Memetic search for composing medical crews with equity and efficiency

Qing Zhou [a], Jin-Kao Hao [a,b], Zhe Sun [c], Qinghua Wu [c,*]

[a] *LERIA, Université d'Angers, 2 bd Lavoisier, 49045 Angers, France*
email: qingzhou@hust.edu.cn; jin-kao.hao@univ-angers.fr

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

[c] *School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China*
email: qinghuawu1005@gmail.com

Applied Soft Computing, May 2020 (In Press)

Abstract

Composing medical crews with equity and efficiency is an important practical problem commonly arising from health care system management. This work presents the first hybrid memetic algorithm for this problem. The proposed approach integrates an original backbone-based crossover for generating promising offspring solutions and a tabu search based local optimization algorithm exploring both feasible and infeasible search regions. Computational experiments on two sets of benchmark instances in the literature are conducted to assess the proposed algorithm with reference to existing methods. This study advances the state-of-the-art of solving this relevant practical problem and is expected to inspire new solution methods to similar problems.

Keywords: Health care service management; memetic algorithm; hybrid search; tabu search; heuristics.

1 Introduction

The quality of medical services critically depends on the delivery of health care and is directly connected to the equity and the efficiency of the health care system. Efficiency mainly concerns providing high quality health care services,

* Corresponding author.

while equity mainly refers to the access to health care systems which should be fairly given to all citizens regardless of age, gender, income and residence.

In general, health care is delivered by medical crews composed of practitioners working together by sharing their skills, experience and knowledge. Among the important issues in health care service management, the pre-hospital care provided by the Emergency Medical Services (EMS) is primordially critical [1]. EMS is responsible for providing timely and appropriate first aid services in case of emergency situations such as accidents. The purpose of such services is to reduce patient death, prevent disability and increase the likelihood of recovery [3]. In [5], Aringhieri presented the first study of the problem of composing medical crews such that the health care service provided complies with the equity and efficiency principles. This problem naturally arises in health care service management, such as building crews for heart surgery or other specific surgeries. A real case from the operation center of the Emergency Medical Service of Milan is described in [4], where the operation center deals with each demand (emergency call) in a timely manner and for urgent calls the response must be fulfilled in less than 8 minutes in the urban regions according to the Italian law. Generally, this problem can be used as a model to formulate practical problems in other settings, such as composing suitable teams with different skills and knowledge that are assigned to a new project.

The problem of composing medical crews with equity and efficiency (CMCEE) studied in this work can be described as follows [5]. Given a set of n individuals $N = \{1, ..., n\}$, for each individual $p \in N$, its efficiency is modeled with a positive value $e_p \in \mathbb{R}^+$ which represents the proficiency of individual p to do a job. Skill difference between two individuals p and q is modeled by a value $d_{pq} \in \mathbb{R}^+$ ($p, q \in N$, $d_{pq} = d_{qp}$ and $d_{pp} = 0$) representing how much the skills of p and q are heterogeneous. The problem is to select a subset of individuals to compose T (T is given) distinct crews such that each crew t ($t = 1, ..., T$) has exactly M_t individuals and the skill diversity of each crew is guaranteed to attain a diversity threshold D_{min} while the efficiency of the crew with the smallest efficiency is maximized.

Formally, the CMCEE can be expressed as the following mathematical model with each binary variable $x_{pt} = 1$ ($p = 1, ..., n$, $t = 1, ..., T$) if the individual p is allocated to crew t and $x_{pt} = 0$ otherwise [5,9]:

$$maximize \qquad \min_{t=1,...,T} \sum_{p=1}^{n} e_p x_{pt} \tag{1}$$

$$subject \quad to \quad \sum_{t=1}^{T} x_{pt} \leq 1, \quad p = 1, ..., n \tag{2}$$

$$\sum_{p=1}^{n} x_{pt} = M_t, \quad t = 1, ..., T \tag{3}$$

$$\sum_{p=1}^{n-1} \sum_{q=p+1}^{n} d_{pq} x_{pt} x_{qt} \geq D_{min}, \quad t = 1, ..., T \tag{4}$$

$$x_{pt} \in \{0, 1\}, \quad p = 1, ..., n, \quad t = 1, ..., T \tag{5}$$

where the objective (1) commits to maximize the efficiency of the crew with the smallest efficiency. Constraint (2) guarantees that each individual is allocated to at most one crew and constraint (3) ensures that the cardinality of crew t is exactly M_t, while constraint (4) forces the sum of skill differences of each crew to be at least D_{min}.

As the literature review of Section 2 shows, there are a number of studies to address the issue of tradeoff between equity and efficiency of health care service management. In addition to its practical relevance, the CMCEE is known to be NP-hard [5] and consequently computationally challenging. Given that exact methods will have an exponential time complexity on the size of the problem, this work focuses thus on developing effective heuristic methods for the CMCEE able to find high-quality solutions with a reasonable time frame. The main contributions of this work are summarized as follows.

- This work presents the first population-based memetic algorithm (MA) for solving the CMCEE, which possesses two original features. First, it integrates an original crossover operator for generating promising offspring solutions. The crossover operator maximally preserves common individuals ('building blocks') grouped in the same crew of parent solutions and favors the inheritance of desirable 'building blocks' through the recombination process. Second, the proposed algorithm uses an effective local optimization procedure combining feasible and infeasible searches driven by the following consideration. By relaxing the diversity threshold constraint in a controlled manner, the algorithm is able to tunnel through feasible and infeasible regions to locate high quality solutions which are difficult to attain otherwise.
- Computational experiments on two sets of benchmark instances in the literature show the competitiveness of the proposed algorithm with respect to existing methods. This implies that our method is able to compose medical crews with a higher equity and a better efficiency. Moreover, the code of our algorithm will be publicly available, which can serve as an useful tool for researchers and practitioners in health care service management. Finally, as the underlying search strategies of the proposed method are rather general, they can be beneficially adapted to similar settings where equity and efficiency are to be considered.

The rest of the paper is structured as follows. Section 2 presents a detailed

review of related works on the studied problem. Section 3 shows the general framework of the proposed algorithm, while its two key search components (crossover and local optimization) are presented in Sections 4 and 5, respectively. Section 6 is dedicated to computational assessments of the proposed algorithm and comparisons with the best performing algorithms in the literature, followed by an experimental analysis of the key components of the algorithm in Section 7. Conclusions are drawn in Section 8.

2 Related works

This section presents a detailed review of the related studies on the composing medical crews with equity and efficiency problem (CMCEE), which originates from a real application from the health care service management. Since the 80s, several studies have considered the issue of efficiency and equity or fairness in health care services.

Mayhew and Leonardi [22] proposed a model that enables a tradeoff between equity and efficiency with application to a regional health-care resource allocation problem in London. Cho [8] presented an equity-efficiency tradeoff model for the location of medical care facilities where equity is represented by the chance to receive medical services whereas efficiency is represented by consumer and producer welfare. Aringhieri [5] proposed both a mathematical model and a graph model following the principles of equity and efficiency that arises from the EMS system of Milan [4]. The author also proposed an effective heuristic algorithm (denoted as GTS) by hybridizing a greedy initialization procedure, a local improvement method and a tabu search procedure. Within GTS, the local improvement method confines its search process in feasible regions, while the tabu search method is allowed to visit infeasible solutions by using a penalized evaluation function. Experimental results showed that the GTS algorithm was able to achieve much better results than the commercial solver Cplex with a shorter computation time. Later, Smith et al. [30] introduced a series of hierarchical location models with bi-objectives (efficiency and equity) for a public service application. Khodaparasti et al. [18] presented an integrated location-allocation model that considers both efficiency and equity for an EMS application. More recently, Delgado-Osuna et al. [9] developed an artificial bee colony (ABC) algorithm for the CMCEE, which deals with infeasible solutions by using a destructive-constructive neighborhood operator and a specialized local search procedure. The reported computational results showed that ABC outperformed the reference algorithms GGA [9] and GTS [5]. Artificial bee colony is a relative new population-based algorithm inspired by the foraging behavior of honeybees. Although ABC has been successfully used in many problems [17] due to its simplicity, it may suffer from the problem of premature

convergence and getting stuck on local optima easily in some scenarios. Recently, Abualigah et al. [2] reviewed sentiment analysis methods and techniques in the health care management to improve health care quality.

According to the computational studies reported in the literature on the CMCEE, the GTS algorithm [5] and the ABC algorithm [9] represent the state of the art of solving the CMCEE. Meanwhile, it is known that the population-based memetic search framework is among the most powerful general methods for grouping problems such as graph coloring [11,16,26], graph partitioning [7,12] and bin packing [10]. Until now, this approach remains unexplored for the CMCEE. This work aims thus to fill the gap by investigating for the first time the potential of the memetic search framework for solving the CMCEE. As the computational results (Section 6) demonstrate, this approach is indeed very successful by finding many improved best solutions for a majority of the tested CMCEE benchmark instances available in the literature.

3 Memetic algorithm for the CMCEE

3.1 General procedure

Memetic algorithms are a powerful and general framework for difficult combinatorial optimization problems [24]. Typically, a memetic algorithm promotes the idea of combining the diversification power of population-based evolutionary search and the intensification strength of local search optimization [14]. The flowchart (Fig. 1) illustrates the general procedure of our proposed memetic algorithm (MA) for the problem of composing medical crews with equity and efficiency. As Algorithm 1 (see Appendix) shows, MA starts with an initial population where each solution is produced by a greedy construction algorithm (Section 3.2). Then the algorithm enters the 'while' loop to perform a number of evolutionary generations until a given cutoff time limit is reached. At each generation, MA selects two parents at random from the population, which are recombined by the backbone-based crossover operator (Section 4) to generate an offspring solution. Subsequently, the generated offspring is improved by the feasible and infeasible tabu search algorithm (Section 5). Finally, the improved offspring solution is used to update the population (Section 3.3).

The remaining of this section is dedicated to population initialization and its update while the crossover operator and the local optimization procedure are presented in two other sections. As it will be evidenced, compared to the two leading CMCEE algorithms (i.e., the tabu search algorithm [5] and the artificial bee colony algorithm [9]), our approach relies on a

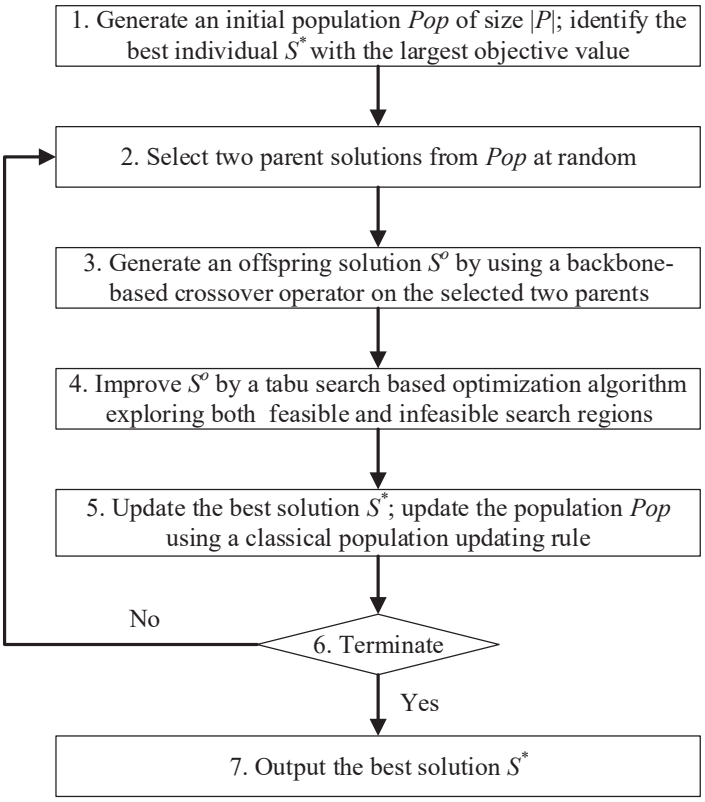

Fig. 1. Flowchart of the proposed MA algorithm.

different search framework (i.e., memetic search) and integrates different search components (i.e., crossover, specific local optimizer), leading to a very competitive algorithm.

