SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2022, Vol. 65 182101:1–182101:16
https://doi.org/10.1007/s11432-019-3017-9

Solving multi-objective constrained minimum weighted bipartite assignment problem: a case study on energy-aware radio broadcast scheduling

Yupeng ZHOU[1], Mingjie FAN[1], Feifei MA[2,3*†] & Minghao YIN[1*†]

[1]*College of Information Science and Technology, Northeast Normal University, Changchun* 130117*, China;*
[2]*Laboratory of Parallel Software and Computational Science, Institute of Software Chinese Academy of Sciences, Beijing* 100190*, China;*
[3]*State Key laboratory of Computer Science, Institute of Software Chinese Academy of Sciences, Beijing* 100190*, China*

Received 3 December 2019/Revised 16 March 2020/Accepted 4 June 2020/Published online 25 July 2022

Abstract This paper proposes a multi-objective constrained minimum weighted bipartite assignment problem (MCMWBAP), which is considered an extension of the classical bipartite matching problem (BMP). We first provide the formulation of the MCMWBAP and prove that it is an NP-hard combinatorial optimization problem. Based on this formulation, multi-objective energy-aware shortwave radio broadcast resource allocation problem (MSRBRAP) application is studied. The goal of this problem is to allocate radio programs to transmission devices to broadcast all radio programs felicitously with a maximized objective of total qualified monitoring sites and a minimized objective of energy consumption. Then, a novel multi-objective hybrid evolutionary algorithm (MOHEA), which is integrated with push and pull initialization, the dynamic resource allocation strategy, and the aggregate local search procedure, is developed to solve the problem. The proposed method is evaluated using two categories of benchmarks for MCMWBAP together with a real scenario case study for MSRBRAP. Furthermore, the key components of MOHEA are analyzed, and the experimental results demonstrate that MOHEA outperforms two classical multi-objective evolutionary algorithms (NSGA-II and MOEA/D), improving working efficiency.

Keywords constrained minimum weighted bipartite assignment, multi-objective energy-aware radio broadcast scheduling, evolutionary algorithms, dynamic resource allocation, aggregate local search

Citation Zhou Y P, Fan M J, Ma F F, et al. Solving multi-objective constrained minimum weighted bipartite assignment problem: a case study on energy-aware radio broadcast scheduling. Sci China Inf Sci, 2022, 65(8): 182101, https://doi.org/10.1007/s11432-019-3017-9

1 Introduction

Given an undirected weighted bipartite graph $G = (U \cup V, E, W_1, W_2)$ with two disjoint vertex sets U, V and the edge set $E \subseteq U \times V$, let $W_1 : E \to \mathbb{R}$ and $W_2 : E \to \mathbb{R}$ be two weighting functions that evaluate the respective weight of each edge $e = (u, v) \in E$. A matching of G is a subset $M \subseteq E$ such that all pairs of edges $\{e_1, e_2\} \in M \times M$ are not attached to the same vertice. However, it is considered extremely strict for the matching restriction to formulate some practical applications. For example, in a shortwave radio broadcast resource allocation problem (SRBRAP) [1], each transmission device can broadcast multiple programs with the purpose of maximizing the sound quality, which distinctly violates the matching constraint. Thus, bipartite assignment is introduced to relax this constraint. Here, an assignment of G attempts to determine a subset A that satisfies the condition $A \subseteq E$. In other words, each vertex can connect to more than one vertex from its adjacent set. Note that the bipartite assignment can be easily solved using greedy algorithms; therefore, additional constraints are introduced to restrict the concurrent selection of some edges. Taking the SRBRAP as an example, as programs overlap the broadcast time and some devices may share hardware components, a constraint is added to

* Corresponding author (email: maff@ios.ac.cn, ymh@nenu.edu.cn)
† Feifei MA and Minghao YIN have the same contribution to this work.

© Science China Press and Springer-Verlag GmbH Germany, part of Springer Nature 2022

ensure that overlapped programs cannot be transmitted on devices with shared components. Thus, the constrained minimum weighted bipartite assignment problem attempts to determine an assignment of G that can minimize the total weights of edges while satisfying constraints C, which formulate the conflicts of all pairs of edges. In addition, as most real-world applications typically have two or more concerned objectives, we further propose a multi-objective constrained minimum weighted bipartite assignment problem (MCMWBAP) by simultaneously optimizing two weighting functions, i.e., a maximized objective of total qualified monitoring sites and a minimized objective of energy consumption.

The weighted bipartite matching problem (BMP) has been studied extensively from both theoretical and practical perspectives. The most popular algorithm for the BMP is the Kuhn-Munkres algorithm [2, 3], which makes this problem solvable in polynomial time. Representative applications include shape matching and objective recognition [4], and the data path allocation problem [5]. Following that, the primary contributions have been related to solving variants of the BMP. One of the most valuable problems is online weighted bipartite matching, where right-hand vertices are given in advance, and left-hand vertices appear online in random order. In response, several online approximation algorithms have been developed to determine which edges should be selected in the matching with a competitive ratio [6]. Online features can be used to describe dynamic combinatorial auctions, where items are assigned to bidders that arrive in an online manner [6]. Recently, a weighted bipartite B-matching problem has been presented, where a maximum degree constraint for each vertex is required for matching. In addition, Chen et al. [7] developed a greedy algorithm with a theoretical guarantee to solve this problem, and they applied the model in an e-commerce scenario. Recently, another study [8] utilized weighted bipartite b-matching to deal with lexical sense alignment in natural language processing applications. In summary, many models and applications have revealed the importance of the bipartite graph optimization problem, and we extend the bipartite graph family by introducing the MCMWBAP.

Based on the MCMWBAP model, a real-world multi-objective energy-aware SRBRAP is first presented in this paper.

In 2016, Ma et al. [1] proposed the SRBRAP from their cooperative project with the State Administration of Press, Publication, Radio, Film, and Television in China. They attempted to replace manual adjustment of staff using pseudo-Boolean constraint and local search techniques. The experimental results demonstrated quality improvement and improved efficiency compared to manual work. The frequency assignment to radio programs was well considered in SRBRAP, and it integrated integer linear programming and satisfiability modulo theories solving, as well as a local search algorithm, to solve the problem effectively [9]. Recently, Wang et al. [10] studied a radio broadcast scheduling problem and proposed a complete (branch-and-bound) algorithm to solve it. In their study, three rules were introduced to select devices, and an upper bound estimation function was introduced to achieve fast convergence. Thus, in this study, we focus on optimizing the multi-objective energy-aware shortwave radio broadcast resource allocation problem (MSRBRAP) as a case study of MCMWBAP in consideration of energy consumption and qualified monitoring sites. Generally speaking, high-power equipment typically generates strong signal strengths, and their energy consumption increases in a corresponding manner. For the sake of compromise, multi-objective optimization techniques are applied to provide a set of nondominated solutions for decision makers. Owing to the characteristics of multi-objective optimization, a group of evolutionary algorithms is preferable because such algorithms can deal with multiple solutions at each iteration. Among these algorithms, multi-objective evolutionary algorithms with a local search procedure have been studied extensively relative to the trade-off between exploitation and exploration [11]. In consideration of communication delay and throughput, the broadcasting scheduling problem is solved by a mixed neural-genetic algorithm [12], which suggests that genetic operators may work to solve the MSRBRAP. To enhance exploitation, the local search method has been integrated into the genetic algorithm to tackle broadcasting scheduling in wireless multi-hop networks [13]. We refer readers to the literature for more information [14].

In this study, we exploit the features of two representative algorithms, i.e., NSGA-II [15] and MOEA/D [16], by combining them. Here, the non-dominating relationship and crowding distance techniques of NSGA-II are integrated into the decomposition-based updating strategies of MOEA/D. In addition, a dynamic computational resource allocation strategy motivated by [17] is applied to determine which search regions should be refined in each generation. Another barrier to this constrained multi-objective optimization problem is dealing with the constraints effectively. In the MCMWBAP, infeasible regions make it difficult to approach the Pareto front; thus, a push and pull strategy is aroused. As suggested by Fan et al. [18], the push operator attempts to rapidly approximate the unconstrained Pareto front without considering the constraints. Then, the pull operator deals with the obtained Pareto front with constraint-handling approaches to make the solution set feasible. As a result, the repaired population begins to evolve from a closer initialized Pareto front. Note that efficient and effective local search is critical to improving solution quality and the distribution of the Pareto front [19]. Therefore, aggregate local search using a weighted-sum decomposition method has been applied to conduct an intensification search. By defining neighborhood structures with Exchange and Reallocate operators, both of which are helpful in the single-objective SRBRAP, we hope to improve solutions and the current Pareto front.

Our primary contributions are summarized as follows. (1) A new constrained minimum weighted bipartite assignment problem is formulated and solved using an evolutionary algorithm. (2) Three valid strategies are designed with the evolutionary framework for this constrained combinatorial optimization problem, including dynamic resource allocation, push and pull initialization, and aggregate local search. (3) A real-world application (MSRBRAP), which considers energy consumption, is solved by the proposed algorithm. (4) Detailed experiments are performed on three categories of benchmarks for the MCMWBAP, and a MSRBRAP case study is presented. The computational results demonstrate that the proposed algorithm is extremely competitive with NSGA-II and MOEA/D algorithms.

