances, and detailed results are available in (*URL available upon acceptance*).

4.1. Multi-objective metrics

In order to evaluate the quality of solutions in multi-objective optimization problems, various researchers suggest that the results of

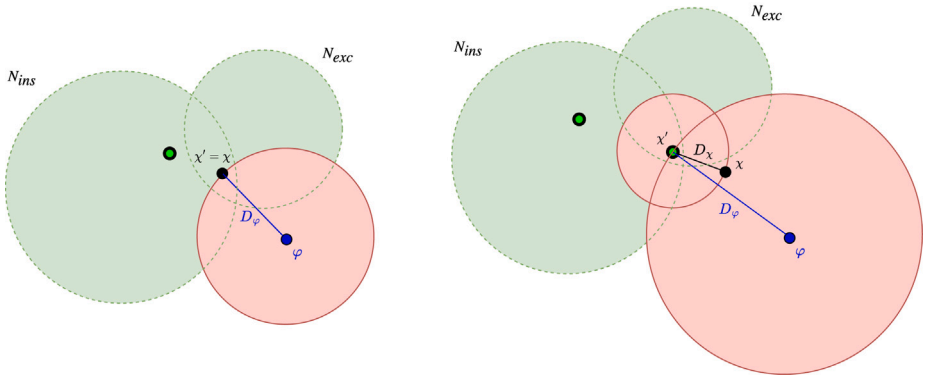


Fig. 8. EPR procedure: First iteration (left); iteration i (right).

Algorithm 3: CBPR($maxCons, maxIters, \alpha, \beta$)

$ND \leftarrow \emptyset$
$S_1 \leftarrow$ Greedy($maxCons, \mathcal{F}_{MHC}$)
$S_2 \leftarrow$ Greedy($maxCons, \mathcal{F}_{CL}$)
for $\varphi_1 \in S_1$ **do**
  for $\varphi_2 \in S_2$ **do**
    $P_1 \leftarrow$ InteriorPR($\varphi_1, \varphi_2, 1, \mathcal{F}_{MHC}$)
    $P_2 \leftarrow$ InteriorPR($\varphi_1, \varphi_2, 1, \mathcal{F}_{CL}$)
    $P_3 \leftarrow$ InteriorPR($\varphi_2, \varphi_1, 1, \mathcal{F}_{MHC}$)
    $P_4 \leftarrow$ InteriorPR($\varphi_2, \varphi_1, 1, \mathcal{F}_{CL}$)
    $ND \leftarrow$ Update($ND, P_1 \cup P_2 \cup P_3 \cup P_4$)
improve $\leftarrow$ true
while *improve* **do**
  improve $\leftarrow$ false
  for $i = 1$ **to** $|ND| - 1$ **do**
    for $j = i + 1$ **to** $|ND|$ **do**
      $P_1 \leftarrow$ InteriorPR($ND_i, ND_j, \alpha, \mathcal{F}_{MHC}$)
      $P_2 \leftarrow$ InteriorPR($ND_i, ND_j, \alpha, \mathcal{F}_{CL}$)
      $P_3 \leftarrow$ InteriorPR($ND_j, ND_i, \alpha, \mathcal{F}_{MHC}$)
      $P_4 \leftarrow$ InteriorPR($ND_j, ND_i, \alpha, \mathcal{F}_{CL}$)
      $P_5 \leftarrow$ ExteriorPR($ND_i, ND_j, \beta, maxIters$)
      $P_6 \leftarrow$ ExteriorPR($ND_j, ND_i, \beta, maxIters$)
      $ND' \leftarrow$ Update($ND, P_1 \cup P_2 \cup P_3 \cup P_4 \cup P_5 \cup P_6$)
  if $ND \neq ND'$ **then**
    improve $\leftarrow$ true
    $ND \leftarrow ND'$
return ND

different algorithms should be compared using suitable Quality Indicators (QIs) rather than evaluating individual objectives separately, as this could lead to misleading conclusions [27,28].

