Heuristic search to the capacitated clustering problem

Qing Zhou, Una Benlic, Qinghua Wu, Jin-Kao Hao

Heuristic search to the capacitated clustering problem

Qing Zhou [a], Una Benlic [b], Qinghua Wu [a,*], Jin-Kao Hao [c,d]

[a] *School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China, email: qingzhou@hust.edu.cn; qinghuawu1005@gmail.com*

[b] *SATALIA, 40 High Street, Islington High Street, London, United Kingdom, N1 8XB, email: una@satalia.com*

[c] *LERIA, Université d'Angers, 2 bd Lavoisier, 49045 Angers Cedex 01, France, email: jin-kao.hao@univ-angers.fr*

[d] *Institut Universitaire de France, 1 rue Descartes, 75231 Paris, France*

Abstract

Given a weighted graph, the capacitated clustering problem (CCP) is to partition a set of nodes into a given number of distinct clusters (or groups) with restricted capacities, while maximizing the sum of edge weights corresponding to two nodes from the same cluster. CCP is an NP-hard problem with many relevant applications. This paper proposes two effective algorithms for CCP: a Tabu Search (denoted as FITS) that alternates between exploration in feasible and infeasible search space regions, and a Memetic Algorithm (MA) that combines FITS with a dedicated cluster-based crossover. Extensive computational results on five sets of 183 benchmark instances from the literature indicate that the proposed FITS competes favorably with the state-of-the-art algorithms. Additionally, an experimental comparison between FITS and MA under an extended time limit demonstrates that further improvements in terms of the solution quality can be achieved with MA in most cases. We also analyze several essential components of the proposed algorithms to understand their importance to the success of these approaches.

Keywords: Tabu search; memetic algorithm; infeasible local search; capacitated clustering.

* Corresponding author.

1 Introduction

Given a weighted graph $G = (V, E)$ where V is a set of n nodes and E is a set of edges, let $w_i \geq 0$ be the weight of node $i \in V$ and let c_{ij} ($\{i, j\} \in E$) be the edge weight between nodes i and j ($c_{ij} = 0$, if $\{i, j\} \notin E$). The Capacitated Clustering Problem (CCP) is to partition V into a given number p ($p \leq n$) of disjoint clusters or groups such that the sum of node weights in each cluster is constrained by an upper and a lower capacity limit, while maximizing the sum of edge weights whose two associated endpoints belong to the same cluster.

Formally, let the binary variable X_{ig} take the value 1 if node i is assigned to group g ($g \in \{1, 2, ..., p\}$), and 0 otherwise. CCP can then be expressed as the following quadratic program [7,10]:

$$\textbf{maximize} \quad \sum_{g=1}^{p} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} X_{ig} X_{jg} \tag{1}$$

$$\textbf{subject to} \quad \sum_{g=1}^{p} X_{ig} = 1, \qquad \forall i \in V \tag{2}$$

$$L_g \leq \sum_{i=1}^{n} w_i X_{ig} \leq U_g, \qquad \forall g \in \{1, 2, ..., p\} \tag{3}$$

$$X_{ig} \in \{0, 1\}, \qquad \forall i \in V, g \in \{1, 2, ..., p\} \tag{4}$$

Constraint (2) guarantees that every node is assigned to exactly one cluster, while constraint (3) ensures that the minimum capacity (L_g) and the maximum capacity (U_g) requirements of each cluster are satisfied.

Note that CCP is closely related to the Graph Partitioning Problem (GPP) [4,5,16] where the lower and the upper capacity limits of the clusters are respectively set to 0 and a predetermined imbalance parameter. Moreover, the Maximally Diverse Grouping Problem (MDGP) [6,17,22,25,34,37,38] is a special case of CCP, when G is a complete graph with unit cost node weights. Consequently, CCP is an NP-hard problem as MDGP is known to be NP-hard. Furthermore, CCP is equivalent to the Handover Minimization Problem (HMP) in mobility networks [29] where the objective is to minimize the sum of weights of the edges with endpoints in different clusters. In addition to its application in mobility networks, CCP arises in several different contexts including mail delivery [2], VLSI design [38] and vehicle routing [14,23].

Since it was first defined in 1984 by Mulvey and Beck [33], a variety of solution approaches have been proposed for CCP given its practical importance and NP-hard nature. State-of-the-art approaches include a Greedy Randomized

Adaptive Search Procedure with Path Relinking (GRASP-PR) by Deng and Bard [10]. In 2013, Morán-Mirabal et al. [31] proposed three algorithms for the equivalent handover minimization problem: a GRASP with path-relinking (denoted as GQAP in the corresponding paper), a GRASP with evolutionary path-relinking (GevPR-HMP) and a population-based biased random-key genetic algorithm (BRKGA). According to their computational results, GevPR-HMP exhibits the best performance among those three algorithms. In 2015, Martínez-Gavara et al. [29] presented several approaches which may be considered as state-of-the-art methods for CCP, including a Greedy Randomized Adaptive Search Procedure (GRASP), a Tabu Search method (TS), a hybrid method combining GRASP with TS (GRASP+TS), and a Tabu Search with Strategic Oscillation (TS_SO). In 2016, Lai and Hao [24] proposed a highly effective Iterated Variable Neighborhood Search (IVNS) for CCP. More recently, Martínez-Gavara et al. [30] applied several methods to CCP including a GRASP algorithm (denoted as GRASP2-1) and an Iterated Greedy (IG) algorithm, while Brimberg et al. [7] presented two highly effective VNS-based heuristics denoted as GVNS and SGVNS. A comprehensive review on the most representative approaches for CCP prior to 2011 can be found in [10].

CCP is a constrained problem that imposes a lower and an upper capacity limit to the size of the clusters. One notices that most of the existing approaches for CCP restrict their search to the feasible region only, while only few approaches including GRASP-PR [10], GevPR-HMP [31], GQAP [31], and TS_SO [29] are allowed to visit infeasible solutions. In this work, we are interested in search algorithms that examine both the feasible and the infeasible space in search of improved solutions. For this purpose, we introduce a highly effective tabu search (denoted as FITS) that alternates between feasible and infeasible regions, bringing more flexibility into the search process. In addition, we propose the first population-based memetic algorithm (MA) for CCP. It uses FITS as the local optimization mechanism, and incorporates a dedicated cluster-based crossover to transfer pertinent properties ("building blocks") from parents to offspring. Experimental results on five sets of 183 benchmark instances indicate a highly competitive performance of FITS with respect to the existing state-of-the-art algorithms. Given a longer time limit, a computational comparison between FITS and MA reveals that MA is able to further improve on the performance of its underlying FITS in terms of solution quality.

The remainder of the paper is organized as follows. Section 2 presents FITS, followed by a detailed description of the proposed memetic approach in Section 3. Experimental results on a widely-used benchmark are provided in Section 4. Section 5 analyzes the contribution of the key algorithmic ingredient to the performance of the proposed algorithms. Furthermore, we motivate the choice for the crossover used by MA, prior to conclusions drawn in the last section.

2 Feasible and infeasible exploration with tabu search

2.1 Main framework

Algorithm 1 Main scheme of FITS

Require: Graph $G = (V, E)$
Ensure: Best found solution S^*
1: $S \leftarrow InitialSolution(G)$ /*Section 2.2*/
2: Initialize *tabu_list*
3: $S^* \leftarrow S$ /*S is a feasible solution*/
4: **while** stopping condition is not verified **do**
5: /*feasible local search phase*/
6: $(S, S_{local\_best}) \leftarrow feasible\_local\_search(S)$ /*Section 2.3*/
7: **if** $f(S_{local\_best}) > f(S^*)$ **then**
8: $S^* \leftarrow S_{local\_best}$
9: **end if**
10: /*infeasible local search phase*/
11: $(S, S_{local\_best}) \leftarrow infeasible\_local\_search(S)$ /*Section 2.4*/
12: **if** $f(S_{local\_best}) > f(S^*)$ **then**
13: $S^* \leftarrow S_{local\_best}$
14: **end if**
15: **end while**

Instead of confining the search process to feasible regions, a number of studies on highly constrained problems [9,12,21,26,35] have shown that the consideration of infeasible solutions during the search may help to better explore the search space. Based on this observation, the proposed tabu search alternates between a feasible local search phase (FLS for short) that only examines feasible solutions, and an infeasible local search phase (InfLS for short) where the capacity constraint is relaxed in a controlled manner. The two phases play different roles in the search process - FLS ensures an intensified exploitation in a relevant search region, while InfLS is used to introduce more freedom (diversification) into the search. By alternating between these two complementary phases, FITS is expected to explore various zones of the search space without being easily trapped in a local optimum.

Algorithm 1 summarizes the general framework of FITS. Starting from a feasible solution generated with a construction procedure (Section 2.2), FITS first enters the FLS phase that is based on the best-improvement strategy with a joint use of three types of move operators (Section 2.3.2). This phase terminates as soon as the search is deemed to be trapped in a deep local optimum, i.e., if the best found solution cannot be improved for N_{cons} consecutive iterations. The algorithm then switches to InfLS that relies on a penalty-based evaluation function to guide the search to move towards new search regions. The stopping condition is typically a time limit or a fixed number of iterations.

2.2 Initial solution

The starting point for the search is a feasible solution generated by means of two randomized construction methods similar to those used in [17,24]. The first method consists in two stages, where the first stage performs the following steps: (i) randomly select a node v from the set of unassigned nodes, and randomly choose a cluster g from the set of clusters whose lower capacity constraint is not satisfied; (ii) allocate v to cluster g. The two steps are repeated until all clusters satisfy the lower capacity constraint. Once the first stage is completed, the proposed construction method enters the second stage that: (i) randomly picks an unassigned node v and a cluster g such that $size[g] + w_v <= U_g$, where $size[g]$ and w_v represent respectively the current weight of cluster g and the weight of node v; (ii) assign v to g. The second stage of this procedure terminates as soon as all the nodes have been assigned.

As observed in our preliminary experiments, the above described method often fails to find a feasible assignment of all the nodes when the upper capacity limit of clusters is very tight. Consequently, we propose the second construction method which constitutes a slight modification of the first method. Instead of randomly choosing a node v in both stages of the first construction method, an unassigned node v is selected such that v has the largest weight (ties broken randomly). The other steps are kept unchanged.

The time complexity of the construction method is $O(n * p)$.

2.3 Feasible local search (FLS)

FLS searches for the most promising solutions in the feasible space of candidate solutions, thus ensuring that the capacity constraint is verified. It is based on the general tabu search framework [18] and a combined use of three complementary move operators as described in the following subsections.

2.3.1 Feasible search space and evaluation function

A candidate solution to CCP is any partition of the node set V into p subsets C_1, C_2,..., C_p, also called clusters. The search space, including both feasible and infeasible solutions, is then formally defined as:

$$\Omega = \{\{C_1, C_2, ..., C_p\} : \cup_{i=1}^{p} C_i = V,\ C_i \cap C_j = \emptyset\} \tag{5}$$

where $i \neq j$, $1 \leq i, j \leq p$. Notice that an infeasible solution may contain one

or more empty clusters.

The feasible search space includes the set of all the candidate solutions $\Omega_f \subset \Omega$ satisfying the capacity constraints:

$$\Omega_f = \{\{C_1, C_2, ..., C_p\} : L_i \leq |C_i| \leq U_i,\ \cup_{i=1}^{p} C_i = V,\ C_i \cap C_j = \emptyset\} \tag{6}$$

where $i \neq j$, $1 \leq i, j \leq p$ and $|C_i|$ represents the total weight of the nodes in cluster i (i.e., $|C_i| = \sum_{u \in C_i} w_u$).

To evaluate the quality of each candidate solution $s = \{C_1, C_2, ..., C_p\}$ in Ω_f, the evaluation function is equivalent to the objective function which sums up the edge weights associated to endpoints in the same cluster:

$$f(s) = \sum_{g=1}^{p} \sum_{i,j \in C_g, i<j} c_{ij} \tag{7}$$

2.3.2 Neighborhood Structures

As previously mentioned, the neighborhood exploited by FLS is defined by a joint use of three basic move operators, which have previously been employed in [7,10,24,29,30]. These operators are briefly described as follows:

OneMove operator: Given a solution $s = \{C_1, C_2, ..., C_p\}$, $OneMove$ transfers a node v from its original cluster i to another cluster j such that the capacity constraint is respected, i.e., $|C_i| - w_v \geq L_i$ and $|C_j| + w_v \leq U_j$. To rapidly evaluate the gain value for each candidate move, our algorithm employs a fast incremental evaluation technique similar to that used in [6,24,34,37]. The main idea is to maintain an incremental matrix γ, where each element $\gamma[v][g]$ represents the sum of the edge weights between v and other nodes located in cluster g of the current solution, i.e., $\gamma[v][g] = \sum_{u \in C_g} c_{uv}$. Let $OneMove(v, i, j)$ denote a move that consists in transferring a node v from cluster i to cluster j, the corresponding gain value can be conveniently calculated as:

$$\Delta_f(OneMove(v, i, j)) = \gamma[v][j] - \gamma[v][i]$$

After each $OneMove$ operation, a subset of values in γ affected by the move is updated as follows: $\gamma[u][i] = \gamma[u][i] - c_{uv}$, $\gamma[u][j] = \gamma[u][j] + c_{uv}, \forall u \in V$. The complexity to update γ after a $OneMove$ operation is $O(n)$.

SwapMove operator: This move operator swaps two nodes v and u from two different clusters i and j, such that the capacity constraint is maintained. Let $SwapMove(v, u)$ denote a swap move, the associated move gain can be efficiently obtained as:

$$\Delta_f(SwapMove(v,u)) = (\gamma[v][j] - \gamma[v][i]) + (\gamma[u][i] - \gamma[u][j]) - 2c_{vu}$$

Since a $SwapMove$ can be decomposed into two consecutive $OneMove$ operations, γ is updated in two steps as for the corresponding $OneMove$ moves. Clearly, updating γ after a $SwapMove$ operation can also be achieved in $O(n)$ time.

2-1 Exchange operator: Let v, u and z be three nodes where v and u are located in the same cluster i, while z belongs to another cluster j. The 2-1 $Exchange$ transfers v and u from cluster i to cluster j and simultaneously moves z from j to i, while respecting the capacity restriction of i and j. Let $Exchange(v, u, z)$ denote such a move, the resulting move gain can be computed as:

$\Delta_f(Exchange(v,u,z)) = (\gamma[v][j] - \gamma[v][i]) + (\gamma[u][j] - \gamma[u][i]) + (\gamma[z][i] - \gamma[z][j]) + 2(c_{vu} - c_{vz} - c_{uz})$

Since a 2-1 $Exchange$ move can be decomposed into three consecutive $OneMove$ operations, the matrix γ is consecutively updated three times according to the corresponding $OneMove$ moves.

2.3.3 Exploration of the feasible search space

The general scheme of FLS is summarized in Algorithm 2. Starting from a feasible solution, FLS selects at each iteration the best non-prohibited move (i.e., non-tabu move of the highest gain) from the union of $OneMove$, $SwapMove$ and 2-1 $Exchange$ moves, where ties are broken at random. Obviously, such a combined neighborhood ensures an intensified examination of the feasible search space, and thus enhances the capacity of finding improved feasible solutions. To avoid short-term cycling, each time a node v is moved from its original cluster C, it is forbidden to move v back to C for the next tt iterations (tt is called the *tabu tenure*). Along with this rule, an aspiration criterion is applied to allow a move, regardless of its tabu status, if it leads to an improved best found solution. The exploration of the feasible search space terminates if no improvement is achieved in the consecutive N_{cons} iterations (N_{cons} is called the *search depth*). At this stage, the algorithm switches to the Infeasible Local Search (InfLS) phase that introduces a greater diversity, as the search is deemed to be trapped in a deep local optimum.

Aside from the diversification incurred during InfLS, FLS additionally incorporates a shake procedure similar to that used in [24]. The shake procedure is applied periodically, and consists in performing a random move from the combined feasible $OneMove$ and $SwapMove$ neighborhoods. This operation is repeated η times, where η is the shake strength.

2.4 Infeasible local search (InfLS)

The basic idea of InfLS is to relax the capacity constraint so as to allow the algorithm to visit some intermediate infeasible solutions. In this way, a larger number of moves become available, enabling transitions between structurally different high-quality feasible solutions.

Algorithm 2 Feasible Local Search

Require: Initial solution s
Ensure: Final solution s, best solution s_{local_best} found during this phase

$s_{local\_best} \leftarrow s$
$NI \leftarrow 0$ /*number of consecutive iterations without improvement of s_{local_best}*/
$Iter1 \leftarrow 0$ /*iteration counter*/
Initialize $tabu\_list$
while $NI < N_{cons}$ **do**
  Choose the best allowed move $m \in \{OneMove \cup SwapMove \cup$ 2-1 $Exchange\}$
  $s \leftarrow s \oplus m$ /*Perform the best move*/
  Update $tabu\_list$
  if $f(s) > f(s_{local\_best})$ **then**
    $s_{local\_best} \leftarrow s$
    $NI \leftarrow 0$
  else
    $NI \leftarrow NI + 1$
  end if
  if $(Iter1 + 1)\%\delta == 0$ **then**
    $Shake()$
  end if
  $Iter1 \leftarrow Iter1 + 1$
end while
return (s, s_{local_best})

Algorithm 3 Infeasible Local Search (InfLS)

Require: Feasible solution s returned by FLS
Ensure: Best feasible solution found during InfLS s_{local_best}, final feasible solution s_{final}

$s_{tmp} \leftarrow s$ /*s_{tmp} is a duplicate of the starting feasible solution s*/
$flag\_fs \leftarrow false$ /*$flag\_fs$ is a boolean variable that indicates whether a feasible solution was encountered during InfLS*/
$s_{local\_best} \leftarrow s$
$penalty\_count \leftarrow 0$
$penalty\_factor \leftarrow 2$
$MI \leftarrow 0$
Initialize $tabu\_list$
while $MI \leq M$ **do**
  Choose the best allowed move $m \in \{OneMove \cup SwapMove \cup$ 2-1 $Exchange\}$
  $s \leftarrow s \oplus m$ /*Perform the best move*/
  Update $tabu\_list$
  if s is a feasible solution **then**
    if $f(s) > f(s_{local\_best})$ **then**
      $s_{local\_best} \leftarrow s$
    end if
  else
    $penalty\_count \leftarrow penalty\_count + 1$
  end if
  $MI \leftarrow MI + 1$
  if $(MI + 1)\%\lambda == 0$ **then**
    if $penalty\_count > \mu_1$ **then**
      $penalty\_factor \leftarrow penalty\_factor * \tau$
    else if $penalty\_count < \mu_2$ **then**
      $penalty\_factor \leftarrow penalty\_factor/\tau$
    end if
    $penalty\_count \leftarrow 0$
  end if
  if s is a feasible solution **then**
    $s_{final} \leftarrow s$
    $flag\_fs \leftarrow true$
  end if
end while
if $flag\_fs == false$ **then**
  $s_{tmp} \leftarrow Shake()$ /*apply the shake procedure to s_{tmp}, Section 2.3.3*/
  $s_{final} \leftarrow s_{tmp}$
end if
return (s_{final},s_{local_best})

2.4.1 Evaluation function and neighborhood structures

To evaluate the quality of a solution $s \in \Omega$ during InfLS, we employ a penalty-based evaluation function f_p which is a linear combination of the basic eval-

uation function f (Equation. 7) and a penalty function associated with the degree of solution infeasibility:

$$f_p(s) = \sum_{g=1}^{p} \sum_{i,j \in C_g, i<j} c_{ij} - \beta \times EX(s) \tag{8}$$

where β is a self-adjustment penalty parameter that controls the degree of infeasibility introduced into the search. $EX(s)$ is the total degree of infeasibility of s defined as $EX(s) = \sum_{g=1}^{p} o_g$, where

$$o_g = \begin{cases} L_g - |C_g|, & \text{if } |C_g| < L_g \\ |C_g| - U_g, & \text{if } |C_g| > U_g \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

The InfLS phase employs the same three basic move operators ($OneMove$, $SwapMove$ and 2-1 $Exchange$) defined in Section 2.3.2, but without any capacity restriction. We further use a fast incremental evaluation technique to effectively calculate the move gain that corresponds to the change in the penalty-based evaluation function f_p. Specifically, for a given move denoted as mv ($mv = OneMove$, $SwapMove$ or 2-1 $Exchange$), the move gain of mv can be defined as $\Delta_f(mv) = \Delta_f(mv) - \beta \times \Delta_{EX}(mv)$. In addition to the incremental matrix γ (see Section 2.3.2), another vector ω is maintained where each element ω_g represents the total weight of nodes contained in cluster g. Since all moves induced by $OneMove$, $SwapMove$ and 2-1 $Exchange$ operators only involve two clusters, the weight of these two clusters after each move can be directly calculated by adding or subtracting the weight of the added or removed nodes. Furthermore, the move gain associated to this change can easily be updated as described in Equation. 9. The complexity of the above gain update procedure is $O(1)$.