3.2 Population initialization

The $|P|$ initial solutions of the population are generated by applying the greedy construction algorithm (denoted as GC, see Algorithm 2 of the Appendix) proposed in [5]. Let S denote a candidate solution which is a set of $T+1$ crews $S = \{C_0, C_1, ..., C_T\}$ such that each C_t ($t \in \{1, ..., T\}$) is the set of individuals allocated to crew t while $C_0 = \{1, ..., n\} \backslash \{C_1 \cup ... \cup C_T\}$ is a dummy crew containing the unallocated individuals. To ensure an efficient implementation of the GC algorithm, a matrix B with its element B_{pt} is used to represent the diversity contribution of each individual p to any crew t [5], where each B_{pt} sums up the diversities between p and all members allocated to C_t:

$$B_{pt} = \sum_{q \in C_t} d_{pq}, q \neq p, p \in \{1, ..., n\}, t \in \{1, ..., T\} \tag{6}$$

Initially, $C_0 = \{1, ..., n\}, C_t = \emptyset$ $(t \in \{1, ..., T\})$ and $B_{pt} = 0$ $(p \in \{1, ..., n\}, t \in \{1, ..., T\})$. Then at each construction step, GC considers all unallocated individuals having the largest efficiency and all incomplete crews with minimum members. Among all these individuals and crews, GC allocates a member p to a crew C_t $(t \in \{1, ..., T\})$ such that the diversity contribution B_{pt} is maximized. This process is repeated until all the crews are complete. Note that the solution constructed by GC always guarantees the feasibility of the cardinality constraint (Constraint (3)) but it may violate the diversity threshold constraint that the sum of skill diversity of each crew must satisfy a certain threshold. In this case, the solution repair procedure (Section 5.4) is applied to transform the infeasible solution S into a feasible one.

3.3 Population updating strategy

A classical population updating rule is used to decide whether the offspring solution after the tabu search improvement should be inserted into the population or not. If the offspring solution is different from the solutions of the population and has a better objective value than the worst solution in the population, then the offspring replaces the worst solution in the population. Otherwise the offspring is discarded and the population is kept unchanged.

4 Crossover operator

The crossover operator of our MA algorithm plays a critical role of diversification and aims at leading the search process to new promising search areas. As indicated in [14], a successful crossover operator should be able to transfer meaningful features from parent solutions to offspring solutions. To design a semantic crossover operator for the CMCEE, the studied problem is considered as a grouping problem whose purpose is to seek a particular partition of n individuals into $T+1$ crews (groups). For grouping problems, it is preferable and natural for a crossover to handle groups of individuals rather than individuals. Crossover operators based on this idea have been successfully used to solve several grouping problems [10,11,16,26]. In case of the CMCEE, a preliminary analysis (Section 7.2) discloses that high quality solutions always contain crews that share the same individuals across the solutions. It thus can be expected that these shared individuals also have a high chance to stay together in the global optimum. Following this consideration, the main idea of our crossover operator is to preserve the groupings of individual (backbone) from parent solutions to offspring solutions.

The proposed backbone-based crossover operator (denoted as BCX) generates

an offspring solution from two parent solutions selected at random.

Definition 1 (backbone): Given two parent solutions $S^1 = \{C_0^1, C_1^1, ..., C_T^1\}$ and $S^2 = \{C_0^2, C_1^2, ..., C_T^2\}$, the backbone BB of S^1 and S^2 is the set of T subsets of individuals $\{BB_1, ..., BB_T\}$ such that each BB_i, $i \in \{1, ..., T\}$ is the subset of individuals that are grouped together in both S^1 and S^2, i.e., $\exists m, n \in \{1, ..., T\}$, $BB_i = C_m^1 \bigcap C_n^2$, while the size of $BB_1 \bigcup ... \bigcup BB_T$ is as large as possible.

Note that the backbone definition does not take into account the set of unallocated individuals of the parent solutions since they are typically low-efficiency and have a small chance to be part of high quality solutions. Based on the notion of backbone, BCX considers, for both parent solutions, only the T crews of allocated individuals.

Definition 2 (individual contribution): Given a solution $S = \{C_0, C_1, ..., C_T\}$, the individual contribution of a member p ($p \in \{1, ..., n\}$) to a crew t ($t \in \{1, ..., T\}$) of S is defined by taking into consideration both its efficiency and its diversity contribution to crew t:

$$IC(S, p, t) = e_p * \sum_{q \in C_t, q \neq p} d_{pq} \tag{7}$$

Based on the backbone, the proposed BCX crossover generates an offspring $S^o = \{C_0^o, C_1^o, ..., C_T^o\}$ in two steps as illustrated in Algorithm 3 of the Appendix. The first step is to create a partial solution based on the backbone of the two selected parent solutions while the second step is to complete the partial solution by allocating some of the unallocated individuals.

Create a partial solution based on backbone. To create a partial offspring solution based on the backbone, it is necessary to carry out a group matching procedure that aims to determine a proper matching between the crews of the two parents, such that it can preserve as many common individuals grouping together in both parent solutions as possible. This group matching problem can be solved by the classic Hungarian approach [19]. However, this will be too time consuming in our case since a maximum weight matching is needed for each crossover application. Therefore, a fast group matching algorithm is adopted to find a near-optimal matching between the crews of the two parent solutions which is used to create a partial solution. The group matching algorithm builds the partial offspring solution S^o in T steps (lines 4-8), and each step t ($t = 1, ..., T$) constructs a crew C_t^o of allocated individuals for S^o as follows. It first identifies a crew C_i^1, $i \in \{1, ..., T\}$ of S^1 and a crew C_j^2, $j \in \{1, ..., T\}$ of S^2 (line 7) such that C_i^1 and C_j^2 have the largest number of identical individuals across all $i \in \{1, ..., T\}$ and $j \in \{1, ..., T\}$, i.e., $max_{i \in \{1,...,T\}, j \in \{1,...,T\}} |C_i^1 \bigcap C_j^2|$. For each identified pair of matched crews, one

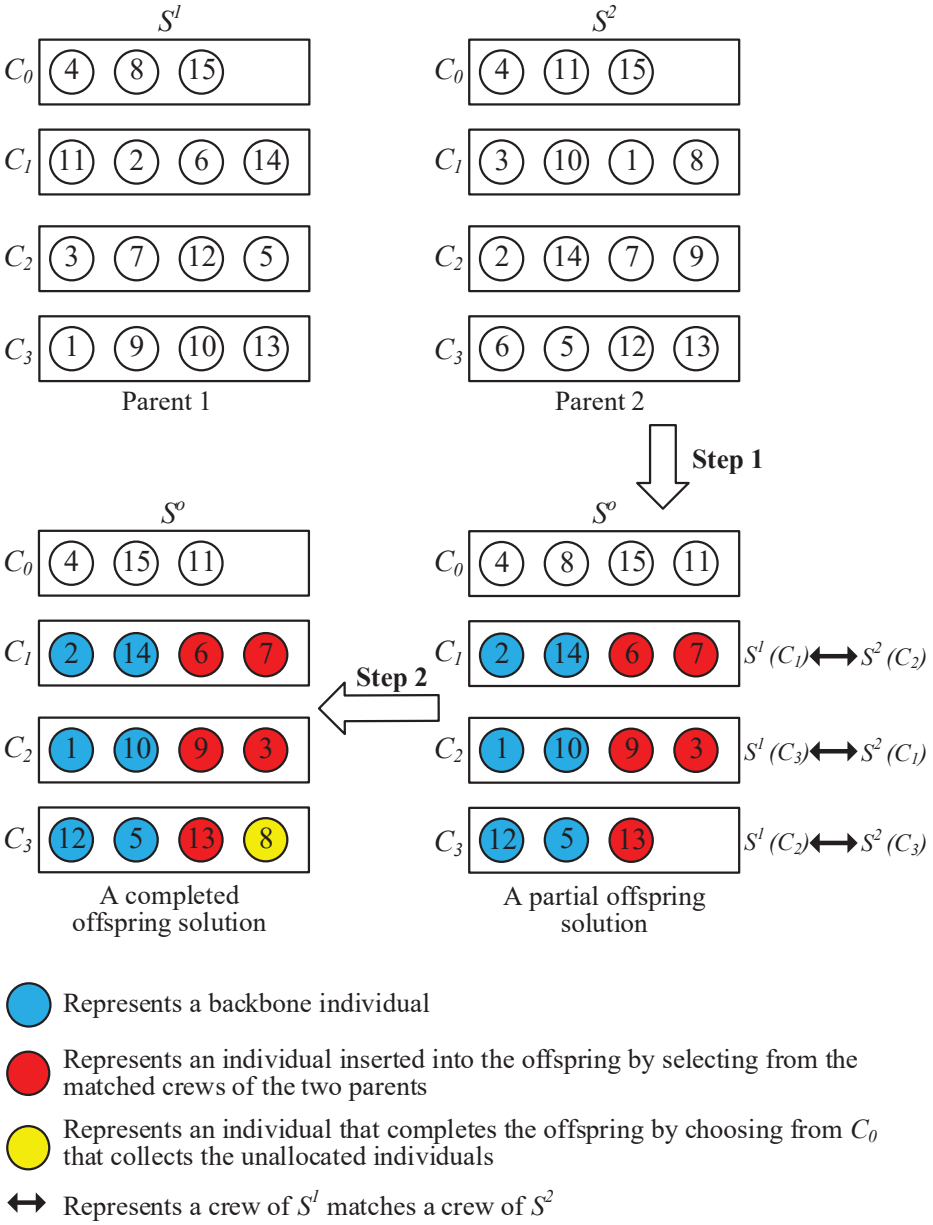

Fig. 2. An illustrative example of backbone crossover steps.

first preserves the backbone (i.e., all common individuals) to the corresponding crew of the offspring solution (line 8), i.e., $C_t^o = C_i^1 \bigcap C_j^2$. Then, C_t^o is further extended by inserting the remaining individuals in C_i^1 and C_j^2 (lines 10-20). Precisely, let $L_1 = C_i^1 \backslash C_t^o$ and $L_2 = C_j^2 \backslash C_t^o$ respectively denote the remaining individuals in C_i^1 and C_j^2 (line 10), the extending procedure alternatively chooses an individual m with the highest individual contribution from L_1 in odd steps and from L_2 in even steps (lines 12-16). Once the selected individual is inserted to C_t^o (line 17), it is removed from the respective set L_1 or L_2 (line 18) . This process is repeated until the number of members in C_t^o reaches M_t or both sets L_1 and L_2 become empty. At the end of step t, all individuals in

C_t^o are removed from all crews of both parents (line 20).

Complete the partial solution. A greedy construct algorithm is applied to complete the partial offspring solution which performs a series of insertion operations until the cardinality of each crew t of S^o reaches M_t (lines 22-27). Let L designate the subset of all remaining unallocated individuals after the first step. At each iteration, the greedy algorithm first selects an individual p from L producing the largest individual contribution $IC(S^o, p, t)$ to a crew t among the ones having the minimum cardinality. Then the selected individual p is displaced from L to crew t of the offspring S^o. Note that the obtained offspring solution after these steps may be an infeasible one violating the diversity threshold constraint. If this happens, a simple solution repair procedure (Section 5.4) is applied to transform the resulting solution into a feasible one (lines 28-30), which is then submitted to the tabu search method for further improvement (Section 5).

