SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

July 2019, Vol. 62 072102:1–072102:14
https://doi.org/10.1007/s11432-017-9324-0

Constrained maximum weighted bipartite matching: a novel approach to radio broadcast scheduling

Shaojiang WANG[1,2], Tianyong WU[1,2], Yuan YAO[2,3], Dongbo BU[4] & Shaowei CAI[1,2*]

[1] *State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China;*
[2] *School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing* 100190*, China;*
[3] *Beijing Key Lab of Human-Computer Interaction, Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China;*
[4] *Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190*, China*

Received 28 July 2017/Revised 25 September 2017/Accepted 29 November 2017/Published online 7 May 2019

Abstract Given a set of radio broadcast programs, the radio broadcast scheduling problem is to allocate a set of devices to transmit the programs to achieve the optimal sound quality. In this article, we propose a complete algorithm to solve the problem, which is based on a branch-and-bound (BnB) algorithm. We formulate the problem with a new model, called constrained maximum weighted bipartite matching (CMBM), i.e., the maximum matching problem on a weighted bipartite graph with constraints. For the reduced matching problem, we propose a novel BnB algorithm by introducing three new strategies, including the highest quality first, the least conflict first and the more edge first. We also establish an upper bound estimating function for pruning the search space of the algorithm. The experimental results show that our new algorithm can quickly find the optimal solution for the radio broadcast scheduling problem at small scales, and has higher scalability for the problems at large scales than the existing complete algorithm.

Keywords radio broadcast scheduling, branch-and-bound algorithm, constrained maximum weighted bipartite matching, Kuhn-Munkres algorithm, strategy combinations

Citation 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(7): 072102, https://doi.org/10.1007/s11432-017-9324-0

1 Introduction

According to the statistics of ITU[1)], near-Earth communication accounts for 99% of human communication, in which wireless communication is one of the main means. In wireless communication, short-wave (the frequency ranges from 2 to 30 MHz) can effectively support long distance communication so that it has been applied in a large number of areas, including radio broadcast of voice.

A central problem of the radio broadcast is how to schedule the programs and available resources (transmitters) to achieve high sound quality. There are great differences in the quality of the programs transmitted by different transmitters to different areas. The users can effectively receive the broadcast programs only if the sound quality reaches some level. In general, this scheduling problem can be naturally translated into a maximum weighted bipartite matching problem, for which a number of algorithms (e.g., [1,2]) have been proposed. However, in the practical scenario, there are several extra constraints on the programs and transmitters, for instance, the same transmitter with different antennas cannot work at the same time (see Subsection 2.1).

* Corresponding author (email: caisw@ios.ac.cn)

1) About ITU. 2016. http://www.itu.int/.

In this paper, we investigate the problem of constrained radio broadcast scheduling (CRBS). At first, we introduce a new optimization problem called constrained maximum weighted bipartite matching (CMBM), and reduce the CRBS problem to the CMBM problem. Then, we show that the deterministic version of CMBM problem is NP-hard by reducing the 3SAT problem to it. We propose an algorithm to solve the CMBM problem based on the branch-and-bound method (BnB), one of the foundational method for optimization problems [3–5]. To make the BnB algorithm more efficient, we introduce three new strategies, namely, the highest quality first, the least conflict first, and the more edges first. Our experimental results show that the proposed algorithm is efficient in real-world applications.

In summary, our contributions in this paper are as follows:

- We propose the new problem of constrained maximum weighted bipartite matching, which well characterizes CRBS problem.
- We formulate the radio scheduling problem to the constrained maximum weighted bipartite matching problem.
- We propose a novel algorithm for the constrained maximum weighted bipartite matching.
- Experiments show that our algorithm is efficient and practical in solving real-world radio broadcast scheduling problem.

The rest of this paper is organized as follows. Section 2 describes some backgrounds and related work of the radio broadcast scheduling problem, as well as the maximum weighted matching on bipartite graphs and pseudo-Boolean optimization. Section 3 introduces how we model the broadcast scheduling problem by CMBM. Section 4 gives the proof that this problem is NP-hard. Sections 5 and 6 show the details of the proposed approaches based on the PBO algorithm and branch-and-bound algorithm, respectively. Our experimental results are shown in Section 7. Finally, we conclude this work and introduce some future directions.

2 Backgrounds

In this section, we introduce the necessary backgrounds about the radio broadcast scheduling problem, maximum weighted matching on bipartite graphs and pseudo-Boolean optimization.

2.1 Radio broadcast scheduling problem

The radio managers have a set of programs and a set of devices (transmitters and antennas) to broadcast the programs. Each program is required to be transmitted to a specific area in a certain period of time. A transmitter can work with several antennas to launch the programs and will have different transmitted effects (sound quality) with different antennas. Note that the devices with the same transmitter (different antennas) cannot work simultaneously, and we call such devices conflicted devices. The quality (like the field strength) of the programs in the associated area must exceed a threshold (e.g., 55 decibels). Since it is infeasible to probe the broadcast quality in all corners of the area, only a few stations are established in each area for monitoring the quality of the program transmitted to the area. The numbers of the stations in different areas are different.

Figure 1 shows a diagram to explain the inputs of the broadcast scheduling problem, where p_i indicates a program; a_i is an area; s_{ij} denotes the jth station in a_i; and d_i represents a device. The interval $[t_{i1}, t_{i2}]$ on the program p_i indicates that the program is broadcasted from time t_{i1} to t_{i2}. The edge from p_i to a_j means the program p_i is required to be transmitted to the area a_j. The value of the edge from d_k to s_{ij} represents the quality of the signal transmitted by d_k to the station s_{ij}. The rectangle surrounding a pair of devices means that the devices in it share the same transmitter and cannot work simultaneously.

Radio broadcast scheduling problem is to allocate a device for each program and maximize the total quality of the whole set of programs. The device allocation solution must satisfy the following three requirements:

- **Unique allocation.** Each device can be allocated to transmit at most one program at each moment, while each program must be transmitted by a unique device.