2.4.2 Exploration with InfLS

While both InfLS and FLS rely on the tabu search strategy, the main difference between these procedures lies in the previously described evaluation functions. The main scheme of InfLS is summarized in Algorithm 3.

During the search process, the variable β of the evaluation function (see Equation. (8)) is periodically updated depending on the penalty counter (*penalty_count*) which records the number of times a feasible solution has been found during λ consecutive iterations (λ is a parameter). Recall that β controls the degree

of infeasibility introduced into the search. More precisely, β is increased by a multiple of τ if $penalty\_count > \mu_1$, and is divided by τ if $penalty\_count < \mu_2$ (τ, μ_1 and μ_2 are parameters). Furthermore, the best found feasible solution s_{local_best} is updated with the current solution s if an improved feasible solution has been found. InfLS terminates after M iterations (M is a parameter), followed by the FLS phase. The starting point for the next round of the FLS search is the most recently encountered feasible solution s_{final} returned by InfLS.

Finally, if no feasible solution is found during the InfLS process, the starting feasible solution s returned by FLS is perturbed with the shake procedure (Section 2.3.3) and is then returned as the output of InfLS. Note that the shake procedure explores feasible solutions only thus resulting in a feasible solution. Algorithm 3 summarizes the general procedure of the InfLS phase, in which we use a boolean variable $flag\_fs$ to indicate whether a feasible solution is encountered during the InfLS phase. It is important to mention that FLS and InfLS use two separate tabu lists, which are critical to the performance of the two local search phases as they prevent the search from short-term cycling. When switching back to FLS, the tabu list is re-initialized before entering the main loop.

3 Memetic algorithm

Algorithm 4 Main scheme of MA

Require: Graph $G = (V, E)$
Ensure: Best solution S^* found so far
1: Initialize population $P = \{S_1, S_2, ..., S_{|P|}\}$
2: $S^* \leftarrow Best(P)$
3: **while** Time does not exceed t_{max} **do**
4: Randomly select two parent solutions $S_i \in P$ and $S_j \in P$
5: $S_c \leftarrow Crossover(S_i, S_j)$ /*Section 3.1*/
6: $S_c \leftarrow FITS\ (S_c)$ /*Section 2.1*/
7: **if** $f(S_c) > f(S^*)$ **then**
8: $S^* \leftarrow S_c$
9: **end if**
10: $POP \leftarrow Pool\_Updating(S_c, P)$ /*Section 3.2 */
11: **end while**

Relying on the combined exploitation power of local optimization and exploration capacity of population-based search, Memetic Algorithm (MA) [32] is an effective hybrid framework for tackling a variety of difficult combinatorial problems.

The main scheme of our MA for CCP is given in Algorithm 4. The algo-

rithm consists of four basic components: a population initializing procedure, a crossover operator, the FITS procedure for local improvement and a population updating rule. Each solution from the initial population is obtained with the two randomized construction methods described in Section 2.2, and then further improved with our FITS method presented in Section 2. At each cycle (generation) of MA, two parent solutions are randomly selected from the population, and then recombined by means of the crossover operator to generate an offspring solution (Section 3.1). This new offspring is then improved by applying a fixed number of iterations of our FITS algorithm (Section 2.1). Finally, the population updating rule decides whether the improved offspring should be inserted into the population and which existing solution should be replaced (Section 3.2). This process is repeated until a predefined stopping condition (usually a fixed number of generations or time limit) is reached.

3.1 Cluster-based crossover

Algorithm 5 Cluster-based crossover

Require: Two randomly selected parents $s^1 = \{C_1^1, C_2^1, ..., C_p^1\}, s^2 = \{C_1^2, C_2^2, ..., C_p^2\}$
Ensure: Offspring $s^o = \{C_1^o, C_2^o, ..., C_p^o\}$

```
l ← 1
while l ≤ p do
    if l%2 == 0 then
        Select a cluster C_* ∈ s^1 with the maximum sum of edge weights
        C_l^o ← C_*
    else
        Select a cluster C_* ∈ s^2 with the maximum sum of edge weights
        C_l^o ← C_*
    end if
    Remove the subset C_l^o of nodes from s^1 and s^2
    l ← l + 1
end while
Assign in a greedy manner the unassigned nodes V − {C_1^o ∪ C_2^o ∪ ... ∪ C_p^o}
```

Crossover operator is one of the key elements of a population-based algorithm. It is well-known that a crossover's efficiency on a given optimization problem crucially depends on its ability to preserve pertinent properties ("building blocks") from parents to offspring [19]. In the context of CCP, a building block may be defined as a cluster (i.e., group or subset of a graph partition), where the aim is to maximize the sum of edge weights whose two associated endpoints belong to the given cluster. As CCP can be classified as a grouping problem [13], it is more natural and straightforward to manipulate groups of objects (i.e., clusters) rather than individual objects. Furthermore,

an analysis on a sample of locally optimal CCP solutions (see Section 5.3.1) discloses a high percentage of nodes that are always grouped together across high quality solutions, which provides a strong motivation for preserving the grouped nodes (i.e., cluster) from parent individuals to offspring solution. The proposed cluster-based crossover (CBX) is further inspired by the Greedy Partition Crossover (GPX) used for the classic graph coloring problem [15].

Given two parent solutions, CBX generates an offspring $s^o = \{C_1^o, C_2^o, ..., C_p^o\}$ in two sequential stages as summarized in Algorithm 5. The first stage performs p iterations (i.e., one iteration per cluster), where each iteration l consists in selecting a cluster C_* from a reference parent such that the weighted sum of edges with both endpoints in C_* is maximized. Cluster C_* then becomes the l^{th} building block of s^o, followed by the removal of all the nodes contained in C_* from both parent individuals. The reference parent is selected between s^1 and s^2 in an alternating manner. Note that the first crossover stage may result in a partial solution as some nodes may have been left unassigned. The second stage of the crossover process is a greedy construction method that consists in selecting an unassigned node v and inserting it into cluster g of the offspring solution, such that $L_g \leq w_v + |C_g^o| \leq U_g$ while maximizing the objective function value defined in Equation. 7. This process is repeated until all the nodes are assigned.

3.2 Population updating strategy

Population update strategy is another key element of a MA algorithm whose main role is to maintain a healthily diversified population throughout the search. To avoid premature convergence, we employ a quality-and-distance pool updating strategy which takes into account both the solution quality and the distance between individuals in the population to decide whether a new offspring should be introduced into the population. For this purpose, the distance $Dist(S_a, S_b)$ between two solutions S_a and S_b is defined as the minimum number of one-move steps required to transform S_b to S_a [28]. Given a population $P = \{S_1, S_2, ..., S_{|P|}\}$, the distance between a solution S_i ($i \in 1, 2, ..., |P|$) and P is computed as:

$$D_{S_i,P} = min\{Dist(S_i, S_j) | S_j \in P, S_j \neq S_i\} \tag{10}$$

The main scheme of the population update procedure, which is similar to that used in [28,39], is provided in Algorithm 6.

First, offspring S_0 is tentatively added to P resulting in $P' = P \bigcup \{S_0\}$. The quality-and-distance score function R is then applied to rank each solution

$S_i \in P'$:

$$R(S_i, P') = \alpha \mathcal{X}(f(S_i)) + (1-\alpha)\mathcal{X}(D_{S_i,P'}) \tag{11}$$

where α is a parameter set to 0.6 according to [28], and $f(S_i)$ is the objective value of S_i. $\mathcal{X}(.)$ is a function defined as:
$\mathcal{X}(y) = \frac{y-y_{min}}{y_{max}-y_{min}+1}$

where y_{max} and y_{min} are the maximum and the minimum possible values of y respectively. Finally, the solution S_w with the smallest score is removed from the population to make space for offspring S_0.

4 Computational experiments

This section provides an extensive assessment of the proposed algorithms on a well-known set of 133 benchmark instances from CCPLIB, as well as on two new groups of 50 large instances recently generated in [7]. To evaluate the effectiveness of the proposed approaches, we perform comparisons with several state-of-the-art algorithms from the literature. For the ease of reading and for the sake of clarity, Table 1 shows the list of the proposed and the reference algorithms from the CCP literature.

Algorithm 6 Population update strategy

Require: Offspring S_0, population $P = \left\{S_1, S_2, ..., S_{|P|}\right\}$
Ensure: Updated population $P = \left\{S_1, S_2, ..., S_{|P|}\right\}$
1: Tentatively add S_0 to P: $P' = P \bigcup \{S_0\}$
2: **for** $i = 0, 1, ..., |P|$ **do**
3: Calculate the distance between S_i and P' according to Eq.(10)
4: Calculate the goodness score $R(S_i, P')$ of S_i according to Eq.(11)
5: **end for**
6: Identify the solution S_w with the smallest goodness score in P' : $S_w = min\{R(S_i, P') | i = 0, ..., |P|\}$
7: **if** $S_w \neq S_0$ **then**
8: Replace S_w with S_0: $P = P \bigcup \{S_0\} \setminus \{S_w\}$
9: **end if**

4.1 Benchmark instances

The instances from CCPLIB [1] can be grouped into three sets:

- DB (10 instances): This set was introduced by Deng and Bard [10] for the Maximally Diverse Grouping Problem (MDGP), and adapted for CCP in [29] by generating node weights with a uniform distribution in the range $[0, 10]$. These instances are characterized by $n = 82, p = 8, L_g = 25, U_g = 75$.
- RanReal (40 instances): This set was originally proposed in [17] and first adapted for CCP in [29] by generating node weights with a uniform distribution in the range $[0, 10]$. It includes 20 instances with $n = 240, p = 12, L_g = 75, U_g = 125$, and another 20 instances with $n = 480, p = 20, L_g = 100, U_g = 150$. The edge weights are real numbers randomly generated in the range $[0, 100]$.
- MM (83 instances): This set was introduced by Morán-Mirabal et al. [31] with $n \in \{20, 30, 40, 100, 200, 400\}$ and $p \in \{5, 10, 15, 25, 50\}$. The edge weights are real numbers, while L_g and U_g are respectively set to 0 and a real number that differs for each instance. The set is widely used in the literature for the handover minimization problem.

Additionally, we perform experiments on the following two groups of 50 large instances [2] from [7]:

- RanReal960 (30 instances): This set consists of 3 subsets. Each subset contains 10 instances characterized as follows:
  · $n = 960, p = 30, L_g = 120, U_g = 180$;
  · $n = 960, p = 40, L_g = 90, U_g = 135$;
  · $n = 960, p = 60, L_g = 60, U_g = 90$;
- MDG (20 instances): This set includes 20 instances with $n = 2000, p = 50, L_g = 200, U_g = 300$. The edge weights and the node weights are generated using the same method as in [11].

4.2 Experimental protocol

The proposed FITS algorithm requires eight parameters: tabu tenure (tt), search depth of FLS (N_{cons}), shake frequency (δ) and shake strength (η) of FLS, update frequency (λ) of the self-adjustment penalty β in InfLS, update coefficients of β (τ, μ_1, μ_2) in InfLS, and the maximum number of InfLS iterations (M). MA requires two additional parameters: the population size ($|P|$) and the number of FITS iterations ($Iter$). To determine the appropriate

[1] `http://www.optsicom.es/ccp/`
[2] `http://www.mi.sanu.ac.rs/~nenad/ccp/`

Table 1
List of the reference algorithms for CCP and its equivalent HMP.

Algorithm name	Reference	Search strategy
FITS	-	A tabu search alternating between exploration in feasible and infeasible search space
MA	-	A memetic algorithm that extends FITS with a dedicated cluster-based crossover operator
GRASP-PR	[10](2011)	A reactive GRASP
GevPR-HMP	[31](2013)	A GRASP combined with an evolutionary path-relinking algorithm in which a repair procedure is applied to achieve feasibility
GQAP	[31](2013)	A GRASP method with a new variant of path-relinking dealing with infeasibilities
BRKGA	[31](2013)	A biased random-key genetic algorithm using a parameterized uniform crossover
GRASP	[29](2015)	A simplified GRASP
TS	[29](2015)	A tabu search algorithm exploiting the 2-1 exchange neighborhood
GRASP+TS	[29](2015)	A hybrid combining GRASP with tabu search
TS_SO	[29](2015)	A tabu search with strategic oscillation that considers infeasible solutions
IVNS	[24](2016)	An iterated variable neighborhood search combining an extended variable neighborhood descent with a randomized shake procedure
GRASP2-1	[30](2017)	A new GRASP method in which the improvement procedure performs 2-1 exchanges
IG	[30](2017)	An iterated greedy method alternating between destructive and constructive phases
IG-GRASP	[30](2017)	A hybrid between GRASP2-1 and iterated greedy method
GVNS	[7](2017)	A general variable neighborhood search that follows the standard VNS approach including more levels of shaking
SGVNS	[7](2017)	A skewed general variable neighborhood search that allows moves to inferior solutions

Table 2
Settings of the parameters.

Parameter	Section	Description	Considered values	Final value
N_{cons}	2.3	search depth of each FLS phase	{500, 700, 1000, 1500, 2000}	1000
M	2.4	maximum number of iterations of each InfLS phase	{100, 150, 200, 250, 300}	200
tt	2.3,2.4	tabu tenure	{5, 7, 10, 12, 15}	10
δ	2.3	frequency of shake	{300, 400, 500, 600, 700}	500
η	2.3	shake strength	{0.06*n, 0.08*n, 0.10*n, 0.12*n, 0.14*n}	0.10*n
$Iter$	3	number of iterations of FITS in MA	{5000, 8000, 10000, 12000, 15000}	10000
$\|P\|$	3	size of population	{5, 7, 10, 13, 15}	5
λ	2.4	update frequency of β in InfLS	-	5
τ	2.4	update coefficients of β in InfLS	-	2
μ_1	2.4	update coefficients of β in InfLS	-	4
μ_2	2.4	update coefficients of β in InfLS	-	1

parameter settings for FITS and MA, we run the Iterated F-race (IFR) method [3], implemented within the IRACE package [27], on a selection of 20 RanReal instances with $n = 240$ and $n = 480$. The tuning budgets of FITS and MA are set to 500 runs with the time limit of $1.0 * n$ seconds for each run. Table 2 shows the tested and the final values obtained in the tuning process. For λ, τ μ_1 and μ_2, we simply adopt the values recommended in [9,20].

The proposed algorithms are coded in C++, compiled with the g++ compiler using option "-O3", and executed on an Intel E5-2670 processor (2.8GHz) with 2GB RAM running under Linux. For time scaling purposes, the execution time of the DIMACS machine benchmark [3] on our system is 0.19s for graph r300.5, 1.17s for graph r400.5 and 4.54s for graph r500.5.

[3] `ftp://dimacs.rutgers.edu/pub/dsj/clique/`

Table 3
Statistical results for FITS and six state-of-the-art algorithms on two sets of CCP instances: DB and RanReal. The best performance is indicated in bold.

Instance set		GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
DB & RanReal	#Best/Avg.	8/2	4/0	9/2	14/10	10/10	**32/40**	29/20
	$p\text{-}value_{best}/p\text{-}value_{avg}$	7.75e-11/1.54e-12	7.75e-11/1.54e-12	2.86e-9/1.54e-10	1.25e-7/7.70e-8	1.87e-9/2.54e-10	0.88/0.02	
	Average Dev_{best}/Dev_{avg}(%)	6.66/7.56	1.33/2.42	1.35/2.00	0.26/0.40	0.22/0.63	0.20/0.36	**0.19/0.33**
	AvgTime(s)	149.25	**50.68**	187.26	224.24	243.04	187.33	201.35

4.3 Comparison between FITS and the state-of-the-art algorithms on the general CCP instances

To evaluate the performance of FITS on the first two sets of instances (DB and RanReal), we provide comparisons with several state-of-the-art algorithms including GRASP [29], TS [29], GRASP+TS [29], IVNS [24], GVNS [7] and SGVNS [7]. For a fair comparison with these state-of-the-art algorithms, we use their corresponding source codes and run them on our computing platform under the same computing conditions as described in Section 4.2. GRASP, TS, GRASP+TS, IVNS were previously re-implemented by Lai et al. [24] and their source codes are available at `http://www.info.univ-angers.fr/pub/hao/ccp.html`. The source codes of GVNS and SGVNS were shared by the corresponding author and are available at `http://www.mi.sanu.ac.rs/~nenad/ccp/`. All the codes were compiled using g++ compiler with the '-O3' option. For all the algorithms in this experiment, the stopping condition is a fixed cutoff time limit t_{max} set to $1.0 * n$ seconds, where n is the number of nodes in the given graph. Due to the stochastic nature of the algorithms, we perform 20 independent runs of each algorithm per instance.

Table 3 summarizes the statistical results for each algorithm on the instances of the DB and RanReal sets. Row '#Best/Avg.' indicates the number of cases that each algorithm outperforms the remaining approaches in terms of the best and the average objective value. The average percent deviation of the best/average result from the best solution obtained within this experiment is provided in row 'Average Dev_{best}/Dev_{avg}', while the average computing time (in seconds) required by each algorithm to reach its final objective value is provided in row 'AvgTime'. For each instance, we calculate the best and the average deviation (Dev_{best} and Dev_{avg}) as $(f^* - f)/f^* \times 100$, where f is the best or the average result and f^* is the best solution obtained with all the compared algorithms. Finally, to determine whether there exists a statistically significant difference in performance between FITS and the six reference algorithms, row '$p\text{-}value_{best}/p\text{-}value_{avg}$' provides the p-values obtained with the pairwise Wilcoxon statistical test on the best/average results. Detailed results on the DB and the RanReal benchmarks are given in the Appendix (Tables

Table 4
Statistical results of FITS and six state-of-the-art algorithms on two large instance sets: RanReal960 and MDG. The best performance is given in bold.

Instance set		GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
RanReal960 & MDG	#Best/Avg.	0/0	0/0	0/0	0/0	0/0	**46/46**	4/4
	$p\text{-}value_{best}/p\text{-}value_{avg}$	1.54e-12/1.54e-12	1.54e-12/1.54e-12	1.54e-12/1.54e-12	1.14e-11/1.54e-12	0.09/0.16	2.86e-9/2.86e-9	
	Average $Dev_{best}/Dev_{avg}(\%)$	16.94/17.44	6.60/7.36	3.77/4.47	1.27/1.70	0.61/0.80	**0.01/0.61**	0.89/1.07
	AvgTime(s)	**694.09**	1115.02	1221.46	1251.36	1202.49	1240.53	1282.67

11, 12).