Fig. 2 shows an example to illustrate the main steps of the backbone-based crossover operator. There are 15 individuals which are grouped into 3 crews with 4 individuals per crew and one crew with 3 individuals. Given two parent solutions S^1, S^2, the offspring solution S^0 is created from the parents in two steps. First, the group matching algorithm is used to find the matched crews of S^1 and S^2, and then each crew of the offspring solution is built by preserving the backbone individuals (individuals colored in blue), and is extended by inserting individuals with the highest individual contribution (individuals colored in red) selected alternately from the matched groups of S^1 and S^2. For instance, since $S^1(C_1)$ matches $S^2(C_2)$ with two backbone individuals 2 and 14, these individuals are preserved in the offspring $S^0(C_1)$. Then individual 6 from $S^1(C_1)$ and 7 from $S^2(C_2)$ are selected to complete crew C_1 of the offspring. The other crews of S^0 are created in a similar way, while noting that C_3 is completed in the second step by including the unallocated individual with the highest contribution (8, indicated in yellow) from C_0 in S^0 that collects its unallocated individuals.

5 Tabu search with feasible and infeasible exploration

5.1 Main framework

Apart from the backbone based crossover, the local optimization procedure constitutes another critical component of our MA approach. As observed in many studies on strongly constrained problems [23,25,27,31], considering intermediary infeasible solutions during the search process may help to better explore the search space, because doing this may facilitate transitions between

structurally different solutions. Following this idea, the proposed tabu search algorithm (denoted as FITS) alternates between a feasible local search phase (denoted as FLS) where only feasible solutions are examined, and an infeasible local search phase (denoted as ILS) which permits a controlled exploration of infeasible solutions. These two local search phases play different roles in the FITS procedure. FLS is applied to intensify the search by focusing on the most relevant feasible solutions, while ILS is applied to diversify the search by introducing more search freedom through constraint relaxation.

The main framework of FITS is given in Algorithm 4 of the Appendix. Beginning with a feasible solution, FITS first performs the feasible local search phase ensured by the FLS procedure. The FLS procedure is based on the general tabu search framework [13] and relies on the exchange move operator to explore the most relevant feasible search space (Section 5.2). It stops when the best solution cannot be improved for N_{cons} consecutive iterations (N_{cons} is called the search depth of FLS). In this case, the search is considered to be stagnating and switches to the infeasible local search phase to bring more search freedom by relaxing the diversity threshold constraint in a controlled manner. The ILS procedure, which is also based on tabu search, relies on a penalty-based evaluation function to guide the search for an effective examination of the infeasible search space. It stops when a fixed number of iterations M_{ILS} is reached. The FITS procedure iterates these two complementary phases until an allowed maximum number of iterations β_{max} is reached. Computational results (see Section 7.1) indicate that the combined use of these complementary phases constitutes a highly effective hybridization for obtaining high quality solutions.

5.2 The feasible local search phase

The feasible local search procedure (FLS) aggressively examines the feasible search regions to seek improved solutions. This is achieved by using the following exchange move operator.

5.2.1 The Exchange move operator and solution evaluation

Given a feasible solution $S = \{C_0, C_1, ..., C_T\}$, the exchange move swaps two individuals from different crews. To make the search more focused, FLS uses a constrained exchange move (denoted by $Exchange(p, q)$) [5,9] that exchanges an individual p from the crew C_w ($w > 0$) having the smallest efficiency with another individual q from a different crew C_t ($t \geq 0$). Notice that C_0 represents a dummy crew containing the unallocated individuals, and using C_0 in an exchange move enables the swap of an allocated individual and an

unallocated individual. For a fast calculation of the objective function of a candidate solution S' induced by swapping $p \in C_w$ and $q \in C_t$, a T-dimensional vector $E = \{E_1, ..., E_T\}$ is used where $E_i = \sum_{k \in C_i} e_k$ denotes the efficiency of crew C_i. Since an exchange move only involves two crews, the efficiency of C_w (C_t) in S' can be easily computed as $E_w - e_p + e_q$ ($E_t - e_q + e_p$) with E. Similarly, to quickly verify the feasibility of the diversity threshold constraint of the candidate solution S', a T-dimensional vector $D = \{D_1, ..., D_T\}$ is maintained where $D_i = \frac{1}{2} \sum_{m,n \in C_i} d_{mn}$ denotes the total diversity of crew C_i, and hold a matrix B with $B_{pt} = \sum_{q \in C_t} d_{pq}$ representing the diversity contribution of individual p to a given crew t. With D and B, the diversity of C_w (C_t) in S' can be easily calculated as $D_w - B_{pw} + B_{qw} - d_{pq}$ ($D_t + B_{pt} - B_{qt} - d_{pq}$), and the verification of the feasibility of S' can be achieved in $O(1)$.

After performing an exchange move involving $p \in C_w$ and $q \in C_t$, the efficiency and diversity of the two crews can be conveniently updated by the following equations [5,9]:

$$E_w = E_w - e_p + e_q \tag{8}$$

$$E_t = E_t - e_q + e_p \tag{9}$$

$$D_w = D_w - B_{pw} + B_{qw} - d_{pq} \tag{10}$$

$$D_t = D_t + B_{pt} - B_{qt} - d_{pq} \tag{11}$$

Meanwhile, the diversity contribution of each individual with respect to crews w and t can be updated as follows [5,9]:

$$B_{rw} = B_{rw} - d_{pr} + d_{qr}, r \in \{1, ..., n\} \tag{12}$$

$$B_{rt} = B_{rt} - d_{qr} + d_{pr}, r \in \{1, ..., n\} \tag{13}$$

Therefore, for each performed exchange move, the matrix B is updated in $O(n)$, while the vector E and D are updated in $O(1)$.

5.2.2 Exploration with feasible local search

The tabu search based FLS procedure is summarized in Algorithm 5 of the Appendix. Starting from an initial input solution, FLS iteratively replaces the current solution S by a best admissible feasible neighboring solution S' obtained by applying the constrained exchange move operator. To avoid a short-term search cycling, each time an individual p is removed from its original crew t, it is forbidden to bring p back to crew t for the next tt iterations, where tt is a parameter called tabu tenure. A move is considered to be admissible if it is not forbidden by the tabu list or it produces a solution better than the best solution ever found during the search (aspiration criterion). Meanwhile, the best solution found is updated whenever an improved solution is obtained. The search stops when the best solution cannot

be updated for N_{cons} consecutive iterations where N_{cons} is the search depth of FLS. At this point, the search is deemed to be trapped into a deep local optimum and then it switches to the ILS phase to unlock the situation.

5.3 Infeasible local search phase

The basic idea behind ILS is to allow the algorithm to visit intermediary infeasible solutions by relaxing the diversity threshold constraint and enable the algorithm to visit an enlarged search space including both feasible and infeasible solutions. Following the general idea of penalty function for constrained optimization, ILS uses an extended penalty-based evaluation function F to evaluate the quality of both feasible and infeasible solutions. The penalty-based evaluation function F enriches the objective function with a penalty function DE.

Let $S = \{C_0, C_1, ..., C_T\}$ be a candidate solution in the enlarged search space, the penalty function $DE(S)$ is defined as the degree of infeasibility of S measured by the total overloaded parts of all the crews to the diversity threshold, i.e., $DE(S) = \sum_{t=1}^{T} o_t$, where

$$o_t = \begin{cases} D_{min} - D_t, & if\ D_t < D_{min} \\ 0, & otherwise \end{cases} \tag{14}$$

Then the extended evaluation function F is composed of the basic objective function f and penalty function DE:

$$F(S) = f(S) - \beta * DE(S) \tag{15}$$

where $f(S) = \underset{t=\{1,...,T\}}{min} \sum_{p \in C_t} e_p$ gives the objective value, $\beta \geq 0$ is a parameter that controls the relative importance given to DE and is dynamically tuned according to a self-adjustment technique [31].

ILS uses the same constrained exchange move defined in Section 5.2, but without any diversity threshold restriction when choosing a candidate solution. To efficiently assess the quality of a candidate solution evaluated by F, the same vectors E, D, and matrix B are maintained in the same manner as FLS. With these data structures, the extended penalty based function value of each neighbor solution S' can be conveniently computed in a similar manner as in FLS in constant time.

ILS is also based on the tabu search framework and is guided by the extended

penalty-based evaluation function F. Algorithm 6 of the Appendix presents the general scheme of the ILS algorithm. At each iteration of ILS, a best admissible candidate solution S' in terms of F is selected to replace the current solution S. During the search process, the value of the self-adjustment parameter β in F is deducted (added) by 1 if all λ consecutive solutions are feasible (infeasible) and its initiating value is set equal to 0. In this work, the value of λ is set to 5 empirically. In general, a small (large) value of β weakly (strongly) penalizes infeasible solutions and leads to the search process to give more importance to infeasible (feasible) solutions. Moreover, the current best feasible solution S^{local_best} is updated by the incumbent solution S if S is a feasible solution and it is better than S^{local_best} in terms of the basic objective value f. The ILS procedure stops after M_{ILS} iterations (M_{ILS} is a parameter), at this point, the FLS procedure is triggered to bring the search back again to the feasible search. Finally, if the solution at the end of the ILS procedure is an infeasible one, the solution repair procedure (Section 5.4) is applied to convert it into a feasible one, which is served as the starting point for the next round of FLS.

5.4 Solution repair procedure

The solution repair procedure is applied to transform an infeasible solution violating the diversity threshold constraint into a feasible one. Specifically, at each step of the solution repair procedure, an individual is selected from the crew C_{min} with the minimum diversity and exchanged with another individual belonging to a different crew C_q satisfying the diversity threshold constraint, such that the diversity of C_{min} increases the most while C_q still complies with the diversity threshold constraint. The above steps are repeated until the solution becomes feasible.

6 Computational experiments

This section is dedicated to a computational assessment of the proposed MA algorithm on a large number of commonly used benchmark instances and comparisons with two heuristic algorithms in the literature: GTS [5] and ABC [9], which are, to our knowledge, the best existing methods for the CMCEE.

6.1 Benchmark instances and experimental protocol

Two sets of benchmark instances were used to test the proposed method. Proposed in [5], the first set (denoted by $I1$, 80 instances) is adapted from the benchmarks originally designed for the Maximum Diversity Problem introduced in [29] with the following characteristics.

- the number of individuals $n = \{100, 200, 300, 400, 500\}$;
- the total number of selected individuals $M = \{10\%, 20\%, 30\%, 40\%\}$ of n;
- the diversity matrix $DM = d_{pq}$ $(p, q \in \{1, ..., n\})$ is a real number in the range [0,9] with uniform distribution.

The CMCEE instances of set $I1$ are then created as follows.

- keep the parameters n and DM unchanged;
- define the quantity of crews T in {5, 10};
- define the efficiency vector $EV = e_p$ of the individuals by using random values from an uniform distribution [1, 100];
- define the diversity threshold D_{min} for each crew;
- define M_t by $M_t = \{\frac{0.6*n}{T}, \frac{0.8*n}{T}\}$.