QIs encompass four Quality Aspects (QAs): convergence, spread, uniformity, and cardinality. Convergence indicates how close to the Pareto front are the solutions found. An effective multi-objective optimization algorithm should generate solutions that are very close to the Pareto front. Spread assesses how the solutions are distributed along the Pareto front. A good distribution implies that the solutions are evenly spread and cover the entire space of potential solutions. Uniformity indicates the consistency in the spacing between consecutive solutions on the Pareto front. High uniformity ensures that the solutions are equally spaced, offering a uniform representation of the Pareto front. Cardinality quantifies the number of solutions in the Pareto front. In multi-objective optimization, a greater cardinality is advantageous as it offers the decision-maker a broader spectrum of options.

Table 2
Quality indicators and their quality aspects. Table adapted from [28].

Quality aspect	HV	IGD^+	ϵ	Size
Convergence	+	+	+	
Spread	+	+	+	
Uniformity	+	−	+	
Cardinality	−	−	−	+

Table 3
Instances in the state-of-the-art. The ones highlighted in bold have been used as benchmark instances.

Instance	n	Instance	n	Instance	n	Instance	n
S9	9	**Am13a**	13	N25_01	25	N30_01	30
S9H	9	Am13b	13	N25_02	25	**N30_02**	30
S10	10	**Am15**	15	N25_03	25	N30_03	30
S11	11	Am17	17	**N25_04**	25	N30_04	30
Am12a	12	Am18	18	**N25_05**	25	N30_05	30
Am12b	12	H20	20				

Since selecting the appropriate QIs is not a trivial task, we have followed the suggestions in [27]. Our goal is to cover all the QAs mentioned above. For this purpose, we have chosen the following QIs: hypervolume (HV), inverted generational distance (IGD^+), ϵ and *Size* [29]. The HV, IGD^+ and ϵ indicators evaluate the convergence, spread, and uniformity of solutions, although the IGD^+ indicator only partially covers uniformity. The *Size* indicator evaluates the cardinality. For the HV and *Size* metrics, the higher its value, the better the front. For IGD^+ and ϵ, just the other way around.

We have summarized the QIs and theirs QAs in Table 2. This table presents quality indicators alongside different quality aspects. A "+" symbol means that the quality aspect is fully addressed by a QI, while a "−" symbol indicates that it is only partially addressed, as indicated in [28,30].

Finally, in this work, we have implemented the referred metrics using the `jMetal` framework [31]. The calculation of these metrics is also described in [23,32].

4.2. Instances from the state-of-the-art

Since we are comparing the performance of our algorithmic proposal with the state of the art for this problem [15], we conduct experiments on the 22 proposed instances, whose name and size are shown in Table 3.

To identify the best configuration and parameter settings for our proposal, we employed an automatic configuration tool on a representative subset of instances, referred to as benchmark instances. In this context, we selected 7 representative instances (30%) following the method described in [33], using a 90% PCA ratio and the recommended characteristics for the matrix weight. The chosen benchmark instances are indicated in bold in Table 3.

Table 4
Configurations returned by `irace`.

Config.	α	β
#1	0.9	0.8
#2	0.5	0.6
#3	0.9	0.9

4.3. Parameter setting

After selecting the benchmark instances, the parameter values for the proposed algorithm were automatically obtained. For this purpose, we used `irace`, a tool based on the iterated F-race method, which determines the best values of the parameters according to this statistical procedure as described in [34,35]. For each parameter setup, the execution quality will be evaluated using the *HV*, which is the unique multi-objective metric that does not require a reference set of solutions [27].

As detailed in Algorithm 3, our method has four input parameters: $maxCons$, $maxIter$, α and β. The first two parameters are adjusted by us rather than `irace`, because `irace` tends to maximize them, thus increasing the execution time by increasing the number of algorithm iterations. Therefore, to constrain execution time, we assign $maxCons = 5 \times n$, and $maxIter = 0.3 \times n$, where n is the size of the instance.