From Table 3, we observe that GRASP, TS, GRASP+TS, IVNS, GVNS, SGVNS and FITS respectively outperform the other algorithms on 8, 4, 9, 14, 10, 32 and 29 instances in terms of the best objective value. In terms of the average results, FITS achieves better performance on 20 instances, while GRASP, TS, GRASP+TS, IVNS, GVNS and SGVNS outperform the other methods on 2, 0, 2, 10, 10 and 40 instances respectively. In terms of the average percent deviation (Average Dev_{best}/Dev_{avg}), FITS reports the smallest deviation from the best solutions obtained within this experiment (0.19%/0.33%). Except for the comparison with SGVNS, the statistical test reveals a significant difference in performance between each of the reference algorithms (p-value ≤ 0.05), demonstrating the efficiency of FITS on the DB and the RanReal instances.

4.4 Comparison between FITS and the state-of-the-art algorithms on the large CCP instances

RanReal960 and MDG benchmarks consist of large instances recently used to assess the performance of the algorithms for CCP in [7]. A summary of the statistical results for these instances, obtained with FITS and the reference algorithms, are shown in Table 4. Detailed results are given in Tables 13 and 14 of the Appendix.

Row $p\text{-}value_{best}/p\text{-}value_{avg}$ reveals a statistically significant difference in performance between FITS and all the reference algorithms except GVNS. When considering the average percent deviation from the best-found solutions within the experiments, SGVNS outperforms all the algorithms with $Dev_{best} = 0.01\%$ and $Dev_{avg} = 0.61\%$, while FITS exhibits a better performance than GRASP, TS, GRASP+TS and IVNS.

Table 5
Statistical results of FITS and the six state-of-the-art algorithms on handover minimization instances with $n \geq 100$. The best performance is indicated in bold.

Instance set		GevPR-HMP	GQAP	BRKGA	IVNS	GVNS	SGVNS	FITS
MM ($n \geq 100$)	#Best/Avg.	9/4	11/0	2/0	37/18	23/15	24/25	**43/30**
	$p\text{-}value_{best}/p\text{-}value_{avg}$	1.97e-9/1.68e-8	5.51e-9/1.97e-11	5.47e-11/1.97e-11	0.00/0.02	1.19e-5/0.08	7.44e-5/0.38	
	Average Dev_{best}/Dev_{avg}(%)	1.32/1.74	8.37/10.15	3.19/3.94	0.01/0.28	0.20/1.53	0.19/0.33	**0.00/0.20**
	AvgTime(s)	-	-	-	97.82	**66.18**	75.77	120.48

4.5 Comparison between FITS and the state-of-the-art algorithms on the handover minimization instances

Table 5 summarizes the statistical results reported with FITS and the six state-of-the-art algorithms on the handover minimization instances with $n \geq 100$. These reference algorithms include IVNS [24], GVNS [7], SGVNS [7] and three algorithms proposed in [31]: GevPR-HMP, GQAP and BRKGA. The handover minimization instances with $n \in \{20, 30, 40\}$ do not appear to be challenging as all the considered algorithms are able to attain the best-known solution within a very short computing time for each case. For completeness, the computational results for these instances are provided in Appendix (Table 22). In Table 5, the results reported with GevPR-HMP, GQAP and BRKGA are directly compiled from [31], and were obtained over 5 independent runs with a cutoff time set to 24 hours. This is significantly longer than the time limit used for IVNS, GVNS, SGVNS and FITS, which is set to $1.0 * n$ seconds (24 hours vs. $n \leq 400$ seconds). The results for IVNS, GVNS, SGVNS and FITS were obtained across 20 independent runs under the same computing conditions as described in Section 4.2. When handling the handover minimization instances, we use the results in the form of minimization for a direct comparison. The following relation is used to transform the CCP objective function into the equivalent objective for handover minimization: $f_{min} = 2(\sum_{i<j} c_{ij} - f_{max})$, where f_{min} and f_{max} represent the objective values of handover minimization and CCP respectively. The symbol "-" denotes the cases when the result is not reported in the literature. Detailed results are given in Tables 15 and 16 of the Appendix.

From Table 5, we observe that GevPR-HMP, GQAP, BRKGA, IVNS, GVNS, SGVNS and FITS outperform the other reference algorithms on 9, 11, 2, 37, 23, 24 and 43 instances respectively in terms of the best objective value. In terms of the average results, FITS achieves a better performance on 30 instances, while GevPR-HMP, GQAP, BRKGA, IVNS, GVNS and SGVNS outperform the other methods on 4, 0, 0, 18, 15, 25 instances respectively. Notice that if we sum up the number of times each algorithm performed best,

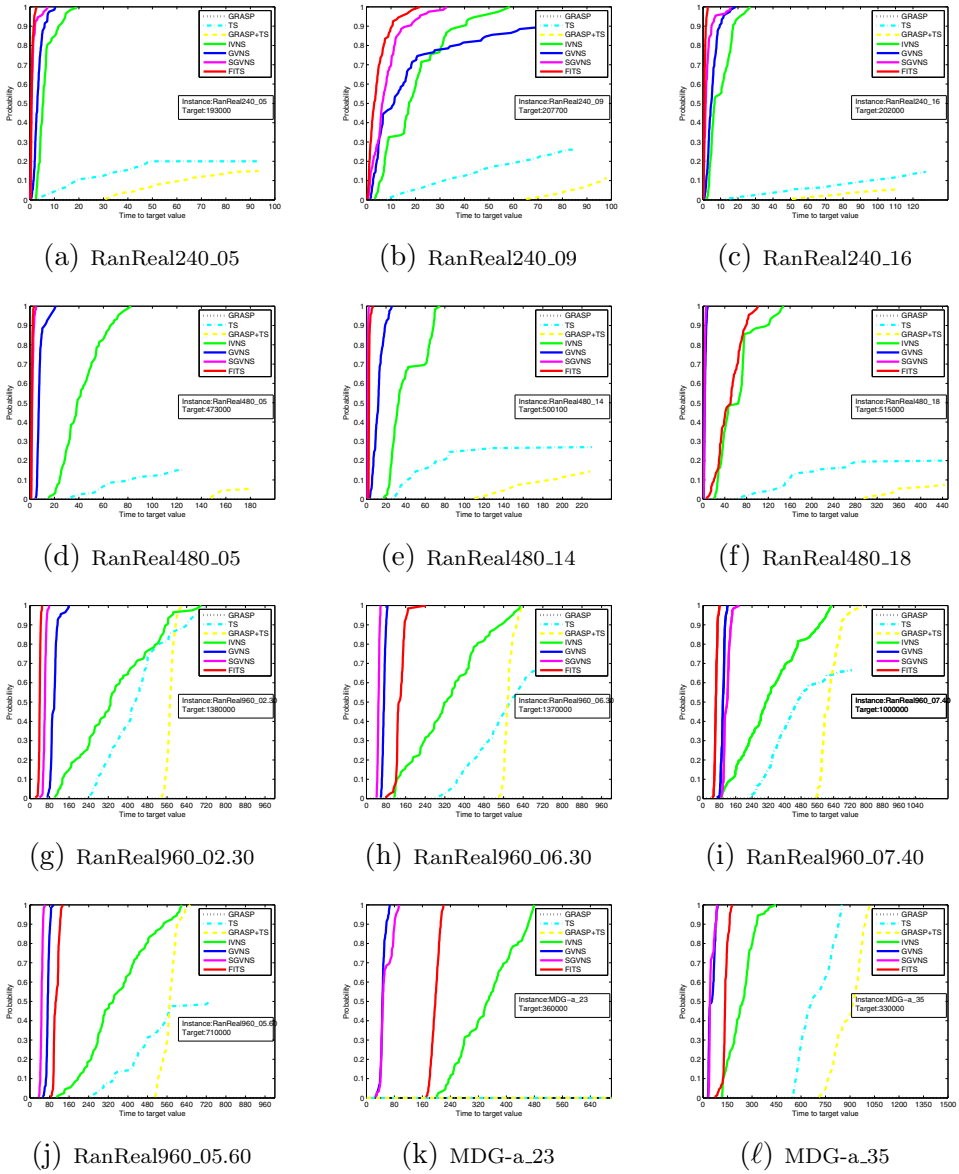
we get 149 for Best and 92 Avg respectively. This is because several algorithms obtain the same best objective value and the same average objective value on some instances. In terms of the average percent deviation of the best/average results from the best solutions obtained within this experiment, FITS shows the best performance with $Dev_{best} = 0.00\%$ and $Dev_{avg} = 0.20\%$. Finally, the p-values of the Wilcoxon pairwise test (row '$p\text{-}value_{best}/p\text{-}value_{avg}$') show a statistically significant difference in the best performance between FITS and the six reference approaches with p-value ≤ 0.05, which shows the benefit of FITS on the handover minimization instances with $n \geq 100$.

4.6 Time-to-target analysis

To further compare the performance between FITS and the reference algorithms, we apply the time-to-target (TTT) analysis which identifies the empirical probability distribution of the time required to achieve a given target value [1]. We conduct this TTT experiment by executing 100 independent runs of GRASP, TS, GRASP+TS, IVNS, GVNS, SGVNS and FITS on each instance. For each instance/target pair, the running times are sorted in an increasing order. We associate with the i-th sorted running time t_i a probability $p_i = (i-0.5)/100$, and plot the points (t_i, p_i). In this experiment, in order to allow all the algorithms to reach the target in all runs, the target value is set to be a value slightly smaller than the best obtained objective value. Fig. 1 illustrates the results for the compared algorithms on instances RanReal240_05, RanReal240_09, RanReal240_16, RanReal480_05, RanReal480_14, RanReal480_18, RanReal960_02.30, RanReal960_06.30, RanReal960_07.40, RanReal960_05.60, MDG-a_23 and MDG-a_35 which are randomly selected from three groups of the largest benchmark instances (RanReal, RanReal960 and MDG). From Fig. 1, we observe that FITS has the highest probability of reaching the target result with the shortest computing time on instances RanReal240_05, RanReal240_09, RanReal240_16, RanReal480_05, RanReal960_02.30 and RanReal960_07.40, and is the third best performing algorithm (after SGVNS and GVNS) on instances RanReal480_18, RanReal960_06.30, RanReal960_05.60, MDG-a_23 and MDG-a_35. We conducted the TTT experiments on other instances and observed similar behavior. Therefore, this experiment confirms that FITS competes very favorably with the reference algorithms GRASP, TS, GRASP+TS, IVNS, GVNS and SGVNS.

4.7 Comparison between FITS and MA

In this section, we additionally compare FITS and MA on the first two sets (DB and RanReal) of CCP instances and the handover minimization instances with $n \geq 100$. For this comparison, each algorithm is executed 20 times per



(a) RanReal240_05 (b) RanReal240_09 (c) RanReal240_16

(d) RanReal480_05 (e) RanReal480_14 (f) RanReal480_18

(g) RanReal960_02.30 (h) RanReal960_06.30 (i) RanReal960_07.40

(j) RanReal960_05.60 (k) MDG-a_23 (ℓ) MDG-a_35

Fig. 1. Probability distribution of the time required to achieve a target value.

Table 6
Comparative statistical results between FITS and MA on the first two sets (DB and RanReal) of CCP instances and the handover minimization instances with $n \geq 100$. The better performances are indicated in bold.

Instance set		FITS	MA
DB & RanReal	#Better1/Better2	3/7	**25/33**
	Average $Dev_{best}/Dev_{avg}(\%)$	0.03/0.11	**0.00/0.10**
MM ($n \geq 100$)	#Better1/Better2	0/8	**3/19**
	Average $Dev_{best}/Dev_{avg}(\%)$	**0.00**/0.15	**0.00/0.05**

Table 7
Summary of statistical results obtained with FITS and its two underlying components FLS and InfLS for the RanReal benchmark set. The best performance is indicated in bold.

Instance set		FITS	FLS	InfLS
RanReal	#Best/Avg.	**35/36**	12/4	0/0
	$p\text{-}value_{best}/p\text{-}value_{avg}$		6.23e-5/4.20e-7	2.54e-10/2.54e-10
	Average $Dev_{best}/Dev_{avg}(\%)$	**0.01/0.18**	0.03/0.21	0.33/0.53

instance with a prolonged computing time of $15.0 * n$ seconds. The reason behind an extended cutoff time is due to our experiences and observations from previous studies [19] indicating a slower convergence pace of an MA compared to a local search algorithm.

Table 6 summarizes the statistical comparative results between FITS and MA, while detailed results are given in Tables 17 and 18 in the Appendix. In a nutshell, MA improves on the best result reported by FITS for 28 instances, and fails to match the best solution obtained with FITS for only three instances. In terms of the average performance, MA outperforms FITS on 52 instances, and is outperformed by FITS on 15 instances. When considering the average percent deviation of the best/average results from the best solutions found within this experiment (Average Dev_{best}/Dev_{avg}), MA achieves a better performance than FITS on all the three sets of instances. Finally, for all the three sets of instances, the Wilcoxon test indicates a statistically significant difference in the best and the average performances with p-value = 6.437e-4 and p-value = 0.049 respectively. Thus, we can conclude that MA should be considered over FITS given a longer time limit.

5 Analysis

This section evaluates the importance of the key elements of the proposed FITS and MA algorithms: (i) the joint exploitation of feasible and infeasible search space, (ii) the best improvement strategy vs. the first improvement strategy, and (iii) the cluster-based crossover operator. The experiments presented below are carried out on a set of 40 instances from the RanReal benchmark with 20 independent executions per instance under the same computing platform as described in Section 4.2.

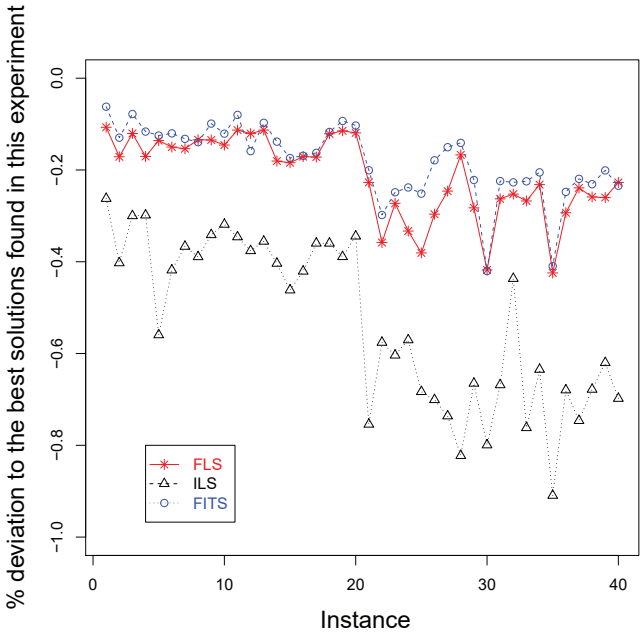


Fig. 2. Percentage deviation of the average result reported with FITS, FLS and ILS from the best solutions found in this experiment for the 40 RanReal instances.

5.1 Analysis of the combined exploitation of feasible and infeasible search space

While a number of existing heuristics for CCP including TS [29], IVNS [24], GVNS [7] and SGVNS [7] restrict their search to the feasible regions only, a key feature of FITS and several other heuristics like GRASP-PR [10], TS_SO [6], GevPR-HMP [31] and GQAP [31] is the consideration of infeasible solutions. By alternating between feasible and infeasible local searches, FITS is able to visit various zones of the search space without being easily trapped in a local optimum. To evaluate the effectiveness of this hybrid scheme, we compare FITS with its two underlying components, namely the feasible local search (FLS) and the infeasible local search (InfLS).

Table 7 summarizes the statistical results of this analysis, while detailed results are given in the Appendix (Table 19). As observed in Table 7, it is evident that FITS outperforms both FLS and InfLS. More precisely, in terms of the best and the average performance, FITS respectively reports matching or better results than both algorithms on 35 and 36 out of the 40 instances with p-values of 6.23e-5/4.20e-7 and 2.54e-10/2.54e-10. We further observe that InfLS alone appears to be the weakest of the three versions on all of the tested instances. To complement this comparison, Fig. 2 plots the performances of the three algorithms on these 40 instances. For each instance and each algorithm, the y-axis shows the percent deviation of the average result from the best

Table 8
Summary of statistical results obtained with FITS and its variant FITS_FI on the 40 RanReal instances. The best performance is indicated in bold.

Instance set		FITS	FITS_FI
RanReal	#Best/Avg.	**40/40**	0/0
	$p\text{-}value_{best}/p\text{-}value_{avg}$		2.54e-10/2.54e-10
	Average Dev_{best}/Dev_{avg}(%)	**0.00/0.15**	0.56/0.71
	AvgTime(s)	249.39	**157.44**

solutions found in this experiment. The figure further highlights the benefit of the combined use of the feasible and the infeasible local search.

5.2 The best improvement strategy v.s. the first improvement strategy

As described in Section 2.1, FITS applies the best improvement strategy to select a solution from the neighborhoods induced by the three move operators, i.e., *OneMove*, *SwapMove* and 2-1 *Exchange*. To verify the importance of this feature, we create a variant of FITS (denoted by FITS_FI) that uses the first improvement strategy, where the earliest visited neighboring solution of improved quality replaces the current solution. The summary of statistical results of this comparison on the 40 RanReal instances are shown in Table 8, while detailed results are given in Table 20 in the Appendix.

From Table 8, one notices that FITS outperforms FITS_FI on all of the instances both in terms of the best and the average result (#Best/Avg.). In terms of the average percent deviation of the best/average results from the best solutions obtained within this experiment (Average Dev_{best}/Dev_{avg}), FITS shows a better performance than FITS_FI with p-value = 2.54e-10 and p-value = 2.54e-10 respectively. These observations confirm the usefulness of the best improvement strategy within the proposed tabu search framework. However, as expected, FITS_FI shows to be faster than FITS in terms of the average time required to reach the final solution.

5.3 Analysis of the cluster-based crossover

5.3.1 Motivation behind the cluster-based crossover

As explained in Section 3.1, the basic idea behind the cluster-based crossover is to preserve building blocks (clusters of nodes) from parent individuals to offspring solution. Such crossovers have shown to be effective when there is a large percentage of nodes that are always grouped together between high-quality local optima (including global optima which are technically speaking

Table 9
Percentage of nodes that are grouped together in local optima of various qualities

Instance	S_{hq}	S_{all}	S_{lo}	Instance	S_{hq}	S_{all}	S_{lo}
RanReal240_01	72	54	45	RanReal480_01	82	29	14
RanReal240_02	85	49	39	RanReal480_02	88	26	14
RanReal240_03	75	48	38	RanReal480_03	88	28	14
RanReal240_04	80	48	41	RanReal480_04	92	29	14
RanReal240_05	86	48	38	RanReal480_05	83	26	14

also local optima) [4].

Given two local optima $s^1 = \{C_1^1, C_2^1, ..., C_p^1\}$ and $s^2 = \{C_1^2, C_2^2, ..., C_p^2\}$, let $E' = \{(C_i^1, C_j^2) | i \in \{1, 2, .., p\}, j \in \{1, 2, .., p\}\}$ denote the set of all the $p \times p$ cluster combinations of s^1 and s^2, and let J denote the set of nodes that are grouped together in both s^1 and s^2. Starting from $J = \emptyset$, we use an iterative procedure that determines the largest percentage of nodes grouped together in s^1 and s^2 with the following steps: (i) for each cluster combination $(C_i^1, C_j^2) \in E'$, compute the number of identical nodes $k_{C_i^1 C_j^2} = C_i^1 \cap C_j^2$; (ii) select a combination $(C_i^1, C_j^2) \in E'$ with the largest $k_{C_i^1 C_j^2}$ and place the common nodes into J (i.e., $J = J \cup \{C_i^1 \cap C_j^2\}$); (iii) remove from E' all combinations associated with C_i^1 and C_j^2. This process is repeated until E' becomes empty. The percentage of nodes that are grouped together in both s^1 and s^2 is then expressed as $100\% * \frac{|J|}{|V|}$.