The remainder of this paper is organized as follows. Definitions and notations for the MCMWBAP and MSRBRAP are presented in Sections 2 and 3, respectively. Then, the proposed algorithm is introduced in Section 4. In Section 5, we discuss experiments conducted to verify the effectiveness of the multi-objective hybrid evolutionary algorithm (MOHEA) in both academic and practical instances. Finally, Section 6 concludes the paper.

2 Multi-objective constrained minimum weighted bipartite assignment problem

Here, the formulation is given with an example to illustrate the problem. Then, the NP-hardness of the MCMWBAP is proven.

2.1 Problem definitions

Here, the definitions and notations used to describe the problem are presented. Assume $G = (U \cup V, E, W_1, W_2)$ is a bipartite graph, U, V are two vertex sets that satisfy $U \cap V = \Phi$, and E is the edge set that contains all edges connecting U and V. Weight vectors W_1 and W_2 are associated with edge $e \in E$ to indicate different contributions. Note that edge weight plays a more flexible role compared to vertex weight because the vertex weight can be represented by the edge weight, where each edge connected to the same vertex is given the same value as the vertex weight. However, the vertex weight cannot well represent the edge weight because it can only describe the information of one endpoint of the edge. Therefore, without loss of generality, two edge weight vectors are defined in this problem. In addition, a constraint set $C = \{c\langle e, e'\rangle |\ e, e' \in E \text{ and } e \neq e'\}$ is maintained. Here, for each pair of edges $\langle e, e'\rangle$ with $c\langle e, e'\rangle = 1$, at most one edge can be assigned to the solutions. In other words, e and e' cannot be selected simultaneously. For condition $c\langle e, e'\rangle = 0$, no constraints are imported on edges e and e'. Given the above notations, the goal of a MCMWBAP is to allocate each left-hand vertex (U) to a right-hand vertex (V) without breaking the conflict rules such that the total weights of both objectives are minimized concurrently. Figure 1 illustrates how the matching problem is converted to the constrained assignment problem.

The left side of Figure 1 shows a feasible solution to the BMP, where each pair of edges has no intersection. In the middle figure, u_3 and u_4 are permitted to be allocated to v_3 under the assignment definition. Thus, a better solution can be generated by adding two red lines. When the constraint is introduced (i.e., the blue lines) in the right figure, the dashed line is deleted and a red line is joined.

Assume $x_{i,j}$ denotes allocation from vertex u_i to vertex v_j, and $c\langle e, e'\rangle \in \{0, 1\}$ is a constraint acting on two conflicting edges e and e'. Here, $w_{(i,j)}$ and $w'_{(i,j)}$ denote the various weights of the edge $e = (i, j)$. Accordingly, the formulations of the MCMWBAP are given as follows.

(1) Each left-hand vertex u_i should be assigned to a single right-hand vertex v_j,

$$\sum_{j=1}^{|V|} x_{i,j} = 1, \quad \text{for each } u_i \in U. \tag{1}$$

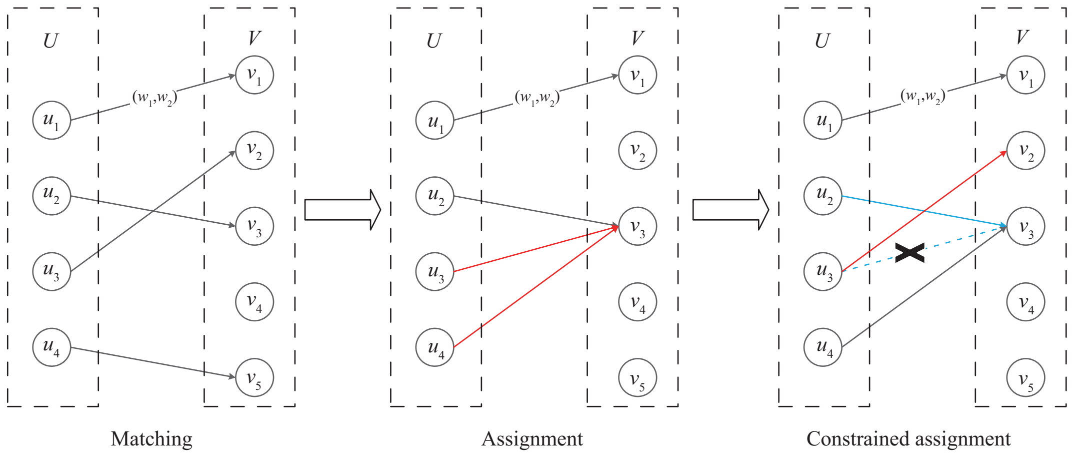

Figure 1 (Color online) Transformation from weighted bipartite matching to constrained weighted bipartite assignment.

(2) Conflicting edges $e = (i, j), e' = (k, l)$ cannot be selected simultaneously,

$$(x_{i,j} + x_{k,l}) \times c\langle e, e'\rangle \leqslant 1, \quad \forall e, e' \in E. \tag{2}$$

(3) The objective is to minimize the total weights of the selected edges:

$$\text{Minimize} \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} x_{i,j} \times w_{(i,j)}, \tag{3}$$

$$\text{Minimize} \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} x_{i,j} \times w'_{(i,j)}. \tag{4}$$

2.2 NP-hardness of MCMWBAP

We prove that the MCMWBAP is NP-hard by reducing from the independent set problem to it, which cannot be solved using exact optimization methods for large-scale instances.

Proposition 1. The multi-objective constrained minimum weighted bipartite assignment problem is NP-hard.

Proof. For an independent set problem (ISP) with $G' = (V', E')$, vertex v is referred to as the adjacent vertex of u if there exists an edge between u and v. The goal of ISP is to select a subset of vertices V such that all adjacent vertices of V are excluded in the candidate solution. In other words, for each pair of adjacent vertices, at most one can be selected in the solution. For the CMWBAP, each edge is considered as a vertex in the ISP, and we add an edge between vertices in G' if the pair of vertices conflict in the CMWBAP. In addition, edges in the CMWBAP with the same left-hand vertex are also linked by edges in the ISP. Then, all weights are set to 1. As a result, the CMWBAP is transformed into a classical NP-complete problem, i.e., the ISP. Clearly, the procedure is a polynomial time reduction process. Assume there exists a solution to the CMWBAP. Then, this solution is also a solution to the original ISP. The construction from CMWBAP to ISP guarantees that conflicting edges will not be selected simultaneously, and one left-hand vertex can be assigned only once. Any vertices in ISP that violate these constraints will be connected by edges, which cannot be selected concurrently. Conversely, if the ISP has a solution, the CMWBAP is also feasible. As a result, the extended multi-objective problem is also NP-hard. Now that we have proven the NP-completeness of the decision version of the MCMWBAP, it easily follows that the problem itself is NP-hard. Similarly, the MSRBRAP is also NP-hard.

3 Multi-objective energy-aware shortwave radio broadcast resource allocation problem

Generally, in a large country like China, there can be thousands of radio transmission devices distributed in dozens of shortwave radio stations. For a radio program, the broadcasting effect in its target area may vary from one device to another. Each radio program should be transmitted with a proper transmission device such that the broadcast satisfies certain criteria in the target area. In addition, it is desirable to maximize the overall broadcasting effect and minimize the total energy consumption of all devices. Given the definition of the MCMWBAP, we introduce a case study called the MSRBRAP, which can be transmitted into the MCMWBAP with small adjustments.

Here, assume the left-hand vertex set U includes all programs to be assigned, and the right-hand vertex set V is the set of all devices. The edges in E represent allocations from programs to devices. To measure the broadcasting effect, each allocation is associated with weight w_1 representing the number of qualified sites if this program is allocated to a device. The total maximum number of qualified sites is considered as the objective of the MSRBRAP. To promote power consumption efficiency, the other objective is to minimize the total energy consumption, where each allocation corresponds to an energy rate. Finally, the constraint in MCMWBAP is not directly formulated in the MSRBRAP. Here, it is formulated through the definitions of competitive programs and conflicting devices. Particularly, each program is labeled with a broadcasting time span $[t_i, t'_i]$ that indicates the start and end times of the given program. When two different programs overlap, they are considered competitive programs. In addition, two distinct devices are referred to as conflicting devices if they share the same transmitter, antenna, or electric switch. Based on this information, competitive programs are not permitted to be allocated to conflicting devices. Therefore, we refine the original constraint (2) as follows:

$$c_{i,k} \wedge c'_{u,v} \rightarrow c_{i,k,u,v}, \tag{5}$$

$$\sum_{x_i,x_k \in U} \sum_{x_u,x_v \in V} (x_{i,u} + x_{k,v} + x_{i,v} + x_{k,u}) \times c_{i,k,u,v} \leqslant 1, \tag{6}$$

where Boolean variables $c_{i,k}$ and $c'_{u,v}$ are for the competitive programs and conflicting devices, respectively, and $c_{i,j,u,v}$ represents the restriction that competitive programs are not permitted to be allocated to conflicting devices. In addition, the objective equation (3) is transformed into a maximization one as follows:

$$\text{Maximize} \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} x_{i,j} \times w_{(i,j)}. \tag{7}$$

To facilitate comprehension, an example is shown in Figure 2. As can be seen, the MSRBRAP can be represented as a variant of the weighted bipartite graph, where P and D are two disjoint sets of vertices. If allocation $\langle P_i, D_j \rangle$ is admissible, there is an edge with pairwise weights (W, W') connecting corresponding vertices. Here, W is the broadcasting effect, W' is the energy consumption per allocation, and the vertices connected by dashed lines represent competitive programs or conflicting devices.