The second set of benchmark instances originates from [9] and includes 240 instances. These instances are generated based on the 80 instances of set $I1$ by using three different diversity thresholds. The diversity thresholds for set $I2$ are set to be a percentage ($\rho \in \{80\%, 100\%, 105\%\}$) of the expected mean diversity of the crews calculated by the following equation [9]:

$$D_{min} = \rho \cdot \frac{(M_t - 1) \cdot M_t \cdot \overline{D}}{2}, \quad with \quad \overline{D} = \frac{\sum_{p,q \in N} d_{pq}}{n \cdot (n-1)} \tag{16}$$

Note that a larger ρ generally makes the diversity threshold constraint more difficult to satisfy. The instances in set $I2$ can be divided into three classes according to ρ [9]: easy ($\rho = 80\%$), challenging ($\rho = 100\%$) and difficult ($\rho = 105\%$). All the input data and the detailed numerical results achieved by the proposed algorithm are available at Mendeley's public repository [1].

The MA algorithm was programmed in C++ [2] and compiled by GNU g++ compiler with the "-O3" option, running on a computer with an Intel Xeon-E5 2695 processor (2.10GHz) and 2GB RAM under Linux operating system. Due to its stochastic nature, the proposed algorithm was ran independently

[1] `https://data.mendeley.com/datasets/zmbnybg9g4/1`

[2] The source code of our memetic algorithm will be available at: `http://www.info.univ-angers.fr/pub/pub/hao/CMCEE.html`

Table 1
Settings of parameters.

Parameter	Section	Description	Considered values	Final value
$\|P\|$	3.1, 3.2	size of population	{5, 10, 15, 20, 30}	10
β_{max}	5.1	exploration strength of FITS	{5, 10, 15, 20, 30}	15
N_{cons}	5.1, 5.2.2	search depth of each FLS phase	{1000, 2000, 3000, 4000, 5000}	4000
M_{ILS}	5.1, 5.3	maximum number of iterations of each ILS phase	{300, 500, 1000, 2000, 3000}	1000
tt	5.2.2	tabu tenure	{5, 7, 10, 15, 20}	10

30 times on each instance.

6.2 Parameter setting

The MA algorithm requires 5 parameters: population size $|P|$, exploration strength of the FITS algorithm β_{max}, search depth of the FLS phase N_{cons}, maximum number of iterations of the ILS phase M_{ILS}, tabu tenure tt. To identify a proper parameter settings, the popular 'IRACE' package for automatic parameter tuning [21] is used on a set of 10 randomly selected instances from set $I1$ (02-P100T10M6, 04-P100T5M12, 05-P200T5M24, 10-P300T5M36, 12-P300T10M18, 13-P400T10M24, 15-P400T10M24, 16-P400T5M48, 18-P500T10M30, 19-P500T5M80). For the experiment, the tuning budget was set to 1000 runs, each with a time limit of $n/2$ seconds where n is the number of individuals of the instance. The studied values and final values (suggested by 'IRACE') of these parameters are given in Table 1. The same parameter values determined by 'IRACE' were used for all experiments conducted in this work.

6.3 Comparative results on the $I1$ instances

This section shows a comparison between the proposed MA algorithm and the reference algorithm GTS [5]. In fact, the source codes of the reference algorithms are unavailable and ABC [9] did not report its results on the $I1$ instances. To make the comparison as fair as possible, our MA algorithm was ran 30 times on each instance under the same time limit (stopping condition) as in [5] (running Linux on an Intel Core 2 Duo 2.0GHz T7200 processor and 2GB RAM). Table 2 summarizes the comparative results between the two algorithms.

In Table 2, columns 1-2 indicate the instance name (Instance) and the best known results (f_{BK}) [5]. Columns 3-4 present the results obtained by GTS, including the best objective value (f_{best}) and the average computation time

Table 2
Comparative results of the proposed MA algorithm and the GTS algorithm [5] on $I1$ instances (part I). The results of the ABC algorithm [9] are unavailable.

Instance	f_{BK}	GTS		MA		
		f_{best}	t_{max}	f_{best}	f_{avg}	t_{avg}
01-P100T10M6	399	399	0.16	**420**	408.33	0.11
01-P100T10M8	458	458	0.23	**501**	498.47	0.15
01-P100T5M12	**871**	**871**	0.18	846	836.43	0.16
01-P100T5M16	1011	999	0.25	**1016**	1014.53	0.23
02-P100T10M6	415	398	0.16	**429**	423.67	0.12
02-P100T10M8	488	469	0.22	**504**	500.27	0.18
02-P100T5M12	**889**	**889**	0.19	876	870.40	0.16
02-P100T5M16	1008	1008	0.24	**1016**	1015.17	0.23
03-P100T10M6	397	397	0.17	**421**	411.50	0.12
03-P100T10M8	475	475	0.22	**487**	470.77	0.21
03-P100T5M12	867	867	0.18	**873**	862.03	0.18
03-P100T5M16	1002	1002	0.24	**1010**	1006.97	0.20
04-P100T10M6	409	393	0.15	**435**	428.20	0.12
04-P100T10M8	490	467	0.22	**504**	502.53	0.15
04-P100T5M12	**863**	**863**	0.17	**863**	851.50	0.16
04-P100T5M16	990	990	0.24	**1010**	1007.10	0.21
05-P200T10M12	800	781	0.61	**817**	812.53	0.50
05-P200T10M16	938	903	0.83	**943**	941.90	0.56
05-P200T5M24	**1645**	**1645**	0.69	1587	1575.63	0.68
05-P200T5M32	1872	1870	0.94	**1887**	1884.97	0.83
06-P200T10M12	797	781	0.62	**817**	811.73	0.56
06-P200T10M16	931	905	0.87	**942**	941.13	0.65
06-P200T5M24	1626	1626	0.67	**1638**	1629.73	0.58
06-P200T5M32	1876	1864	0.94	**1889**	1886.63	0.89
07-P200T10M12	788	749	0.61	**820**	816.40	0.52
07-P200T10M16	936	883	0.91	**943**	942.67	0.67
07-P200T5M24	1623	1623	0.70	**1635**	1628.67	0.70
07-P200T5M32	1884	1860	0.93	**1890**	1889.27	0.82
08-P200T10M16	928	887	0.88	**943**	940.97	0.70
08-P200T5M24	1641	1641	0.69	**1642**	1629.70	0.65
08-P200T5M32	1883	1871	0.94	**1890**	1888.93	0.89
09-P300T10M18	1245	1245	1.38	**1269**	1260.33	1.34
09-P300T10M24	1468	1440	2.06	**1478**	1477.57	1.58
09-P300T5M36	**2551**	**2551**	1.60	2465	2425.83	1.54
09-P300T5M48	2932	2932	2.12	**2954**	2950.57	2.08
10-P300T10M18	1233	1233	1.39	**1258**	1238.10	1.35
10-P300T10M24	1434	1433	2.26	**1472**	1466.73	1.92
10-P300T5M36	**2559**	**2559**	1.87	2486	2464.07	1.81
10-P300T5M48	2947	2938	2.63	**2953**	2950.70	2.59
11-P300T10M18	1265	1265	1.78	**1276**	1269.33	1.28
11-P300T10M24	1471	1455	2.32	**1478**	1476.43	1.69

Table 2
Comparative results of the proposed MA algorithm and the GTS algorithm [5] on $I1$ instances (part II). The results of the ABC algorithm [9] are unavailable.

Instance	f_{BK}	GTS		MA		
		f_{best}	t_{max}	f_{best}	f_{avg}	t_{avg}
11-P300T5M36	2556	2556	1.93	**2570**	2559.33	1.86
11-P300T5M48	2957	2937	2.67	**2959**	2959.00	0.84
12-P300T10M18	1242	1242	1.68	**1273**	1265.57	1.36
12-P300T10M24	1469	1438	2.32	**1478**	1477.17	1.55
12-P300T5M36	**2542**	**2542**	1.90	2505	2479.17	1.84
12-P300T5M48	2955	2921	2.62	**2959**	2957.50	2.39
13-P400T10M24	**1707**	**1707**	3.09	**1707**	1626.33	2.69
13-P400T10M32	1998	1979	5.02	**2011**	2009.07	3.10
13-P400T5M48	**3471**	**3471**	3.28	3414	3399.43	3.08
13-P400T5M64	4004	4000	4.81	**4023**	4019.90	4.73
14-P400T10M24	1697	1697	2.93	**1723**	1712.70	2.77
14-P400T10M32	2000	1961	4.40	**2012**	2009.30	3.49
14-P400T5M48	**3466**	**3466**	3.52	3329	3278.60	3.38
14-P400T5M64	4008	3998	4.81	**4025**	4022.13	4.49
15-P400T10M24	1702	1702	3.02	**1709**	1703.70	2.39
15-P400T10M32	1999	1974	4.24	**2011**	2009.67	3.83
15-P400T5M48	**3473**	**3473**	3.48	3414	3387.80	3.25
15-P400T5M64	4018	4001	4.71	**4026**	4025.23	4.66
16-P400T10M24	1685	1685	2.94	**1713**	1697.33	2.30
16-P400T10M32	1989	1952	4.36	**2010**	2007.47	3.84
16-P400T5M48	**3470**	**3470**	3.66	3428	3400.10	3.57
16-P400T5M64	4024	3993	4.62	**4026**	4025.33	3.93
17-P500T10M30	**2145**	**2145**	5.39	**2145**	2070.30	4.00
17-P500T10M40	2483	2483	6.67	**2501**	2496.27	5.95
17-P500T5M60	**4332**	**4332**	5.64	4082	4023.37	5.58
17-P500T5M80	4989	4989	7.58	**4992**	4975.80	7.24
18-P500T10M30	2132	2132	5.03	**2135**	2082.10	4.20
18-P500T10M40	2486	2461	6.75	**2505**	2502.10	6.17
18-P500T5M60	**4354**	**4354**	5.64	4109	4045.40	5.51
18-P500T5M80	5000	5000	7.54	**5008**	5001.57	7.47
19-P500T10M30	2133	2133	5.44	**2141**	2072.17	4.03
19-P500T10M40	2470	2470	6.95	**2497**	2489.43	6.54
19-P500T5M60	**4335**	**4335**	5.58	4199	4165.47	5.55
19-P500T5M80	4987	4987	7.58	**5006**	4992.57	7.38
20-P500T10M30	2131	2114	4.96	**2179**	2165.73	4.49
20-P500T10M40	2504	2446	6.56	**2509**	2508.23	4.17
20-P500T5M60	**4346**	**4346**	5.51	4217	4179.47	5.33
20-P500T5M80	4995	4995	7.40	**5009**	5002.37	7.29
#Improve		0		63		
#Match		17		3		
#Total		80		80		
p-value		1.59e-3				

(t_{max}) in seconds to complete its execution. The results of GTS are directly taken from [5]. The last three columns report the results produced by our MA algorithm, including the best objective value (f_{best}), the average objective value (f_{avg}) and average run time (t_{avg}) in seconds to reach the best objective value. Rows '#Improve' and '#Match' report respectively the number of cases for which each algorithm improves or matches the best-known results from the literature. Row '#Total' denotes the total number of instances. Row p-value indicates the results from the Wilcoxon signed-rank test with a confidence level of 95% between the compared algorithms. The results in bold are the best results among f_{BK} and f_{best} of the compared algorithms.

The comparison mainly focuses on solution quality in terms of the objective values, while the timing information is provided only for indicative purposes. Since the two compared algorithms were run on different computing platforms, the Standard Performance Evaluation Cooperation tool (www.spec.org) is used to obtain the scale ratio (2.10/2.00 = 1.05) of the CPU frequencies, which indicates that our computer is slightly (1.05 time) faster.

Table 2 indicates that MA performs better than GTS by finding 63 improved lower bounds and matching the best known results on 3 cases while MA is outperformed by GTS for the remaining 14 instances. The Wilcoxon signed-rank test (p-value< 0.05) also confirms the dominance of MA over GTS. This experiment shows that MA is highly efficient for the $I1$ instances compared to the GTS algorithm.