For this analysis, we employ a selection of 10 hard instances from the RanReal set. For each instance, we collect a set S_{all} of local optima of different qualities, obtained after 500 independent runs of MA and FITS with different time limits. We select the top 20% (100) local optima with the largest objective values from S_{all} to form the subset S_{hq} of 'high-quality solutions'. Similarly, we take the bottom 20% (100) with the smallest objective values to form the subset S_{lo} of 'low-quality solutions'. Table 9 shows the percentages of common node groupings across all the local optima in S_{hq}, S_{all} and S_{lo} respectively. From these results, we conclude that the percentage of nodes that are grouped together throughout each of the high-quality local optima from S_{hq} is very large, ranging from 72% to 92%. Assuming that high-quality solutions might be close to an optimal solution or could themselves constitute optimal solutions, it is very likely that these clusters might form the building blocks of a global optimum.

5.3.2 Comparison with the uniform crossover

To evaluate the benefit of the cluster-based crossover that transfers pertinent properties from parents to offspring, we compare it with a standard uniform crossover where each node is randomly transferred to the same cluster as in one of the two parents. As the resulting offspring may violate the capacity constraint, the uniform crossover proceeds next by moving a randomly selected

Table 10
Comparison of the two MA versions using a uniform and a cluster-based crossover respectively on the 40 RanReal instances. The best performance is indicated in bold.

Instance set		Uniform CX	Cluster-based CX
RanReal	#Best/Avg.	2/0	**31/40**
	$p\text{-}value_{best}/p\text{-}value_{avg}$	4.46e-7/2.54e-10	
	Average Dev_{best}/Dev_{avg}(%)	0.56/0.71	**0.00/0.15**
	AvgTime(s)	**2898.81**	3856.70

node from the highest to the lowest weight cluster until the solution feasibility is reached. For this experiment, the other MA components are left unchanged. The time limit is set to $15.0 * n$ seconds per execution.

Table 10 summarizes the statistical results for each MA version on the 40 RanReal instances, while detailed results are given in Table 21 in the Appendix. Although the cluster-based crossover results in longer computing time than the uniform crossover due to a higher complexity, it clearly outperforms the uniform crossover in terms of both the best and the average results with p-value = 4.46e-7 and p-value = 2.54e-10 respectively. Indeed, out of the 40 instances, the cluster-based crossover is outperformed by the uniform crossover only on two instances. These observations highlight the importance of preserving important building blocks from parents to children in case of CCP.

6 Conclusion

We presented two highly effective heuristics for the Capacitated Clustering Problem (CCP): a tabu search approach (denoted as FITS) that alternates between exploration in feasible and infeasible search space regions, and a Memetic Algorithm (MA) that extends FITS with a dedicated cluster-based crossover and a quality-and-distance pool updating strategy. The computational results on five sets of 183 CCP instances indicate that both FITS and MA compete favorably with the current state-of-the-art algorithms. The investigation of several essential components of the proposed algorithms sheds light on the following points. First, the consideration of both feasible and infeasible search space regions can greatly enhance the neighborhood search for CCP. Second, the best improvement strategy is able to outperform the first improvement strategy within the tabu search framework for CCP. Third, given an extended time limit, MA can further improve upon the performance of FITS which is greatly due to the cluster-based crossover that transfers pertinent properties from parents to offspring. The use of this crossover within MA was motivated by a large degree of similarity between high-quality CCP solutions. Finally, this work demonstrates the effectiveness of exploring both feasible and infeasible spaces for CCP - an idea that certainly deserves to be investigated on other highly constrained problems in the future.

Acknowledgments

We are grateful to the reviewers for their valuable comments which helped us to improve the paper. We thank the authors of [7,24] for kindly sharing with us their source codes that enabled fair comparisons with the proposed algorithms. This work is partially supported by the National Natural Science Foundation Program of China [Grant No. 71401059, 71771099, 71810107003, 71620107002, 71531009] and the Huazhong University of Science and Technology (5001300001).

References

[1] Aiex, R. M., Resende, M. G. C., & Ribeiro, C. C. (2007). Ttt plots: a perl program to create time-to-target plots. Optimization Letters, 1(4), 355-366.

[2] Bard, J. F., & Jarrah, A. I. (2009). Large-scale constrained clustering for rationalizing pickup and delivery operations. Transportation Research Part B: Methodological, 43(5), 542-561.

[3] Bartz-Beielstein, T., Chiarandini, M., Paquete, L., & Preuss, M. (2010). Experimental Methods for the Analysis of Optimization Algorithms. Springer Berlin Heidelberg.

[4] Benlic, U., & Hao, J. K. (2011). A multilevel memetic approach for improving graph k-partitions. IEEE Transactions on Evolutionary Computation, 15(5), 624-642.

[5] Benlic, U., & Hao, J. K. (2013). Hybrid Metaheuristics for the Graph Partitioning Problem. Hybrid Metaheuristics, 157-185.

[6] Brimberg, J., Mladenović, N., & Urošević, D. (2015). Solving the maximally diverse grouping problem by skewed general variable neighborhood search. Information Sciences, 295, 650-675.

[7] Brimberg, J., Mladenović, N., Todosijević, R., & Urošević, D. (2017). Solving the capacitated clustering problem with variable neighborhood search. Annals of Operations Research(2), 1-33.

[8] Chen, X., Ong, Y. S., Lim, M. H., & Tan, K. C. (2011). A multi-facet survey on memetic computation. IEEE Transactions on Evolutionary Computation, 15(5), 591-607.

[9] Chen Y.N., Hao J.K., Glover F. (2016). A hybrid metaheuristic approach for the capacitated arc routing problem, European Journal of Operational Research, 253(1): 25–39.

[10] Deng, Y., & Bard, J. F. (2011). A reactive GRASP with path relinking for capacitated clustering. Journal of Heuristics, 17(2), 119-152.

[11] Duarte, A., & Martí, R. (2007). Tabu search and grasp for the maximum diversity problem. European Journal of Operational Research, 178(1), 71-84.

[12] Errico, F., Desaulniers, G., Gendreau, M., Rei, W., & Rousseau, L.M. (2016). A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times, European Journal of Operational Research, 249(1):55-66.

[13] Falkenauer, E. Genetic algorithms and grouping problems. New York: Wiley, 1998.

[14] Fisher, M. L., & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. Networks, 11(2), 109-124.

[15] Galinier, P., & Hao, J.k. (1999). Hybrid evolutionary algorithms for graph coloring. Journal of Combinatorial Optimization, 3(4): 379-397.

[16] Galinier, P., Boujbel, Z., & Fernandes, M. C. (2011). An efficient memetic algorithm for the graph partitioning problem. Annals of Operations Research, 191(1), 1-22.

[17] Gallego, M., & Duarte, A. (2013). Tabu search with strategic oscillation for the maximally diverse grouping problem. Journal of the Operational Research Society, 64(5), 724-734.

[18] Laguna, M. & Glover, F. (1997). Tabu Search. Springer.

[19] Hao, J. K. (2012). Memetic Algorithms in Discrete Optimization. In F. Neri, C. Cotta, P. Moscato (Eds.) Handbook of Memetic Algorithms. Studies in Computational Intelligence 379, Chapter 6, pages 73-94, 2012. Springer Berlin Heidelberg.

[20] Hertz, A., Laporte, G., & Mittaz, M. (2000). A tabu search heuristic for the capacitated arc routing problem. Operations Research, 48(1), 129-135.

[21] Higgins, A. J. (2001). A dynamic tabu search for large-scale generalised assignment problems. Computers & Operations Research, 28(10), 1039-1048.

[22] Johnes, J. (2015). Operational research in education. European Journal of Operational Research, 243(3), 683-696.

[23] Koskosidis, Y. A., & Powell, W. B. (1992). Clustering algorithms for consolidation of customer orders into vehicle shipments. Transportation Research Part B Methodological, 26(5), 365-379.

[24] Lai, X., & Hao, J. K. (2016). Iterated variable neighborhood search for the capacitated clustering problem. Engineering Applications of Artificial Intelligence, 56, 102-120.

[25] Lai, X., & Hao, J. K. (2016). Iterated maxima search for the maximally diverse grouping problem. European Journal of Operational Research, 254(3):780-800.

[26] Lapierre, S. D., Ruiz, A., & Soriano, P. (2006). Balancing assembly lines with tabu search. European Journal of Operational Research, 168(3), 826-837.

[27] López-Ibánez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., & Stützle, T. (2016). The irace package: iterated racing for automatic algorithm configuration.Operations Research Perspectives, 3, 43-58.

[28] Lü, Z., Glover, F., & Hao, J. K. (2010). A hybrid metaheuristic approach to solving the ubqp problem. European Journal of Operational Research, 207(3), 1254-1262.

[29] Martínez-Gavara, A., Campos, V., Gallego, M., Laguna, M., & Martí, R. (2015). Tabu search and GRASP for the capacitated clustering problem. Computational Optimization and Applications, 62(2), 589-607.

[30] Martínez-Gavara, A., Landa-Silva, D., Campos, V., & Martí, R. (2017). Randomized heuristics for the capacitated clustering problem. Information Sciences, 417.

[31] Morán-Mirabal, L. F., González-Velarde, J. L., Resende, M. G., & Silva, R. M. (2013). Randomized heuristics for handover minimization in mobility networks. Journal of Heuristics, 19(6), 845-880.

[32] Moscato, P., & Cotta, C. (2006). A Gentle Introduction to Memetic Algorithms. Handbook of Metaheuristics. Springer US.

[33] Mulvey, J. M., & Beck, M. P. (1984). Solving capacitated clustering problems . European Journal of Operational Research, 18(3), 339-348.

[34] Palubeckis, G., Ostreika, A., & Rubliauskas, D. (2015). Maximally diverse grouping: an iterated tabu search approach. Journal of the Operational Research Society, 66(4), 579-592.

[35] Paraskevopoulos, D. C., Laporte, G., Repoussis, P. P., & Tarantilis, C. D. (2017). Resource Constrained Routing and Scheduling: Review and Research Prospects. European Journal of Operational Research, 263(3):737-754.

[36] Porumbel, D. C., Hao, J. K., & Kuntz, P. (2010). An evolutionary approach with diversity guarantee and well-informed grouping recombination for graph coloring. Computers & Operations Research, 37(10), 1822-1832.

[37] Rodriguez, F. J., Lozano, M., García-Martínez, C., & González-Barrera, J. D. (2013). An artificial bee colony algorithm for the maximally diverse grouping problem. Information Sciences, 230(5), 183-196.

[38] Weitz, R. R., & Lakshminarayanan, S. (1998). An empirical comparison of heuristic methods for creating maximally diverse groups. Journal of the Operational Research Society, 25(6), 473-482.

[39] Wu, Q., & Hao, J. K. (2013). A hybrid metaheuristic method for the maximum diversity problem. European Journal of Operational Research, 231(2), 452-464.

Appendix

The purpose of this appendix is to show detailed computational results and comparisons between our two proposed algorithms (FITS and MA) and the state-of-the-art algorithms on the complete CCP benchmark consisting of 183 instances (Tables 11-22). For each instance and approach, columns 'f_{best}', 'f_{avg}' and 't_{avg}' show respectively the best objective value, the average objective value and the average computing time in seconds required to reach the final solution (see Section 4.2 for the used experimental protocol). Column 'Dev_{best}/Dev_{avg}' indicates the percent deviation between the best or the average result and the best solutions obtained within each experiment. Row '#Best' gives the number of cases when each algorithm outperforms the remaining approaches, while row 'Average' shows the average result for a given subset of instances. Finally, row 'p-value' indicates the outcome of the non-parametric Friedman tests on the results obtained with FITS and the reference algorithms. The best results are highlighted in bold.

Table 11. Comparison of FITS with six state-of-the-art algorithms on the first two sets of CCP instances (RanReal and DB).

	f_{best}							f_{avg}						
Name	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
Sparse82_01	**1342.17**	1336.18	**1342.17**	**1342.17**	**1342.17**	**1342.17**	**1342.17**	1342.13	1319.78	1342.08	**1342.17**	**1342.17**	**1342.17**	**1342.17**
Sparse82_02	**1306.64**	1304.07	**1306.64**	**1306.64**	**1306.64**	**1306.64**	**1306.64**	1305.13	1281.86	1305.82	**1306.64**	**1306.64**	**1306.64**	**1306.64**
Sparse82_03	1352.60	**1353.94**	**1353.94**	**1353.94**	**1353.94**	**1353.94**	**1353.94**	1349.32	1340.87	1351.97	**1353.94**	**1353.94**	**1353.94**	**1353.94**
Sparse82_04	1289.41	1289.85	**1291.22**	**1291.22**	**1291.22**	**1291.22**	**1291.22**	1286.76	1275.32	1286.79	**1291.22**	**1291.22**	**1291.22**	**1291.22**
Sparse82_05	**1352.35**	1351.93	**1352.35**	**1352.35**	**1352.35**	**1352.35**	**1352.35**	**1352.35**	1326.91	**1352.35**	**1352.35**	**1352.35**	**1352.35**	**1352.35**
Sparse82_06	**1354.61**	**1354.61**	**1354.61**	**1354.61**	**1354.61**	**1354.61**	**1354.61**	1348.27	1338.99	1354.53	**1354.61**	**1354.61**	**1354.61**	**1354.61**
Sparse82_07	**1266.94**	1263.46	**1266.94**	**1266.94**	**1266.94**	**1266.94**	**1266.94**	1266.41	1240.36	1266.49	**1266.94**	**1266.94**	**1266.94**	**1266.94**
Sparse82_08	**1393.02**	1391.80	**1393.02**	**1393.02**	**1393.02**	**1393.02**	**1393.02**	**1393.02**	1356.97	**1393.02**	**1393.02**	**1393.02**	**1393.02**	**1393.02**
Sparse82_09	**1294.12**	**1294.12**	1293.39	**1294.12**	**1294.12**	**1294.12**	**1294.12**	1293.50	1283.51	1293.39	**1294.12**	**1294.12**	**1294.12**	**1294.12**
Sparse82_10	**1356.98**	**1356.98**	**1356.98**	**1356.98**	**1356.98**	**1356.98**	**1356.98**	1356.64	1331.70	1356.92	**1356.98**	**1356.98**	**1356.98**	**1356.98**
RanReal240_01	208042.53	223612.67	222583.14	224831.56	224580.56	**224968.01**	224941.48	205301.82	221895.65	221577.41	224571.29	224207.09	224769.72	**224802.06**
RanReal240_02	192176.29	202712.60	202267.35	**204624.36**	204205.41	**204624.36**	**204624.36**	190315.97	201126.44	200804.77	204275.49	203682.12	**204444.31**	204359.38
RanReal240_03	189174.76	196190.26	196219.48	198861.68	198472.78	**199059.56**	198954.91	184911.36	193715.43	194896.16	198606.95	197806.87	**198849.22**	198799.84
RanReal240_04	211858.14	222757.74	223014.44	225390.88	225144.59	**225627.16**	**225627.16**	208108.02	220498.11	221213.35	225069.14	224463.81	**225389.88**	225364.97
RanReal240_05	185068.56	193354.26	193366.04	195540.41	194911.72	195516.57	**195564.48**	182679.54	191838.80	191419.12	195184.84	194361.91	195306.51	**195320.28**
RanReal240_06	204852.60	214549.17	214795.75	216713.91	216383.13	216733.31	**216747.32**	201609.66	213162.89	213268.67	216355.53	215849.06	**216584.23**	216487.02
RanReal240_07	199266.17	207885.74	207651.76	209216.90	209118.64	209223.34	**209305.70**	197642.75	206463.38	206283.07	208992.44	208329.21	**209080.57**	209029.23
RanReal240_08	191567.87	203825.77	202751.94	**205246.82**	204754.36	205154.20	**205246.82**	188619.12	201792.09	201223.78	204842.79	204137.78	204951.20	**204961.05**
RanReal240_09	198371.82	208130.74	208084.10	209142.07	208702.16	209007.44	**209159.16**	195694.73	206905.51	206706.66	208720.16	208276.52	208904.78	**208952.48**
RanReal240_10	179776.60	190295.58	189990.45	192885.48	192343.75	**193062.60**	192986.21	178618.43	188282.57	188638.25	192598.79	191874.34	**192842.05**	192811.13
RanReal240_11	192080.94	203400.14	202566.12	204647.20	204399.04	204615.71	**204722.75**	190809.39	201583.11	201425.46	204377.08	203900.12	204480.92	**204559.39**
RanReal240_12	190392.42	200045.55	199587.27	201028.32	200822.69	201076.30	**201117.11**	188016.22	198745.49	198735.77	200763.75	200150.52	**200938.30**	200797.67
RanReal240_13	188179.08	200362.63	200707.10	202331.20	201977.87	202321.58	**202335.99**	185699.95	199258.93	199400.56	202027.28	201356.20	**202198.63**	202139.57
RanReal240_14	209616.67	227192.94	226818.46	**228870.89**	228661.60	228775.14	**228870.89**	205710.79	225659.19	225340.11	228520.04	228054.32	**228569.78**	228554.78
RanReal240_15	179818.51	189389.87	188797.36	191152.17	190575.48	191238.53	**191255.87**	177071.08	187190.02	187392.49	190827.68	189965.59	**191058.62**	190923.28
RanReal240_16	192596.39	201462.83	202285.47	**204074.95**	203816.48	203991.53	204054.99	189519.11	200135.28	200807.00	203668.49	203270.11	203649.04	**203710.39**
RanReal240_17	187556.75	194012.92	194388.66	195206.73	194840.79	195423.83	**195561.36**	186122.62	192037.08	192104.74	194950.57	194404.70	195241.19	**195243.32**
RanReal240_18	180830.22	193593.81	192869.78	194916.37	194915.62	**195120.98**	195100.39	179180.63	191159.08	191152.82	194704.23	194114.09	**194967.73**	194872.13
RanReal240_19	188396.09	196746.61	196285.24	199200.03	198828.82	**199307.33**	199225.98	186251.65	194628.91	195112.90	198905.05	198119.14	**199093.86**	199040.43
RanReal240_20	201830.01	210191.93	210396.12	212264.10	211984.80	212268.46	**212268.52**	199866.05	208860.59	208700.04	211871.74	211458.51	212037.43	**212049.85**
RanReal480_01	489977.75	545089.31	546703.95	554337.23	553224.53	555430.60	**555489.92**	483244.45	538552.42	540798.54	553795.85	552326.59	**554994.68**	554376.54
RanReal480_02	464078.50	500020.24	502865.44	510066.41	508711.62	510718.79	**511280.50**	460760.58	494706.50	497535.90	509058.46	507540.62	**510304.78**	509757.15
RanReal480_03	449196.02	486459.38	486913.79	496334.51	495140.23	**497725.86**	497295.19	446619.33	480858.95	482487.60	495409.98	493706.92	**496785.80**	496059.50
RanReal480_04	472680.46	512815.26	510347.72	521669.00	520653.34	**522572.81**	522305.16	468798.75	504488.99	505766.67	520051.55	519051.60	**521952.28**	521062.13
RanReal480_05	434381.50	471071.30	473774.42	483670.19	481803.95	483819.77	**484084.66**	428924.10	467096.07	469365.07	482390.51	480508.80	482603.84	**482867.74**
RanReal480_06	479994.22	523619.23	524093.48	533589.61	532702.72	**534515.67**	533991.27	476225.99	518672.22	519996.88	532462.25	531627.12	**533916.00**	533036.36
RanReal480_07	484189.08	535454.67	533884.71	545343.81	544445.60	**545812.49**	545470.73	477599.50	530962.69	531611.12	544060.83	542318.56	**545302.82**	544651.12
RanReal480_08	482171.91	521615.37	521459.15	531974.48	531287.92	**532736.12**	532417.42	477609.32	516060.40	517411.97	531023.25	529525.39	**532109.62**	531667.91
RanReal480_09	492676.73	546829.14	545707.17	555604.38	555163.27	556865.18	**556868.85**	481867.19	539150.61	541281.01	554820.43	553394.39	**556081.91**	555634.40
RanReal480_10	466084.48	508095.64	508999.35	519066.57	517431.34	520014.70	**520257.54**	461032.38	503102.84	504603.14	518412.32	516228.20	518024.16	**518071.71**
RanReal480_11	473865.90	517123.38	514661.24	523463.33	522626.50	**524124.60**	523991.29	467783.06	508603.89	510647.48	522201.99	521218.97	**523508.02**	522816.94
RanReal480_12	451806.42	493221.32	492811.38	501462.57	499914.17	**502570.10**	501915.56	448852.16	489093.31	488561.87	500055.30	498596.97	**501632.20**	500776.79
RanReal480_13	487797.57	527920.79	524454.74	534294.24	533672.27	**535411.94**	535025.51	482511.37	519991.80	520834.94	533478.80	532322.68	**534651.24**	533823.79
RanReal480_14	451992.66	503936.70	504472.18	513186.65	512764.33	**514537.52**	514107.62	444843.33	497303.47	499403.40	512501.71	511212.78	**513935.11**	513053.25
RanReal480_15	465846.65	509749.02	507379.12	516657.20	515607.47	**518029.18**	517205.02	461078.21	503685.19	502983.82	515416.17	514465.20	**517189.00**	516018.38
RanReal480_16	487514.16	540214.26	541062.43	549230.25	549033.57	**549840.64**	549552.63	483468.24	533478.98	536063.21	548274.66	547075.83	**549253.54**	548462.13
RanReal480_17	482452.36	530089.83	529061.73	537223.44	536402.45	**537993.79**	537924.55	476468.76	523792.28	523877.97	536149.06	534770.20	**537568.15**	536745.39
RanReal480_18	478578.77	518280.43	517080.81	525490.09	524631.54	**526349.49**	525822.76	474635.66	510154.88	510238.83	524515.11	522908.72	**525453.50**	524712.42
RanReal480_19	474559.35	512590.03	513182.29	522280.40	521672.59	**522757.15**	522316.22	469860.30	508273.65	509272.24	521442.89	519720.44	**522218.13**	521267.22
RanReal480_20	461494.13	508105.69	511126.25	518436.63	516488.64	**518847.01**	518349.10	457017.22	504721.35	505910.27	516935.15	515450.41	**518202.52**	517430.77
#Best	8	4	9	14	10	**32**	**29**	2	0	2	10	10	40	**20**
p-value	7.75e-11	7.75e-11	2.86e-9	1.25e-7	1.87e-9	0.88		1.54e-12	1.54e-12	1.54e-10	7.70e-8	2.54e-10	0.02	

Table 12. Comparison between FITS and six state-of-the-art algorithms on the first two sets of CCP instances (RanReal and DB).