4 Multi-objective hybrid evolutionary algorithms for MCMWBAP

The MOHEA is introduced in this section. The MOHEA framework is described first with detailed explanations. Then, the key components of the MOHEA, i.e., fast push and pull initialization, dynamic resource allocation strategy, and aggregate local search, are described.

4.1 Framework of MOHEA

The framework of MOHEA is described by the pseudocode presented in Algorithm 1.

At the beginning of the algorithm, weight sets W_1 and W_2 are normalized to $[0, 1]$ (line 1). Next, the MOHEA decomposes the MCMWBAP into N subproblems by initializing the uniform weight vector λ as MOEA/D does (line 2). Based on the Manhattan distance of λ, each subproblem is associated with T closest subproblems as its neighborhoods. Then, GP is initialized by the push and pull initialization operator to better cope with the constraints (line 3). The initial external population EP is set to GP, and the offspring population OP is initialized as $\emptyset$ (lines 4 and 5).

The main loop of the MOHEA iterates until the cut-off time is reached (lines 6–25). At each iteration, MOHEA allocates computing resource to various subproblems dynamically through the experience of the competition. Specifically, one subproblem (i.e., an individual) is selected to evolve each time (line 8). When this individual is determined, two parents are selected randomly from its T neighbors in GP and EP, respectively (lines 9 and 10). Next, the crossover operator is applied to combine good genes, and the mutation operator is applied to strengthen the diversity of the population (lines 11 and 12). Then,

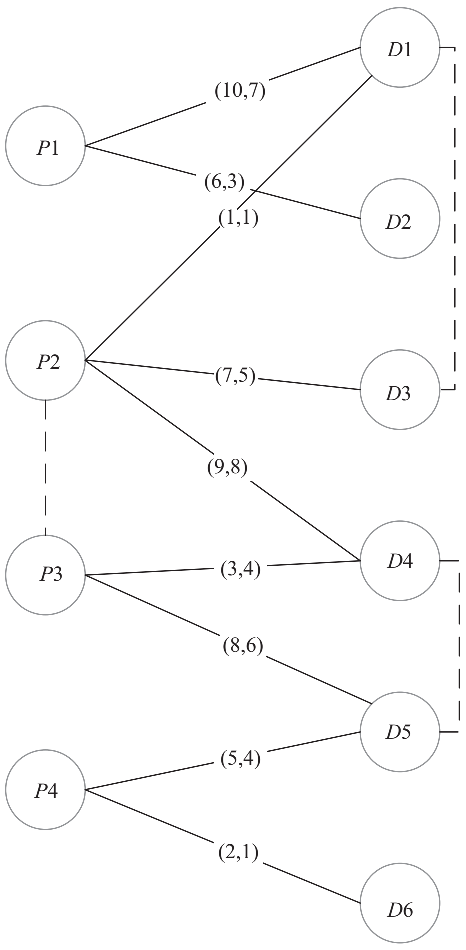

Figure 2 Example of the MSRBRAP.

Algorithm 1 Framework of MOHEA

```
1: $(W_1, W_2) \leftarrow$ initializeEdgeWeight($W_1, W_2$);
2: $\lambda \leftarrow$ initializeWeightVectors($N$);
3: GP $\leftarrow$ PushPullInitialize($\lambda$); // Subsection 4.2
4: EP $\leftarrow P$;
5: OP $\leftarrow \emptyset$;
6: while (elapsed_time < cutoff_time) do
7:     for $n = 1$ to $N$ do
8:         $j$ = selete_a_subproblem(); // Subsection 4.3
9:         $P_1 \leftarrow$ random_neighbor(GP, $\lambda^j, T$);
10:        $P_2 \leftarrow$ random_neighbor(EP, $\lambda^j, T$);
11:        OP$_i \leftarrow$ crossover($P_1, P_2$);
12:        OP$_i \leftarrow$ mutation(OP$_i$);
13:    end for
14:    if No_Excellent_Generated($L$) then
15:        for $i = 1$ to pop_size do
16:            OP$_i \leftarrow$ local_search(OP$_i$); // Subsection 4.4
17:        end for
18:    end if
19:    for $i = 1$ to pop_size do
20:        update_neighborhood(GP, OP$_i$, $T$);
21:    end for
22:    EP $\leftarrow$ OP + EP;
23:    EP $\leftarrow$ fast_non_dominanted_sort(EP);
24:    EP $\leftarrow$ first_N_ind(EP);
25: end while
26: Return EP;
```

the algorithm performs fast aggregated local search to enhance the exploitation of MOHEA under the condition that no excellent individuals were generated in previous L generations (lines 14–18). GP is then updated by those newly generated solutions through a neighborhood updating procedure (lines 19–21). Finally, the external population is updated by the fast nondominated sorting process on the union set of EP and OP, and only the first N individuals are selected (lines 22–24). When the cut-off time is reached, the external population EP is returned as the result of the MCMWBAP.

4.2 Push and pull initialization

For evolutionary algorithms, initialization quality plays a key role in the entire process, especially for constrained optimizations. Generally, abundant diversity in the initialized population is desirable to cover the searching space as much as possible. Thus, random initialization is often employed in evolutionary algorithms by ignoring each individual's fitness value, which improves the initial solutions in the evolutionary process. However, randomized solutions typically result in slow convergence, which is a vital challenge in time-sensitive environments. In addition, some infeasible regions formed by constraints are considered to be the obstacle of advancing the frontier; thus, they should be considered carefully. Based on the above considerations, push and pull initialization (PPI) is proposed, which greedily pushes the frontier to the true PF of the unconstrained version of the MCMWBAP, and then pulls it back to feasible regions by a repair operator in random order. As a result, the initialized population can generate stochastic solutions that can approach the true PF better.

Unlike the generic evolutionary algorithms used in the push search reported in [18], the push operator in the MOHEA is an a priori heuristic. To be specific, each vertex of U can select those vertices in V independently if all constraints are ignored. Thus, for a single vertex u in U, we greedily select vertex v in V with the minimum aggregated weights of $e \in E$. As a result, it is easy to see that the obtained solution S is an exact solution of the unconstrained aggregated single-objective problem. Algorithm 2 outlines the process of the push operator. To generate feasible solutions, a pull operator is applied immediately after the push operator, whose aim is to produce a high-quality as well as diversified population within acceptable time. The pull operator is described in Algorithm 3.

Algorithm 2 Push operator

```
1: UP = $\emptyset$;
2: for each weight vector $\lambda^i \in \lambda$ do
3:     $S_{uc} = \emptyset$;
4:     for each element $u$ in $U$ do
5:         $V' = \{v|(u,v)\in E\}$;
6:         $v = \arg\min_{v\in V'}\{g^{ws}((u,v)|\lambda^i)\}$;
7:         $S_{uc} = S_{uc} \cup \{(u,v)\}$;
8:     end for
9:     UP = UP $\cup\{S_{uc}\}$;
10: end for
11: return $UP$;
```

Algorithm 3 Pull operator

```
1: $P_c = \emptyset$;
2: for $(i = 1; i \leqslant |\mathrm{UP}|; i = i + 1)$ do
3:     $S_c = \emptyset$;
4:     $S_{uc} = \mathrm{UP}_i$;
5:     while $S_{uc} \neq \emptyset$ do
6:         $(u,v) = \mathrm{random\_edge}(S_{uc})$;
7:         if Consist($(u,v), S_c$) then
8:             $S_c = S_c \cup \{(u,v)\}$;
9:         else
10:            $V' = \{v'|(u,v')\in E$ and Consist$((u,v'),S_c) = $ true$\}$;
11:            $v' = \arg\min_{v'\in V'}\{g^{ws}((u,v')|\lambda^i)\}$;
12:            $S_c = S_c \cup \{(u,v')\}$;
13:        end if
14:        $S_{uc} = S_{uc} \setminus\{(u,v)\}$;
15:    end while
16:    $P_c = P_c \cup\{S_c\}$;
17: end for
18: return $P_c$;
```

In the process of the pull operator, all allocations in unconstrained solution S_{uc} are evaluated to determine whether they break the conflicting rules. Here, edge (u,v) is added to S_c directly if it does not conflict with any other allocations in S_c (lines 7 and 8); otherwise, it will be repaired greedily by selecting the best qualified edge (u,v') that is consistent with S_c (lines 9–13). After all solutions have been examined and adjusted, the current individual becomes feasible and gets close to the true PF.

Figure 3 shows the initialization process of the MOHEA. First, the push operator quickly obtains the unconstrained true PF. Then, the pull operator pulls those solutions over the infeasible region and arrives at a close position. The MOHEA then performs evolution procedures to approach the final constrained PF. Note that each allocation is checked in random order; thus, the diversity of the population is maintained. In addition, the uniform weight vectors used by PPI ensure a good population distribution in most cases. These push and pull operators are efficient because they only scan individuals once and replace them in a greedy mode.