Hence, the parameters to be set by `irace` are α and β. We have set both parameters as categorical types whose categories are defined every 0.1 in the range [0.0, 0.9]. Moreover, in order to have a representative number of experiments, we have set `irace` to run 10,000 experiments.

Table 4 presents the parameter configurations generated by `irace`. Each row in the table represents a specific configuration of parameter values produced by `irace`. These configurations are ordered from highest to lowest performance according to `irace`. It is evident that the α values exhibit a relatively wider range, while the β values show a narrower range. This suggests that tuning α is less critical compared to the adjustment of β. It is also evident that both parameters must have a high value.

Since the parameter values are so close, we will study the behavior of the three configurations obtained for our CBPR proposal.

4.4. Comparison with state of the art

In this section, we examine the sets of non-dominated solutions generated by the three configurations of our proposal, as presented in the previous section, and contrast them with the outcomes from [15]. Despite the previous work does not present results of multi-objective metrics, we have calculated them for the solutions the authors provide. Since we need a Pareto front for the multi-objective metrics but for the *HV* [27], and no Pareto fronts are available for the instances, we generated a reference set for the multi-objective metrics. This set is generated for each instance, and for each pair of our proposal and the non-dominated solutions from [15]. Consequently, we can employ the multi-objective metrics outlined in Section 4.1.

Table 5 presents the results that compare our proposal with the state-of-the-art method. The first column, labeled as *Algorithm*, specifies the algorithm used. Our proposal is labeled CBPR followed by the configuration number, and the state-of-the-art method is labeled GAVNS, as referenced in [15]. The second to the fifth columns display the metrics used for comparison (see Section 4.1). For these metrics, except for *HV*, a reference set is required [27]. As a reference set is not available for the current instances, we created one using a non-dominated set of solutions. More precisely, for each instance, we obtained the set of non-dominated solutions from those obtained by the executions of our algorithm and with those reported in [15]. Thus, this reference set only includes non-dominated solutions. The final column T (s) indicates the execution time in seconds for the algorithms. All metrics, except IGD^+, use the average value of all instances, whereas IGD^+ employs the geometric mean due to the wide range of values. In order to skip the stochastic bias, we have run each CBPR configuration 10 times for each instance. The bold figures in the table highlight the values that improve the GAVNS performance.

Table 5
Overview for comparative analysis against the state-of-the-art.

Algorithm	HV	IGD^+	ϵ	Size	T (s)
CBPR #1	**0.47**	7126.12	**0.04**	**16.32**	**419.23**
CBPR #2	**0.47**	7125.31	**0.04**	**16.11**	**454.76**
CBPR #3	**0.47**	**7120.07**	**0.04**	**16.76**	**449.60**
GAVNS	0.45	7121.43	0.07	8.86	1987.52

Table 6
Proposed instances for future research.

Instance	n	Instance	n
ste36_01	36	N40_01	40
ste36_02	36	N40_02	40
ste36_03	36	N40_03	40
ste36_04	36	N40_04	40
ste36_05	36	N40_05	40

As shown in the table, our algorithm achieves better metrics in all configurations, except for IGD^+ in the first and second configurations. It is notable that all three configurations yield similar values across each metric, a trend that continues in the following experiment. The metric values are detailed instance by instance in Appendix for the third configuration compared to the GAVNS method. Notice that our metaheuristic approach does not verify optimally. However, the authors in [15] presented a model that was able to obtain the ends of the sets of non-dominated solutions for each instance up to size 15 using a MILP model. Given that we obtained a same set of non-dominated solutions for instances S9 and S10 (see Table A.8), we may assume that they are optimal fronts. However, this assumption cannot be made for any other instance.