6.4 Comparative results on the $I2$ instances

To further assess the performance of our MA algorithm, this section compares MA with the two reference algorithms GTS [5] and ABC [9] on the $I2$ instances. A summary of the comparative results is shown in Table 3. Row '#Best' denotes the number of instances for which the corresponding algorithm produces the best result among all the compared algorithms. The last row gives the p-values from the Wilcoxon signed-rank test with a confidence level of 95%. The symbol '-' indicates that the corresponding algorithm cannot obtain a feasible solution on the instance within the time limit with $n/2$ seconds and other symbols have the same meanings as those in Table 2. The results for GTS and ABC are directly compiled from [9] and were obtained on a platform equipped with an Intel i5 quad-core processor (2.90GHz) and 8 GB RAM running on MAC OS X operating system. Note that the results on the $I2$ instances reported for GTS were based on the implementation by the authors of [9]. The scale ratio (2.10/2.90 $\approx$ 0.72) of the CPU frequencies from SPEC (www.spec.org) indicates that our computer is slower than the computer used in [9].

Table 3. Comparative results of the proposed MA algorithm and the reference algorithms (ABC [9] and GTS [5]) on *I2* instances (part I).

Instance	$\rho = 80\%$					$\rho = 100\%$					$\rho = 105\%$				
	ABC	GTS	MA			ABC	GTS	MA			ABC	GTS	MA		
	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}
01-P100T10M6	301	243	**449**	449.00	0.04	-	243	**449**	449.00	0.06	-	-	**449**	449.00	0.04
01-P100T10M8	401	406	**508**	508.00	0.01	-	421	**508**	508.00	0.03	-	-	**508**	508.00	0.06
01-P100T5M12	704	654	**899**	899.00	0.03	690	538	**899**	899.00	0.05	-	-	**899**	899.00	0.06
01-P100T5M16	896	802	**1016**	1016.00	0.09	903	837	**1016**	1016.00	0.13	926	-	**1016**	1016.00	0.15
02-P100T10M6	319	292	**449**	449.00	0.05	323	347	**449**	449.00	0.07	-	-	**449**	449.00	0.08
02-P100T10M8	418	412	**508**	508.00	0.04	406	430	**508**	508.00	0.05	-	-	**508**	508.00	0.06
02-P100T5M12	733	665	**899**	899.00	0.03	737	680	**899**	899.00	0.03	704	-	**899**	899.00	0.04
02-P100T5M16	932	881	**1016**	1016.00	0.02	915	856	**1016**	1016.00	0.03	902	-	**1016**	1016.00	0.03
03-P100T10M6	308	333	**449**	449.00	0.06	-	160	**449**	449.00	0.05	-	-	**449**	449.00	0.06
03-P100T10M8	416	438	**508**	508.00	0.07	397	246	**508**	508.00	0.07	-	-	**508**	508.00	0.09
03-P100T5M12	736	719	**899**	899.00	0.30	732	682	**899**	899.00	0.47	-	-	**899**	899.00	0.51
03-P100T5M16	919	861	**1016**	1016.00	0.06	947	835	**1016**	1016.00	0.06	-	-	**1016**	1016.00	0.08
04-P100T10M6	316	314	**449**	449.00	0.04	-	-	**449**	449.00	0.07	-	-	**449**	449.00	0.10
04-P100T10M8	417	387	**508**	508.00	0.05	416	-	**508**	508.00	0.12	401	-	**508**	508.00	0.15
04-P100T5M12	718	648	**899**	899.00	0.15	731	674	**899**	899.00	0.10	-	-	**899**	899.00	0.18
04-P100T5M16	915	879	**1016**	1016.00	0.02	915	871	**1016**	1016.00	0.04	-	-	**1016**	1016.00	0.02
05-P200T10M12	584	550	**833**	833.00	0.30	589	594	**833**	833.00	0.36	-	-	**833**	833.00	0.50
05-P200T10M16	768	742	**945**	945.00	0.28	767	705	**945**	945.00	0.29	-	-	**945**	945.00	0.34
05-P200T5M24	1294	1171	**1667**	1667.00	0.18	1279	1190	**1667**	1667.00	0.33	-	-	**1667**	1667.00	0.73
05-P200T5M32	1651	1553	**1890**	1890.00	0.20	1661	1499	**1890**	1890.00	0.20	-	-	**1890**	1890.00	0.26
06-P200T10M12	600	585	**833**	833.00	0.22	586	597	**833**	833.00	0.23	-	-	**833**	833.00	0.33
06-P200T10M16	774	781	**945**	945.00	0.31	760	777	**945**	945.00	0.34	-	-	**945**	945.00	0.47
06-P200T5M24	1316	1187	**1667**	1667.00	0.64	1323	1204	**1667**	1667.00	0.89	-	-	**1667**	1667.00	1.04
06-P200T5M32	1669	1615	**1890**	1890.00	2.67	1677	1626	**1890**	1890.00	3.13	-	-	**1890**	1890.00	3.51
07-P200T10M12	568	525	**833**	833.00	0.20	574	550	**833**	833.00	0.25	-	-	**833**	833.00	0.29
07-P200T10M16	753	724	**945**	945.00	0.21	760	799	**945**	945.00	0.32	-	-	**945**	945.00	0.43
07-P200T5M24	1299	1249	**1667**	1667.00	0.70	1344	1315	**1667**	1667.00	1.35	-	-	**1667**	1667.00	1.73
07-P200T5M32	1684	1624	**1890**	1890.00	0.34	1688	-	**1890**	1890.00	0.75	-	-	**1890**	1890.00	0.78
08-P200T10M12	593	477	**833**	833.00	0.21	-	558	**833**	833.00	0.24	-	-	**833**	833.00	0.36
08-P200T10M16	791	704	**945**	945.00	0.34	784	714	**945**	945.00	0.32	-	-	**945**	945.00	0.38
08-P200T5M24	1320	1166	**1667**	1667.00	1.26	1310	1221	**1667**	1667.00	1.96	-	-	**1667**	1667.00	2.30
08-P200T5M32	1687	1613	**1890**	1890.00	0.18	1665	1571	**1890**	1890.00	0.20	-	-	**1890**	1890.00	0.34
09-P300T10M18	914	820	**1294**	1294.00	0.41	900	857	**1294**	1294.00	0.55	-	-	**1294**	1294.00	0.84
09-P300T10M24	1192	1147	**1479**	1479.00	0.77	1194	1083	**1479**	1479.00	0.79	-	-	**1479**	1479.00	0.82
09-P300T5M36	2005	1890	**2589**	2589.00	1.37	2011	1958	**2589**	2589.00	1.89	-	-	**2589**	2589.00	2.96
09-P300T5M48	2580	2479	**2959**	2959.00	0.45	2529	2563	**2959**	2959.00	0.42	-	-	**2959**	2959.00	1.13
10-P300T10M18	882	875	**1294**	1294.00	0.67	865	-	**1294**	1294.00	0.47	-	-	**1294**	1294.00	0.77
10-P300T10M24	1164	1171	**1479**	1479.00	0.74	1204	-	**1479**	1479.00	0.82	-	-	**1479**	1479.00	0.61
10-P300T5M36	1958	1820	**2589**	2589.00	0.42	1955	1808	**2589**	2589.00	0.46	-	-	**2589**	2589.00	0.47
10-P300T5M48	2583	2412	**2959**	2959.00	0.44	2585	2430	**2959**	2959.00	0.48	-	-	**2959**	2959.00	0.39
11-P300T10M18	914	939	**1294**	1294.00	0.57	899	896	**1294**	1294.00	0.62	-	-	**1294**	1294.00	1.14
11-P300T10M24	1209	1213	**1479**	1479.00	0.72	1195	-	**1479**	1479.00	0.71	-	-	**1479**	1479.00	0.66

Table 3. Comparative results of the proposed MA algorithm and the reference algorithms (ABC [9] and GTS [5]) on $I2$ instances (part II).

Instance	$\rho = 80\%$					$\rho = 100\%$					$\rho = 105\%$				
	ABC	GTS	MA			ABC	GTS	MA			ABC	GTS	MA		
	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{best}	f_{best}	f_{avg}	t_{avg}
11-P300T5M36	2046	1856	**2589**	2589.00	2.85	2006	1712	**2589**	2589.00	3.09	-	-	**2589**	2589.00	3.50
11-P300T5M48	2644	2351	**2959**	2959.00	2.09	2522	2412	**2959**	2959.00	2.31	-	-	**2959**	2959.00	3.47
12-P300T10M18	948	783	**1294**	1294.00	0.50	899	791	**1294**	1294.00	0.58	-	-	**1294**	1294.00	0.46
12-P300T10M24	1200	1126	**1479**	1479.00	0.79	1197	1165	**1479**	1479.00	0.92	-	-	**1479**	1479.00	0.61
12-P300T5M36	1951	1937	**2589**	2589.00	1.89	2010	1778	**2589**	2589.00	1.93	-	-	**2589**	2589.00	2.71
12-P300T5M48	2621	2398	**2959**	2959.00	1.47	2572	2525	**2959**	2959.00	1.87	-	-	**2959**	2959.00	3.28
13-P400T10M24	1233	1174	**1754**	1754.00	1.09	-	-	**1754**	1754.00	1.18	-	-	**1754**	1754.00	1.24
13-P400T10M32	1645	1602	**2013**	2013.00	1.99	-	1694	**2013**	2013.00	2.20	-	-	**2013**	2013.00	2.52
13-P400T5M48	2715	2324	**3509**	3509.00	2.65	2702	2526	**3509**	3509.00	4.08	-	-	**3509**	3509.00	4.67
13-P400T5M64	3521	3314	**4026**	4026.00	1.60	3550	3339	**4026**	4026.00	1.87	-	-	**4026**	4026.00	2.30
14-P400T10M24	1228	1118	**1754**	1754.00	1.04	1229	1168	**1754**	1754.00	1.11	-	-	**1754**	1754.00	1.04
14-P400T10M32	1634	1581	**2013**	2013.00	1.84	1658	1466	**2013**	2013.00	2.24	-	-	**2013**	2013.00	2.45
14-P400T5M48	2666	2420	**3509**	3509.00	5.92	2666	2437	**3509**	3509.00	7.66	-	-	**3509**	3509.00	9.04
14-P400T5M64	3506	3268	**4026**	4026.00	1.33	3523	3143	**4026**	4026.00	0.96	-	-	**4026**	4026.00	1.64
15-P400T10M24	1271	1238	**1754**	1754.00	1.07	1241	1322	**1754**	1754.00	1.21	-	-	**1754**	1754.00	1.69
15-P400T10M32	1651	1632	**2013**	2013.00	1.78	1657	1662	**2013**	2013.00	2.53	-	-	**2013**	2013.00	2.89
15-P400T5M48	2720	2482	**3509**	3509.00	2.82	2668	2535	**3509**	3509.00	4.28	-	-	**3509**	3509.00	4.83
15-P400T5M64	3563	3307	**4026**	4026.00	1.57	3562	3238	**4026**	4026.00	2.25	-	-	**4026**	4026.00	3.13
16-P400T10M24	1265	1156	**1754**	1754.00	1.03	-	-	**1754**	1754.00	1.43	-	-	**1754**	1754.00	1.96
16-P400T10M32	1637	1597	**2013**	2013.00	1.86	-	-	**2013**	2013.00	2.30	-	-	**2013**	2013.00	2.70
16-P400T5M48	2708	2538	**3509**	3509.00	1.78	2784	2557	**3509**	3509.00	1.96	-	-	**3509**	3509.00	2.63
16-P400T5M64	3472	3278	**4026**	4026.00	0.86	3459	3427	**4026**	4026.00	1.13	-	-	**4026**	4026.00	3.72
17-P500T10M30	1530	1537	**2189**	2189.00	1.73	1507	1554	**2189**	2189.00	2.02	-	-	**2189**	2189.00	2.97
17-P500T10M40	2037	1972	**2509**	2508.20	2.10	-	2065	**2509**	2509.00	2.76	-	-	**2509**	2509.00	3.89
17-P500T5M60	3233	3200	**4378**	4378.00	4.13	3289	-	**4378**	4378.00	5.71	-	-	**4378**	4378.00	6.43
17-P500T5M80	4289	4250	**5019**	5019.00	1.97	4266	-	**5019**	5019.00	2.41	-	-	**5019**	5019.00	3.63
18-P500T10M30	1574	1526	**2189**	2189.00	2.17	-	1663	**2189**	2189.00	3.08	-	-	**2189**	2189.00	3.77
18-P500T10M40	2067	1982	**2509**	2506.70	2.78	-	2107	**2509**	2509.00	4.24	-	-	**2509**	2509.00	5.14
18-P500T5M60	3392	2836	**4378**	4378.00	1.79	3326	3081	**4378**	4378.00	1.81	-	-	**4378**	4378.00	3.06
18-P500T5M80	4349	3960	**5019**	5019.00	0.90	4355	4072	**5019**	5019.00	2.12	-	-	**5019**	5019.00	4.20
19-P500T10M30	1535	1455	**2189**	2189.00	2.87	1528	1446	**2189**	2189.00	5.08	-	-	**2189**	2189.00	5.37
19-P500T10M40	2031	1906	**2509**	2508.27	4.49	-	2011	**2509**	2509.00	6.62	-	-	**2509**	2509.00	8.83
19-P500T5M60	3288	3107	**4378**	4378.00	4.68	3304	3247	**4378**	4378.00	5.26	-	-	**4378**	4378.00	5.41
19-P500T5M80	4325	4144	**5019**	5019.00	0.50	4391	4243	**5019**	5019.00	1.88	-	-	**5019**	5019.00	5.56
20-P500T10M30	1551	1481	**2189**	2189.00	2.78	1543	1535	**2189**	2189.00	4.35	-	-	**2189**	2189.00	5.20
20-P500T10M40	2023	2002	**2509**	2507.17	2.18	2026	2069	**2509**	2509.00	6.87	-	-	**2509**	2509.00	8.28
20-P500T5M60	3339	3289	**4378**	4378.00	0.88	3408	3236	**4378**	4378.00	1.20	-	-	**4378**	4378.00	2.63
20-P500T5M80	4305	4216	**5019**	5019.00	0.59	4358	4363	**5019**	5019.00	2.00	-	-	**5019**	5019.00	6.61
#Best	0	0	80			0	0	80			0	0	80		
p-value	7.85e-15	7.84e-15				7.84e-15	7.85e-15				7.73e-15	7.71e-15			