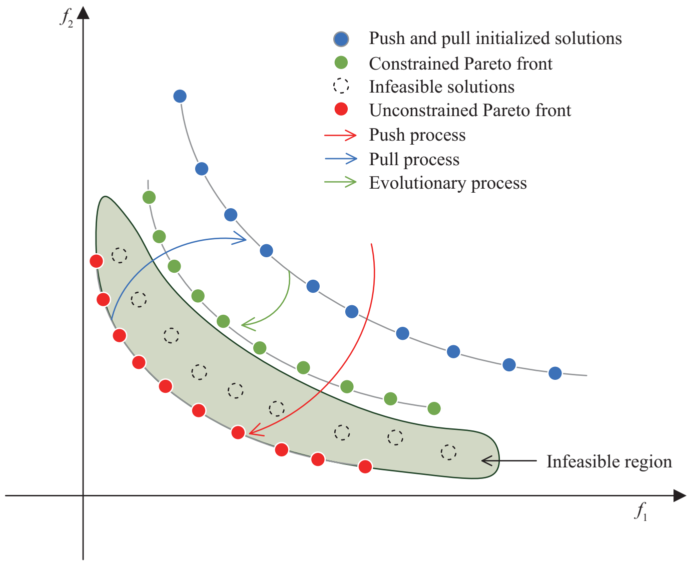

Figure 3 (Color online) Push and pull initialization.

4.3 Dynamic resource allocation strategy

As resource competition exists extensively in nature, an individual who is stronger typically gains more resources. In the basic MOEA/D, all subproblems evolve together and are treated equally; therefore, computational resources are distributed evenly among the various subproblems. However, different subproblems may have different contributions to optimization at different stages. Thus, the dynamic resource allocation strategy is applied in the MOHEA to exploit elites.

One solution can be considered excellent if its fitness and distance from others are promising. To filter these solutions, all children generated in this iteration are merged into EP, and the best N individuals are selected to be the new EP by fast nondominated sorting [15]. Then, the surviving children in EP are referred to as excellent solutions. Next, the subproblems that yield excellent solutions are likely to gain more computational resources in the next L generations. Here, parameter L controls the dynamic allocation frequency. Specifically, roulette wheel selection is applied to determine the subproblem i to be optimized in generation g according to the following equation:

$$p_i^g = \begin{cases} \frac{1}{N}, & 1 \leqslant g \leqslant L, \\ \frac{\mathrm{ES}_i^g + \Delta}{\sum_{j=1}^N (\mathrm{ES}_j^g + \Delta)}, & g > L, \end{cases} \tag{8}$$

$$\mathrm{ES}_i^g = \sum_{k=g-L}^{g-1} \mathrm{es}_i^k, \quad i = 1, 2, \ldots, N, \tag{9}$$

$$\Delta = \varepsilon \sum_{i=1}^N \mathrm{ES}_i^g, \tag{10}$$

where es_i^k is the number of excellent solutions generated by subproblem i in the kth generation, and Δ is a perturbation factor with $\varepsilon = 0.002$. In each generation, the MOHEA samples subproblems N times according to the probability p_i^g calculated by (8). Note that the selection probability p_i^g is equal and static within the first L generations because insufficient information is provided for learning, and MOHEA is iterated without distinction. When g exceeds L, the solution information of the previous L iterations is gathered (denoted ES_i^g) to affect p_i^g. Briefly, all subproblems compete to generate a single child through their performance in the previous L iterations. As the probability is updated along with the generation, only better individuals survive.

4.4 Fast aggregate local search procedure

Local search is an efficient method to improve one individual by exploiting its surroundings. Here, a fast aggregate local search technique is proposed for the MCMWBAP to quickly improve solution quality. First, two neighborhood operators are introduced, i.e., the Exchange and Reallocate operators.

Exchange operator. The Exchange operator (Algorithm 4) focuses on internal optimization by rationalizing the selected vertices. Note that no other vertices are considered in this structure. In each iteration, the operator attempts to select two distinct edges with no shared vertices, and then exchanges their right-hand vertices. To ensure the feasibility and convergence of solution S, the following conditions should be satisfied. (1) Each left-hand node should only be allocated to a single attachable right-hand node (line 6). (2) Both allocations after Exchange should be consistent with others (line 15). (3) Only those swaps with positive benefits are acceptable (line 10), where the benefit is calculated according to the gap of the aggregate fitness functions (line 9). The first two conditions ensure the feasibility of the exchange, and the third condition guarantees convergence. Once the Exchange operator finds two feasible and acceptable left-hand nodes to exchange, it performs the operator and returns a better solution (line 16). Otherwise, the operator will terminate and return the original solution.

Reallocate operator. The Reallocate operator is complementary to the Exchange operator, which stresses external optimizations. It attempts to improve allocation from a reallocation among vertices not in the current solution. As Algorithm 5 shows, for each edge (u, v) in the solution, each right-hand vertex v is evaluated to determine whether it can be replaced by vertex v' such that edge (u, v') is more suitable for the current solution. Here, the Reallocate operator uses a different searching trajectory compared to the Exchange operator and can help avoid the local optimum. Here, the calculation of the benefit and consistent inspection are similar to that of the Exchange operator. Note that the Reallocate operator terminates immediately when it finds a better solution.

Algorithm 4 Exchange

```
1: visited ← $\emptyset$;
2: while $|S\backslash\text{visited}| > 0$ do
3:     $(u, v) \leftarrow$ random_alloc($S\backslash$visited);
4:     visited = visited $\cup \{(u, v)\}$;
5:     for each allocation $(u', v') \in S\backslash$visited do
6:         if attachable($(u, v')$) and attachable($(u', v)$) then
7:             old_pair ← $\{(u, v), (u', v')\}$;
8:             new_pair ← $\{(u, v'), (u', v)\}$;
9:             benefit = $g^{ws}(\text{old\_pair}|\lambda^j) - g^{ws}(\text{new\_pair}|\lambda^j)$;
10:            if benefit <= 0 then
11:                Continue;
12:            end if
13:            $S_1 = S\backslash\text{old\_pair} \cup \{(u, v')\}$;
14:            $S_2 = S\backslash\text{old\_pair} \cup \{(u', v)\}$;
15:            if Consist($(u, v'), S_2$) $\wedge$ Consist($(u', v), S_1$) then
16:                Return ($S\backslash$old_pair) $\cup$ new_pair;
17:            end if
18:        end if
19:    end for
20: end while
21: Return $S$;
```

Algorithm 5 Reallocate

```
1: for each device allocation $(u, v) \in S$ do
2:     old_alloc = $(u, v)$;
3:     for each right node $v' \in$ attachable_nodes($u$) do
4:         new_alloc = $(u, v')$;
5:         benefit = $g^{ws}(\text{old\_alloc}|\lambda^j) - g^{ws}(\text{new\_alloc}|\lambda^j)$;
6:         if benefit <= 0 then
7:             Continue;
8:         end if
9:         if Consit($S\backslash$old_alloc, new_alloc) then
10:            Return ($S\backslash\{$old_alloc$\}) \cup \{$new_alloc$\}$;
11:        end if
12:    end for
13: end for
14: Return $S$;
```

Local search framework. Given the above descriptions of the basic operators, the aggregate local search framework is described in Algorithm 6. During the search process, only a single operator is employed each time. Here, two flags, i.e., CanExch and CanReal, are used to indicate whether the Exchange and Reallocate operators can be applied, respectively. In addition, the oper variable is used to indicate whether Exchange (oper = 0) or Reallocate (oper = 1) is employed.

First, both switches are initialized as true, oper as 0, and the counters SearchNum as 0 (line 1). Then, the loop iterates until SearchNum reaches MaxSearchNum (lines 2–30). In each loop, the algorithm determines whether internal or external optimization is executed on the solution S (lines 3–11). According to the selected oper, related operations are executed as described above, i.e., Exchange (line 13) or Reallocate (line 21). In addition, the update procedure is accomplished by weighted-sum decomposition of the multiple objective functions (lines 14–16 and 22–24). Note that each local search is associated with a weight vector λ^j, which represents a specific optimized direction of the multi-objective optimization problems (MOP). The operator is forbidden in the next loop when no improved solutions are produced, which helps avoid repetitive search procedures, thereby making the local search efficient (lines 18 and 26). At the end of the local search, S is returned as the best solution under the weight vector λ^j (line 31).

Algorithm 6 Local search

```
1: CanExch = CanReal = true; oper = SearchNum = 0;
2: while (SearchNum < MaxSearchNum) do
3:     if CanExch = false and CanReal = false then
4:         break;
5:     else if CanExch = true and CanReal = true then
6:         oper = rand{0, 1}; //Randomly select 0 or 1
7:     else if CanExch = true then
8:         oper = 0;
9:     else
10:        oper = 1;
11:    end if
12:    if oper = 0 then
13:        $S'$ = Exchange($S, \lambda^j$);
14:        if $g^{ws}(S'|\lambda^j) < g^{ws}(S|\lambda^j)$ then
15:            $S = S'$;
16:            CanReal = true;
17:        else
18:            CanExch = false;
19:        end if
20:    else
21:        $S'$ = Reallocate($S, \lambda^j$);
22:        if $g^{ws}(S'|\lambda^j) < g^{ws}(S|\lambda^j)$ then
23:            $S = S'$;
24:            CanExch = true;
25:        else
26:            CanReal = false;
27:        end if
28:    end if
29:    SearchNum = SearchNum + 1;
30: end while
31: Return $S$;
```

5 Experimental studies

In this section, we first describe three categories of MCMWBAP benchmarks to fully test the proposed algorithm and analyze the characteristics of each instance. Then, the parameter settings are described, and two popular metrics for evaluation are introduced. All details can be found online[1]. Here, we compared MOHEA to competitive algorithms to evaluate and compare its performance. In addition, the effectiveness of the key components of MOHEA are demonstrated with further discussion. Finally, a real-world case, i.e., the MSRBRAP, is examined to improve broadcasting quality and reduce energy consumption.