	$Dev_{best}/Dev_{avg}(\%)$							t_{avg}						
Name	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
Sparse82_01	0.00/0.00	0.45/1.67	0.00/0.01	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	32.67	5.82	25.73	0.32	30.83	31.25	12.51
Sparse82_02	0.00/0.12	0.20/1.90	0.00/0.06	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	41.34	4.71	32.60	1.32	25.10	22.08	20.30
Sparse82_03	0.10/0.34	0.00/0.97	0.00/0.15	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	27.53	5.30	40.18	0.18	24.29	30.60	4.28
Sparse82_04	0.14/0.35	0.11/1.23	0.00/0.34	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	36.28	11.28	30.47	6.89	25.90	29.40	26.75
Sparse82_05	0.00/0.00	0.03/1.88	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	16.31	6.81	8.53	0.11	42.73	25.40	3.95
Sparse82_06	0.00/0.47	0.00/1.15	0.00/0.01	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	45.16	5.36	41.17	0.14	34.59	28.80	3.57
Sparse82_07	0.00/0.04	0.27/2.10	0.00/0.04	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	32.73	3.66	35.85	0.64	31.43	35.06	4.27
Sparse82_08	0.00/0.00	0.09/2.59	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.42	15.25	0.38	0.04	31.41	26.69	10.18
Sparse82_09	0.00/0.05	0.00/0.82	0.06/0.06	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	13.57	5.79	16.82	0.99	25.39	31.97	4.53
Sparse82_10	0.00/0.03	0.00/1.86	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	35.45	12.64	41.41	1.00	13.99	29.50	1.50
RanReal240_01	7.53/8.75	0.61/1.38	1.07/1.52	0.07/0.19	0.18/0.35	0.00/0.18	0.02/0.08	116.19	24.61	138.34	155.57	155.92	127.16	149.09
RanReal240_02	14.29/15.12	9.59/10.29	9.78/10.44	8.73/8.89	0.00/9.15	8.78/8.97	8.73/8.85	95.06	31.06	152.77	184.44	159.56	158.20	150.75
RanReal240_03	4.92/7.07	1.40/2.64	1.38/2.05	0.05/0.18	0.25/0.58	0.00/0.27	0.01/0.09	142.89	8.67	127.14	186.06	164.11	139.26	169.95
RanReal240_04	6.10/7.76	1.27/2.27	1.16/1.96	0.10/0.25	0.21/0.52	0.04/0.19	0.00/0.12	102.72	22.69	147.86	159.70	175.00	137.87	118.58
RanReal240_05	5.37/6.59	1.13/1.91	1.12/2.12	0.01/0.19	0.33/0.61	0.06/0.31	0.00/0.12	112.52	12.77	133.26	155.02	179.73	136.48	132.33
RanReal240_06	5.49/6.98	1.01/1.65	0.90/1.60	0.02/0.18	0.17/0.41	0.08/0.27	0.00/0.12	113.64	34.23	146.87	174.51	187.59	126.61	155.99
RanReal240_07	4.80/5.57	0.68/1.36	0.79/1.44	0.04/0.15	0.09/0.47	0.02/0.23	0.00/0.13	98.82	48.81	171.68	160.67	195.33	144.35	144.96
RanReal240_08	6.66/8.10	0.69/1.68	1.22/1.96	0.00/0.20	0.24/0.54	0.11/0.35	0.00/0.14	114.33	20.45	137.72	145.94	193.88	122.79	124.22
RanReal240_09	5.16/6.44	0.49/1.08	0.51/1.17	0.01/0.21	0.22/0.42	0.07/0.23	0.00/0.10	145.30	43.70	158.65	194.36	171.09	136.24	141.69
RanReal240_10	6.84/7.44	1.39/2.44	1.55/2.25	0.05/0.20	0.33/0.58	0.02/0.25	0.00/0.09	112.01	12.19	129.47	183.49	145.39	162.77	134.20
RanReal240_11	6.18/6.80	0.65/1.53	1.05/1.61	0.04/0.17	0.16/0.40	0.04/0.18	0.00/0.08	122.37	65.47	161.82	157.98	155.12	136.25	129.91
RanReal240_12	5.33/6.51	0.53/1.18	0.76/1.18	0.04/0.18	0.15/0.48	0.00/0.25	0.00/0.16	147.93	96.77	163.87	186.71	157.43	123.13	132.03
RanReal240_13	7.00/8.22	0.98/1.52	0.81/1.45	0.00/0.15	0.18/0.48	0.10/0.31	0.00/0.10	107.97	30.38	150.83	173.46	171.44	127.25	131.34
RanReal240_14	8.41/10.12	0.73/1.40	0.90/1.54	0.00/0.15	0.09/0.36	0.06/0.24	0.00/0.14	104.70	12.33	131.31	147.56	164.87	112.27	135.87
RanReal240_15	5.98/7.42	0.98/2.13	1.29/2.02	0.05/0.22	0.36/0.67	0.17/0.41	0.00/0.17	147.90	11.13	135.10	167.88	188.75	154.19	131.83
RanReal240_16	5.62/7.13	1.28/1.93	0.88/1.60	0.00/0.20	0.13/0.39	0.07/0.21	0.01/0.18	134.19	20.82	153.67	165.22	181.82	130.93	89.11
RanReal240_17	4.09/4.83	0.79/1.80	0.60/1.77	0.18/0.31	0.37/0.59	0.06/0.31	0.00/0.16	97.91	31.38	146.50	171.75	179.25	149.06	171.03
RanReal240_18	7.31/8.16	0.77/2.02	1.14/2.02	0.09/0.20	0.09/0.51	0.04/0.20	0.00/0.12	134.95	20.60	133.33	154.60	169.52	167.15	152.83
RanReal240_19	5.44/6.51	1.24/2.31	1.48/2.06	0.01/0.16	0.20/0.56	0.11/0.30	0.00/0.09	118.97	10.95	132.71	173.65	179.99	140.21	98.09
RanReal240_20	4.93/5.85	0.99/1.62	0.89/1.69	0.01/0.20	0.15/0.39	0.00/0.18	0.01/0.11	139.31	33.98	151.07	188.64	175.52	112.33	115.38
RanReal480_01	11.83/13.04	1.91/3.09	1.62/2.69	0.25/0.35	0.45/0.61	0.00/0.26	0.04/0.24	269.94	91.12	333.56	391.12	419.98	301.25	340.60
RanReal480_02	9.23/9.88	2.20/3.24	1.65/2.69	0.24/0.43	0.50/0.73	0.05/0.20	0.00/0.30	264.17	78.83	289.46	403.38	414.24	293.36	355.59
RanReal480_03	9.72/10.24	2.23/3.36	2.14/3.03	0.25/0.43	0.49/0.77	0.00/0.18	0.05/0.30	202.03	53.59	299.70	420.24	427.12	331.81	352.06
RanReal480_04	9.54/10.28	1.86/3.45	2.33/3.21	0.17/0.48	0.36/0.67	0.00/0.21	0.04/0.28	221.32	93.30	302.04	383.19	428.09	332.13	399.51
RanReal480_05	10.27/11.39	2.69/3.51	2.13/3.04	0.09/0.35	0.47/0.74	0.10/0.24	0.00/0.25	235.49	57.96	288.63	417.55	448.15	325.05	335.84
RanReal480_06	10.26/10.97	2.11/3.03	2.02/2.79	0.24/0.45	0.41/0.61	0.00/0.22	0.17/0.35	222.09	103.33	310.30	393.37	428.17	300.77	390.06
RanReal480_07	11.28/12.49	1.89/2.71	2.18/2.60	0.08/0.31	0.24/0.63	0.00/0.18	0.06/0.21	282.40	101.60	312.15	395.01	423.31	342.07	390.23
RanReal480_08	9.56/10.42	2.16/3.20	2.19/2.95	0.22/0.40	0.35/0.68	0.00/0.21	0.14/0.28	239.63	88.35	328.43	388.45	398.32	356.39	380.64
RanReal480_09	11.59/13.53	1.87/3.25	2.07/2.86	0.29/0.43	0.37/0.69	0.00/0.26	0.07/0.29	237.49	78.39	315.08	376.21	421.61	350.35	383.67
RanReal480_10	10.41/11.38	2.34/3.30	2.16/3.01	0.23/0.35	0.54/0.77	0.12/0.27	0.00/0.42	241.70	83.45	290.39	383.71	428.06	300.43	328.85
RanReal480_11	9.62/10.78	1.37/3.00	1.84/2.61	0.17/0.41	0.33/0.59	0.00/0.17	0.07/0.29	227.89	94.12	312.81	383.91	421.15	320.08	383.65
RanReal480_12	10.04/10.63	1.79/2.61	1.87/2.72	0.15/0.43	0.46/0.72	0.00/0.19	0.06/0.29	271.59	89.29	309.44	397.43	409.11	270.86	347.32
RanReal480_13	8.87/9.86	1.37/2.85	2.02/2.70	0.18/0.33	0.30/0.55	0.00/0.18	0.05/0.27	208.64	138.35	313.17	404.23	390.76	327.70	330.84
RanReal480_14	12.10/13.49	2.00/3.29	1.89/2.88	0.20/0.33	0.28/0.58	0.00/0.18	0.02/0.23	276.22	73.53	311.36	405.04	419.87	319.14	352.07
RanReal480_15	9.99/10.91	1.51/2.68	1.97/2.82	0.18/0.42	0.38/0.60	0.00/0.17	0.07/0.30	235.52	118.86	319.34	384.96	418.77	313.91	387.02
RanReal480_16	11.34/12.08	1.76/2.98	1.60/2.51	0.12/0.29	0.15/0.51	0.00/0.14	0.06/0.26	227.75	104.65	322.10	345.11	419.82	301.16	392.78
RanReal480_17	10.32/11.43	1.46/2.63	1.65/2.62	0.14/0.33	0.29/0.59	0.00/0.18	0.00/0.22	229.01	109.71	287.42	386.16	411.52	320.59	360.93
RanReal480_18	8.98/9.73	1.43/2.98	1.66/2.96	0.06/0.25	0.23/0.55	0.02/0.18	0.00/0.21	227.34	106.08	292.13	368.20	431.30	278.26	320.60
RanReal480_19	9.20/10.10	1.92/2.75	1.81/2.55	0.07/0.23	0.18/0.55	0.00/0.17	0.06/0.26	249.78	75.37	303.77	400.78	435.58	306.80	380.69
RanReal480_20	11.03/11.90	2.05/2.70	1.46/2.47	0.05/0.34	0.43/0.63	0.00/0.19	0.07/0.25	201.47	124.30	344.80	385.30	420.08	339.09	353.52
Average	6.66/7.56	1.33/2.42	1.35/2.00	0.26/0.40	0.22/0.63	0.20/0.36	0.19/0.33	149.25	50.68	187.26	224.24	243.04	187.33	201.35

Table 13. Comparison of FITS with six state-of-the-art algorithms on the large CCP instances (RanReal960 and MDG).

	f_{best}							f_{avg}						
Name	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
RanReal960_01.30	1160656.62	1293137.12	1295744.06	1331323.28	1331996.73	**1337853.23**	1333878.00	1151895.17	1283288.50	1290241.93	1329095.10	1328415.40	**1335679.54**	1332712.78
RanReal960_02.30	1216485.40	1389456.81	1392709.68	1426870.24	1427037.60	1433071.63	**1434529.49**	1202250.66	1384366.48	1388067.85	1423903.72	1422960.00	1430808.90	**1433886.30**
RanReal960_03.30	1189286.33	1351373.72	1357014.34	1390084.19	1390367.98	**1395846.80**	1392101.18	1182522.35	1344238.06	1351727.20	1387508.88	1386753.98	**1394012.26**	1390924.21
RanReal960_04.30	1219509.95	1370623.88	1371234.23	1407607.65	1406916.02	1413478.82	**1414344.67**	1212573.34	1362351.05	1368276.84	1405972.04	1403198.73	1410438.07	**1412460.01**
RanReal960_05.30	1188052.95	1325522.71	1329439.01	1363405.33	1362240.30	**1370560.76**	1365975.96	1182095.82	1319156.72	1324439.72	1360883.59	1359314.24	**1367959.84**	1365612.74
RanReal960_06.30	1182929.35	1376617.75	1381506.52	1413074.62	1410969.99	**1417338.38**	1413476.58	1176910.55	1369018.83	1377468.14	1409803.60	1408670.37	**1415708.82**	1412750.08
RanReal960_07.30	1137446.74	1300578.66	1298460.40	1332205.25	1332748.04	**1339735.87**	1334504.35	1129944.15	1294554.26	1295020.62	1329503.04	1329170.21	**1337029.01**	1334263.93
RanReal960_08.30	1257726.73	1425314.30	1429319.26	1462280.35	1460407.11	**1466738.95**	1463737.39	1245787.22	1416728.40	1424868.01	1458524.76	1457531.52	**1464441.53**	1462602.72
RanReal960_09.30	1208989.95	1337671.23	1346330.61	1378445.37	1380206.02	**1385287.77**	1381577.32	1198263.71	1331119.48	1339752.81	1376305.91	1374713.94	**1382960.30**	1379280.98
RanReal960_10.30	1177285.83	1343535.01	1343597.60	1377646.33	1377009.66	**1384154.95**	1379905.83	1166762.04	1338126.35	1339495.54	1374311.23	1373790.24	**1381870.76**	1378772.67
RanReal960_01.40	891572.56	1004227.73	1004270.26	1034548.05	1032872.42	**1041148.42**	1035642.67	886644.65	993064.32	998964.36	1031462.28	1030512.99	**1038487.54**	1034626.82
RanReal960_02.40	935401.94	1085305.47	1085823.96	1108588.66	1109086.02	**1115789.54**	1110547.59	928159.94	1077230.13	1080504.37	1106671.47	1106469.64	**1113937.77**	1110074.36
RanReal960_03.40	921042.44	1044295.09	1052879.25	1081509.08	1080281.13	**1086488.77**	1083240.15	915374.21	1038465.67	1048423.29	1079553.82	1077244.61	**1085060.19**	1082948.77
RanReal960_04.40	937109.10	1059684.54	1070116.28	1096347.39	1096438.26	1100866.71	**1103897.12**	933821.48	1056026.26	1061981.40	1092785.74	1091633.05	1098759.10	**1101073.37**
RanReal960_05.40	906482.91	1029269.47	1027437.77	1056103.80	1057478.73	**1063682.56**	1059158.09	903666.87	1022127.35	1022735.81	1054445.69	1054478.20	**1062213.71**	1058478.46
RanReal960_06.40	897340.52	1069823.35	1074709.94	1096895.36	1099861.44	**1104590.22**	1100368.74	895656.35	1064068.17	1069231.66	1095721.54	1093932.85	**1102651.28**	1100064.66
RanReal960_07.40	867071.23	1004440.73	1005791.58	1034299.55	1035241.97	1041064.58	**1043376.95**	863943.79	999659.95	1001132.07	1032667.93	1031362.33	1038879.34	**1039933.23**
RanReal960_08.40	966280.23	1102339.42	1110622.81	1137464.75	1136567.63	**1142282.86**	1139865.05	955984.19	1096241.59	1106224.45	1136002.24	1133556.70	**1141171.70**	1138054.46
RanReal960_09.40	924938.50	1036368.74	1042262.50	1068288.24	1068892.92	**1076229.18**	1072116.14	921990.58	1028766.23	1034766.78	1066496.30	1066427.06	**1073798.91**	1071426.07
RanReal960_10.40	904198.86	1035597.43	1049773.93	1069556.59	1069985.94	**1077400.98**	1072919.65	899148.57	1031513.70	1040754.01	1067889.46	1067548.97	**1074400.36**	1071480.21
RanReal960_01.60	619038.29	700687.54	701813.93	726465.37	725912.84	**732096.63**	727690.58	612713.55	695227.62	697743.35	724244.42	723156.84	**730601.10**	727222.73
RanReal960_02.60	647084.93	748934.65	746043.75	770060.90	770477.46	**776289.95**	773921.97	641266.62	738650.06	742288.71	768413.74	768062.71	**775060.74**	772572.49
RanReal960_03.60	632413.61	724699.78	726919.69	753090.65	753094.36	**760248.25**	756677.95	628405.90	720449.00	723759.32	751419.90	750898.37	**758432.10**	755442.64
RanReal960_04.60	645023.75	738689.79	739337.91	762952.42	763837.98	**769112.25**	765253.27	642799.40	730470.51	734819.23	761387.83	760835.94	**767780.08**	764696.30
RanReal960_05.60	630215.36	716742.73	719501.92	741248.79	741932.85	**748581.43**	743715.56	624979.06	709477.81	714588.65	739443.77	738830.28	**746014.20**	743165.17
RanReal960_06.60	622892.55	734541.49	738718.50	761947.94	762260.95	**767679.61**	763029.06	618513.44	727566.01	732184.86	760369.56	759048.94	**765628.88**	761957.51
RanReal960_07.60	600734.82	700703.94	701940.30	723439.86	723786.02	**728827.33**	725993.23	597456.58	693719.70	697375.56	721167.69	720360.15	**727427.19**	725733.17
RanReal960_08.60	664742.63	762712.02	763064.60	789552.19	787775.07	**794363.93**	791334.42	658736.52	754557.15	757767.15	786344.58	785553.15	**792538.90**	790285.98
RanReal960_09.60	638160.86	723165.49	721784.75	747509.96	746710.78	**753943.93**	750858.18	634848.33	715936.10	718335.78	745313.38	744341.99	**751871.94**	749209.83
RanReal960_10.60	618919.01	719787.60	724270.46	748812.21	747565.74	**754666.01**	749883.45	616548.20	715319.84	720345.30	746701.78	745004.51	**752583.88**	748985.05
MDG-a_21	315160.00	362695.00	374102.00	379241.00	388647.00	**389269.00**	383164.00	312644.80	360104.20	373209.70	376249.80	387189.20	**389208.15**	382387.05
MDG-a_22	317268.00	328093.00	371724.00	380665.00	384487.00	**386354.00**	380817.00	316499.40	325587.15	368874.70	377418.75	384018.35	**386007.50**	380489.60
MDG-a_23	317128.00	360151.00	369080.00	375697.00	386154.00	**387267.00**	380630.00	315288.70	359117.55	364125.35	374941.20	385489.20	**387019.40**	379334.85
MDG-a_24	312916.00	332128.00	372676.00	378258.00	387095.00	**388423.00**	381783.00	310704.15	326971.20	368878.30	376540.75	386473.15	**388008.45**	380560.55
MDG-a_25	320905.00	370090.00	382627.00	389242.00	387121.00	**398108.00**	390623.00	319286.50	368531.30	374075.45	384685.70	395749.55	**397471.50**	386346.05
MDG-a_26	320433.00	374776.00	386554.00	392407.00	396756.00	**402135.00**	393222.00	318889.70	372769.80	378871.70	389113.50	400015.60	**401471.95**	392916.10
MDG-a_27	312370.00	324348.00	356856.00	375566.00	380243.00	**381896.00**	377617.00	310857.55	322750.50	355431.50	373113.95	379248.65	**381208.70**	376721.60
MDG-a_28	314843.00	330349.00	372681.00	378486.00	386018.00	**387594.00**	380748.00	313527.20	328473.25	368721.65	375546.50	385207.20	**387004.55**	379732.95
MDG-a_29	311435.00	357683.00	363780.00	377175.00	382108.00	**383937.00**	378239.00	310036.20	356208.50	361418.90	372663.75	381301.50	**383008.55**	377540.40
MDG-a_30	320558.00	343222.00	381167.00	389615.00	394817.00	**396678.00**	389452.00	318338.05	338243.30	375183.25	386318.15	394281.50	**396041.70**	389254.10
MDG-a_31	313035.00	332137.00	372007.00	376920.00	384718.00	**386587.00**	379407.00	311684.20	328719.90	367440.60	374627.70	384018.25	**385749.40**	378922.70
MDG-a_32	318574.00	345815.00	369324.00	382405.00	391693.00	**393098.00**	383333.00	317281.50	336858.50	362748.20	379790.40	391004.60	**392578.60**	382737.35
MDG-a_33	315383.00	357377.00	368858.00	377689.00	382576.00	**384291.00**	379047.00	313647.90	356012.45	366607.00	373096.60	381876.80	**383273.45**	376641.80
MDG-a_34	318048.00	344939.00	372511.00	384412.00	393135.00	**394676.00**	386544.00	316882.65	336415.90	370375.60	381472.65	392409.55	**394176.80**	385459.25
MDG-a_35	319670.00	334549.00	379479.00	385251.00	393384.00	**394687.00**	386347.00	318614.15	331542.20	378821.80	383302.15	392174.20	**394059.40**	385302.60
MDG-a_36	323168.00	344521.00	382566.00	394867.00	400271.00	**402087.00**	396368.00	322207.50	340160.80	374720.45	389332.30	400019.40	**401287.65**	394103.05
MDG-a_37	315054.00	344145.00	369934.00	384811.00	387006.00	**388411.00**	386195.00	313427.95	339023.50	363213.40	380906.40	386276.45	**387875.60**	385797.30
MDG-a_38	324112.00	367162.00	380306.00	387472.00	394264.00	**395865.00**	388712.00	319654.20	361682.75	376996.50	384752.55	393879.15	**395287.90**	387807.15
MDG-a_39	317407.00	364448.00	372179.00	381738.00	390008.00	**391415.00**	385975.00	315434.25	363026.80	368562.90	378840.90	389147.50	**391074.95**	383036.10
MDG-a_40	324559.00	348432.00	390438.00	392751.00	403662.00	**405184.00**	396321.00	322889.20	344485.80	389367.10	391517.70	403108.90	**405008.90**	395505.80
#Best	0	0	0	0	0	**46**	4	0	0	0	0	0	**46**	4
p-value	1.54e-12	1.54e-12	1.54e-12	1.14e-11	0.09	2.86e-9		1.54e-12	1.54e-12	1.54e-12	1.54e-12	0.16	2.86e-9	