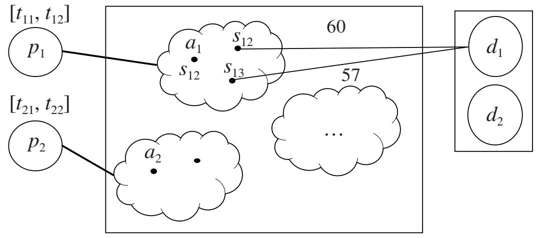

Figure 1 An example of radio broadcast scheduling problem.

- **Area cover.** If the device d_i is allocated to transmit the program p_j that is required to be presented in area a_k, then d_i must cover the area a_k.
- **Device conflict.** If the device d_i is allocated to transmit some program p_j, then the other devices that share the same transmitter of d_i cannot be allocated to transmit the program whose interval of broadcast time overlaps with that of program p_j.

2.2 Maximum weighted bipartite matching problem

A bipartite graph $G = \langle V, E\rangle$ is a graph whose vertices can be categorized into two disjoint sets such that every edge crosses over from one to the other. A weighted bipartite graph is extended by attaching a weight to each edge.

A matching of a graph is a set of edges that no two edges share a common end. In addition, a vertex is matched (or saturated) if it is an endpoint of an edge in the matching. The maximum weighted bipartite matching problem is to find in a given weighted bipartite graph a matching whose total weight is maximized. There are several algorithms [2, 6, 7] to solve this problem in polynomial time. In this paper, we use the Kuhn-Munkres algorithm [6, 7].

2.3 Pseudo-Boolean optimization

The pseudo-Boolean optimization (PBO) problem [8] consists of a pseudo-Boolean formula and an objective function. The pseudo-Boolean formula is a conjunction of pseudo-Boolean constraints, each of which is an inequality with Boolean variables. The objective function is a linear arithmetic expression over the Boolean variables in the pseudo-Boolean formula. A PBO problem can be expressed in the following form, where x_i is a Boolean variable and w_i, c_{ij} and b_i are positive integers.

$$\text{Maximize} : \sum_{i=1}^{n} w_i x_i,$$

such that

$$\sum_{i=1}^{n} c_{ij} x_i \leqslant b_j, \quad \text{for each } 1 \leqslant j \leqslant m.$$

2.4 Related work

Ma et al. [9] proposed a local search algorithm, by using swap and substitute two operators. They calculated the approximate solution in less time, but local search would be impossible to guarantee optimal solutions. Pan et al. [10] developed, with additional consideration of radio frequency, this algorithm by integrating ILP and SMT solving into the local search approach. However, both these algorithms [9, 10] are based on local search, and thus no optimal solution can be guaranteed.

Zhan et al. [5] proposed a wireless broadcast real-time scheduling model, focusing on real-time scheduling and coding processing. Beale [3] proposed a branch-and-bound algorithm on constraint satisfaction optimization (CSP) problem, and did not deal with wireless broadcast scheduling. Li et al. [11] proposed a branch-and-bound algorithm on max-sat for the maximum clique problem, without involving wireless broadcast scheduling. In this paper, we have reduced the wireless broadcast scheduling problem to a constrained maximum weighted bipartite matching problem and then proposed a new BnB algorithm. The experimental results confirm the advantages of the new approach.

3 Constrained maximum weighted bipartite matching

In this section, we formally describe the radio broadcast scheduling problem as our proposed model CMBM. Before giving the detailed formal descriptions, we introduce several notations.

Let $P = \{p_1, p_2, \ldots, p_m\}$ denote the set of programs, $D = \{d_1, d_2, \ldots, d_n\}$ denote the set of devices, $T = \{t_1, t_2, \ldots, t_k\}$ denote the set of time intervals of the programs, $A = \{a_1, a_2, \ldots, a_l\}$ denote the set of areas, and $D_c \subset D \times D$ denote all the pairs of conflicted devices that cannot work at the same time. The item $\mathrm{time}(p_i) \in T$ denotes the broadcast time interval of program p_i. Let $\mathrm{area}(p_i) \in A$ be the area that program p_i is required to be transmitted to, $S_i = \mathrm{stat}(a_i)$ be the station set of area a_i and $S = \bigcup_{i=1}^{l} S_i$ be the set of all stations. The function $\mathrm{qual} : D \times S \to \mathbb{N}^+$ gives the quality of the signal transmitted by the devices to the stations. L denotes the necessary transmitted quality of a program for users.

With the above notations, we define the function $\mathrm{sc} : D \times S \to \{0, 1\}$ and $\mathrm{ac} : D \times A \to \{0, 1\}$ to represent whether a device cover a station or an area as follows:

$$\mathrm{sc}(d, s) = \begin{cases} 1, & \mathrm{qual}(d, s) \geqslant L, \\ 0, & \text{otherwise}, \end{cases}$$

$$\mathrm{ac}(d, a) = \begin{cases} 1, & \frac{|CS|}{|\mathrm{stat}(a)|} \geqslant 0.8, \\ 0, & \text{otherwise}, \end{cases}$$

where $d \in D$, $s \in S$, $a \in A$, and $\mathrm{CS} = \{s\ |s \in \mathrm{stat}(a) \wedge \mathrm{sc}(d, s) = 1\}$. Finally, we define the function $f : P \times D \to \mathbb{N}$ to denote the quality of a program transmitted by a device as follows:

$$f(p, d) = \begin{cases} 0, & \mathrm{ac}(d, \mathrm{area}(p)) = 0, \\ \mathrm{avg}(p, d), & \text{otherwise}, \end{cases}$$

where p denotes a program, d indicates a device, and $\mathrm{avg}(p, d) = \mathrm{Avg}\{\mathrm{qual}(d, s)|s \in \mathrm{area}(p) \wedge \mathrm{sc}(d, s) = 1\}$ calculates the average value of the quality of program p transmitted by device d.