All experiments were conducted 10 times on a computer with Intel(R) Core(TM) i7-6700 (3.4 GHz and 8 GB RAM) processors running the Windows 10 operating system. Note that the time limit for all optimizers was 3600 s.

5.1 Comparison with different algorithms

To demonstrate the superiority of MOHEA, a set of MOEAs were compared, including MOEA/D-WS, MOEA/D-TF, MOEAD-BI [16], and NSGAII [15]. In addition, a heuristic method PPI is also recorded, which is proposed for the initialization of the MOHEA. The HV and IGD results are shown in Tables 1 and 2, respectively.

The results obtained with † indicate that the algorithm is significantly inferior or superior to MOHEA. The last row in these tables summarizes the statistical Mann-Whitney U test with a 0.05 significance level, where "S-D-I" indicates that the compared algorithm is superior to, not significantly different from, or inferior to the proposed MOHEA. In addition, the PF distribution figures are illustrated to compare different methods explicitly. From the results, we can conclude the following.

(1) MOHEA outperformed PPI on all s-type instances, medium-sized w-type instances, and the MovieLens 100k network. There was no significant difference on the Crime network and those small and large w-type MCMWBAP instances.

This situation is further analyzed in Figure A1. (i) For the Crime network and w-type MCMWBAP instances with small scale, PPI could produce a high-quality PF that was close to the true PF. Thus,

1) http://ai.nenu.edu.cn/yinmh/data/supplementary.pdf.

Table 1 Mean HV values obtained by MOHEA and compared algorithms on MCMWBAP

Instances	MOHEA	PPI	MOEAD-WS	MOEAD-TF	MOEAD-BI	NSGAII
MCMWBAP-1-s	**5.8396E−1**	4.4892E−1†	5.5722E−1	5.3716E−1	5.2621E−1	5.2545E−1
MCMWBAP-2-s	**6.0079E−1**	4.6342E−1†	4.7437E−1†	4.3876E−1†	4.1689E−1†	4.3476E−1†
MCMWBAP-3-s	**6.4629E−1**	5.2771E−1†	4.5262E−1†	4.1568E−1†	3.9871E−1†	4.1363E−1†
MCMWBAP-4-s	**6.5023E−1**	5.4152E−1†	4.0005E−1†	3.6537E−1†	3.4920E−1†	3.5310E−1†
MCMWBAP-5-s	**6.4556E−1**	5.4352E−1†	3.6390E−1†	3.2740E−1†	3.1224E−1†	3.0027E−1†
MCMWBAP-6-s	**6.2836E−1**	5.4641E−1†	3.3146E−1†	2.9415E−1†	2.7947E−1†	2.5233E−1†
MCMWBAP-7-s	**6.1880E−1**	5.4629E−1†	3.0913E−1†	2.7256E−1†	2.5770E−1†	2.2373E−1†
MCMWBAP-8-s	**6.0409E−1**	5.4773E−1†	2.8785E−1†	2.5365E−1†	2.4270E−1†	2.1122E−1†
MCMWBAP-1-w	**9.3708E−1**	9.3115E−1	8.2378E−1†	7.0098E−1†	6.5082E−1†	6.3499E−1†
MCMWBAP-2-w	**9.9634E−1**	9.9511E−1	6.0068E−1†	5.6900E−1†	5.7419E−1†	6.0225E−1†
MCMWBAP-3-w	**9.9697E−1**	9.9540E−1†	4.2184E−1†	3.9529E−1†	3.7825E−1†	4.7223E−1†
MCMWBAP-4-w	**9.9678E−1**	9.9473E−1†	2.6014E−1†	2.4907E−1†	2.1046E−1†	3.2539E−1†
MCMWBAP-5-w	**9.9643E−1**	9.9475E−1†	1.8761E−1†	1.7442E−1†	1.5319E−1†	1.7565E−1†
MCMWBAP-6-w	**9.9547E−1**	9.9423E−1†	1.1682E−1†	1.1025E−1†	9.1071E−2†	4.6930E−2†
MCMWBAP-7-w	**9.9404E−1**	9.9364E−1	8.4797E−2†	7.8850E−2†	6.7609E−2†	1.9429E−2†
MCMWBAP-8-w	**9.9338E−1**	9.9324E−1	6.7013E−2†	5.7740E−2†	4.7944E−2†	1.3281E−2†
Crime network	**7.7949E−1**	7.7677E−1	7.3294E−1†	7.0555E−1†	6.8073E−1†	6.9036E−1†
MovieLens 100k network	**9.8414E−1**	9.1082E−1†	2.2023E−1†	2.1065E−1†	1.8034E−1†	2.7058E−1†
U test (S-D-I)	–	0-5-13	0-1-17	0-1-17	0-1-17	0-1-17

Table 2 Mean IGD values obtained by MOHEA and compared algorithms on MCMWBAP

Instances	MOHEA	PPI	MOEAD-WS	MOEAD-TF	MOEAD-BI	NSGAII
MCMWBAP-1-s	**3.1707E−3**	7.9421E−2†	1.8986E−2†	6.7727E−2†	8.3482E−2†	7.7280E−2†
MCMWBAP-2-s	**2.1645E−3**	7.0450E−2†	1.2978E−1†	1.9602E−1†	2.2492E−1†	2.0306E−1†
MCMWBAP-3-s	**1.8449E−3**	6.8309E−2†	1.9242E−1†	2.5507E−1†	2.7620E−1†	2.6019E−1†
MCMWBAP-4-s	**1.4899E−3**	6.4362E−2†	2.5396E−1†	3.1308E−1†	3.2846E−1†	3.2882E−1†
MCMWBAP-5-s	**9.5887E−4**	5.9059E−2†	2.9071E−1†	3.4397E−1†	3.6067E−1†	3.7185E−1†
MCMWBAP-6-s	**1.0091E−3**	6.8174E−2†	3.1799E−1†	3.7376E−1†	3.9054E−1†	4.1418E−1†
MCMWBAP-7-s	**6.3509E−4**	7.0581E−2†	3.5131E−1†	4.0799E−1†	4.2384E−1†	4.5523E−1†
MCMWBAP-8-s	**4.5254E−4**	7.7587E−2†	3.6480E−1†	4.1507E−1†	4.2797E−1†	4.5682E−1†
MCMWBAP-1-w	**7.2343E−4**	1.1379E−2†	7.8226E−2†	1.8931E−1†	2.2878E−1†	2.0758E−1†
MCMWBAP-2-w	**1.2414E−4**	6.7648E−3†	3.3816E−1†	3.2441E−1†	3.0284E−1†	2.7151E−1†
MCMWBAP-3-w	**8.3947E−6**	2.2745E−2†	5.2068E−1†	5.0292E−1†	5.1670E−1†	4.0213E−1†
MCMWBAP-4-w	**0.0000E+0**	2.5956E−2†	7.2475E−1†	6.8890E−1†	7.2909E−1†	5.6196E−1†
MCMWBAP-5-w	**0.0000E+0**	2.3233E−2†	8.1030E−1†	8.0161E−1†	8.3201E−1†	7.8021E−1†
MCMWBAP-6-w	**6.8671E-07**	1.3207E−2†	9.1679E−1†	9.1805E−1†	9.5834E−1†	1.0582E+0†
MCMWBAP-7-w	**0.0000E+0**	6.2384E−3†	9.9231E−1†	9.8518E−1†	1.0184E+0†	1.1625E+0†
MCMWBAP-8-w	**0.0000E+0**	2.1145E−3†	1.0207E+0†	1.0341E+0†	1.0597E+0†	1.1904E+0†
Crime network	**8.1571E−4**	3.2781E−3†	4.9216E−2†	1.0664E−1†	1.5444E−1†	1.2182E−1†
MovieLens 100k network	**0.0000E+0**	6.7626E−2†	7.3852E−1†	7.3472E−1†	7.4854E−1†	7.0154E−1†
U test (S-D-I)	–	0-0-18	0-0-18	0-0-18	0-0-18	0-0-18

based on the initialization of PPI, the MOHEA can explore the search space with little improvement. This is the main reason that there is no significant difference in the HV indicator for small-scale instances. Of course, this also explains why the PPI concept is effective. For the large w-type MCMWBAP instances, the MOHEA required too much time in each iteration and converged slowly. Therefore, given the limited time, PPI achieved performance that was similar to that of the MOHEA. (ii) For w-type MCMWBAP instances of medium size, the results of PPI gathered in the origin region, and the MOHEA could seek a more dispersed PF through its exploration capacity. For s-type MCMWBAP instances, Figure 4 shows the limitation of the weighted-sum decomposition methods. Owing to the evolutionary process and local search procedure, the MOHEA can deal with the challenge better than PPI. For the MovieLens 100k network, Figure 4 shows that the PF obtained by the MOHEA is denser and better than that obtained by PPI. Although it is easier for EAs to be trapped in a local optimum, both PPI and MOHEA approximated the true PF well, which demonstrates the effectiveness of PPI.