Table 14. Comparison between FITS and the six state-of-the-art algorithms on the large CCP instances (RanReal960 and MDG).

	$Dev_{best}/Dev_{avg}(\%)$							t_{avg}						
Name	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS	GRASP	TS	GRASP+TS	IVNS	GVNS	SGVNS	FITS
RanReal960_01.30	13.24/13.90	3.34/4.08	3.15/3.56	0.49/0.65	0.44/0.71	0.00/0.44	0.30/0.38	643.28	659.97	746.31	918.26	811.54	856.21	915.53
RanReal960_02.30	15.20/16.19	3.14/3.50	2.92/3.24	0.53/0.74	0.52/0.81	0.10/0.52	0.00/0.04	589.33	507.99	730.14	797.61	842.95	835.98	904.23
RanReal960_03.30	14.80/15.28	3.19/3.70	2.78/3.16	0.41/0.60	0.39/0.65	0.00/0.39	0.27/0.35	495.00	716.62	700.85	883.41	860.63	829.85	836.81
RanReal960_04.30	13.78/14.27	3.09/3.68	3.05/3.26	0.48/0.59	0.53/0.79	0.06/0.53	0.00/0.13	634.29	355.14	693.39	919.96	832.99	822.40	929.49
RanReal960_05.30	13.32/13.75	3.29/3.75	3.00/3.37	0.52/0.71	0.61/0.82	0.00/0.61	0.33/0.36	472.15	639.50	738.05	804.13	883.85	840.42	957.25
RanReal960_06.30	16.54/16.96	2.87/3.41	2.53/2.81	0.30/0.53	0.45/0.61	0.00/0.45	0.27/0.32	405.17	678.17	713.02	857.85	801.30	834.76	935.81
RanReal960_07.30	15.10/15.66	2.92/3.37	3.08/3.34	0.56/0.76	0.52/0.79	0.00/0.52	0.39/0.41	475.25	422.65	713.50	875.58	850.22	816.60	894.98
RanReal960_08.30	14.25/15.06	2.82/3.41	2.55/2.85	0.30/0.56	0.43/0.63	0.00/0.43	0.20/0.28	365.27	707.84	796.19	804.15	822.22	831.19	948.47
RanReal960_09.30	12.73/13.50	3.44/3.91	2.81/3.29	0.49/0.65	0.37/0.76	0.00/0.37	0.27/0.43	575.87	602.61	710.25	815.06	863.40	812.81	884.89
RanReal960_10.30	14.95/15.71	2.93/3.33	2.93/3.23	0.47/0.71	0.52/0.75	0.00/0.52	0.31/0.39	355.56	758.63	692.78	878.66	850.02	792.14	897.68
RanReal960_01.40	14.37/14.84	3.55/4.62	3.54/4.05	0.63/0.93	0.79/1.02	0.00/0.79	0.53/0.63	560.30	527.60	735.84	877.73	847.87	832.62	913.51
RanReal960_02.40	16.17/16.82	2.73/3.46	2.69/3.16	0.65/0.82	0.60/0.84	0.00/0.60	0.47/0.51	446.57	450.16	765.06	824.06	876.23	830.26	938.57
RanReal960_03.40	15.23/15.75	3.88/4.42	3.09/3.50	0.46/0.64	0.57/0.85	0.00/0.57	0.30/0.33	455.66	480.50	753.60	905.64	850.33	816.43	829.18
RanReal960_04.40	15.11/15.41	4.01/4.34	3.06/3.80	0.68/1.01	0.68/1.11	0.27/0.68	0.00/0.26	370.04	620.19	729.88	800.47	904.06	761.94	931.93
RanReal960_05.40	14.78/15.04	3.24/3.91	3.41/3.85	0.71/0.87	0.58/0.87	0.00/0.58	0.43/0.49	546.96	599.63	652.52	826.35	857.74	803.47	882.95
RanReal960_06.40	18.76/18.92	3.15/3.67	2.71/3.20	0.70/0.80	0.43/0.96	0.00/0.43	0.38/0.41	463.89	638.31	701.67	876.40	835.26	840.43	795.61
RanReal960_07.40	16.90/17.20	3.73/4.19	3.60/4.05	0.87/1.03	0.78/1.15	0.22/0.78	0.00/0.33	415.17	567.45	770.42	882.22	884.53	871.15	891.65
RanReal960_08.40	15.41/16.31	3.50/4.03	2.77/3.16	0.42/0.55	0.50/0.76	0.00/0.50	0.21/0.37	693.30	562.11	680.24	807.24	851.96	827.89	884.17
RanReal960_09.40	14.06/14.33	3.70/4.41	3.16/3.85	0.74/0.90	0.68/0.91	0.00/0.68	0.38/0.45	524.53	586.74	712.38	818.16	878.54	847.56	873.44
RanReal960_10.40	16.08/16.54	3.88/4.26	2.56/3.40	0.73/0.88	0.69/0.91	0.00/0.69	0.42/0.55	641.03	484.57	679.54	872.90	863.94	860.78	943.53
RanReal960_01.60	15.44/16.31	4.29/5.04	4.14/4.69	0.77/1.07	0.84/1.22	0.00/0.84	0.60/0.67	495.14	312.03	715.60	789.02	901.12	907.89	901.04
RanReal960_02.60	16.64/17.39	3.52/4.85	3.90/4.38	0.80/1.01	0.75/1.06	0.00/0.75	0.31/0.48	550.63	576.58	623.63	826.25	874.07	873.06	887.61
RanReal960_03.60	16.81/17.34	4.68/5.24	4.38/4.80	0.94/1.16	0.94/1.23	0.00/0.94	0.47/0.63	427.92	440.24	668.37	808.81	852.82	873.26	923.81
RanReal960_04.60	16.13/16.42	3.96/5.02	3.87/4.46	0.80/1.00	0.69/1.08	0.00/0.69	0.50/0.57	493.87	537.67	671.38	853.85	830.99	839.14	819.90
RanReal960_05.60	15.81/16.51	4.25/5.22	3.88/4.54	0.98/1.22	0.89/1.30	0.00/0.89	0.65/0.72	460.62	462.68	663.19	831.31	895.34	868.22	889.58
RanReal960_06.60	18.86/19.43	4.32/5.23	3.77/4.62	0.75/0.95	0.71/1.12	0.00/0.71	0.61/0.75	464.96	443.56	625.41	766.03	872.94	836.76	864.12
RanReal960_07.60	17.58/18.02	3.86/4.82	3.69/4.32	0.74/1.05	0.69/1.16	0.00/0.69	0.39/0.42	382.58	287.23	684.55	918.06	909.03	878.59	875.95
RanReal960_08.60	16.32/17.07	3.98/5.01	3.94/4.61	0.61/1.01	0.83/1.11	0.00/0.83	0.38/0.51	616.02	344.11	750.21	851.60	911.42	859.76	870.01
RanReal960_09.60	15.36/15.80	4.08/5.04	4.27/4.72	0.85/1.14	0.96/1.27	0.00/0.96	0.41/0.63	454.24	454.70	685.44	770.73	849.73	868.30	900.64
RanReal960_10.60	17.99/18.30	4.62/5.21	4.03/4.55	0.78/1.06	0.94/1.28	0.00/0.94	0.63/0.75	403.71	367.32	617.20	855.35	898.95	857.38	870.89
MDG-a_21	19.04/19.68	6.83/7.49	3.90/4.13	2.58/3.34	0.16/0.53	0.00/0.16	1.57/1.77	1113.87	1996.39	1997.42	1938.05	1518.30	1867.83	1872.28
MDG-a_22	17.88/18.08	15.08/15.73	3.79/4.52	1.47/2.31	0.48/0.60	0.00/0.48	1.43/1.52	1067.71	1999.48	1996.70	1840.87	1806.64	1801.35	1931.75
MDG-a_23	18.11/18.59	7.00/7.27	4.70/5.98	2.99/3.18	0.29/0.46	0.00/0.29	1.71/2.05	1082.92	1997.29	1997.69	1868.53	1741.74	1839.26	1937.18
MDG-a_24	19.44/20.01	14.49/15.82	4.05/5.03	2.62/3.06	0.34/0.50	0.00/0.34	1.71/2.02	911.13	1997.83	1996.65	1892.93	1697.71	1830.54	1903.31
MDG-a_25	19.39/19.80	7.04/7.43	3.89/6.04	2.23/3.37	2.76/0.59	0.00/2.76	1.88/2.95	1074.60	1996.68	1997.50	1829.42	1663.95	1870.12	1824.79
MDG-a_26	20.32/20.70	6.80/7.30	3.87/5.78	2.42/3.24	1.34/0.53	0.00/1.34	2.22/2.29	1149.10	1996.75	1997.02	1923.60	1665.89	1790.90	1926.98
MDG-a_27	18.21/18.60	15.07/15.49	6.56/6.93	1.66/2.30	0.43/0.69	0.00/0.43	1.12/1.35	854.07	1999.38	1998.81	1790.23	1635.81	1890.33	1837.59
MDG-a_28	18.77/19.11	14.77/15.25	3.85/4.87	2.35/3.11	0.41/0.62	0.00/0.41	1.77/2.03	924.16	1999.27	1997.85	1920.76	1681.61	1879.05	1733.96
MDG-a_29	18.88/19.25	6.84/7.22	5.25/5.87	1.76/2.94	0.48/0.69	0.00/0.48	1.48/1.67	1026.38	1996.99	1997.68	1903.37	1835.42	1859.38	1913.92
MDG-a_30	19.19/19.75	13.48/14.73	3.91/5.42	1.78/2.61	0.47/0.60	0.00/0.47	1.82/1.87	1028.09	1998.93	1998.08	1820.66	1692.74	1793.70	1867.39
MDG-a_31	19.03/19.38	14.08/14.97	3.77/4.95	2.50/3.09	0.48/0.66	0.00/0.48	1.86/1.98	1302.25	1998.60	1997.17	1912.22	1691.17	1789.28	1703.24
MDG-a_32	18.96/19.29	12.03/14.31	6.05/7.72	2.72/3.39	0.36/0.53	0.00/0.36	2.48/2.64	575.50	1998.49	1997.63	1888.87	1663.50	1853.43	1913.13
MDG-a_33	17.93/18.38	7.00/7.36	4.02/4.60	1.72/2.91	0.45/0.63	0.00/0.45	1.36/1.99	1160.48	1997.66	1997.26	1908.54	1816.25	1908.27	1960.62
MDG-a_34	19.42/19.71	12.60/14.76	5.62/6.16	2.60/3.35	0.39/0.57	0.00/0.39	2.06/2.34	630.93	1998.63	1996.86	1812.31	1759.61	1840.43	1820.16
MDG-a_35	19.01/19.27	15.24/16.00	3.85/4.02	2.39/2.88	0.33/0.64	0.00/0.33	2.11/2.38	1051.55	1998.13	1996.08	1810.68	1661.62	1797.99	1895.51
MDG-a_36	19.63/19.87	14.32/15.40	4.85/6.81	1.80/3.17	0.45/0.51	0.00/0.45	1.42/1.99	899.31	1998.23	1999.26	1838.15	1700.47	1842.45	1715.11
MDG-a_37	18.89/19.31	11.40/12.72	4.76/6.49	0.93/1.93	0.36/0.55	0.00/0.36	0.57/0.67	752.54	1997.55	1999.13	1851.54	1752.65	1833.76	1935.18
MDG-a_38	18.13/19.25	7.25/8.63	3.93/4.77	2.12/2.81	0.40/0.50	0.00/0.40	1.81/2.04	1105.56	1996.57	1997.76	1863.12	1769.98	1804.21	1909.01
MDG-a_39	18.91/19.41	6.89/7.25	4.91/5.84	2.47/3.21	0.36/0.58	0.00/0.36	1.39/2.14	1018.71	1997.23	1998.49	1901.77	1732.95	1887.26	1874.83
MDG-a_40	19.90/20.31	14.01/14.98	3.64/3.90	3.07/3.37	0.38/0.51	0.00/0.38	2.19/2.39	1097.38	1998.28	1997.11	1735.43	1770.49	1819.48	1864.35
Average	16.94/17.44	6.60/7.36	3.77/4.47	1.27/1.70	0.61/0.80	0.01/0.61	0.89/1.07	694.09	1115.02	1221.46	1251.36	1202.49	1240.53	1282.67

Table 15. Comparison of FITS with the six state-of-art algorithms on handover minimization instances with $n \geq 100$, where the objective is to minimize the total cost function.