Now we model the input of the broadcast scheduling problem with a constrained weighted bipartite graph $G = \langle V_1, V_2, E, C\rangle$, which is constructed by the following steps. Each element $v_1 \in V_1$ corresponds an element $p \in P$ as defined above (denoted by $g(v_1) = p$). Each element $v_2 \in V_2$ corresponds to a tuple $\langle d, t\rangle \in D \times T$ (denoted by $h(v_2) = \langle d, t\rangle$), which means using the device d to broadcast the program in the time interval t. The edge set E consists of pairs of the elements in V_1 and V_2, i.e., $E = \{\langle v_1, v_2\rangle | v_1 \in V_1 \wedge v_2 \in V_2 \wedge g(v_1) = p \wedge h(v_2) = \langle d, t\rangle \wedge \mathrm{time}(p) = t \wedge f(p, d) > 0\}$. The weight of each edge $e = \langle v_1, v_2\rangle \in E$ is denoted by $w(e) = f(p, d)$. The set $C = C_1 \cup C_2$ denotes the conflicted device pairs, where $C_1 = \{\langle v, v'\rangle | v, v' \in V_2 \wedge \exists d, t, t'(h(v) = \langle d, t\rangle \wedge h(v') = \langle d, t'\rangle \wedge t \cap t' \neq \emptyset)\}$ indicates the conflictions of the same device with overlapped time intervals and $C_2 = \{\langle v, v'\rangle | v, v' \in V_2 \wedge \exists d, d', t, t'(h(v) = \langle d, t\rangle \wedge h(v') = \langle d', t'\rangle \wedge \langle d, d'\rangle \in D_c \wedge t \cap t' \neq \emptyset)\}$ represents the conflicted devices with overlapped time intervals. The scale of the graph G is shown as follows:

- $|V_1| = |P|$;
- $|V_2| = |D| \cdot |T| \leqslant |D| \cdot |P|$;
- $|E| \leqslant |D| \cdot |P|$;
- $|C| \leqslant (|D_c| + |D|) \cdot |T|^2 \leqslant (|D_c| + |D|) \cdot |P|^2$.

Then the broadcast scheduling problem can be translated to a CMBM problem, i.e., to find a subset $E' \subset E$ in the graph G that satisfies the following properties.

- P1: E' is a matching.
- P2: $|E'| = |V_1|$.
- P3: For any two edges $e_1 = \langle v_1, v_2\rangle$, $e_2 = \langle v_1', v_2'\rangle \in E'$, the condition $\langle v_2, v_2'\rangle \notin C$ always holds.

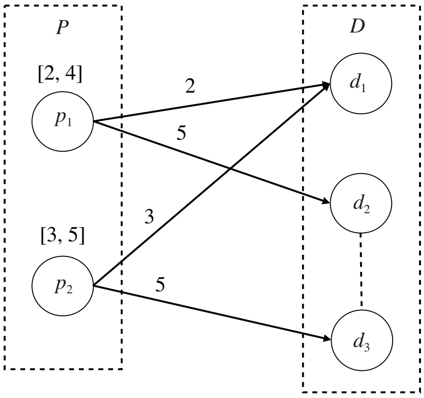

Figure 2 Example CBRS problem.

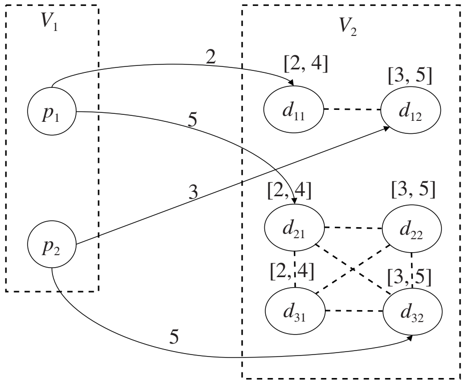

Figure 3 CMBM model for CBRS problem.

- P4: The weight sum of E' is maximized.

Example 1. An instance of radio broadcast scheduling problem is shown in Figure 2, where a solid line with a weight represents an edge between a program and a device, while a dashed line describes two conflicted devices, and the intervals above the P nodes denote the broadcasting time.

Figure 3 demonstrates a CMBM model of the above radio broadcast scheduling problem. We translate the programs clashing in the broadcasting time to the confliction of devices. Specifically, we split each device into two new devices with different time intervals of the programs, and add the corresponding edges.

4 Complexity analysis

We prove that the CMBM problem is NP-hard by reducing from 3SAT problem. The 3SAT problem is described as follows: for a given collection of clauses $c_1, c_2, \ldots, c_m$, each of which is a disjunction of exactly three literals, decides whether there is an assignment that can evaluate all the clauses true.

Theorem 1. The CMBM problem is NP-hard.

Proof. For a 3SAT instance, for example, the variables are $x_1, x_2, \ldots, x_n$ and the clauses are $c_1, c_2, \ldots,$ c_m, we construct a constrained weighted bipartite graph $G = \langle V_1, V_2, E, C\rangle$ by the following steps. First, we construct m nodes in V_1 to represent m clauses. Then we construct a node in V_2 for each literal in a clause, that is, we totally construct $3m$ nodes in V_2. Note that the different nodes in V_2 may correspond to the same literal, if the literal occurs multiple times in the clauses. And we construct $3m$ edges between the nodes in V_1 and V_2, where each node v in V_1 has three edges to three nodes in V_2 that represent the literals of the corresponding clause of v. All the weights of edges are set as 1. The conflicted pairs of nodes in C are set as the nodes in V_2 that corresponds to the complementary variables (e.g., x_1 and $\bar{x_1}$). Figure 4 shows the constructed constrained weighted bipartite graph for the 3SAT instance $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x_1} \vee \bar{x_2} \vee x_3)$.

On the one hand, if the 3SAT instance has a solution (denoted by A), then there is at least one literal is evaluated true under the assignment A in each clause. We can construct an edge set E' by the following method. For each node $v_1 \in V_1$, we can always find an edge $\langle v_1, v_2\rangle \in E$ such that the corresponding literal of v_2 is evaluated true under A, and add the edge to E'. Then we need to prove E' satisfies the four properties in the previous section. We can easily observe E' satisfies the properties P1 and P2. Assume that E' does not satisfy P3, that is, there exists $v_1, v_1' \in V_1, \langle v_2, v_2'\rangle \in C$ such that $\langle v_1, v_2\rangle, \langle v_1', v_2'\rangle \in E'$. According to the construction of C, we can get that the corresponding literals of v_2 and v_2' are complementary literals. According to the construction of E', we can get that the corresponding literals of v_2 and v_2' are both evaluated true under A, that is contradiction. It is obvious that the number of edges in a match of G is no more than m, and the weight sum of a match is no more than m. Since the weight sum of E' is m, E' satisfies the property P4.