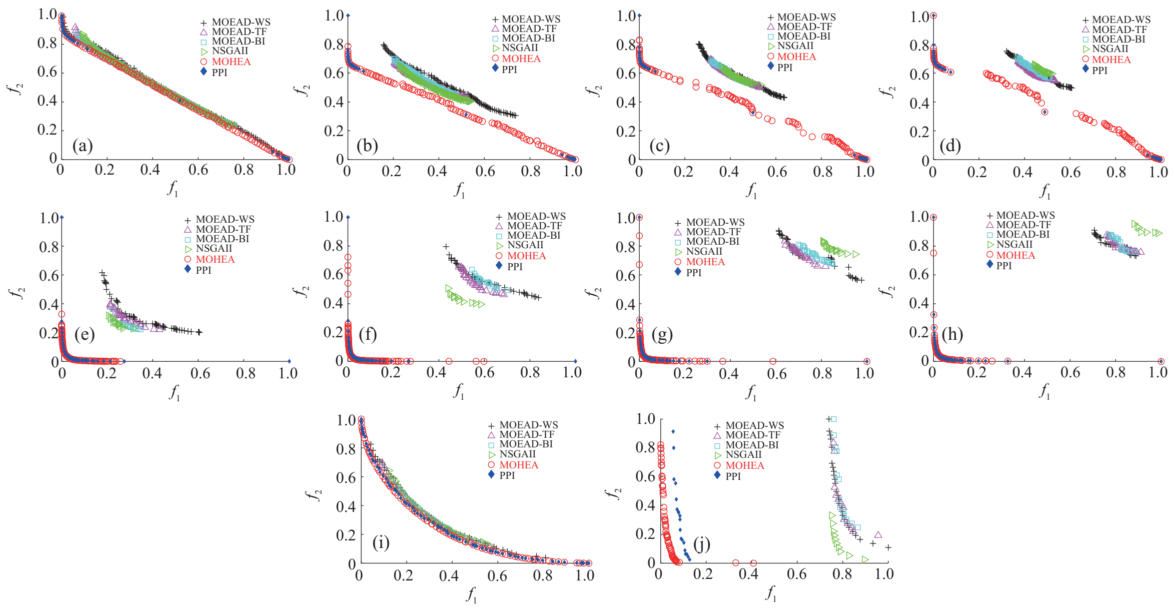

Figure 4 (Color online) Nondominated solutions obtained by various algorithms on 10 instances. (a) MCMWBAP-1-s; (b) MCMWBAP-3-s; (c) MCMWBAP-5-s; (d) MCMWBAP-7-s; (e) MCMWBAP-2-w; (f) MCMWBAP-4-w; (g) MCMWBAP-6-w; (h) MCMWBAP-8-w; (i) Crime network; (j) MovieLens 100k network.

(2) Compared to other MOEAs, the MOHEA worked best on all instances relative to IGD, and most instances regarding HV except for MCMWBAP-1-s, on which MOHEA is considered to be statistically equal to other MOEAs. (i) For MCMWBAP-1-s, which is a small-sized instance with simple structures but is difficult for PPI, MOHEA performed slightly better than others for the HV metric. Figure 4 shows that the distribution of the PF generated by the MOHEA has a wider spread than others. (ii) For the other large-scale s-type MCMWBAP instances, the MOHEA is clearly superior to other algorithms relative to both the distribution and convergence of the PF. For w-type MCMWBAP instances, PPI could obtain excellent solutions along the PF, thereby making MOHEA the most promising algorithm. As the scale increased, the gap between MOHEA and others increased. The MOHEA demonstrated the best performance on the Crime network instance because it found a wider PF. In addition, the compared MOEAs obtained poor performance relative to optimizing the first objective of the MovieLens 100k network.

5.2 Analysis of key MOHEA components

To clarify the performance of the major MOHEA components, three variants were obtained by deleting each key MOHEA component, i.e., dynamic resource allocation, aggregate local search, and push and pull initialization. Detailed descriptions are given as follows.

- MOHEA$_{\mathrm{noDRA}}$: This alternative version of MOHEA uses a static method to allocate computing resources; i.e., each subproblem is allocated to the same computing resource.
- MOHEA$_{\mathrm{noLS}}$: This alternative version of MOHEA does not include the local search process.
- MOHEA$_{\mathrm{noPPI}}$: This alternative version of MOHEA uses random initialization rather than PPI.

Tables 3 and 4 show the results of all MOHEA variants relative to HV and IGD, respectively.

Here, compared to MOHEA$_{\mathrm{noDRA}}$, MOHEA demonstrates its advantage relative to solving s-type MCMWBAP instances, especially for large cases. As was analyzed previously, the central part of the s-type PF is more difficult to explore. Therefore, the dynamic resource allocation component offers more computing resources to subproblems that focus on optimizing the challenging part of the PF.

From the comparison of MOHEA and MOHEA$_{\mathrm{noLS}}$, we observe that MOHEA outperformed MOHEA$_{\mathrm{noLS}}$ on six instances relative to the HV indicator and 13 instances in terms of the IGD indicator. These results reveal the benefits of the local search procedure to move forward the PF and maintain a good distribution. There is no doubt that local search will influence the efficiency of MOHEA more or less; however, the overall performance of MOHEA is promoted distinctly.

Table 3 HV results of strategy validation on MCMWBAP instances

Instances	MOHEA	MOHEA$_{\mathrm{noDRA}}$	MOHEA$_{\mathrm{noLS}}$	MOHEA$_{\mathrm{noPPI}}$
MCMWBAP-1-s	5.6750E−1	5.6592E−1	5.6479E−1	**5.6778E−1**
MCMWBAP-2-s	**5.7376E−1**	5.7010E−1	5.6935E−1	5.7358E−1
MCMWBAP-3-s	6.4711E−1	6.4068E−1	6.4005E−1	**6.4734E−1**
MCMWBAP-4-s	**6.5021E−1**	6.3991E−1†	6.3943E−1†	6.4947E−1
MCMWBAP-5-s	**6.4549E−1**	6.3274E−1†	6.3465E−1	6.4160E−1
MCMWBAP-6-s	6.2843E−1	6.1631E−1†	6.2683E−1	**6.3063E−1**
MCMWBAP-7-s	**6.1876E−1**	6.0183E−1†	6.1669E−1	6.1680E−1
MCMWBAP-8-s	6.0413E−1	5.9606E−1†	**6.0918E−1**	5.9545E−1
MCMWBAP-1-w	9.1505E−1	9.1498E−1	**9.1525E−1**	9.1507E−1
MCMWBAP-2-w	9.9551E−1	9.9546E−1	9.9406E−1	**9.9553E−1**
MCMWBAP-3-w	**9.9693E−1**	9.9690E−1	9.9537E−1†	9.9687E−1
MCMWBAP-4-w	**9.9674E−1**	9.9673E−1	9.9469E−1†	9.9666E−1
MCMWBAP-5-w	**9.9637E−1**	9.9635E−1	9.9471E−1†	9.9623E−1
MCMWBAP-6-w	**9.9546E−1**	9.9523E−1	9.9421E−1†	9.9443E−1†
MCMWBAP-7-w	**9.9403E−1**	9.9391E−1	9.9362E−1	9.8392E−1†
MCMWBAP-8-w	**9.9337E−1**	9.9336E−1	9.9323E−1	9.6067E−1†
Crime network	7.7949E−1	**7.7956E−1**	7.7716E−1	7.7952E−1
MovieLens 100k network	**9.5060E−1**	9.4470E−1	7.9954E−1†	9.3667E−1†
U test (S-D-I)	–	0-13-5	0-12-6	0-14-4

Table 4 IGD results of strategy validation on MCMWBAP instances

Instances	MOHEA	MOHEA$_{\mathrm{noDRA}}$	MOHEA$_{\mathrm{noLS}}$	MOHEA$_{\mathrm{noPPI}}$
MCMWBAP-1-s	3.6735E−3	4.3584E−3†	4.9839E−3†	**3.4894E−3**
MCMWBAP-2-s	**3.3629E−3**	5.2651E−3†	6.2142E−3†	3.5836E−3
MCMWBAP-3-s	**3.2633E−3**	7.1112E−3†	8.2386E−3†	3.3184E−3
MCMWBAP-4-s	**4.2192E−3**	8.7073E−3†	1.0220E−2†	4.6876E−3
MCMWBAP-5-s	**4.3455E−3**	8.9654E−3†	9.8731E−3†	6.0900E−3
MCMWBAP-6-s	**5.4491E−3**	1.1240E−2†	6.6261E−3	6.9909E−3
MCMWBAP-7-s	6.6201E−3	1.2629E−2†	**6.2984E−3**	1.1045E−2†
MCMWBAP-8-s	8.1218E−3	1.3476E−2†	**5.0405E−3**	2.2470E−2†
MCMWBAP-1-w	3.7817E−3	3.7424E−3	**3.5642E−3**	3.8038E−3
MCMWBAP-2-w	**1.4407E−3**	1.6335E−3	6.1857E−3†	1.5133E−3
MCMWBAP-3-w	**1.4398E−3**	1.8671E−3†	1.5610E−2†	2.0619E−3†
MCMWBAP-4-w	2.9367E−3	**2.3618E−3**	2.2767E−2†	3.1127E−3
MCMWBAP-5-w	3.7147E−3	**3.2218E−3**	2.2241E−2†	4.2163E−3
MCMWBAP-6-w	**2.4408E−3**	4.5104E−3†	1.6365E−2†	9.6764E−3†
MCMWBAP-7-w	**1.5288E−3**	4.1766E−3	8.1211E−3†	2.9236E−2†
MCMWBAP-8-w	**2.3255E−3**	3.1052E−3	4.3917E−3	4.6742E−2†
Crime network	2.5517E−3	2.6828E−3	3.0229E−3†	**2.4679E−3**
MovieLens 100k network	**3.0504E−3**	1.2533E−2†	1.3694E−1†	1.8139E−2†
U test (S-D-I)	–	0-7-11	0-5-13	0-11-7