Name	GevPR-HMP		GQAP		BRKGA		IVNS		GVNS		SGVNS		FITS	
	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}
100_15_270001	19174.00	19174.00	**19000.00**	19114.80	**19000.00**	19533.20	**19000.00**	**19000.00**	**19000.00**	**19000.00**	**19000.00**	**19000.00**	**19000.00**	**19000.00**
100_15_270002	**22686.00**	**22686.00**	**22686.00**	22805.60	23288.00	23478.00	**22686.00**	**22686.00**	**22686.00**	**22686.00**	**22686.00**	**22686.00**	**22686.00**	**22686.00**
100_15_270003	**14558.00**	**14558.00**	**14558.00**	14568.80	14616.00	14655.20	**14558.00**	**14558.00**	15184.00	15184.00	15184.00	15184.00	**14558.00**	**14558.00**
100_15_270004	19762.00	19762.00	**19700.00**	19711.20	19882.00	20177.60	**19700.00**	**19700.00**	**19700.00**	**19700.00**	**19700.00**	**19700.00**	**19700.00**	**19700.00**
100_15_270005	22892.00	22892.00	**22746.00**	22885.20	23092.00	23430.80	**22746.00**	**22746.00**	22894.00	22894.00	22894.00	22894.00	**22746.00**	**22746.00**
100_25_270001	**36412.00**	36665.20	36448.00	36608.00	36752.00	37107.20	**36412.00**	**36412.00**	**36412.00**	**36412.00**	**36412.00**	**36412.00**	**36412.00**	**36412.00**
100_25_270002	39144.00	39199.20	**38608.00**	38677.20	39256.00	39515.60	**38608.00**	**38608.00**	**38608.00**	**38608.00**	**38608.00**	**38608.00**	**38608.00**	**38608.00**
100_25_270003	32966.00	33098.00	**32686.00**	32717.60	32708.00	33356.00	**32686.00**	**32686.00**	**32686.00**	**32686.00**	**32686.00**	**32686.00**	**32686.00**	**32686.00**
100_25_270004	35678.00	35801.20	**35322.00**	35433.20	35954.00	36068.40	**35322.00**	**35322.00**	**35322.00**	**35322.00**	**35322.00**	**35322.00**	**35322.00**	**35322.00**
100_25_270005	36906.00	36911.20	36878.00	36968.00	37100.00	37363.60	**36690.00**	**36690.00**	37092.00	37092.00	37092.00	37092.00	**36690.00**	**36690.00**
100_50_270001	**60922.00**	61074.00	61172.00	61234.80	61554.00	61845.20	**60922.00**	**60922.00**	**60922.00**	**60922.00**	**60922.00**	**60922.00**	**60922.00**	**60922.00**
100_50_270002	62046.00	62065.20	**62022.00**	62090.80	62524.00	62684.80	**62022.00**	**62022.00**	**62022.00**	**62022.00**	**62022.00**	**62022.00**	**62022.00**	**62022.00**
100_50_270003	54618.00	54707.60	**54596.00**	54661.60	55192.00	55390.80	**54596.00**	**54596.00**	**54596.00**	**54596.00**	**54596.00**	**54596.00**	**54596.00**	**54596.00**
100_50_270004	**57894.00**	57906.40	**57894.00**	57903.60	58208.00	58358.00	**57894.00**	**57894.00**	**57894.00**	**57894.00**	**57894.00**	**57894.00**	**57894.00**	**57894.00**
100_50_270005	61088.00	61283.20	61318.00	61318.00	62784.00	63042.00	**61080.00**	61092.00	**61080.00**	61102.10	**61080.00**	**61080.00**	**61080.00**	61240.10
200_15_270001	**81558.00**	81915.20	82834.00	84327.20	**81558.00**	81946.00	**81558.00**	**81558.00**	**81558.00**	**81558.00**	**81558.00**	**81558.00**	**81558.00**	**81558.00**
200_15_270002	89810.00	90949.60	90620.00	93462.00	90506.00	92372.40	**89492.00**	90407.40	**89492.00**	89863.60	**89492.00**	**89725.20**	**89492.00**	90201.50
200_15_270003	**79232.00**	**79232.00**	80980.00	81716.80	79548.00	79584.00	**79232.00**	79252.10	**79232.00**	**79232.00**	**79232.00**	**79232.00**	**79232.00**	79242.90
200_15_270004	**78324.00**	**78324.00**	80538.00	84737.60	80026.00	81019.20	**78324.00**	78549.10	**78324.00**	78376.10	**78324.00**	**78324.00**	**78324.00**	78376.10
200_15_270005	95998.00	96492.20	98826.00	100146.80	98830.00	99065.60	**95680.00**	96333.00	95734.00	**95734.00**	95734.00	**95734.00**	**95680.00**	96026.00
200_25_270001	**133168.00**	133674.00	138454.00	141961.60	140492.00	141938.80	**133168.00**	**133168.00**	133774.00	133774.00	133774.00	133774.00	**133168.00**	**133168.00**
200_25_270002	136038.00	137514.00	140066.00	141666.40	140690.00	141012.80	**133778.00**	133954.40	133876.00	133876.00	133876.00	133876.00	**133778.00**	**133844.60**
200_25_270003	139438.00	139962.40	144120.00	145647.60	143724.00	144409.60	**136782.00**	136801.50	**136782.00**	136802.70	**136782.00**	**136792.50**	**136782.00**	136870.10
200_25_270004	128554.00	129508.00	134054.00	136128.40	131786.00	133894.00	**128246.00**	128274.60	128472.00	128491.80	128472.00	128486.30	**128246.00**	**128246.00**
200_25_270005	148402.00	149298.80	154260.00	157307.20	152934.00	154275.20	**147844.00**	**147844.00**	148050.00	148050.00	148050.00	148050.00	**147844.00**	**147844.00**
200_50_270001	221550.00	221821.60	223096.00	223556.80	223098.00	224034.00	**215388.00**	215687.20	**215388.00**	215459.40	**215388.00**	**215449.50**	**215388.00**	215722.70
200_50_270002	218254.00	218761.20	219910.00	221346.80	219834.00	221131.20	**212798.00**	213005.10	213178.00	213251.80	213178.00	213239.50	**212798.00**	**212946.20**
200_50_270003	221500.00	222315.60	222404.00	223175.20	221110.00	221568.00	**214364.00**	214678.60	**214364.00**	214451.10	**214364.00**	**214394.80**	**214364.00**	214834.70
200_50_270004	212044.00	212626.80	212544.00	213880.80	213170.00	213509.20	**206476.00**	206985.40	207254.00	270343.60	207254.00	207301.70	**206476.00**	**206692.90**
200_50_270005	231890.00	232938.00	236136.00	238228.80	237156.00	237939.60	229918.00	230107.30	230484.00	230630.30	230484.00	230543.60	**229900.00**	**229980.40**
400_15_270001	372694.00	375429.20	456158.00	475012.00	375650.00	378718.00	**369048.00**	373670.80	**369048.00**	370847.20	**369048.00**	**370787.50**	**369048.00**	371752.40
400_15_270002	370274.00	373304.80	460232.00	465669.20	383096.00	386282.80	**365878.00**	370147.60	366062.00	369675.00	366062.00	**368351.10**	**365878.00**	369214.50
400_15_270003	358684.00	360152.80	448830.00	456513.60	366314.00	369552.80	**352588.00**	358234.90	353078.00	355399.30	353078.00	355002.50	**352588.00**	**354776.20**
400_15_270004	334430.00	336826.40	406834.00	447753.20	346282.00	349110.00	**331888.00**	339014.80	**331888.00**	**335504.80**	**331888.00**	337685.30	**331888.00**	337899.40
400_15_270005	361904.00	365974.00	457274.00	476186.00	377094.00	380204.40	**360422.00**	364127.30	360714.00	364176.50	360714.00	362735.10	**360422.00**	**362651.80**
400_25_270001	570852.00	571930.00	663908.00	694715.60	579130.00	584584.80	545662.00	547000.40	545674.00	547537.20	**545228.00**	**546479.10**	545540.00	547541.30
400_25_270002	544568.00	547953.60	658440.00	679772.40	554840.00	560692.00	**528470.00**	530441.40	**528470.00**	530204.70	**528470.00**	529495.00	**528470.00**	**529055.90**
400_25_270003	548000.00	554179.60	667982.00	680754.80	553162.00	555433.20	525016.00	527990.20	525016.00	527434.30	525016.00	526789.10	**524678.00**	**526193.40**
400_25_270004	501750.00	504474.40	607672.00	633011.20	516416.00	517828.00	481660.00	482858.10	481920.00	482922.10	481994.00	483238.40	**481436.00**	**481994.10**
400_25_270005	556044.00	561315.20	679848.00	703440.00	585070.00	589444.00	**548100.00**	**548744.40**	548450.00	549737.80	548450.00	549170.50	**548100.00**	549106.70
400_50_270001	851412.00	854656.00	951882.00	957526.00	879438.00	881239.60	824868.00	825682.50	826252.00	826962.10	825956.00	826809.80	**824656.00**	**825547.00**
400_50_270002	845496.00	848217.60	949562.00	953687.60	874226.00	877662.00	822562.00	824563.50	824468.00	826207.90	823762.00	825441.00	**822336.00**	**824048.70**
400_50_270003	819242.00	824118.80	919140.00	927517.60	843242.00	850384.80	801722.00	802977.20	802936.00	804269.90	801824.00	803537.30	**801472.00**	**802646.30**
400_50_270004	774564.00	777953.60	878912.00	885893.20	806690.00	810186.80	760514.00	761484.60	**760044.00**	762079.10	**760044.00**	**761455.70**	760232.00	763379.60
400_50_270005	854726.00	857133.20	940358.00	950535.60	882060.00	885058.80	**828398.00**	829882.60	828902.00	829946.80	828450.00	**829578.20**	**828398.00**	830872.10
#Best	9	4	11	0	2	0	37	18	23	15	**24**	**25**	**43**	30
p-value	1.97e-9	1.68e-8	5.51e-9	1.97e-11	5.47e-11	1.97e-11	0.00	0.02	1.19e-5	0.08	7.44e-5	0.38		

Table 16. Comparison between FITS and six state-of-the-art algorithms on the handover minimization instances with $n \geq 100$.

Name	$Dev_{best}/Dev_{avg}(\%)$							t_{avg}			
	GevPR-HMP	GQAP	BRKGA	IVNS	GVNS	SGVNS	FITS	IVNS	GVNS	SGVNS	FITS
100_15_270001	0.92/0.92	0.00/0.60	0.00/2.81	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	1.03	0.24	0.28	3.84
100_15_270002	0.00/0.00	0.00/0.53	2.65/3.49	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	1.54	0.39	0.49	25.06
100_15_270003	0.00/0.00	0.00/0.07	0.40/0.67	0.00/0.00	4.30/4.30	4.30/4.30	0.00/0.00	0.08	0.06	0.10	0.14
100_15_270004	0.31/0.31	0.00/0.06	0.92/2.42	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.30	0.13	0.11	2.03
100_15_270005	0.64/0.64	0.00/0.61	1.52/3.01	0.00/0.00	0.65/0.65	0.65/0.65	0.00/0.00	1.62	0.24	0.36	11.35
100_25_270001	0.00/0.70	0.10/0.54	0.93/1.91	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	1.54	0.64	0.82	4.38
100_25_270002	1.39/1.53	0.00/0.18	1.68/2.35	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	1.62	1.33	0.66	5.57
100_25_270003	0.86/1.26	0.00/0.10	0.07/2.05	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	5.42	1.40	0.86	12.78
100_25_270004	1.01/1.36	0.00/0.31	1.79/2.11	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	6.18	0.13	0.21	19.05
100_25_270005	0.59/0.60	0.51/0.76	1.12/1.84	0.00/0.00	1.10/1.10	1.10/1.10	0.00/0.00	6.89	0.15	0.23	14.61
100_50_270001	0.00/0.25	0.41/0.51	1.04/1.52	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	9.19	0.45	0.48	25.75
100_50_270002	0.04/0.07	0.00/0.11	0.81/1.07	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	10.47	0.87	0.71	7.41
100_50_270003	0.04/0.20	0.00/0.12	1.09/1.46	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	7.61	0.68	0.59	8.70
100_50_270004	0.00/0.02	0.00/0.02	0.54/0.80	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	5.63	1.26	0.05	14.18
100_50_270005	0.01/0.33	0.39/0.39	2.79/3.21	0.00/0.02	0.00/0.04	0.00/0.00	0.00/0.26	9.45	1.42	0.43	42.71
200_15_270001	0.00/0.44	1.56/3.40	0.00/0.48	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	25.19	2.23	2.77	67.35
200_15_270002	0.36/1.63	1.26/4.44	1.13/3.22	0.00/1.02	0.00/0.42	0.00/0.26	0.00/0.79	70.30	20.16	55.81	116.10
200_15_270003	0.00/0.00	2.21/3.14	0.40/0.44	0.00/0.03	0.00/0.00	0.00/0.00	0.00/0.01	38.27	2.08	1.85	13.10
200_15_270004	0.00/0.00	2.83/8.19	2.17/3.44	0.00/0.29	0.00/0.07	0.00/0.00	0.00/0.07	48.99	13.94	20.44	43.07
200_15_270005	0.33/0.85	3.29/4.67	3.29/3.54	0.00/0.68	0.06/0.06	0.06/0.06	0.00/0.36	11.33	29.97	31.91	40.97
200_25_270001	0.00/0.38	3.97/6.60	5.50/6.59	0.00/0.00	0.46/0.46	0.46/0.46	0.00/0.00	41.78	24.60	54.73	83.74
200_25_270002	1.69/2.79	4.70/5.90	5.17/5.41	0.00/0.13	0.07/0.07	0.07/0.07	0.00/0.05	87.71	47.07	19.05	91.89
200_25_270003	1.94/2.33	5.36/6.48	5.08/5.58	0.00/0.01	0.00/21.95	0.00/0.01	0.00/0.06	53.89	86.94	51.56	103.30
200_25_270004	0.24/0.98	4.53/6.15	2.76/4.40	0.00/0.02	0.18/0.19	0.18/0.19	0.00/0.00	49.99	35.98	19.00	92.27
200_25_270005	0.38/0.98	4.34/6.40	3.44/4.35	0.00/0.00	0.14/0.14	0.14/0.14	0.00/0.00	9.64	6.86	5.17	78.63
200_50_270001	2.86/2.99	3.58/3.79	3.58/4.01	0.00/0.14	0.00/0.03	0.00/0.03	0.00/0.16	76.15	62.88	65.42	87.34
200_50_270002	2.56/2.80	3.34/4.02	3.31/3.92	0.00/0.10	0.18/0.21	0.18/0.21	0.00/0.07	69.51	25.08	14.92	105.89
200_50_270003	3.33/3.71	3.75/4.11	3.15/3.36	0.00/0.15	0.00/0.04	0.00/0.01	0.00/0.22	69.87	46.94	83.88	101.50
200_50_270004	2.70/2.98	2.94/3.59	3.24/3.41	0.00/0.25	0.38/30.93	0.38/0.40	0.00/0.11	81.66	64.90	55.80	111.74
200_50_270005	0.87/1.32	2.71/3.62	3.16/3.50	0.01/0.09	0.25/0.32	0.25/0.28	0.00/0.03	86.92	65.05	68.95	131.04
400_15_270001	0.99/1.73	23.60/28.71	1.79/2.62	0.00/1.25	0.00/0.49	0.00/0.47	0.00/0.73	184.94	114.10	154.72	218.52
400_15_270002	1.20/2.03	25.79/27.27	4.71/5.58	0.00/1.17	0.05/1.04	0.05/0.68	0.00/0.91	197.33	195.42	162.95	230.06
400_15_270003	1.73/2.15	27.30/29.48	3.89/4.81	0.00/1.60	0.14/0.80	0.14/0.68	0.00/0.62	156.99	73.55	125.33	253.96
400_15_270004	0.77/1.49	22.58/34.91	4.34/5.19	0.00/2.15	0.00/1.09	0.00/1.75	0.00/1.81	245.49	123.96	171.49	226.90
400_15_270005	0.41/1.54	26.87/32.12	4.63/5.49	0.00/1.03	0.08/1.04	0.08/0.64	0.00/0.62	234.97	136.54	243.64	271.41
400_25_270001	4.70/4.90	21.77/27.42	6.22/7.22	0.08/0.33	0.08/0.42	0.00/0.23	0.06/0.42	239.15	124.71	127.27	280.77
400_25_270002	3.05/3.69	24.59/28.63	4.99/6.10	0.00/0.37	0.00/0.33	0.00/0.19	0.00/0.11	206.04	109.35	114.10	263.82
400_25_270003	4.45/5.62	27.31/29.75	5.43/5.86	0.06/0.63	0.06/0.53	0.06/0.40	0.00/0.29	213.34	176.03	182.30	259.13
400_25_270004	4.22/4.79	26.22/31.48	7.27/7.56	0.05/0.30	0.10/0.31	0.12/0.37	0.00/0.12	209.56	108.17	142.02	274.96
400_25_270005	1.45/2.41	24.04/28.34	6.75/7.54	0.00/0.12	0.06/0.30	0.06/0.20	0.00/0.18	243.63	192.24	215.43	295.26
400_50_270001	3.24/3.64	15.43/16.11	6.64/6.86	0.03/0.12	0.19/0.28	0.16/0.26	0.00/0.11	260.81	214.55	244.55	255.22
400_50_270002	2.82/3.15	15.47/15.97	6.31/6.73	0.03/0.27	0.26/0.47	0.17/0.38	0.00/0.21	284.52	228.68	229.22	280.60
400_50_270003	2.22/2.83	14.68/15.73	5.21/6.10	0.03/0.19	0.18/0.35	0.04/0.26	0.00/0.15	302.17	197.58	258.58	267.40
400_50_270004	1.91/2.36	15.64/16.56	6.14/6.60	0.06/0.19	0.00/0.27	0.00/0.19	0.02/0.44	265.09	228.35	230.35	294.04
400_50_270005	3.18/3.47	13.52/14.74	6.48/6.84	0.00/0.18	0.06/0.19	0.01/0.14	0.00/0.30	267.93	210.97	249.88	284.11
Average	1.32/1.74	8.37/10.15	3.19/3.94	0.01/0.28	0.20/1.53	0.19/0.33	0.00/0.20	97.82	66.18	75.77	120.48

Table 17
Comparison between FITS and MA on two sets of CCP instances (RanReal and DB).

Name	FITS			MA		
	f_{best}	f_{avg}	t_{avg}(s)	f_{best}	f_{avg}	t_{avg}(s)
Sparse82_01	1342.17	1342.17	7.70	1342.17	1342.17	9.81
Sparse82_02	1306.64	1306.64	15.92	1306.64	1306.64	14.24
Sparse82_03	1353.94	1353.94	3.88	1353.94	1353.94	4.86
Sparse82_04	1291.22	1291.22	45.02	1291.22	1291.22	43.75
Sparse82_05	1352.35	1352.35	1.13	1352.35	1352.35	1.26
Sparse82_06	1354.61	1354.61	1.63	1354.61	1354.61	1.92
Sparse82_07	1266.94	1266.94	4.13	1266.94	1266.94	4.42
Sparse82_08	1393.02	1393.02	3.59	1393.02	1393.02	3.36
Sparse82_09	1294.12	1294.12	2.75	1294.12	1294.12	2.60
Sparse82_10	1356.98	1356.98	0.89	1356.98	1356.98	0.99
RanReal240_01	225003.53	224926.96	2348.59	225003.53	**224937.30**	2348.57
RanReal240_02	204624.36	204525.77	1580.11	204624.36	**204553.10**	1948.50
RanReal240_03	198976.88	198941.44	1572.29	**199079.37**	**198956.17**	2276.32
RanReal240_04	225667.77	225542.39	1151.94	**225683.17**	**225547.69**	2524.26
RanReal240_05	195621.68	195480.29	1631.35	195621.68	**195481.00**	1943.62
RanReal240_06	216747.32	216697.36	1667.33	**216747.39**	**216699.09**	1945.37
RanReal240_07	**209316.85**	**209245.15**	1562.66	209305.70	209213.53	1642.42
RanReal240_08	205246.82	**205175.77**	1768.87	205246.82	205134.15	2146.29
RanReal240_09	209186.90	209145.93	2038.81	209186.90	**209152.19**	2123.53
RanReal240_10	193062.60	192969.92	1856.91	193062.60	**192972.14**	1945.36
RanReal240_11	204722.75	204627.27	1975.32	204722.75	**204635.30**	1990.59
RanReal240_12	201117.11	201022.63	1197.68	201117.11	**201037.08**	2295.96
RanReal240_13	202338.43	202312.16	1218.02	202338.43	**202312.50**	1613.35
RanReal240_14	228870.89	**228657.39**	2027.09	228870.89	228601.66	1278.33
RanReal240_15	191263.28	191149.81	1653.79	191263.28	**191177.81**	2281.96
RanReal240_16	204072.57	203960.10	1710.90	**204081.46**	**203971.34**	1986.64
RanReal240_17	195623.04	195532.19	1791.09	**195638.95**	**195544.67**	2031.51
RanReal240_18	195127.57	195035.21	1887.10	**195167.12**	**195070.91**	2355.17
RanReal240_19	199307.33	**199191.03**	1317.83	199307.33	199175.59	1848.30
RanReal240_20	212321.06	212234.67	1798.98	**212322.48**	**212236.65**	2139.52
RanReal480_01	556259.42	555579.98	3785.15	**556401.65**	**555586.79**	5465.92
RanReal480_02	511504.92	510798.1	4735.30	**511832.51**	**511033.52**	5928.92
RanReal480_03	497973.87	497377.76	4213.93	**498197.75**	**497467.34**	6199.20
RanReal480_04	523131.64	**522494.45**	4786.27	**523428.57**	522493.58	6273.48
RanReal480_05	484701.85	483989.96	3463.80	**484856.17**	**484089.84**	5828.84
RanReal480_06	534897.14	534283.07	4402.99	**535110.60**	**534324.91**	5859.55
RanReal480_07	546052.12	545744.17	4340.06	**546671.53**	**545774.12**	5779.21
RanReal480_08	533042.97	532676.82	4098.46	**533593.65**	**532710.69**	5438.86
RanReal480_09	557169.08	556629.74	4028.24	**557558.00**	**556641.92**	5221.96
RanReal480_10	**520714.86**	**519959.24**	3871.66	520657.09	519841.12	5395.04
RanReal480_11	**524479.86**	**524065.89**	4859.36	524356.65	524044.88	5087.50
RanReal480_12	502547.16	502141.47	4282.61	**502888.85**	**502143.97**	5675.84
RanReal480_13	535381.87	535028.27	3833.55	**536024.76**	**535078.71**	5648.80
RanReal480_14	514731.74	514378.83	4568.84	**515137.95**	**514415.58**	5395.97
RanReal480_15	518140.43	517471.49	4449.48	**518686.16**	**517480.73**	5247.89
RanReal480_16	550260.76	549765.36	4074.76	**550566.81**	**549796.27**	5959.42
RanReal480_17	538339.63	537969.48	4540.85	**538584.62**	**537997.22**	5623.36
RanReal480_18	526419.74	525880.41	4768.13	**527089.59**	**526065.80**	5575.99
RanReal480_19	522972.89	522598.39	5210.14	**523302.33**	**522624.73**	5832.21
RanReal480_20	519148.25	518650.30	4097.54	**519626.64**	**518717.70**	6164.32
#Best	3	7		**25**	**33**	
p-value	3.22e-5	3.94e-5				

Table 18
Comparison between FITS and MA on the large handover minimization instances with $n \geq 100$. For direct comparisons, we present the results in the minimization form.