Table 4
Comparison results between FITS, FLS and ILS on the 10 randomly selected instances of $I1$ set.

Instance	FITS			FLS			ILS		
	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}
02-P100T10M6	**435**	**424.27**	0.12	431	405.23	0.03	420	362.77	0.12
04-P100T5M12	**863**	**849.40**	0.16	857	839.97	0.12	859	794.07	0.13
05-P200T5M24	**1591**	**1575.50**	0.68	1586	1572.43	0.63	1563	1372.83	0.62
10-P300T5M36	**2492**	**2470.83**	1.75	2488	2467.80	1.52	2425	2275.53	1.27
12-P300T10M18	**1279**	**1273.50**	1.44	1276	1171.50	0.75	1246	1138.47	1.27
13-P400T10M24	**1707**	**1640.40**	2.73	1677	1442.20	1.10	1456	1325.40	1.91
15-P400T10M24	**1726**	**1708.77**	2.82	1717	1663.80	1.65	1622	1475.27	1.87
16-P400T5M48	**3439**	**3413.33**	3.52	3435	3404.10	3.38	3348	3050.07	3.37
18-P500T10M30	**2135**	**2096.73**	4.69	2120	1997.20	2.79	1975	1752.27	3.54
19-P500T5M80	**5011**	**4998.87**	7.33	5008	4996.10	6.96	4949	4762.20	4.82
#Best	**10**	**10**		0	0		0	0	
$Avg_t(s)$			2.52			1.89			1.89

One observes from Table 3 that our MA algorithm outperforms these two reference algorithms on the $I2$ set by producing the best result for all 240 instances in terms of the best objective value. One also notices that, for all 240 instances, even our *average* results (f_{avg}) across the 30 independent runs are better than the *best* results of the two reference algorithms. Furthermore, the superiority of MA over each reference algorithm is confirmed by the Wilcoxon signed-rank test with p-values less than 0.05. This experiment demonstrates that MA competes very favorably with the reference algorithms on this set of instances.

7 Analysis

This section is dedicated to an analysis of several key ingredients of the proposed algorithm to illustrate their influences on the performance of the algorithm, which includes the combined use of feasible and infeasible local searches, the motivation behind the backbone-based crossover and the effect of the memetic framework. The experiments were conducted on 10 randomly selected instances from the $I1$ set and 30 instances from the $I2$ set, covering three different diversity thresholds $\rho = 80\%$, $\rho = 100\%$ and $\rho = 105\%$.

Table 5
Comparison results between FITS, FLS and ILS on 30 instances of $I2$ set with different thresholds $\rho = 80\%$, $\rho = 100\%$ and $\rho = 105\%$.

ρ	Instance	FITS			FLS			ILS		
		f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}
80%										
	02-P100T10M6	**449**	**449.00**	0.03	**449**	**449.00**	0.01	**449**	**449.00**	0.01
	04-P100T5M12	**899**	**898.93**	0.28	**899**	898.87	0.06	**899**	898.70	0.01
	05-P200T5M24	**1667**	**1666.97**	0.13	**1667**	**1666.97**	0.05	**1667**	1666.93	0.04
	10-P300T5M36	**2589**	**2589.00**	0.30	**2589**	2588.97	0.25	**2589**	2588.93	0.26
	12-P300T10M18	**1294**	**1293.97**	0.29	**1294**	**1293.97**	0.13	**1294**	**1293.97**	0.11
	13-P400T10M24	**1754**	**1754.00**	0.87	**1754**	**1754.00**	0.68	**1754**	**1754.00**	0.65
	15-P400T10M24	**1754**	**1753.97**	0.93	**1754**	**1753.97**	1.04	**1754**	1753.93	0.74
	16-P400T5M48	**3509**	**3508.43**	0.89	**3509**	3508.23	0.68	**3509**	3508.33	0.67
	18-P500T10M30	**2189**	**2188.80**	1.20	**2189**	**2188.80**	2.15	**2189**	2188.33	0.36
	19-P500T5M80	**5019**	**5018.70**	0.33	**5019**	5018.67	0.75	**5019**	5018.07	0.25
100%										
	02-P100T10M6	**449**	**449.00**	0.06	**449**	**449.00**	0.03	**449**	**449.00**	0.02
	04-P100T5M12	**899**	**899.00**	0.05	**899**	**899.00**	0.03	**899**	**899.00**	0.02
	05-P200T5M24	**1667**	**1666.93**	0.12	**1667**	1666.90	0.12	**1667**	1666.87	0.10
	10-P300T5M36	**2589**	**2589.00**	0.35	**2589**	**2589.00**	0.32	**2589**	**2589.00**	0.33
	12-P300T10M18	**1294**	**1293.93**	0.38	**1294**	1293.90	0.29	**1294**	1293.17	0.25
	13-P400T10M24	**1754**	**1754.00**	0.98	**1754**	**1754.00**	0.76	**1754**	**1754.00**	0.69
	15-P400T10M24	**1754**	**1753.97**	1.03	**1754**	**1753.97**	0.93	**1754**	**1753.97**	0.76
	16-P400T5M48	**3509**	**3508.60**	1.21	**3509**	3508.33	0.76	**3509**	3508.53	0.74
	18-P500T10M30	**2189**	**2189.00**	2.11	**2189**	**2189.00**	2.15	**2189**	**2189.00**	0.69
	19-P500T5M80	**5019**	**5018.70**	1.02	**5019**	5018.67	0.93	**5019**	5018.67	0.47
105%										
	02-P100T10M6	**449**	**449.00**	0.06	**449**	**449.00**	0.04	**449**	**449.00**	0.02
	04-P100T5M12	**899**	**899.00**	0.16	**899**	**899.00**	0.06	**899**	**899.00**	0.07
	05-P200T5M24	**1667**	**1667.00**	0.58	**1667**	1666.83	0.24	**1667**	1666.47	0.16
	10-P300T5M36	**2589**	**2589.00**	0.43	**2589**	2588.97	0.36	**2589**	**2589.00**	0.38
	12-P300T10M18	**1294**	**1294.00**	0.39	**1294**	**1294.00**	0.31	**1294**	1293.93	0.28
	13-P400T10M24	**1754**	**1754.00**	1.02	**1754**	**1754.00**	0.84	**1754**	**1754.00**	0.81
	15-P400T10M24	**1754**	**1753.93**	1.28	**1754**	1753.83	1.03	**1754**	1753.67	0.87
	16-P400T5M48	**3509**	**3508.70**	2.12	**3509**	3508.67	0.89	**3509**	3508.60	1.04
	18-P500T10M30	**2189**	**2188.97**	2.78	**2189**	2188.90	2.34	**2189**	2188.83	1.46
	19-P500T5M80	**5019**	**5017.33**	3.56	**5019**	5017.23	3.38	**5019**	5017.13	1.26
#Best		**30**	**30**		**30**	17		**30**	14	
$Avg_t(s)$				0.83			0.72			0.45

7.1 Effect of the combined use of feasible and infeasible local searches

A key feature of FITS is the hybrid scheme integrating both feasible and infeasible local search methods. To examine the merit of the hybrid scheme, an experiment was conducted to compare FITS and its two underlying local search (FLS and ILS) procedures.

The comparative results are summarized in Tables 4 and 5. The average run time in seconds to reach the best objective value for all the test instances is given in the last row ('Avg_t'). Tables 4 and 5 show that FITS produced the best result on all 40 instances in terms of the best objective value whereas FLS

Table 6
Percentage of individuals in the same crews in local optima of different qualities on a random selection of 10 instances from $I1$ set.

Instance	S_{hq}	S_{all}	S_{lo}	Instance	S_{hq}	S_{all}	S_{lo}
02-P100T10M6	0.84	0.71	0.55	13-P400T10M24	0.92	0.64	0.29
04-P100T5M12	0.86	0.69	0.57	15-P400T10M24	0.85	0.63	0.35
05-P200T5M24	0.81	0.64	0.35	16-P400T5M48	0.87	0.68	0.43
10-P300T5M36	0.79	0.58	0.41	18-P500T10M30	0.82	0.59	0.25
12-P300T10M18	0.76	0.57	0.38	19-P500T5M80	0.83	0.63	0.40

Table 7
Percentage of individuals in the same crews in local optima of different qualities on 30 instances from $I2$ set with different diversity thresholds ρ.