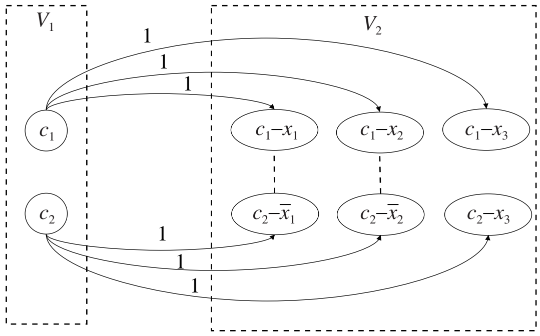

Figure 4 CMBM model for 3SAT problem.

On the other hand, assume the CMBM instance has a solution (denoted by E'). Since E' satisfies the properties P1 and P2, we can get that for each node in P, there exists a unique edge in E' that connects the node. We can construct an assignment A by the following steps. For each edge $\langle v_1, v_2\rangle \in E'$, if the corresponding literal of v_2 is the negation of the variable, we assign this variable false, otherwise, we assign this variable true. We need to prove that each variable is not assigned true and false simultaneously for two different edges $\langle v_1, v_2\rangle, \langle v_1', v_2'\rangle \in E'$. If the above situation occurs, we can get that the corresponding literals of v_2 and v_2' are complementary literals, and $\langle v_2, v_2'\rangle \in C$. So E' does not satisfies the property P3, that is contradiction. If some variables are not assigned in the above process, we can assign them either true or false. Assignment A make all the clause true, since for each clause there exists at least one literal that is true.

This shows that the CMBM problem is NP-hard.

5 A method based on PBO

In this section, we introduce a baseline algorithm to solve the broadcast scheduling problem, which is designed based on PBO technique.

As mentioned in Subsection 2.3, a PBO instance consists of an optimization function and a pseudo-Boolean formula over a number of Boolean variables. In the weighted bipartite graph (Section 3), for each edge $\langle p_i, d_j\rangle \in E$, we define a Boolean variable x_{ij}. If x_{ij} equals 1, it indicates the edge is selected in the targeted matching, otherwise, it is not selected. Then we can easily get the optimization function as follows:

$$\text{Maximize}: \sum_{\langle p_i,d_j\rangle\in E} w_{ij}x_{ij},$$

where w_{ij} is the weight of the edge $\langle p_i, d_j\rangle$.

Next, let us consider the pseudo-Boolean formula that includes a series of constraints. For the property P1, according to the definition of matching, we have two constraints shown as follows:

$$\sum_{\langle p_i,d_j\rangle\in E} x_{ij} \leqslant 1, \quad \text{for each } p_i \in P,$$

$$\sum_{\langle p_i,d_j\rangle\in E} x_{ij} \leqslant 1, \quad \text{for each } d_j \in D,$$

where the 1st constraint indicates that there is at most one edge in the selected edge set that contains p_i, and the 2nd one indicates that there is at most one edge in the selected edge set that contains d_j.

The property P2 can be represented by the following formula:

$$\sum_{\langle p_i,d_j\rangle\in E} x_{ij} = m.$$

Finally let us consider the property P3. Assume $\langle d_u, d_v\rangle \in C$, then this property can be expressed by the following formula:

$$\sum_{\langle p_i,d_u\rangle\in E} x_{iu} + \sum_{\langle p_i,d_v\rangle\in E} x_{iv} \leqslant 1.$$

To sum up, the broadcast scheduling problem can be transformed to a PBO instance shown as follows:

$$\begin{aligned}
\text{Maximize}: \quad & \sum_{\langle p_i,d_j\rangle\in E} w_{ij}x_{ij} \\
\text{s.t.} \quad & \bigwedge_{i=1}^{m}\left(\sum_{\langle p_i,d_j\rangle\in E} x_{ij} \leqslant 1\right), \\
& \bigwedge_{j=1}^{n}\left(\sum_{\langle p_i,d_j\rangle\in E} x_{ij} \leqslant 1\right), \\
& \sum_{\langle p_i,d_j\rangle\in E} x_{ij} = m, \\
& \bigwedge_{\langle d_u,d_v\rangle\in C}\left(\sum_{\langle p_i,d_u\rangle\in E} x_{iu} + \sum_{\langle p_i,d_v\rangle\in E} x_{iv} \leqslant 1\right).
\end{aligned}$$

6 An algorithm for constrained maximum weighted bipartite matching

In this section, we propose a new algorithm for the problem of constrained maximum weighted bipartite matching. Our algorithm is based on the branch-and-bound method, which exploits a number of observations the structure of the solution space of the constrained maximum weighted bipartite matchings. Our overall idea is to divide the problem into a number of maximum weighted matching problems that can be solved by the Kuhn-Munkres algorithm [6, 7]. Then we leverage the branch-and-bound technique to decide the exploration order of these sub-problems and prune some of the unnecessary sub-problems. The following two subsections illustrate these two steps in detail.

6.1 Formulation

We first define several notations. Given a constrained weighted bipartite graph $G = \langle V_1, V_2, E, C\rangle$, let M denote the maximum weight of the edges in G, i.e., $M = \max\{w(e)|e \in E\}$. We introduce a notation, called maximal non-conflict device set, to represent the subset $V' \subset V_2$ that satisfies the properties: (1) for two different nodes $v, v' \in V'$, $\langle v, v'\rangle \notin C$; (2) for each $v \in V_2 \setminus V'$, there always exists $v' \in V'$ such that $\langle v, v'\rangle \in C$. The set $\mathcal{V} = \{V^1, V^2, \ldots, V^r\}$ is composed of all the maximal non-conflict device set. Then we translate the graph G to a series of weighted bipartite graphs $\mathcal{G} = \{G^1, G^2, \ldots, G^r\}$, where $G^i = \langle V_1, V^i, E^i\rangle$. The edge set $E^i = \{\langle v_1, v_2\rangle | v_1 \in V_1 \wedge v_2 \in V^i\}$, and for each edge $e \in E^i$, the weight $w^i(e)$ is defined with the following formula:

$$w^i(e) = \begin{cases} w(e), & \text{if} \quad e \in E, \\ (-M)\cdot |V_1|, & \text{otherwise.} \end{cases}$$