Name	FITS			MA		
	f_{best}	f_{avg}	t_{avg}(s)	f_{best}	f_{avg}	t_{avg}(s)
100_15_270001	19000	19000.00	7.59	19000	19000.00	4.52
100_15_270002	22686	22686.00	143.84	22686	22686.00	70.77
100_15_270003	14558	14558.00	0.20	14558	14558.00	0.08
100_15_270004	19700	19700.00	3.34	19700	19700.00	0.84
100_15_270005	22746	22746.00	15.68	22746	22746.00	4.78
100_25_270001	36412	36412.00	5.73	36412	36412.00	2.53
100_25_270002	38608	38608.00	72.92	38608	38608.00	11.48
100_25_270003	32686	32686.00	84.02	32686	32686.00	28.65
100_25_270004	35322	35322.00	20.94	35322	35322.00	10.55
100_25_270005	36690	36690.00	9.81	36690	36690.00	6.19
100_50_270001	60922	60922.00	103.80	60922	60922.00	21.03
100_50_270002	62022	62022.00	42.17	62022	62022.00	8.20
100_50_270003	54596	54596.00	94.73	54596	54596.00	4.43
100_50_270004	57894	57894.00	121.33	57894	57894.00	55.42
100_50_270005	61080	61099.90	620.30	61080	**61080.00**	158.34
200_15_270001	81558	81558.00	544.44	81558	81558.00	348.50
200_15_270002	89492	90134.50	966.93	89492	**89653.40**	628.38
200_15_270003	79232	79232.00	165.95	79232	79232.00	56.59
200_15_270004	78324	78372.30	345.14	78324	**78324.00**	344.47
200_15_270005	95680	96023.90	398.99	95680	**95680.00**	115.01
200_25_270001	133168	133168.00	512.23	133168	133168.00	206.01
200_25_270002	133778	**133818.20**	742.93	133778	133829.00	686.27
200_25_270003	136782	136828.10	885.09	136782	**136796.70**	870.38
200_25_270004	128246	128246.00	1341.78	128246	128246.00	767.18
200_25_270005	147844	147844.00	132.42	147844	147844.00	189.92
200_50_270001	215388	**215560.80**	962.60	215388	215616.40	802.60
200_50_270002	212798	**212848.70**	731.01	212798	212858.40	1055.20
200_50_270003	214364	214555.20	1237.59	214364	**214550.10**	1333.02
200_50_270004	206476	**206574.40**	1097.72	206476	206595.90	797.46
200_50_270005	229900	**229936.70**	757.27	229900	229945.20	1044.29
400_15_270001	369048	371437.90	1659.67	369048	**369317.60**	1904.87
400_15_270002	365878	368928.70	2880.39	365878	**366497.10**	1558.38
400_15_270003	352588	353787.40	1394.43	352588	**352600.70**	1101.44
400_15_270004	331888	336731.60	1517.02	331888	**332017.50**	1631.73
400_15_270005	360422	361852.50	2456.41	360422	**360949.90**	2080.04
400_25_270001	545556	546421.40	2447.34	**545540**	**546420.60**	2447.35
400_25_270002	528470	**528792.60**	2686.55	528470	528862.50	2615.13
400_25_270003	524678	525339.10	2455.44	524678	**525226.00**	2204.49
400_25_270004	481436	**481925.10**	2715.72	481436	481974.50	2351.65
400_25_270005	548100	**548123.60**	2104.69	548100	548199.70	1808.02
400_50_270001	824604	825539.90	2749.00	**824518**	**825488.10**	3086.45
400_50_270002	822336	823753.30	3002.52	822336	**823699.90**	3006.55
400_50_270003	801432	802627.00	2814.66	**801346**	**802484.30**	3022.16
400_50_270004	760232	762474.40	3003.92	760232	**761699.40**	2924.69
400_50_270005	828398	829700.80	2739.78	828398	**829226.00**	3801.92
#Best	0	8		**3**	**19**	
p-value	0.08	0.05				

Table 19
Comparison between FITS and its two underlying components FLS and InfLS on the RanReal benchmark set.

Name	FITS		FLS		InfLS	
	f_{best}	f_{avg}	f_{best}	f_{avg}	f_{best}	f_{avg}
RanReal240_01	**224941.48**	**224802.06**	224941.28	224700.99	224673.38	224351.31
RanReal240_02	**204624.36**	**204359.38**	**204624.36**	204273.94	204191.72	203800.45
RanReal240_03	**198954.91**	**198799.84**	198896.78	198715.24	198612.31	198358.12
RanReal240_04	**225627.16**	**225364.97**	**225627.16**	225242.26	225382.26	224953.72
RanReal240_05	**195564.48**	**195320.28**	195472.49	195298.72	195165.72	194470.36
RanReal240_06	**216747.32**	**216487.02**	216736.00	216422.13	216244.78	215840.77
RanReal240_07	**209305.70**	**209029.23**	209286.63	208984.08	208904.93	208538.21
RanReal240_08	**205246.82**	204961.05	**205246.82**	**204971.43**	204867.72	204447.79
RanReal240_09	**209159.16**	**208952.48**	209073.60	208877.53	208782.77	208445.36
RanReal240_10	192986.21	**192811.13**	**193044.16**	192762.94	192765.27	192429.05
RanReal240_11	**204722.75**	**204559.39**	**204722.75**	204491.34	204458.26	204014.37
RanReal240_12	**201117.11**	200797.67	201074.54	**200874.41**	200687.09	200360.62
RanReal240_13	**202335.99**	**202139.57**	**202335.99**	202107.26	201879.46	201616.58
RanReal240_14	**228870.89**	**228554.78**	228844.44	228457.67	228316.81	227946.95
RanReal240_15	**191255.87**	**190923.28**	191202.77	190903.62	190761.09	190372.67
RanReal240_16	**204054.99**	**203710.39**	204019.08	203706.22	203513.73	203196.32
RanReal240_17	**195561.36**	**195243.32**	195447.87	195225.20	195223.74	194857.78
RanReal240_18	**195100.39**	**194872.13**	**195100.39**	194864.12	194689.78	194398.32
RanReal240_19	**199225.98**	**199040.43**	199216.94	198997.54	199006.08	198450.68
RanReal240_20	**212268.52**	**212049.85**	**212268.52**	212015.41	211849.93	211537.55
RanReal480_01	**555489.92**	**554376.54**	555300.10	554228.16	552864.61	551299.72
RanReal480_02	**511280.50**	**509757.15**	510466.38	509449.52	509294.32	508336.14
RanReal480_03	**497295.19**	**496059.50**	496683.63	495938.74	495664.06	494292.73
RanReal480_04	**522305.16**	**521062.13**	522000.18	520565.30	520451.39	519325.61
RanReal480_05	**484084.66**	**482867.74**	483514.31	482241.30	481725.02	480777.24
RanReal480_06	**533991.27**	**533036.36**	533843.95	532409.06	531163.12	530249.35
RanReal480_07	**545470.73**	**544651.12**	545209.27	544128.41	542724.00	541454.69
RanReal480_08	**532417.42**	**531667.91**	532357.71	531527.82	529288.64	528038.04
RanReal480_09	**556868.85**	**555634.40**	556552.85	555295.47	554085.91	553165.52
RanReal480_10	**520257.54**	518071.71	519857.07	**518083.48**	517835.50	516098.53
RanReal480_11	**523991.29**	**522816.94**	523587.47	522613.58	521964.67	520488.83
RanReal480_12	**501915.56**	**500776.79**	501683.69	500647.73	500922.44	499723.69
RanReal480_13	**535025.51**	**533823.79**	534867.67	533594.29	531960.39	530949.44
RanReal480_14	**514107.62**	**513053.25**	514039.43	512915.41	512043.79	510844.33
RanReal480_15	517205.02	**516018.38**	**518140.43**	515941.80	514508.48	513428.89
RanReal480_16	549552.63	**548462.13**	**549824.00**	548211.77	547509.72	546087.39
RanReal480_17	**537924.55**	**536745.39**	537702.09	536639.04	534980.65	533909.81
RanReal480_18	525822.76	**524712.42**	**525926.74**	524565.69	523702.51	522359.71
RanReal480_19	**522316.22**	**521267.22**	522291.27	520957.84	520173.80	519077.31
RanReal480_20	518349.10	517430.77	**518645.84**	**517467.88**	515626.91	515025.41
#Best	**35**	**36**	12	4	0	0
p-value			6.23e-5	4.20e-7	2.54e-10	2.54e-10

Table 20
Comparison between FITS and its variation FITS_FI (using the first-improvement strategy) on the RanReal benchmark.

Name	FITS			FITS_FI		
	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}
RanReal240_01	**224941.48**	**224802.06**	149.09	224182.68	223791.06	97.10
RanReal240_02	**204624.36**	**204359.38**	150.75	204024.55	203748.76	115.86
RanReal240_03	**198954.91**	**198799.84**	169.95	198312.80	198070.31	107.81
RanReal240_04	**225627.16**	**225364.97**	118.58	224995.76	224412.34	116.39
RanReal240_05	**195564.48**	**195320.28**	132.33	195102.26	194867.72	117.86
RanReal240_06	**216747.32**	**216487.02**	155.99	215788.46	215440.78	99.28
RanReal240_07	**209305.70**	**209029.23**	144.96	208530.22	208114.22	96.03
RanReal240_08	**205246.82**	**204961.05**	124.22	204428.30	204277.73	108.99
RanReal240_09	**209159.16**	**208952.48**	141.69	208338.57	208175.46	116.96
RanReal240_10	**192986.21**	**192811.13**	134.20	192232.44	192071.19	120.39
RanReal240_11	**204722.75**	**204559.39**	129.91	204217.59	204001.29	89.54
RanReal240_12	**201117.11**	**200797.67**	132.03	200498.08	200344.22	78.75
RanReal240_13	**202335.99**	**202139.57**	131.34	201813.49	201615.62	69.13
RanReal240_14	**228870.89**	**228554.78**	135.87	228231.53	228117.87	92.49
RanReal240_15	**191255.87**	**190923.28**	131.83	190537.12	190180.38	81.39
RanReal240_16	**204054.99**	**203710.39**	89.11	203428.14	203258.65	100.98
RanReal240_17	**195561.36**	**195243.32**	171.03	194568.09	194356.29	64.85
RanReal240_18	**195100.39**	**194872.13**	152.83	194458.12	194254.79	89.27
RanReal240_19	**199225.98**	**199040.43**	98.09	198479.56	198337.34	102.39
RanReal240_20	**212268.52**	**212049.85**	115.38	211474.47	211194.85	66.20
RanReal480_01	**555489.92**	**554376.54**	340.60	551254.15	550147.26	236.79
RanReal480_02	**511280.50**	**509757.15**	355.59	507215.91	506193.99	251.43
RanReal480_03	**497295.19**	**496059.50**	352.06	494017.02	493425.88	251.20
RanReal480_04	**522305.16**	**521062.13**	399.51	516643.64	515925.45	112.97
RanReal480_05	**484084.66**	**482867.74**	335.84	480982.54	480210.46	240.39
RanReal480_06	**533991.27**	**533036.36**	390.06	530142.87	529021.63	230.79
RanReal480_07	**545470.73**	**544651.12**	390.23	541115.04	539994.76	258.65
RanReal480_08	**532417.42**	**531667.91**	380.64	529566.14	527736.78	223.86
RanReal480_09	**556868.85**	**555634.40**	383.67	551863.24	551045.16	204.35
RanReal480_10	**520257.54**	**518071.71**	328.85	517484.26	515898.39	272.95
RanReal480_11	**523991.29**	**522816.94**	383.65	520367.63	519570.84	204.96
RanReal480_12	**501915.56**	**500776.79**	347.32	498568.75	498030.75	227.79
RanReal480_13	**535025.51**	**533823.79**	330.84	530707.31	529726.48	147.77
RanReal480_14	**514107.62**	**513053.25**	352.07	510789.76	509613.70	252.60
RanReal480_15	**517205.02**	**516018.38**	387.02	514108.83	513515.25	195.55
RanReal480_16	**549552.63**	**548462.13**	392.78	543160.92	542154.24	252.46
RanReal480_17	**537924.55**	**536745.39**	360.93	533616.73	532759.09	254.00
RanReal480_18	**525822.76**	**524712.42**	320.60	521450.47	520006.79	108.58
RanReal480_19	**522316.22**	**521267.22**	380.69	518719.33	517865.80	214.04
RanReal480_20	**518349.10**	**517430.77**	353.52	514528.29	514097.85	224.97
#Best	**40**	**40**		0	0	
p-value				2.54e-10	2.54e-10	

Table 21
Comparison of two MA versions using uniform and cluster-based crossover respectively on the RanReal benchmark.

Name	Uniform CX			Cluster-based CX		
	f_{best}	f_{avg}	t_{avg}(s)	f_{best}	f_{avg}	t_{avg}(s)
RanReal240_01	224921.21	224821.36	1295.14	**225003.53**	**224937.30**	2348.57
RanReal240_02	204624.36	204493.54	1952.30	204624.36	**204553.10**	1948.50
RanReal240_03	198981.64	198893.17	1943.37	**199079.37**	**198956.17**	2276.32
RanReal240_04	225566.15	225454.43	1480.25	**225683.17**	**225547.69**	2524.26
RanReal240_05	195533.32	195428.43	1598.60	**195621.68**	**195481.00**	1943.62
RanReal240_06	216747.32	216618.54	1889.24	**216747.39**	**216699.09**	1945.37
RanReal240_07	209288.75	209194.38	1710.89	**209305.70**	**209213.53**	1642.42
RanReal240_08	205246.82	205108.07	1645.73	205246.82	**205134.15**	2146.29
RanReal240_09	209186.90	209094.98	1974.60	209186.90	**209152.19**	2123.53
RanReal240_10	192986.21	192903.33	2459.53	**193062.60**	**192972.14**	1945.36
RanReal240_11	204680.80	204587.50	1672.03	**204722.75**	**204635.30**	1990.59
RanReal240_12	201117.11	200944.04	1924.83	201117.11	**201037.08**	2295.96
RanReal240_13	202338.43	202281.95	1813.21	202338.43	**202312.50**	1613.35
RanReal240_14	228646.52	228555.05	1560.60	**228870.89**	**228601.66**	1278.33
RanReal240_15	191230.33	191094.96	1835.09	**191263.28**	**191177.81**	2281.96
RanReal240_16	203909.40	203808.95	1537.05	**204081.46**	**203971.34**	1986.64
RanReal240_17	195638.95	195464.27	1929.86	195638.95	**195544.67**	2031.51
RanReal240_18	195100.39	194968.12	1876.05	**195167.12**	**195070.91**	2355.17
RanReal240_19	199307.33	199162.91	1660.33	199307.33	**199175.59**	1848.30
RanReal240_20	212268.52	212159.63	1373.99	**212322.48**	**212236.65**	2139.52
RanReal480_01	555377.78	554896.75	3868.70	**556401.65**	**555586.79**	5465.92
RanReal480_02	511021.69	510398.02	4410.56	**511832.51**	**511033.52**	5928.92
RanReal480_03	497675.92	496992.13	4234.89	**498197.75**	**497467.34**	6199.20
RanReal480_04	522425.81	521851.94	3692.03	**523428.57**	**522493.58**	6273.48
RanReal480_05	484496.11	483529.47	3896.52	**484856.17**	**484089.84**	5828.84
RanReal480_06	534140.09	533526.58	4560.41	**535110.60**	**534324.91**	5859.55
RanReal480_07	545781.35	545111.15	4012.66	**546671.53**	**545774.12**	5779.21
RanReal480_08	532566.57	532224.38	3723.32	**533593.65**	**532710.69**	5438.86
RanReal480_09	556523.87	556084.63	4458.78	**557558.00**	**556641.92**	5221.96
RanReal480_10	520148.06	519300.68	4485.14	**520657.09**	**519841.12**	5395.04
RanReal480_11	**524676.35**	523771.15	3951.08	524356.65	**524044.88**	5087.50
RanReal480_12	502329.51	501725.88	4251.32	**502888.85**	**502143.97**	5675.84
RanReal480_13	535017.14	534562.12	4163.93	**536024.76**	**535078.71**	5648.80
RanReal480_14	514814.34	514085.50	4657.32	**515137.95**	**514415.58**	5395.97
RanReal480_15	517835.64	517051.95	4075.11	**518686.16**	**517480.73**	5247.89
RanReal480_16	550021.73	549159.13	3883.47	**550566.81**	**549796.27**	5959.42
RanReal480_17	**538667.77**	537830.76	3244.27	538584.62	**537997.22**	5623.36
RanReal480_18	526142.15	525543.15	4510.03	**527089.59**	**526065.80**	5575.99
RanReal480_19	522705.19	521971.51	3469.05	**523302.33**	**522624.73**	5832.21
RanReal480_20	518857.89	518260.21	3271.27	**519626.64**	**518717.70**	6164.32
#Best	2	0		**31**	**40**	
p-value	4.46e-7	2.54e-10				

Table 22
Computational results on small handover minimization instances. For direct comparison, the results are converted into the minimization form.

Name	FITS		
	f_{best}	f_{avg}	$t_{avg}(s)$
20_5_270001	540	540.00	0.00
20_5_270002	54	54.00	0.00
20_5_270003	816	816.00	0.00
20_5_270004	126	126.00	0.00
20_5_270005	372	372.00	0.00
20_10_270001	2148	2148.00	0.00
20_10_270002	1426	1426.00	0.00
20_10_270003	2458	2458.00	0.00
20_10_270004	1570	1570.00	0.00
30_5_270001	772	772.00	0.00
30_5_270002	136	136.00	0.00
30_5_270003	920	920.00	0.01
30_5_270004	52	52.00	0.00
30_5_270005	410	410.00	0.01
30_10_270001	3276	3276.00	0.00
30_10_270002	1404	1404.00	0.00
30_10_270003	2214	2214.00	0.00
30_10_270004	2150	2150.00	0.02
30_10_270005	2540	2540.00	0.04
30_15_270001	6178	6178.00	0.01
30_15_270002	4042	4042.00	0.00
30_15_270003	4126	4126.00	0.00
30_15_270004	3920	3920.00	0.01
40_5_270001	610	610.00	0.07
40_5_270002	136	136.00	0.05
40_5_270003	234	234.00	0.12
40_5_270004	232	232.00	1.30
40_5_270005	774	774.00	0.00
40_10_270001	4544	4544.00	0.08
40_10_270002	2068	2068.00	0.00
40_10_270003	2090	2090.00	0.01
40_10_270004	1650	1650.00	0.00
40_10_270005	4316	4316.00	0.01
40_15_270001	8646	8646.00	0.24
40_15_270002	4586	4586.00	0.26
40_15_270003	5396	5396.00	0.02
40_15_270004	4800	4800.00	0.00
40_15_270005	6272	6272.00	0.05