Instance	$\rho = 80\%$			$\rho = 100\%$			$\rho = 105\%$		
	S_{hq}	S_{all}	S_{lo}	S_{hq}	S_{all}	S_{lo}	S_{hq}	S_{all}	S_{lo}
02-P100T10M6	0.84	0.70	0.46	0.72	0.50	0.35	0.78	0.59	0.46
04-P100T5M12	0.64	0.46	0.35	0.81	0.58	0.44	0.89	0.62	0.50
05-P200T5M24	0.87	0.68	0.48	0.76	0.61	0.48	0.94	0.70	0.49
10-P300T5M36	0.85	0.57	0.36	0.74	0.50	0.35	0.95	0.76	0.53
12-P300T10M18	0.64	0.49	0.31	0.81	0.66	0.46	0.94	0.69	0.51
13-P400T10M24	0.89	0.69	0.45	0.90	0.65	0.42	0.77	0.65	0.40
15-P400T10M24	0.87	0.66	0.43	0.89	0.58	0.36	0.85	0.58	0.42
16-P400T5M48	0.67	0.49	0.32	0.84	0.63	0.45	0.91	0.67	0.41
18-P500T10M30	0.80	0.60	0.38	0.87	0.67	0.52	0.88	0.64	0.37
19-P500T5M80	0.69	0.47	0.29	0.89	0.62	0.51	0.81	0.58	0.34

and ILS yielded the best result on 30, 30 cases respectively. When comparing the average objective value over 30 independent runs, FITS obtained the best result on all the instances, while FLS and ILS produced the best result on 17, 14 out of the 40 instances respectively. The Wilcoxon signed-rank test leads to small p-values in terms of the best solution values (average solution values): 4.92e-3 (1.81e-5) for FITS v.s. FLS, and 5.06e-3 (5.60e-6) for FITS v.s. ILS. This experiment demonstrates the effectiveness of the hybrid scheme integrating both the feasible and infeasible local searches.

7.2 Motivation behind the backbone-based crossover

To explain the use of the proposed backbone-based crossover, this section investigates the structural similarity between local optima having different qualities. Given two local optima S^1 and S^2, the similarity between them is defined as the percentage of individuals grouped together in both S^1 and S^2: $sim(S^1, S^2) = \frac{|J|}{n}$ where J denotes the set of individuals grouped together in S^1 and S^2. J is identified by using the group matching algorithm proposed in Section 4.

Table 8
Comparison results between FITS and MA on the 10 randomly selected instances of $I1$ set.

Instance	FITS			MA		
	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}
02-P100T10M6	**435**	428.83	0.88	**435**	**429.17**	1.17
04-P100T5M12	**863**	858.13	1.02	**863**	**858.97**	1.42
05-P200T5M24	**1591**	1585.97	5.24	**1591**	**1586.63**	6.61
10-P300T5M36	2499	2491.93	15.34	**2505**	**2493.77**	17.08
12-P300T10M18	1281	1278.70	8.73	**1283**	**1279.90**	12.09
13-P400T10M24	**1707**	1661.97	17.52	**1707**	**1678.13**	18.57
15-P400T10M24	**1729**	1722.93	19.32	**1729**	**1724.20**	19.45
16-P400T5M48	3445	3433.93	31.16	**3446**	**3437.83**	30.39
18-P500T10M30	2147	2118.33	29.13	**2151**	**2129.17**	30.01
19-P500T5M80	5011	5009.53	71.33	**5013**	**5009.97**	73.90
#Best	5	0		10	10	
$Avg_t(s)$			19.97			21.07

For this analysis, the same 40 instances (10 $I1$ instances and 30 $I2$ instances) as before were used. For each instance, 1000 local optima of different qualities were produced by using FITS and MA. Then the top 10% (100) local optima having the largest objective values are selected to form the set of 'high-quality solutions', and the bottom 10% (100) solutions with the smallest objective values are used to build the set of 'low-quality solutions'.

Tables 6 and 7 show the experimental results. Columns S_{hq}, S_{all} and S_{lo} represent respectively the percentage of common individuals grouped across the set of 100 high-quality solutions, the set of 1000 sampled local optima and the set of 100 low-quality solutions. It is observed that the percentage of individuals grouped together across the solutions from the set of high-quality solutions is large, ranging from 0.64 to 0.95. This observation provides the basis for the proposed backbone crossover operator, which preserves the common individuals grouped together between two parent solutions.

7.3 Usefulness of the memetic framework

To analyze the effect of the memetic framework, the MA algorithm was compared with a multi-start version of the FITS algorithm. For this experiment, both FITS and MA were executed independently 30 times on each instance with a time limit of $n/2$ seconds per run. Notice that the FITS algorithm was run in a multi-start way by generating a random initial solution for each restart until the cutoff time was reached.

Table 9
Comparisons between FITS and MA on 30 instances of $I2$ set with different thresholds $\rho = 80\%$, $\rho = 100\%$ and $\rho = 105\%$.

ρ	Instance	FITS			MA		
		f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}
80%							
	02-P100T10M6	**449**	**449.00**	0.03	**449**	**449.00**	0.05
	04-P100T5M12	**899**	898.83	0.13	**899**	**899.00**	0.15
	05-P200T5M24	**1667**	1666.87	0.11	**1667**	**1667.00**	0.18
	10-P300T5M36	**2589**	**2589.00**	0.32	**2589**	**2589.00**	0.42
	12-P300T10M18	**1294**	1293.67	0.26	**1294**	**1294.00**	0.50
	13-P400T10M24	**1754**	**1754.00**	0.82	**1754**	**1754.00**	1.09
	15-P400T10M24	**1754**	1753.87	0.83	**1754**	**1754.00**	1.07
	16-P400T5M48	**3509**	3508.63	0.98	**3509**	**3509.00**	1.78
	18-P500T10M30	**2189**	2188.87	1.40	**2189**	**2189.00**	2.87
	19-P500T5M80	**5019**	5018.73	0.36	**5019**	**5019.00**	0.50
100%							
	02-P100T10M6	**449**	**449.00**	0.05	**449**	**449.00**	0.07
	04-P100T5M12	**899**	**899.00**	0.06	**899**	**899.00**	0.10
	05-P200T5M24	**1667**	1666.97	0.18	**1667**	**1667.00**	0.33
	10-P300T5M36	**2589**	**2589.00**	0.41	**2589**	**2589.00**	0.46
	12-P300T10M18	**1294**	1293.83	0.31	**1294**	**1294.00**	0.58
	13-P400T10M24	**1754**	**1754.00**	0.93	**1754**	**1754.00**	1.18
	15-P400T10M24	**1754**	1753.97	1.07	**1754**	**1754.00**	1.21
	16-P400T5M48	**3509**	3508.33	1.06	**3509**	**3509.00**	1.96
	18-P500T10M30	**2189**	**2189.00**	1.97	**2189**	**2189.00**	3.08
	19-P500T5M80	**5019**	5018.87	1.31	**5019**	**5019.00**	1.88
105%							
	02-P100T10M6	**449**	**449.00**	0.06	**449**	**449.00**	0.08
	04-P100T5M12	**899**	**899.00**	0.14	**899**	**899.00**	0.18
	05-P200T5M24	**1667**	**1667.00**	0.63	**1667**	**1667.00**	0.73
	10-P300T5M36	**2589**	**2589.00**	0.44	**2589**	**2589.00**	0.47
	12-P300T10M18	**1294**	**1294.00**	0.41	**1294**	**1294.00**	0.46
	13-P400T10M24	**1754**	**1754.00**	1.12	**1754**	**1754.00**	1.24
	15-P400T10M24	**1754**	1753.97	1.38	**1754**	**1754.00**	1.69
	16-P400T5M48	**3509**	3508.87	2.34	**3509**	**3509.00**	2.63
	18-P500T10M30	**2189**	**2189**	3.18	**2189**	**2189.00**	3.77
	19-P500T5M80	**5019**	5018.83	4.56	**5019**	**5019.00**	5.56
#Best		30	15		30	30	
$Avg_t(s)$				0.89			1.21

Tables 8 and 9 summarize the computational results of the compared algorithms. One observes that MA performs better than FITS in terms of both the best and the average objective values. Specifically, MA obtained the best result on all 40 instances considering the best objective value, whereas

FITS yielded the best result on 35 out of 40 instances. In terms of the average objective value, MA produced the best result on all the 40 instances, while FITS only obtained the best result on 15 instances. The average computation time to reach the best objective value for all 40 instances indicates that MA is a little slower than FITS. The Wilcoxon signed-rank test (p-value of $4.22e-2$ and $1.20e-5$ in terms of best and average results) supports the superiority of MA over FITS.

8 Conclusions

This work introduced the first population-based memetic algorithm for solving the problem of composing medical crews with equity and efficiency. The proposed algorithm combines a backbone-based crossover for generating new promising solutions and a powerful local optimization procedure mixing feasible search and infeasible search to ensure an effective examination of the search space. The computational experiments demonstrated that our algorithm dominates the best performing approaches in the literature. Specifically, it discovers improved best solutions (new lower bounds) for 303 out of the 320 test instances ($\approx 95\%$). This work thus advances the state of the art for solving the benchmark instances of the problem. More importantly, the publicly available code of our algorithm can be freely used by researchers and practitioners in health care service management (e.g., to compose medical crews with a high equity and efficiency) and could incite more research on this important application as well.

Meanwhile, given that the proposed algorithm is a heuristic method, one does not know how far the reported solutions are from the optimal solutions. As a result, more research is needed to investigate exact and approximation approaches with quality guarantee. Moreover, to further improve the search capacity of the proposed algorithm, it would be interesting to investigate machine learning techniques [6,20,28,32] to make the search process more effective. In addition, the studied problem requires the simultaneous consideration of two conflicting criteria (equity and efficiency). It is thus a special case of the general multi-objective optimization. Consequently, it would be interesting to investigate popular evolutionary multi-objective optimization approach, in particular scalable algorithms [15] to deal with large-scale problems in modern medical service systems. Finally, the search strategies of the proposed algorithm follow general principles and could be applied to design algorithms for other problems with equity and efficiency requirements.

Acknowledgment

We are grateful to the reviewers for their useful comments and suggestions which helped us to significantly improve the paper.

References

[1] Aboueljinane, L., Sahin, E., & Jemai, Z. (2013). A review on simulation models applied to emergency medical service operations. Computers & Industrial Engineering, 66(4), 734-750.

[2] Abualigah, L., Alfar, H. E., Shehab, M., & Hussein, A. M. A. (2020). Sentiment Analysis in Healthcare: A Brief Review. In: Abd Elaziz M., Al-qaness M., Ewees A., & Dahou A. (Eds.), Recent Advances in NLP: The Case of Arabic Language, Studies in Computational Intelligence, Vol 874 (pp. 129-141). Cham: Springer.

[3] Aringhieri, R., Carello, G., & Morale, D. (2007). Ambulance location through optimization and simulation: the case of Milano urban area. XXXVIII Annual Conference of the Italian Operations Research Society Optimization and Decision Sciences, pp. 1-29.

[4] Aringhieri, R. (2008). Models for the efficient team planning at emergency medical service of Milano. In: Xie, X., Lorca, F., & Marcon, E. (Eds.), Operations Research for Health Care Delivery Engineering, Proceeding of the 33rd international conference on Operational Research Applied to Health Service (ORAHS 2007), pp. 281-288.

[5] Aringhieri, R. (2009). Composing medical crews with equity and efficiency. Central European Journal of Operations Research, 17(3), 343-357.

[6] Bello, I., Pham, H., Le, Q. V., Norouzi, M. & Bengio, S. (2016). Neural Combinatorial Optimization with Reinforcement Learning. arXiv preprint arXiv:1611.09940.

[7] Benlic, U., & Hao, J. K. (2011). A Multilevel Memetic Approach for Improving Graph K-Partitions. IEEE Transactions on Evolutionary Computation, 15(5), 624-642.