The effectiveness of the PPI strategy is confirmed in Tables 1 and 2, where PPI achieves rather competitive performance. We conducted additional experiments to confirm that it is vital to include PPI in the MOHEA. The results indicate that there is no significant difference between MOHEA and MOHEA$_{\mathrm{noPPI}}$ on small and medium instances, while MOHEA$_{\mathrm{noPPI}}$ deteriorated as the scale of the instances increased because PPI is limited in that it can produce a well-distributed PF in only one iteration. In addition, for some difficult instances, e.g., the MovieLens 100k network, PPI failed to get closer to the true PF, which can be remedied by the local search and evolutionary processes.

5.3 Case study

In this subsection, a real-world application is introduced and then solved using MOHEA, which benefits both the program broadcasting effect and energy consumption. As MSRBRAP described in Section 3,

Table 5 HV results of MOHEA and other algorithms on difficult broadcast instances

Instances	MOEAH	PPI	MOEAD-WS	MOEAD-TF	MOEAD-BI	NSGAII
T2000_P30	**6.9465E−1**	6.6134E−1	6.8810E−1	6.8210E−1	6.8185E−1	6.8395E−1
T2000_P40	**6.7100E−1**	6.3564E−1†	6.6200E−1	6.5873E−1	6.5766E−1	6.5599E−1
T2000_P50	**6.8853E−1**	6.4772E−1	6.7653E−1	6.7057E−1	6.6651E−1	6.6200E−1
T2000_P60	**6.3696E−1**	5.9988E−1	6.2141E−1	6.1996E−1	6.1398E−1	6.0951E−1
T2000_P70	**6.1607E−1**	5.7311E−1†	5.8781E−1†	5.8889E−1†	5.8442E−1†	5.7878E−1†
T2000_P87	**6.1214E−1**	5.6287E−1†	5.7062E−1†	5.7233E−1†	5.6100E−1†	5.5424E−1†
T3000_P30	**7.1950E−1**	6.9234E−1	7.1027E−1	7.0696E−1	7.0140E−1	7.0575E−1
T3000_P40	**6.3878E−1**	5.9945E−1	6.3428E−1	6.3014E−1	6.2634E−1	6.2237E−1
T3000_P50	**6.4817E−1**	6.0750E−1	6.3571E−1	6.3150E−1	6.2897E−1	6.2314E−1
T3000_P60	**6.1482E−1**	5.7903E−1†	5.9632E−1†	5.9431E−1†	5.8925E−1†	5.8276E−1†
T3000_P70	**6.0807E−1**	5.6973E−1†	5.8467E−1†	5.8030E−1†	5.7126E−1†	5.6293E−1†
T3000_P87	**6.0902E−1**	5.7107E−1†	5.6693E−1†	5.6417E−1†	5.5001E−1†	5.4596E−1†
T4000_P30	**7.0272E−1**	6.5833E−1	6.9786E−1	6.9268E−1	6.9242E−1	6.9522E−1
T4000_P40	**6.4123E−1**	6.0370E−1	6.3132E−1	6.3130E−1	6.2630E−1	6.2508E−1
T4000_P50	**6.1192E−1**	5.7128E−1†	6.0121E−1	5.9580E−1	5.9216E−1	5.8922E−1
T4000_P60	**6.2329E−1**	5.8859E−1†	6.0548E−1†	5.9852E−1†	5.9390E−1†	5.8722E−1†
T4000_P70	**6.2045E−1**	5.8403E−1	5.9186E−1	5.8338E−1	5.7941E−1	5.7055E−1
T4000_P87	**6.2650E−1**	5.9048E−1	5.7592E−1†	5.6910E−1†	5.5610E−1†	5.5522E−1†
T5000_P30	**6.5210E−1**	5.9466E−1	6.4894E−1	6.4437E−1	6.4225E−1	6.4149E−1
T5000_P40	**5.9858E−1**	5.5398E−1†	5.9453E−1	5.9162E−1	5.8494E−1†	5.8510E−1
T5000_P50	**6.0840E−1**	5.7364E−1†	5.9837E−1	5.9330E−1	5.8676E−1†	5.8043E−1†
T5000_P60	**6.4448E−1**	6.0967E−1	6.2678E−1	6.1354E−1	6.0477E−1	6.0106E−1
T5000_P70	**6.4265E−1**	6.0663E−1	6.1100E−1	5.9827E−1	5.8700E−1	5.8289E−1
T5000_P87	**6.1801E−1**	5.8275E−1†	5.6350E−1†	5.5318E−1†	5.4344E−1†	5.3332E−1†
T6000_P30	**6.7523E−1**	6.4250E−1	6.7345E−1	6.6430E−1	6.6181E−1	6.6381E−1
T6000_P40	**6.2274E−1**	5.8926E−1†	6.1717E−1	6.1214E−1	6.0487E−1†	6.0359E−1
T6000_P50	**6.2329E−1**	5.8492E−1	6.0703E−1	6.0243E−1	5.8880E−1	5.9080E−1
T6000_P60	**6.1200E−1**	5.7939E−1†	5.9264E−1†	5.7974E−1†	5.7036E−1†	5.6403E−1†
T6000_P70	**5.9798E−1**	5.6973E−1†	5.6357E−1†	5.5611E−1†	5.3918E−1†	5.3339E−1†
T6000_P87	**6.1254E−1**	5.8344E−1†	5.4716E−1†	5.3960E−1†	5.2132E−1†	5.1561E−1†
T7061_P30	**6.9799E−1**	6.7591E−1	6.9508E−1	6.8563E−1	6.8663E−1	6.8622E−1
T7061_P40	**6.2893E−1**	5.9586E−1	6.2193E−1	6.1472E−1	6.0727E−1	6.0574E−1
T7061_P50	**6.1305E−1**	5.7541E−1†	5.9840E−1†	5.9220E−1†	5.8187E−1†	5.7708E−1†
T7061_P60	**6.3370E−1**	5.9872E−1†	6.0717E−1†	5.9271E−1†	5.8148E−1†	5.7668E−1†
T7061_P70	**6.1806E−1**	5.8513E−1†	5.7870E−1†	5.6576E−1†	5.4823E−1†	5.4199E−1†
T7061_P87	**6.3911E−1**	6.0562E−1	5.6896E−1†	5.5310E−1†	5.3603E−1†	5.3304E−1†
U test (S-D-I)	–	0-18-18	0-21-15	0-21-15	0-18-18	0-20-16

here, we only report the optimized plans for decision makers. The HV and IGD results obtained with the MOHEA and other algorithms are shown in Tables 5 and 6, respectively.

Table 5 shows that MOHEA outperforms the other algorithms relative to the HV indicator, and this superiority is more obvious with the IGD indicator (Table 6). In addition, Figure 5 draws the nondominated solutions achieved by MOHEA and other algorithms for different scales of broadcast instances. According to the characteristic of broadcast instances, the top right corner of the PF is the most difficult region to explore. As shown in Figure 5, MOHEA achieves the maximum broadcasting quality and minimum energy consumption simultaneously, and generates a broader range of trade-off solutions than

Table 6 IGD results of MOHEA and other algorithms on difficult broadcast instances