We can perform the Kuhn-Munkres algorithm on the graph G^i to get the maximum weight sum of the matching that is denoted by $\mathrm{KM}(G^i) = (\mathrm{opt}^i, \mathrm{match}^i)$, where opt^i denotes the maximum weight sum and match^i denotes the corresponding matching. Let $G^* \in \mathcal{G}$ denote a graph, and $\mathrm{KM}(G^*) = (\mathrm{opt}^*, \mathrm{match}^*)$. Assume that $\mathrm{opt}^* = \max\{\mathrm{opt}^i | 1 \leqslant i \leqslant |\mathcal{G}| \wedge |\mathrm{match}^i| = |V_1|\}$, and the solution of CMBM on G is denoted by $(\mathrm{opt}, \mathrm{match})$. Then we prove match^* satisfies the four properties. It is obvious that match^* satisfies the properties P1 and P2. Through the construction of G^*, we can also see that match^* satisfies the property P3. So we only need to prove that match^* satisfies the last property, that is, $\mathrm{opt}^* = \mathrm{opt}$. To prove that, we first introduce and prove several propositions as follows.

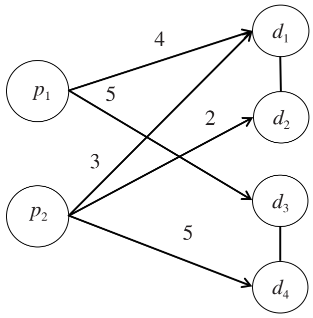

Figure 5 Example to illustrate the translating process.

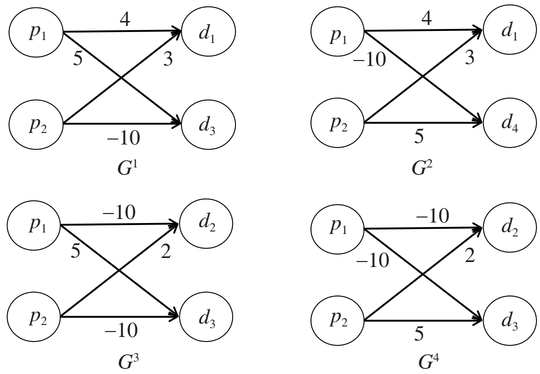

Figure 6 Translated graphs without constraints.

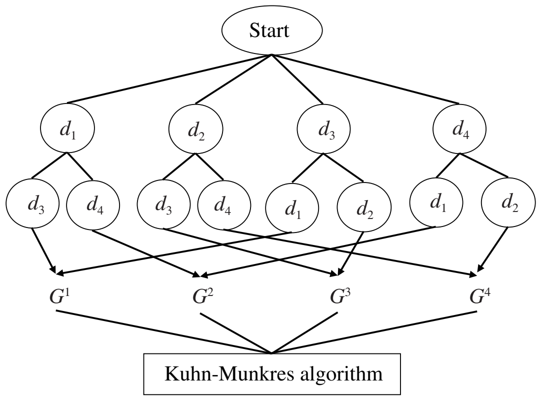

Figure 7 Tree structure of translated graphs.

Proposition 1. $\text{opt}^* \geqslant \text{opt}$.

Proof. Let $V^\dagger = \{v_2 | \exists v_1 \in V_1(\langle v_1, v_2\rangle \in \text{match})\}$ denote the vertexes in V_2 that are covered by match. Sine match satisfies the property P3, for any $v, v' \in D^\dagger$, $\langle v, v'\rangle \notin C$ holds. So there must exists $V^k \in \mathcal{V}$ such that $V^\dagger \subset V^k$ holds. Then match must be a matching in the graph G^k, and so $\text{opt}^* \geqslant \text{opt}^k \geqslant \text{opt}$ holds.

Proposition 2. For any $G^i \in \mathcal{G}$, if $\text{opt}^i > 0$, then match^i is a matching in the graph G.

Proof. To prove this proposition, we only need to prove that all edges in match^i belong to E. Assume that there exists an edge in match^i that does not belong to E. Then we can get that $\text{opt}^i \leqslant (m-1) \cdot M - M \cdot m = -M < 0$. This comes to a contradiction.

With Proposition 2, we have $\text{opt}^* \leqslant \text{opt}$. Then with Proposition 1, we further infer that $\text{opt}^* = \text{opt}$, that is, match^* satisfies the property P4.

Example 2. Figure 5 shows an example of a constrained weighted bipartite graph $G = \langle V_1, V_2, E, C\rangle$, which contains two vertices in V_1 and four vertices in V_2. The value in each edge indicates the weight of the edge. The conflict device set C has four elements, $\langle d_1, d_2\rangle$, $\langle d_2, d_1\rangle$, $\langle d_3, d_4\rangle$, $\langle d_4, d_3\rangle$. The solution of this instance is $(\langle p_1, d_1\rangle, \langle p_2, d_4\rangle)$, and the maximum weight sum is 9.

Figure 6 shows the translated graph G^1, G^2, G^3 and G^4 of the graph G in Figure 5. We can perform the Kuhn-Munkres algorithm [6, 7] to calculate and compare the maximum weight sum for these graph, and find the optimal solution in G^2 that is also the solution on the graph G.

We can organize the search space of device combination as a tree, where each leaf of the tree indicates a maximal non-conflict device set. Figure 7 shows the tree structure of the graph in Figure 5.

6.2 Branch-and-bound algorithm

Before describing our algorithm, we introduce three new strategies for determining which device is explored in the searching process.

- **Highest quality first.** We prefer to choose the device that has higher quality to transmit some program.
- **More edge first.** We prefer to choose the device that is able to transmit more programs with acceptable quality.
- **Least conflict first.** We prefer to choose the device that is conflicted with the least number of devices. Extremely, if a device has no conflict with other devices, it is obvious that this device must be in any maximal non-conflict device set, and we should select it preferentially.

We also estimate the upper bound of a given node in the search tree for pruning. The bound is estimated by summing up the maximum weights of selected devices and the devices that are not conflicted with the selected devices.