[8] Cho, C. J. (1998). An equity-efficiency trade-off model for the optimum location of medical care facilities. Socio-Economic Planning Sciences, 32(2), 99-112.

[9] Delgado-Osuna, J. A., Lozano, M., & García-Martínez, C. (2016). An alternative artificial bee colony algorithm with destructive-constructive neighbourhood operator for the problem of composing medical crews. Information Sciences, 326, 215-226.

[10] Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. Journal of Heuristics, 2(1), 5-30.

[11] Galinier, P., & Hao, J. K. (1999). Hybrid Evolutionary Algorithms for Graph Coloring. Journal of Combinatorial Optimization, 3(4), 379-397.

[12] 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.

[13] Glover, F., & Laguna, M. (1998). Tabu Search. In: Du, DZ., & Pardalos, P. M. (Eds.), Handbook of Combinatorial Optimization, pp. 2093-2229. Boston: Springer.

[14] Hao, J. K. (2012). Memetic Algorithms in Discrete Optimization. In: Neri, F., Cotta, C., & Moscato, P. (Eds.), Handbook of Memetic Algorithms, Studies in Computational Intelligence, Vol 379 (pp. 73-94). Berlin: Springer.

[15] Hong, W., Tang, K., Zhou, A., Ishibuchi, H., & Yao, X. (2018). A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization. IEEE Transactions on Evolutionary Computation, 23(3), 525-537.

[16] Jin, Y., & Hao, J. K. (2016). Hybrid evolutionary search for the minimum sum coloring problem of graphs. Information Sciences, 352-353, 15-34.

[17] Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2014). A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review, 42(1), 21-57.

[18] Khodaparasti, S., Maleki, H. R., Bruni, M. E., Jahedi, S., Beraldi, P., & Conforti, D. (2016). Balancing efficiency and equity in location-allocation models with an application to strategic EMS design. Optimization Letters, 10(5), 1053-1070.

[19] Kuhn, H. W. (1955). The Hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2(1-2), 83-97.

[20] Li, Z., Chen, Q., & Koltun, V. (2018). Combinatorial Optimization with Graph Convolutional Networks and Guided Tree Search. In Advances in Neural Information Processing Systems, pp. 539-548.

[21] López-Ibáñez, 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.

[22] Mayhew, L. D., & Leonardi, G. (1982). Equity, Efficiency, and Accessibility in Urban and Regional Health-Care Systems. Environment and Planning A: Economy and Space, 14(11), 1479-1507.

[23] Moeini, R., Soltani-nezhad, M., & Daei, M. (2017). Constrained gravitational search algorithm for large scale reservoir operation optimization problem. Engineering Applications of Artificial Intelligence, 62, 222-233.

[24] Moscato, P., & Cotta, C. (2003). A Gentle Introduction to Memetic Algorithms. In: Glover, F., & Kochenberger, G. A. (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science, Vol 57 (pp. 105-144). Boston: Springer.

[25] 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.

[26] 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.

[27] Qin, J., Xu, X., Wu, Q., & Cheng, T. C. E. (2016). Hybridization of tabu search with feasible and infeasible local searches for the quadratic multiple knapsack problem. Computers & Operations Research, 66, 199-214.

[28] Rennie, S. J. , Marcheret, E., Mroueh, Y., Ross, J. & Goel, V. (2017). Self-Critical Sequence Training for Image Captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7008-7024.

[29] Silva, G. C., Ochi, L. S., & Martins, S. L. (2004). Experimental Comparison of Greedy Randomized Adaptive Search Procedures for the Maximum Diversity Problem. In: Ribeiro, C. C., & Martins S. L. (Eds.), Experimental and Efficient Algorithms, Lecture Notes in Computer Science, Vol 3059 (pp. 498-512). Berlin: Springer.

[30] Smith, H. K., Harper, P. R., & Potts, C. N. (2013). Bicriteria efficiency/equity hierarchical location models for public service application. Journal of the Operational Research Society, 64(4), 500-512.

[31] Sun, W., Hao, J. K., Lai, X., & Wu, Q. (2018). Adaptive feasible and infeasible tabu search for weighted vertex coloring. Information Sciences, 466, 203-219.

[32] Tian, C., Xu, Y., Fei, L., Wang, J., Wen, J., & Luo, N. (2019). Enhanced CNN for image denoising. CAAI Transactions on Intelligence Technology, 4(1), 17-23.

Appendix

This appendix shows the pseudo codes of the proposed memetic algorithm (Algorithm 1) and its composing procedures, including the greedy construction procedure (Algorithm 2), the backbone-based crossover operator procedure (Algorithm 3), the tabu search procedure with feasible and infeasible exploration (Algorithms 4-6).

Algorithm 1 Main scheme of MA

Input: An instance I, time limit t_{max}, population size $|P|$
Output: Best solution S^* found
Initialize population $Pop = \{S^1, S^2, ..., S^{|P|}\}$
$S^* \leftarrow Best(Pop)$
while Time does not exceed t_{max} **do**
  Select two parents $S^i \in Pop$ and $S^j \in Pop$ at random
  $S^o \leftarrow Crossover(S^i, S^j)$ /* Crossover to generate an offspring solution, Section 4 */
  $S^o \leftarrow FITS\ (S^o)$ /* Improve the offspring solution by the tabu search procedure, Section 5 */
  if $f(S^o) > f(S^*)$ **then**
    $S^* \leftarrow S^o$ /* Update the recorded best solution */
  end if
  $Pop \leftarrow Pool\_Updating(S^o, Pop)$ /* Update the population, Section 3.3 */
end while

Algorithm 2 The greedy construction algorithm

Input: Instance I
Output: A feasible initial solution denoted by $S = \{C_0, C_1, ..., C_T\}$
Initializing $C_0 = \{1, ..., n\}$, $C_t = \emptyset$ $(t \in \{1, ..., T\})$, $B_{pt} = 0$ $(t \in \{1, ..., T\}$, $p \in \{1, ..., n\})$
repeat
  $E_{max} \leftarrow \{p \in C_0 : e_p = max_{v \in C_0}\{e_v\}\}$ /* E_{max} represents the set of unallocated individuals with the largest efficiency */
  $S_{min} \leftarrow \{t = 1, ..., T : |C_t| = min\{|C_m|, m = 1, ..., T\}\}$ /* S_{min} denotes the set of crews with the smallest cardinality */
  select $p \in E_{max}$, $t \in S_{min}$ such that B_{pt} is maximized
  $C_0 \leftarrow C_0 \setminus \{p\}$
  $C_t \leftarrow C_t \cup \{p\}$
  for each $q \in \{1, ..., n\}$ **do**
    $B_{pt} = B_{pt} + d_{pq}$
  end for
until Each crew C_t $(t \in \{1, ..., T\})$ has exactly M_t individuals
if S is an infeasible solution **then**
  $repair\_solution(S)$ /* Solution repair procedure, Section 5.4 */
end if
return $S = \{C_0, C_1, ..., C_T\}$

Algorithm 3 The backbone-based crossover

Input: Two parent solutions $S^1 = \{C_0^1, C_1^1, ..., C_T^1\}$ and $S^2 = \{C_0^2, C_1^2, ..., C_T^2\}$
Output: A feasible offspring solution $S^o = \{C_0^o, C_1^o, ..., C_T^o\}$
/* Step 1: Create a partial solution based on backbone */
/* Group matching procedure */
Initializing $C_0^o = \{1, ..., n\}$, $C_t^o = \emptyset$ $(t \in \{1, ..., T\})$
for $t := 1$ *to* T **do**
    Identify a crew C_i^1 of S^1 and a crew C_j^2 of S^2 $(i, j \in \{1, ..., T\})$ such that C_i^1 and C_j^2 have the maximum number of identical individuals
    $C_t^o \leftarrow C_i^1 \cap C_j^2$
    /* Extending procedure */
    $coin \leftarrow 1$; $L_1 \leftarrow C_i^1 \backslash C_t^o$; $L_2 \leftarrow C_j^2 \backslash C_t^o$
    while $|C_t^o| < M_t$ && ($L_1 \neq \emptyset$ or $L_2 \neq \emptyset$) **do**
        if *coin* is odd **then**
            Choose an individual p from L_1 with the highest individual contribution
        else
            Choose an individual p from L_2 with the highest individual contribution
        end if
        $C_i^o \leftarrow C_i^o \cup \{p\}$
        $L_1 \leftarrow L_1 \backslash \{p\}$; $L_2 \leftarrow L_2 \backslash \{p\}$; $coin \leftarrow coin + 1$
    end while
    Remove all individuals in C_t^o from all crews of S^1 and S^2
end for
/* Step 2: complete the partial offspring solution */
$L \leftarrow \{1, ..., n\} \backslash \{C_1^o \cup ... \cup C_T^o\}$
repeat
    Identify an individual p from L and a crew t among the ones having the smallest cardinality such that the individual contribution $IC(S^o, p, t)$ is maximized
    $C_t^o \leftarrow C_t^o \cup \{p\}$; $L \leftarrow L \backslash \{p\}$
until Each crew C_T^o $(t \in \{1, ..., T\})$ has exactly M_t individuals
if S^o is an infeasible solution **then**
    repair_solution(S^o) /* Repair the infeasible solution, Section 5.4 */
end if
return S^o

Algorithm 4 Main scheme of FITS

Input: An instance I, solution S
Output: Best found feasible solution S^*
Begin
$S^* \leftarrow S$ /* S^* records the best feasible solution found so far */
$\beta \leftarrow 0$
while $\beta < \beta_{max}$ **do**
  /* feasible local search phase */
  $(S^1, S^{local\_best}) \leftarrow feasible\_local\_search(S)$ /* Section 5.2 */
  if $f(S^{local\_best}) > f(S^*)$ **then**
    $S^* \leftarrow S^{local\_best}$ /* Update the best recorded solution */
  end if
  /* infeasible local search phase */
  $(S, S^{local\_best}) \leftarrow infeasible\_local\_search(S^1)$ /* Section 5.3 */
  if $f(S^{local\_best}) > f(S^*)$ **then**
    $S^* \leftarrow S^{local\_best}$
  end if
  $\beta \leftarrow \beta + 1$
end while
return S^*

Algorithm 5 Feasible Local Search

Input: Solution S
Output: Final solution S, best solution S^{local_best} found
$S^{local\_best} \leftarrow S$ /* S^{local_best} records the best solution found so far */
$NI \leftarrow 0$ /* number of consecutive iterations without improvement of S^{local_best} */
Initialize *tabu_list*
while $NI < N_{cons}$ **do**
  Choose a best admissible neighboring solution S'
  $S \leftarrow S'$
  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
end while
return $(S,\ S^{local\_best})$

Algorithm 6 Infeasible Local Search

Input: Feasible solution S returned by the feasible local search phase
Output: Final resulting solution S, best feasible solution found during this phase S^{local_best}
$S^{local\_best} \leftarrow S$
$MI \leftarrow 0$
$\beta \leftarrow 0$
Initialize $tabu\_list$
while $MI \leq M_{ILS}$ **do**
  Select a best admissible neighboring solution S' in terms of the extended penalty-based evaluation function F
  $S \leftarrow S'$
  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
  end if
  $MI \leftarrow MI + 1$
  if All previous λ solutions are feasible **then**
    $\beta \leftarrow \beta - 1$
  else if They are all infeasible solutions **then**
    $\beta \leftarrow \beta + 1$
  end if
  if $\beta < 0$ **then** $\beta \leftarrow 0$ **end if**
end while
if S is not a feasible solution **then**
  $repair\_solution(S)$ /* Solution repair procedure, Section 5.4 */
end if
return (S, S^{local_best})