It is important to note that our algorithm spends less execution time, consuming only 23% of the time compared to the previous algorithm. Given that the previous work was executed in a Windows 10 environment on a desktop PC with an AMD Ryzen 5 2600 Processor at 3.40 GHz and 8 GB RAM, it is apparent that the gains in execution speed are due to the algorithm itself rather than the hardware, since the state-of-the-art setup is comparable. Therefore, there is room for running our proposal in larger instances.

4.5. Proposal for the new instances

Since the largest instance of state-of-the-art has size 30 and our algorithm shown efficient performance, we decide to add 10 new instances with a larger size. These instances are described in Table 6, where the columns *Instance* represent the name of the instance, and the columns *Size* their size. In this way, now the larger instance has 40 facilities.

The discussion of results for the new instance set will cover again the multi-objective metrics described in Section 4.1. Due to the unavailability of the algorithms from [15], this experiment studies the results of the three configurations of our algorithm, along with their comparison. In light of the extended execution time, compared to the former instances, a time limit of 9500 s has been set. This time limit is the longest execution time from the previous research.

Table 7 presents the results for our three configurations using the instances mentioned earlier. This table follows the same structure as Table 5. Due to the lack of access to the previous code, we have compared our three CBPR configurations between them. Due to the absence of the reference set, we applied the same methodology as previously described in Section 4.4. The table illustrates that the first configuration performs slightly better than the other two in *HV*, IGD^+ and ϵ, while

Table 7
Overview for comparative analysis between our three configurations of CBPR.

Algorithm	HV	IGD$^+$	ϵ	Size	T (s)
CBPR #1	**0.40**	**43 590.35**	**0.29**	19.80	6269.24
CBPR #2	0.37	43 557.14	0.41	**20.17**	**6232.61**
CBPR #3	0.38	43 586.74	0.48	19.83	6292.73

the second configuration is better for *Size* and T (s). Similarly, the results for all three configurations are consistent with those in Table 5. Therefore, we can infer that any of these configurations is suitable for the target problem, proving the stability of our algorithm. Once again, the metrics are detailed instance by instance in Appendix.

4.6. Practical applications

The two objectives confronted in this problem, *MHC* and *CL* are able to represent different situations in real-world scenarios. One of the most relevant applications is the design of buildings with two floors and a corridor on each floor, which is very common in services to society such as hospitals, education centers, or public libraries, among others. Having an algorithm able to generate a variety of non-dominated scenarios allows the decision makers to consider different good alternatives for the final design of the building. As an example, the length of the corridor determines the design (and cost) of the heating, ventilation, and air conditioning systems, which could have a huge impact on the energy efficiency of the building. Depending on environmental requirements, budget, or additional considerations, using the proposed algorithm, the decision maker has the possibility to examine different good designs. In addition, given the short computation time of the proposed algorithm, both changes in the layout and in the facilities can also be quickly assessed.

5. Conclusions and future work

The bi-objective Double Floor Corridor Allocation Problem (bDFCAP), an important problem of the FLP family, addresses two distinct objectives, making it applicable to numerous real-world scenarios.

In our study, we approach the bDFCAP with a multi-objective perspective, focusing on both material handling costs and corridor length. For this purpose, we developed a novel approach based on a combination of Path Relinking procedures that we call Combined Bi-objective Path Relinking (CBPR). In particular, an initial Interior Path Relinking phase generates an initial set of non-dominated solutions. These solutions are further refined using a combination of Interior Path Relinking and Exterior Path Relinking strategies. The latter one employs dominance and moves away from initial and guide solutions by utilizing the Kendall–Tau distance.

Previously, this problem had a limited number of instances, which we have increased from 22 to 32. Additionally, the largest instance now contains 40 facilities, up from 30, creating a new benchmark dataset for the research community.

Our extensive experimental analysis shows that our method, after automatic parameter optimization, outperforms the previous benchmarks spending less than a fourth of the computation time on equivalent hardware. We also provide detailed results for the new instances, including standard metrics for multi-objective problems.