Algorithm 1 gives the details of our approach. It accepts the constrained weighted bipartite graph $G = \langle V_1, V_2, E, C\rangle$ and outputs the maximum weighted bipartite matching M and the optimal value opt. We make use of a priority queue to perform the process. In the algorithm, each searching state contains two attributes, including sv and pri. The set sv stores the current selected vertices in V_2, while the pri indicates the search priority of the current state. If the current state is not a leaf node in the search tree (not maximal non-conflict device set), then we estimate the upper bound in line 7. If the upper bound ub is less than the current value of opt, we discard this state, otherwise, we extend this state and calculate the priority of the new state with the above strategies. If the current state is a leaf node, we construct the translated graph with the set sv of the current state, and perform Kuhn-Munkres algorithm [6, 7] to solve the maximum weighted matching.

Algorithm 1 BnB algorithm for the constrained maximum weighted graph matching problem (CMGM)

Input: Constrained weighted bipartite graph G;

Output: A matching M and its weight sum opt such that M satisfies the constraints and opt is maximum.

1: Priority queue $q = \emptyset$, opt $= 0$;
2: Create a new state n such that $n.sv = \emptyset$, n.pri $= 0$;
3: q.push(n);
4: **while** q is not empty **do**
5:   $n = q$.front(), q.pop();
6:   **if** n.sv in not a maximal non-conflict device set **then**
7:     ub = estimate_upper_bound (n, G);
8:     **if** ub > opt **then**
9:       **for** each unvisited and non-conflicted vertex $v \in V_2$ **do**
10:         Create a new state n' such that n'.sv $= n$.sv $\cup \{v\}$, n'.pri = cal_priority(v);
11:         q.push(n);
12:       **end for**
13:     **end if**
14:   **else**
15:     $G^* =$ translate_graph(G, n);
16:     (m, val) = KM(G^*);
17:     **if** val > opt **then**
18:       $M = m$, opt = val;
19:     **end if**
20:   **end if**
21: **end while**

6.3 Strategies for reducing search space

The search space (or the times of calling KM algorithm) in our algorithm is equal to the number of maximal non-conflict device set. Assume the average number of the elements in a non-conflict device set is nc, then the search space scale is $C_{|D|}^{\mathrm{nc}}$. In this subsection, we discuss how to reduce the search space in some specific input data.

6.3.1 *Reduce the non-conflict device set*

The first strategy for reducing search space is to reduce the elements in non-conflict device set. When the input satisfies the property: $\mathrm{nc} \gg |P|$, i.e., the number of elements in a maximal non-conflict device set is much greater than that of programs, we do not need to enumerate all the maximal non-conflict device set. Instead, we can only traverse all the non-conflict device sets, each of which has exactly $|P|$ elements. If we do that, the search space is reduced to $C_{|D|}^{|P|}$. The correctness is easily to be proved as follows. Since the match in the solution has at most $|P|$ device nodes, which must be enumerated in the above step.

6.3.2 *Delete dominated device*

Before giving the second strategy, we first introduce a definition. Assume $d, d' \in D$ denote two device nodes. The sets $C_d, C_{d'}$ are the sets of the conflicted device nodes of d and d'. We call a device d dominate d' if and only if the following properties holds.

- $\langle d, d' \rangle \in C$;
- $C_d \subset C_{d'}$;
- For any $p \in P$, $w(p, d) \geqslant w(p, d')$.

With this definition, we can reduce the device set: for each $d' \in D$, if there exists a device node d such that d dominate d', then we can delete d' from D. The correctness is also easily to be proved. Assume that the match in the solution contains d'. Then the device node d must not be in the solution. We can replace it with d and the weight sum must not decrease, and d must not be conflicted with the rest device nodes in the solution.

6.3.3 *Merge devices in a clique*

We can construct a undirected graph $G^D = \langle D, C \rangle$ by regarding the device set and conflicted device set as the vertex set and edge set. If a subset $D' \subset D$ satisfies the following properties, then we can merge the devices into one.

- D' is a clique in G^D;
- For a device $d \in D$, assume $D^c(d) = \{d' | d' \notin D' \wedge (d, d') \in C\}$. For any two devices $d, d' \in D'$, $D^c(d) = D^c(d')$.

Assume we merge the devices in D' into a new node $d^\dagger$, then we construct new edges in the following way:

$$w(p, d^\dagger) = \max\{w(p, d') | d' \in D'\}.$$

7 Experiments

To evaluate the effectiveness of our approaches, we implement our algorithms in C++ language. The experiments are conducted on a machine with Intel Core 3.6 GHz (4 cores) and 8 GB memory, Windows 10 operating system.

7.1 Design of the experiments

To evaluate the effectiveness of our approach, we raise two research questions as follows.

Q1. Can our new strategies improve the BnB algorithm to find better quality solutions? Which combination of these strategies perform best?

Q2. What is the effectiveness of our approach, compared with existing approaches?

To answer the above questions, we carried out on two benchmarks, including real-world application instances and random instances.

The real-world data is taken from [9]. According to the longitude and latitude, the earth is divided into 85 regions and 911 stations, and the quality of signal launched by a device to a station is calculated according to the transmitter, antenna information and the distance between the device and area. If the quality exceeds the threshold, we call the device cover the station. In addition, if a device can cover more than 80% of stations in an area, we call the device cover the area.

For Q1, we configure our algorithm with different combinations of the strategies (including without any strategy), and apply them to the real-world application instances. Through comparing the best found quality solution, we conclude the best combination of our strategies. For Q2, we configure our algorithm with the best combination to find the quality solution, and compare the results generated by our approach with that of CLASP, CPLEX, and the local search (LS) solver shown in [9].

7.2 Effectiveness of strategy combinations

First of all, we investigate the effectiveness and efficiency of our new strategies. Clearly, there are four arrangements for the priority of the three strategies. To determine which one is best, we have tried all of them and conclude that the order "highest quality, more edge, least conflict" performs best. We evaluate our approach on a benchmark of application instances in the work of Ma et al. [9], and set the time limit as 180 s. The results are shown in Table 1. The first two columns indicate the number of programs and devices. It can be observed that the incremental strategy combinations improve the best found quality solution (#QS) incrementally, and for most cases, the time at which #QS is found decreases a lot, especially for large data sets. The summary results in Table 2, which includes the comparisons "Weight vs. None", "Weight+Edge vs. Weight" and "Weight+Edge+Conflict vs. Weight". The table compares three strategies and their results. "Improving", "equal" and "worse" indicate that the former strategy outperforms, keeps and underperforms the latter one, respectively.