Instances	MOHEA	PPI	MOEAD-WS	MOEAD-TF	MOEAD-BI	NSGAII
T2000_P30	3.8933E−3	2.6969E−2†	**3.6363E−3**	4.3635E−3	4.8536E−3	6.0918E−3†
T2000_P40	**3.4375E−3**	2.3875E−2†	4.2643E−3†	4.6203E−3†	5.5093E−3†	7.7326E−3†
T2000_P50	**3.1706E−3**	2.5865E−2†	4.9152E−3†	5.2124E−3†	6.5016E−3†	1.0305E−2†
T2000_P60	**2.8327E−3**	2.4972E−2†	8.3741E−3†	8.8772E−3†	1.3124E−2†	1.5616E−2†
T2000_P70	**1.1805E−3**	2.7648E−2†	1.9052E−2†	1.8585E−2†	2.3042E−2†	2.8684E−2†
T2000_P87	**2.4592E-05**	3.2536E−2†	2.9100E−2†	3.1556E−2†	4.2944E−2†	4.6986E−2†
T3000_P30	**3.4966E−3**	2.4864E−2†	3.7907E−3	4.1682E−3	4.8904E−3	6.4660E−3†
T3000_P40	**3.6697E−3**	2.9539E−2†	3.6874E−3	4.8510E−3†	6.1356E−3†	8.3321E−3†
T3000_P50	**3.3029E−3**	2.8569E−2†	6.3348E−3†	7.4607E−3†	9.4433E−3†	1.3085E−2†
T3000_P60	**1.6624E−3**	2.4458E−2†	1.2024E−2†	1.4264E−2†	1.7898E−2†	2.2099E−2†
T3000_P70	**6.9994E−4**	2.5704E−2†	1.6704E−2†	2.0710E−2†	2.9304E−2†	3.4639E−2†
T3000_P87	**1.0751E−4**	2.5895E−2†	3.0239E−2†	3.6318E−2†	4.9191E−2†	5.0847E−2†
T4000_P30	3.7073E−3	4.1887E−2†	**3.5460E−3**	4.1280E−3	4.6498E−3	5.4264E−3†
T4000_P40	**3.4583E−3**	2.9150E−2†	4.9751E−3†	5.3025E−3†	6.1704E−3†	8.0481E−3†
T4000_P50	**2.8917E−3**	2.7976E−2†	6.3664E−3†	8.7233E−3†	1.1635E−2†	1.3634E−2†
T4000_P60	**1.9014E−3**	2.3777E−2†	1.1349E−2†	1.6466E−2†	1.9764E−2†	2.4326E−2†
T4000_P70	**5.5149E−4**	2.4487E−2†	2.0051E−2†	2.6785E−2†	3.0901E−2†	3.5950E−2†
T4000_P87	**3.0481E−4**	2.5045E−2†	3.5496E−2†	4.2988E−2†	5.3984E−2†	5.6450E−2†
T5000_P30	3.8314E−3	4.8537E−2†	**3.5636E−3**	4.2043E−3	5.3800E−3†	6.1349E−3†
T5000_P40	**3.6955E−3**	3.5219E−2†	3.9201E−3	5.2047E−3†	8.2166E−3†	8.9155E−3†
T5000_P50	**2.7117E−3**	2.5847E−2†	6.5339E−3†	9.3923E−3†	1.4187E−2†	1.7319E−2†
T5000_P60	**1.0632E−3**	2.3218E−2†	1.1117E−2†	1.7650E−2†	2.4752E−2†	2.7229E−2†
T5000_P70	**5.3344E−4**	2.3299E−2†	1.9522E−2†	2.7994E−2†	3.6834E−2†	4.0239E−2†
T5000_P87	**4.1033E−4**	2.3730E−2†	3.7892E−2†	4.9351E−2†	6.0214E−2†	6.8124E−2†
T6000_P30	3.4536E−3	2.8948E−2†	**3.1238E−3**	4.7136E−3†	6.2024E−3†	6.7241E−3†
T6000_P40	**3.2471E−3**	2.7272E−2†	4.3163E−3†	6.5292E−3†	1.0244E−2†	1.1072E−2†
T6000_P50	**2.1985E−3**	2.7527E−2†	9.8373E−3†	1.2658E−2†	2.2732E−2†	2.0915E−2†
T6000_P60	**8.4822E−4**	2.2834E−2†	1.3733E−2†	2.4859E−2†	3.3657E−2†	3.6888E−2†
T6000_P70	**7.1308E−4**	2.0082E−2†	2.4838E−2†	3.5005E−2†	5.0712E−2†	5.5626E−2†
T6000_P87	**5.2201E−4**	1.9920E−2†	4.6079E−2†	5.9400E−2†	7.5755E−2†	8.0096E−2†
T7061_P30	3.4358E−3	2.0568E−2†	**3.3785E−3**	5.2710E−3†	6.0392E−3†	6.7924E−3†
T7061_P40	**3.2619E−3**	2.3026E−2†	4.7109E−3†	8.4847E−3†	1.2834E−2†	1.4256E−2†
T7061_P50	**1.7365E−3**	2.5713E−2†	9.7501E−3†	1.5989E−2†	2.3651E−2†	2.6697E−2†
T7061_P60	**6.7199E−4**	2.3410E−2†	1.8362E−2†	3.0329E−2†	4.0940E−2†	4.2667E−2†
T7061_P70	**6.3876E−4**	2.2052E−2†	2.7808E−2†	4.3849E−2†	5.8384E−2†	6.2802E−2†
T7061_P87	**2.2327E−4**	2.3333E−2†	4.9467E−2†	6.7619E−2†	8.4872E−2†	8.2721E−2†
U test (S-D-I)	–	0-0-36	0-8-28	0-4-32	0-3-33	0-0-36

others.

6 Conclusion

We conducted a study to model and solve the MCMWBAP. To handle such a constrained optimization problem, three strategies were integrated into an EA framework with problem information. The proposed MOHEA found the current best solutions for various instances and demonstrated superior performance

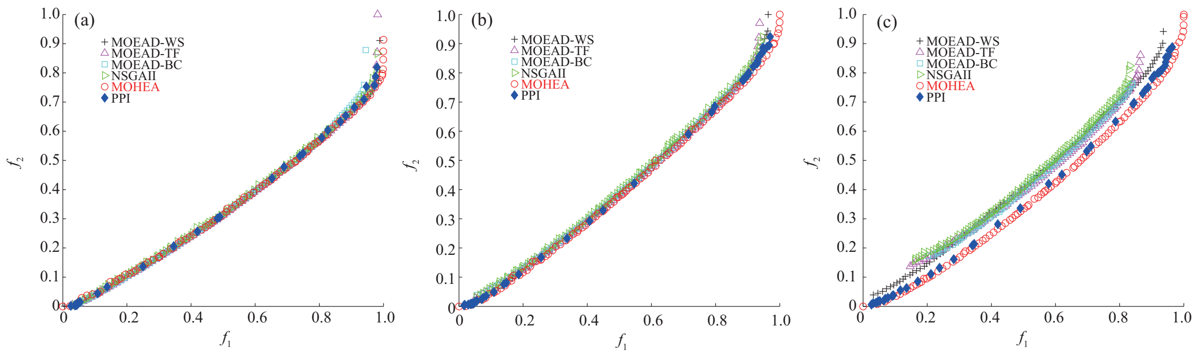

Figure 5 (Color online) Nondominated solutions obtained by MOHEA and other algorithm on some broadcast instances. (a) T2000_P30; (b) T5000_P50; (c) T7061_P87.

compared to conventional MOEAs. In addition, a real-world application of MCMWBAP was investigated to provide appropriate assignments for radio programs in consideration of the broadcasting effect and energy consumption.

References

1 Ma F, Gao X, Yin M, et al. Optimizing shortwave radio broadcast resource allocation via pseudo-boolean constraint solving and local search. In: Proceedings of International Conference on Principles and Practice of Constraint Programming, 2016. 650–665

2 Kuhn H W. The Hungarian method for the assignment problem. Naval Res Logistics, 1955, 2: 83–97

3 Munkres J. Algorithms for the assignment and transportation problems. J Soc Industrial Appl Math, 1957, 5: 32–38

4 Belongie S, Malik J, Puzicha J. Shape matching and object recognition using shape contexts. IEEE Trans Pattern Anal Machine Intell, 2002, 24: 509–522

5 Huang C Y, Chen Y S, Lin Y L, et al. Data path allocation based on bipartite weighted matching. In: Proceedings of the 27th ACM/IEEE Design Automation Conference, 1991. 499–504

6 Reiffenhäuser R. An optimal truthful mechanism for the online weighted bipartite matching problem. In: Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms, 2019. 1982–1993

7 Chen C, Chester S, Srinivasan V, et al. Group-aware weighted bipartite b-matching. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, 2016. 459–468

8 McCrae J, Arcan M, Ahmadi S. Lexical sense alignment using weighted bipartite b-matching. In: Proceedings of the 2nd Conference on Language, Data and Knowledge (LDK 2019), 2019

9 Pan L, Jin J, Gao X, et al. Integrating ILP and SMT for shortwave radio broadcast resource allocation and frequency assignment. In: Proceedings of International Conference on Principles and Practice of Constraint Programming, 2017. 405–413

10 Wang S J, Wu T Y, Yao Y, et al. Constrained maximum weighted bipartite matching: a novel approach to radio broadcast scheduling. Sci China Inf Sci, 2019, 62: 072102

11 Long J, Sun Z, Pardalos P M, et al. A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem. Inf Sci, 2019, 478: 40–61

12 Salcedo-Sanz S, Bousono-Calzon C, Figueiras-Vidal A R. A mixed neural-genetic algorithm for the broadcast scheduling problem. IEEE Trans Wirel Commun, 2003, 2: 277–283

13 Arivudainambi D, Rekha D. An evolutionary algorithm for broadcast scheduling in wireless multihop networks. Wireless Netw, 2012, 18: 787–798

14 Neri F, Cotta C. Memetic algorithms and memetic computing optimization: a literature review. Swarm Evolary Comput, 2012, 2: 1–14

15 Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Computat, 2002, 6: 182–197

16 Zhang Q F, Li H. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Computat, 2007, 11: 712–731

17 Cai X Y, Li Y X, Fan Z, et al. An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization. IEEE Trans Evol Computat, 2015, 19: 508–523

18 Fan Z, Li W, Cai X, et al. Push and pull search for solving constrained multi-objective optimization problems. Swarm Evolary Comput, 2019, 44: 665–679

19 Ishibuchi H, Murata T. A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Trans Syst Man Cybern C, 1998, 28: 392–403