Despite the short execution times of our proposal in relation to the state of the art, we have reached the time limit of 3600 in the largest instances, formed by 40 facilities. Therefore, in order to tackle even larger instances, an effort should be made to reduce the execution time, mainly due to the large neighborhood traversed by External Path Relinking.

Future research will explore different FLP topologies, such as T-row or bay-row layouts. In addition, we will extend the CBPR to other multi-objective problems.

Table A.8
Comparison among the state-of-the-art and the third configuration of our proposal, for the set of previous instances with size [9, 20]. Best values are highlighted with bold.

Instance	Algorithm	HV	IGD$^+$	ϵ	Size	T (s)
S9	CBPR #3	**0.24**	**1924.93**	**0.00**	**5.00**	**0.79**
	GAVNS	**0.24**	**1924.93**	**0.00**	**5.00**	13.80
S9H	CBPR #3	**0.30**	**3994.95**	**0.00**	7.50	**0.82**
	GAVNS	**0.30**	**3994.95**	**0.00**	**8.00**	13.91
S10	CBPR #3	**0.74**	**2139.34**	**0.00**	**5.00**	**1.32**
	GAVNS	**0.74**	**2139.34**	**0.00**	**5.00**	29.00
S11	CBPR #3	**0.58**	5207.71	**0.00**	**11.75**	**2.46**
	GAVNS	0.57	**5202.21**	0.08	10.00	40.92
Am12a	CBPR #3	**0.64**	**2209.87**	0.08	**8.00**	**3.63**
	GAVNS	**0.64**	**2209.87**	**0.00**	**8.00**	69.68
Am12b	CBPR #3	0.57	2471.72	0.06	6.75	**3.58**
	GAVNS	**0.58**	**2464.23**	**0.00**	**8.00**	63.41
Am13a	CBPR #3	0.47	3463.74	0.07	8.25	**5.41**
	GAVNS	**0.49**	**3456.25**	**0.00**	**9.00**	87.16
Am13b	CBPR #3	**0.52**	3785.35	0.05	**13.50**	**6.29**
	GAVNS	**0.52**	**3783.85**	**0.04**	10.00	101.73
Am15	CBPR #3	**0.63**	**4239.55**	**0.02**	**12.00**	**13.46**
	GAVNS	0.57	4424.69	0.19	9.00	159.93
Am17	CBPR #3	**0.49**	**5903.68**	**0.05**	**13.50**	**26.20**
	GAVNS	**0.49**	**5903.68**	0.07	10.00	970.28
Am18	CBPR #3	**0.57**	6709.01	**0.03**	**16.50**	**37.06**
	GAVNS	0.55	**6706.27**	0.09	10.00	1397.50
H20	CBPR #3	**0.51**	9514.64	**0.07**	**19.00**	**71.66**
	GAVNS	0.48	**9426.66**	0.08	10.00	1926.72

CRediT authorship contribution statement

Nicolás R. Uribe: Writing – review & editing, Writing – original draft, Software, Conceptualization. **Alberto Herrán:** Writing – review & editing, Writing – original draft, Methodology, Conceptualization. **J. Manuel Colmenar:** Writing – review & editing, Writing – original draft, Methodology, Conceptualization.

Declaration of competing interest

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

Acknowledgments

This work has been partially supported by the Spanish Ministerio de Ciencia e Innovación (MCIN/AEI/10.13039/501100011033) and FEDER, UE, "ERDF A way of making Europe" under grant refs. PID2021-126605NB-I00 and RED2022-134480-T.

Appendix. Detailed results

Tables A.8 and A.9 show extensive results for the state-of-the-art instance set using the multi-objective metrics described in Section 4.1. Similarly, Table A.10 presents the detailed results of the three configurations of our proposal in the set of proposed instances, also using the same metrics. In this table, TL stands for Time Limit (9500 s). To ensure brevity, the tables display numbers in two decimal places.