7.3 Comparison with other methods

In this section, we compare our method with other methods on application benchmark and random benchmark. This includes the comparisons to the results of CLASP, CPLEX and the local search (LS) solver.

7.3.1 *Results on application benchmark*

Table 3 shows the comparative experimental results. For each method, we list the best found quality solution (#QS) and the time to get it. Note that the local search solver invokes a randomized construction phase and the solution is also random. So we list the #QS column for the local search solver both the best solution quality, and the average solution quality in parentheses. In contrast, our method is deterministic, and also achieves the results as qualitative as theirs. Therefore, it is not convictive to summarize the best solutions directly for comparison.

From the results, we have the following observations. For small sized instances, CPLEX is the best. Both CPLEX and CLASP become futile when the instance size becomes large, e.g., $|P| \geqslant 4000$ and $|D| \geqslant 50$. Our approach and local search survive through the whole benchmark, and they show competitive performance. For medium sized instances, the local search method performs the best, slightly better than our approach. However, for two largest instances (the last two rows), our approach is the best. This might seem a little odd for BnB methods outperforming local search on very large instances. The reason why our approach performs better is below. The local search method used in [9] picks a starting point in search space randomly. However, we choose the starting point of the depth first searching process by the three strategies listed in Subsection 6.3. Moreover, the strategies focus on picking the potential devices

Table 1 Results of different versions of our method on application benchmark[a)]

P	D	None		Weight		Weight+Edge		Weight+Edge+Conflict	
		#QS	Time (s)	#QS	Time (s)	#QS	Time (s)	#QS	Time (s)
2	50	94*	<0.04	94*	<0.05	94*	<0.01	94*	<0.001
5	50	103*	3.92	103*	5.14	103*	5.11	103*	0.141
2	100	94*	0.012	94*	0.011	94*	0.01	94*	<0.001
5	100	103*	9.43	103*	5	103*	11.45	103*	0.23
2	200	121*	0.016	121*	0.013	121*	0.016	121*	<0.001
5	200	206	13.5	238*	5	238*	2.03	238*	2.04
10	200	393	180	645	40.6	645	40.2	645	42.9
2	400	187*	0.017	187*	0.16	187*	0.016	187*	<0.001
5	400	406	142	438*	20.8	438*	21	438	20.8
10	400	416	85.2	845	46.1	845	46.2	845	46.7
20	400	588	84.9	761	175	761	169	761	160
2	800	212*	0.448	212*	0.434	212*	0.427	212*	<0.001
5	800	408	140	447	0.22	447	0.222	447	0.222
50	4000	2837	129	4471	0.043	4471	0.045	4471	0.049
60	4000	3073	103	5143	0.239	5143	0.24	5143	0.238
70	4000	3780	25.2	6057	0.306	6057	0.35	6057	0.326
87	4000	4776	160	6824	0.144	6824	0.138	6824	0.141
50	5000	2802	54.7	4613	0.223	4612	0.22	4612	0.22
60	5000	3073	151	5313	0.284	5313	0.286	5313	0.278
70	5000	3780	37.3	6277	0.143	6280	0.294	6280	0.3
87	5000	4771	96.2	7099	0.19	7099	0.18	7099	0.18
50	6000	2826	72.5	4989	2.6	4989	2.63	4989	2.59
60	6000	3054	158	5784	2.18	5784	2.18	5784	0.189
70	6000	3804	51.4	6861	49.4	6861	49.3	6861	49
87	6000	4795	136	7745	165	7742	0.748	7742	0.754
50	7061	2826	76	5006	5.33	5006	5.43	5006	5.24
60	7061	3054	167	5821	53.1	5821	53.1	5820	0.554
70	7061	3804	54.2	6948	0.189	6948	0.181	6948	0.182
87	7061	4795	144	7893	1.81	7902	0.633	7902	0.619

a) * denotes the optimal value.

Table 2 Summary on comparisons between different versions

Strategy	Improving	Equal	Worse
Weight vs. None	29	0	0
Weight+Edge vs. Weight	18	3	8
Weight+Edge+Conflict vs. Weight	25	2	2

that could be matched to the programs, rather than a concrete matching itself as the local search in [9] does. This idea works since Kuhn-Munkres algorithm finds efficiently the best matching among all the available solutions using these devices.

7.3.2 *Results on random benchmark*

To further evaluate the effectiveness of our branch-and-bound algorithm, we randomly generate 100 instances and apply our two algorithms to calculate the optimal matching solution. Since the PBO algorithm (CLASP) may cost a lot of time to get the optimal solution, we set a time limit (1000 s). Figure 8 shows the experimental results of these 100 instances. From Figure 8, we can see that the PBO algorithm fails to give the solution within the time limit on a quart of these instances, while the branch-and-bound algorithm can give the best solution within 200 s, and 27 s on average.

Table 3 Experimental results on application benchmark[a)]