Data availability

Instances, detailed results and code are available in https://doi.org/10.5281/zenodo.13984255.

Table A.9
Comparison among the state-of-the-art and the third configuration of our proposal, for the set of previous instances with size 25 and 30. Best values are highlighted with bold.

Instance	Algorithm	HV	IGD$^+$	ϵ	Size	T (s)
N25_01	CBPR #3	0.35	3338.99	0.02	**6.00**	**209.38**
	GAVNS	**0.36**	**3326.49**	**0.00**	**6.00**	407.75
N25_02	CBPR #3	**0.46**	19 750.74	**0.02**	**36.50**	**469.33**
	GAVNS	0.43	**19 729.23**	0.07	10.00	3687.02
N25_03	CBPR #3	**0.39**	**13 240.60**	**0.03**	**25.50**	**319.07**
	GAVNS	0.33	**13 240.60**	0.11	10.00	3360.56
N25_04	CBPR #3	**0.48**	25 387.38	**0.02**	**43.00**	**505.47**
	GAVNS	0.44	**25 356.38**	0.10	10.00	4461.00
N25_05	CBPR #3	**0.45**	8958.36	**0.03**	**14.75**	**221.90**
	GAVNS	0.41	**8937.61**	0.13	10.00	2099.13
N30_01	CBPR #3	0.30	5533.92	0.02	**7.00**	**554.47**
	GAVNS	**0.31**	**5513.42**	**0.00**	**7.00**	624.53
N30_02	CBPR #3	**0.39**	12 231.51	**0.03**	**18.00**	**740.02**
	GAVNS	0.35	**12 211.01**	0.12	10.00	3292.60
N30_03	CBPR #3	**0.41**	**24 102.07**	**0.03**	**26.50**	**1152.36**
	GAVNS	0.36	24 106.57	0.13	10.00	5196.52
N30_04	CBPR #3	**0.41**	30 011.89	**0.04**	**33.25**	**1207.05**
	GAVNS	0.36	**30 010.14**	0.11	10.00	6159.69
N30_05	CBPR #3	**0.43**	59 623.06	**0.14**	**30.75**	**1063.50**
	GAVNS	0.34	**59 123.85**	0.19	10.00	9562.64

Table A.10
Comparison among the three configurations of our proposal, for the set of proposed instances with size 36 and 40. Best values are highlighted with bold.

Instance	Config.	HV	IGD$^+$	ϵ	Size	T (s)
ste36_01	#1	**0.43**	6451.73	0.07	8.80	3335.03
	#2	**0.43**	**6369.73**	**0.05**	**8.90**	3037.21
	#3	**0.43**	6385.93	**0.05**	8.70	**2830.55**
ste36_02	#1	**0.47**	**93 004.15**	**0.60**	**7.50**	**2782.29**
	#2	0.16	93 965.23	1.64	7.10	2821.88
	#3	0.25	93 398.33	1.18	6.00	3125.22
ste36_03	#1	0.17	56 599.88	**1.01**	4.80	2984.46
	#2	**0.28**	**56 289.38**	1.20	4.90	**2913.11**
	#3	0.24	56 301.50	2.36	**5.60**	3506.58
ste36_04	#1	0.45	53 250.49	0.41	7.10	3247.52
	#2	**0.51**	**52 753.90**	**0.31**	**9.30**	3131.03
	#3	0.48	53 303.88	0.40	6.50	**3030.77**
ste36_05	#1	**0.37**	**50 160.08**	**0.59**	6.30	**2843.14**
	#2	0.20	50 426.89	0.72	**7.10**	2922.91
	#3	0.28	50 680.08	0.64	5.10	2934.19
N40_01	#1	0.39	57 265.46	0.04	38.60	TL
	#2	0.39	**57 245.46**	0.04	37.50	TL
	#3	**0.40**	57 250.16	**0.03**	**39.60**	TL
N40_02	#1	**0.40**	51 708.00	**0.02**	34.50	TL
	#2	0.39	51 754.00	0.03	**35.30**	TL
	#3	0.39	**51 707.20**	0.03	33.70	TL
N40_03	#1	**0.45**	41 914.67	0.04	24.60	TL
	#2	**0.45**	41 935.67	0.04	**25.30**	TL
	#3	**0.45**	**41 863.77**	**0.03**	25.00	TL
N40_04	#1	**0.46**	**40 775.43**	0.04	29.60	TL
	#2	**0.46**	40 821.53	**0.03**	29.90	TL
	#3	0.45	40 799.23	0.04	**30.30**	TL
N40_05	#1	**0.42**	**54 019.74**	0.03	36.20	TL
	#2	0.41	54 143.64	0.04	36.40	TL
	#3	**0.42**	54 026.34	**0.02**	**37.80**	TL

References

[1] J.M. Pablo Pérez-Gosende, M. Díaz-Madroñero, Facility layout planning. An extended literature review, Int. J. Prod. Res. 59 (2021) 3777–3816.

[2] M. Besbes, M. Zolghadri, R. Costa Affonso, F. Masmoudi, M. Haddar, A methodology for solving facility layout problem considering barriers: Genetic algorithm coupled with A* search, J. Intell. Manuf. 31 (2020) 615–640.

[3] M.F. Anjos, M.V.C. Vieira, Mathematical optimization approaches for facility layout problems: The state-of-the-art and future research directions, European J. Oper. Res. 261 (2017) 1–16.

[4] M. Rubio-Sánchez, M. Gallego, F. Gortázar, A. Duarte, GRASP with path relinking for the single row facility layout problem, Knowl.-Based Syst. 106 (2016) 1–13.

[5] R. Kothari, D. Ghosh, Insertion based Lin–Kernighan heuristic for single row facility layout, Comput. Oper. Res. 40 (2013) 129–136.

[6] P. Burggräf, J. Wagner, B. Heinbach, Bibliometric study on the use of machine learning as resolution technique for facility layout problems, IEEE Access 9 (2021) 22569–22586.

[7] D.M. Simmons, One-dimensional space allocation: An ordering algorithm, Oper. Res. 17 (1969) 812–826.

[8] J. Chung, J.M.A. Tanchoco, The double row layout problem, Int. J. Prod. Res. 48 (2010) 709–727.

[9] P. Hungerländer, M.F. Anjos, A semidefinite optimization-based approach for global optimization of multi-row facility layout, European J. Oper. Res. 245 (2015) 46–61.

[10] Z. Kalita, D. Datta, Solving the bi-objective corridor allocation problem using a permutation-based genetic algorithm, Comput. Oper. Res. 52 (2014) 123–134.

[11] A.R.S. Amaral, The corridor allocation problem, Comput. Oper. Res. 39 (2012) 3325–3330.

[12] D. Datta, A.R.S. Amaral, J.R. Figueira, Single row facility layout problem using a permutation-based genetic algorithm, European J. Oper. Res. 213 (2011) 388–394.

[13] Z. Kalita, D. Datta, G. Palubeckis, Bi-objective corridor allocation problem using a permutation-based genetic algorithm hybridized with a local search technique, Soft Comput. 23 (2019) 961–986.

[14] G. Palubeckis, Fast local search for single row facility layout, European J. Oper. Res. 246 (2015) 800–814.

[15] C. Guan, Z. Zhang, J. Gong, S. Liu, Mixed integer linear programming model and an effective algorithm for the bi-objective double-floor corridor allocation problem, Comput. Oper. Res. 132 (2021) 105283.

[16] N. Uribe, A. Herrán, J.M. Colmenar, A path relinking-based approach for the bi-objective double floor corridor allocation problem, in: Conference of the Spanish Association for Artificial Intelligence, Springer, 2024, pp. 111–120.