P	D	CLASP		CPLEX		LS		Ours	
		#QS	Time (s)	#QS	Time (s)	#QS	Time (s)	#QS	Time (s)
2	50	94*	<0.01	94*	<0.01	94(94)	<0.01	94*	<0.001
5	50	103*	0.171	103*	0.19	103(103)	<0.01	103*	0.141
2	100	94*	0.016	94*	<0.01	94(94)	<0.01	94*	<0.001
5	100	103*	0.171	103*	0.14	103(103)	<0.01	103*	0.23
2	200	121*	0.078	121*	0.14	121(121)	<0.01	121*	<0.001
5	200	238	7.269	238*	0.16	238(238)	<0.01	238*	2.04
10	200	762	273.995	762*	0.27	762(762)	0.014	654	42.9
2	400	187*	0.577	187*	0.13	187(187)	<0.01	187*	<0.001
5	400	438	25.569	438*	0.09	438(438)	0.015	438	20.8
10	400	965	473.975	965*	0.19	965(965)	0.047	845	46.7
20	400	1228	3200.502	1232*	0.28	1232(1231.9)	3.411	761	160
2	800	212*	6.209	212*	0.16	212(212)	0.026	212*	<0.001
5	800	501	1589.128	501*	0.14	501(501)	0.127	447	0.222
50	4000	1060	3004.924	–	–	4577(4541.7)	8.500	4471	0.049
60	4000	–	–	–	–	5387(5329.4)	9.884	5143	0.238
70	4000	–	–	–	–	6284(6249.9)	9.806	6057	0.326
87	4000	–	–	–	–	7029(6986)	9.824	6824	0.141
50	5000	927	2892.373	–	–	4708(4681.5)	9.845	4612	0.22
60	5000	–	–	–	–	5496(5416.4)	9.912	5143	0.238
70	5000	–	–	–	–	6373(6318)	9.795	6280	0.3
87	5000	–	–	–	–	7186(7142.9)	9.712	7099	0.18
50	6000	–	–	–	–	5030(4990.4)	9.803	4989	2.59
60	6000	–	–	–	–	5856(5788.9)	9.826	5784	0.189
70	6000	–	–	–	–	6878(6797.5)	9.842	6861	49
87	6000	–	–	–	–	7854(7667)	9.687	7742	0.754
50	7061	–	–	–	–	5068(5045.9)	9.828	5006	5.24
60	7061	–	–	–	–	5883(5805.7)	9.909	5820	0.554
70	7061	–	–	–	–	6860(6800.0)	9.779	6948	0.182
87	7061	–	–	–	–	7739(7629.6)	9.620	7902	0.619

a) * denotes the optimal value.

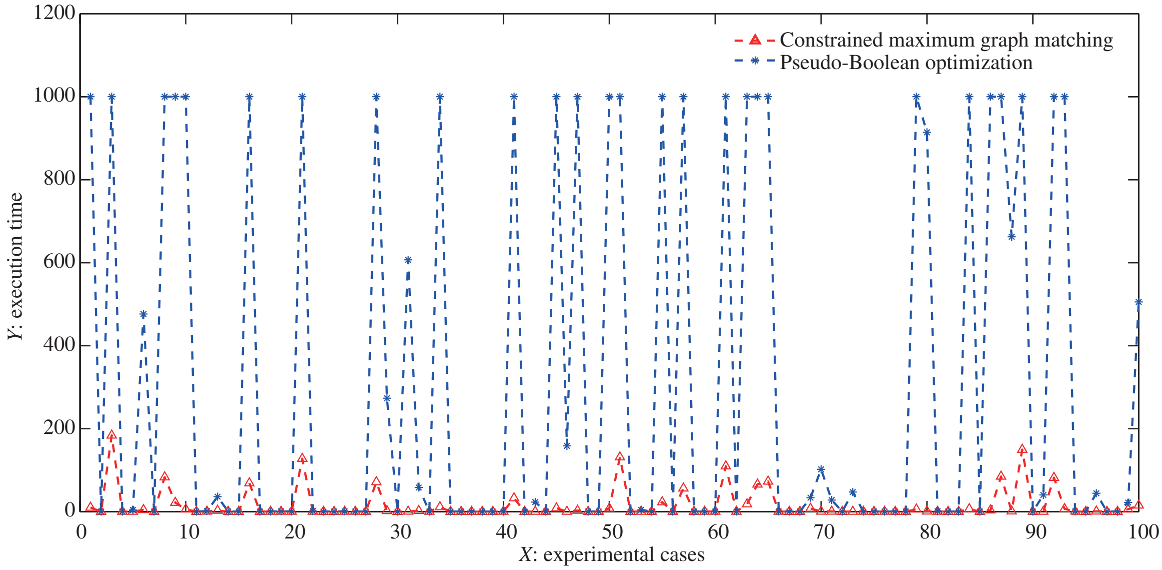

Figure 8 Experimental results of random data.

8 Conclusion

We propose an effective algorithm based on branch-and-bound to solve the resource allocation problem in wireless communication. Specifically, we model the problem to constrained maximum weighted bipartite matching problem. To quickly find the best matching pair, we introduce three rules for selecting the next exploration devices and an upper bound estimation function. Our experimental results show that these techniques can greatly reduce the searching space and decrease the execution time. In future, we will try to apply this to more applications, such as subway scheduling.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 61772503) and National Basic Research Program of China (Grant No. 2014CB340302).

References

1 Li Y, Nie F, Huang H, et al. Large-scale multi-view spectral clustering via bipartite graph. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, 2015. 2750–2756
2 Gu T L, Chang L, Xu Z B. A novel symbolic algorithm for maximum weighted matching in bipartite graphs. Int J Commun Netw Syst Sci, 2011, 4: 111–121
3 Beale S. Using branch-and-bound with constraint satisfaction in optimization problems. In: Proceedings of the 14th National Conference on Artificial Intelligence and 9th Innovative Applications of Artificial Intelligence Conference, Providence, 1997. 209–214
4 Lelis L H S, Otten L, Dechter R. Predicting the size of depth-first branch and bound search trees. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, 2013
5 Zhan C, Xiao F. Coding based wireless broadcast scheduling in real time applications. J Network Comput Appl, 2016, 64: 194–203
6 Kuhn H W. The Hungarian method for the assignment problem. Naval Res Log, 1955, 2: 83–97
7 Munkres J. Algorithms for the assignment and transportation problems. J Soc Industrial Appl Math, 1957, 5: 32–38
8 Mitchell D G. A sat solver primer. Bull EATCS, 2005, 85: 112–132
9 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 the 22nd International Conference on Principles and Practice of Constraint Programming, Toulouse, 2016. 650–665
10 Pan L, Jin J, Gao X, et al. Integrating ILP and SMT for shortwave radio broadcast resource allocation and frequency assignment. In: Proceedings of the 23rd International Conference on Principles and Practice of Constraint Programming, Melbourne, 2017. 405–413
11 Li C M, Quan Z. An efficient branch-and-bound algorithm based on maxsat for the maximum clique problem. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence, Atlanta, 2010