[17] I.F.C. Fernandes, E.F.G. Goldbarg, S.M.D.M. Maia, M.C. Goldbarg, Multi- and many-objective path-relinking: A taxonomy and decomposition approach, Comput. Oper. Res. 133 (2021) 105370.

[18] M.G.C. Resende, C.C. Ribeiro, Scatter search and path relinking: Advances and applications, in: Handbook of Metaheuristics, Springer US, Boston, MA, 2010, pp. 1–37.

[19] F. Glover, Tabu search and scatter search, in: S. Voss, S. Martello, I.H. Osman, C. Roucairol (Eds.), Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer Academic Publishers, 1999, pp. 1–21.

[20] T.A. Feo, M.G.C. Resende, A probabilistic heuristic for a computationally difficult set covering problem, Oper. Res. Lett. 8 (1989) 67–71.

[21] A. Duarte, J. Sánchez-Oro, M.G.C. Resende, F. Glover, R. Martí, Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization, Inform. Sci. 296 (2015) 46–60.

[22] S. Pérez-Peló, J. Sánchez-Oro, A. Duarte, Finding weaknesses in networks using greedy randomized adaptive search procedure and path relinking, Expert Syst. 37 (2020) e12540.

[23] I. Lozano-Osorio, J. Sanchez-Oro, A.D. Lopez-Sanchez, A. Duarte, A reactive path relinking algorithm for solving the bi-objective p-median and p-dispersion problem, Soft Comput. 27 (2023) 8029–8059.

[24] M.G. Kendall, A new measure of rank correlation, Biometrika 30 (1938) 81–93, Full publication date: Jun., 1938.

[25] F.J. Brandenburg, A. Gleißner, A. Hofmeier, Comparing and aggregating partial orders with Kendall Tau distances, in: M.S. Rahman, S.-i. Nakano (Eds.), WALCOM: Algorithms and Computation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 88–99.

[26] A.E.F. Muritiba, M.J.N. Gomes, M.F. de Souza, H.L.G. Oria, Path-relinking with tabu search for the capacitated centered clustering problem, Expert Syst. Appl. 198 (2022) 116766.

[27] M. Li, X. Yao, Quality evaluation of solution sets in multiobjective optimisation: A survey, ACM Comput. Surv. 52 (2019).

[28] S. Ali, P. Arcaini, D. Pradhan, S.A. Safdar, T. Yue, Quality indicators in search-based software engineering: An empirical evaluation, ACM Trans. Softw. Eng. Methodol. 29 (2020).

[29] C. Audet, J. Bigeon, D. Cartier, S. Le Digabel, L. Salomon, Performance indicators in multiobjective optimization, European J. Oper. Res. 292 (2021) 397–422.

[30] J. Yuste, E.G. Pardo, A. Duarte, J.-K. Hao, Multi-objective general variable neighborhood search for software maintainability optimization, Eng. Appl. Artif. Intell. 133 (2024) 108593.

[31] J.J. Durillo, A.J. Nebro, jMetal: A Java framework for multi-objective optimization, Adv. Eng. Softw. 42 (2011) 760–771.

[32] N.R. Uribe, A. Herrán, J.M. Colmenar, A GRASP method for the bi-objective multiple row equal facility layout problem, Appl. Soft Comput. 163 (2024) 111897.

[33] R. Martín-Santamaría, S. Cavero, A. Herrán, A. Duarte, J.M. Colmenar, A practical methodology for reproducible experimentation: An application to the double-row facility layout problem, Evol. Comput. (2022) 1–35.

[34] M.L.-I. nez, J. Dubois-Lacoste, L. Pérez Cáceres, M. Birattari, T. Stützle, The irace package: Iterated racing for automatic algorithm configuration, Oper. Res. Perspect. 3 (2016) 43–58.

[35] T. Stützle, M. López-Ibáñez, Automated algorithm configuration and design, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 959–982.