Received March 2, 2019, accepted March 19, 2019, date of publication March 27, 2019, date of current version April 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2907708

Room Allocation With Capacity Diversity and Budget Constraints

YUNPENG LI[1,2], YICHUAN JIANG[2], (Senior Member, IEEE), WEIWEI WU[2], JIUCHUAN JIANG[3], AND HUI FAN[4]

[1]Innovation Group for Interdisciplinary Computing Technologies, College of Computer Science and Technology, Xi'an University of Science and Technology, Xi'an, China
[2]School of Computer Science and Engineering, Southeast University, Nanjing, China
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[4]Co-Innovation Center of Shandong Colleges and Universities-Future Intelligent Computing, Shandong Technology and Business University, Yantai, China

Corresponding authors: Yunpeng Li (yunpengli.seu@gmail.com) and Yichuan Jiang (yjiang@seu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61472079, Grant 61806053, and Grant 61807008, and in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20171363, Grant BK20180356, and Grant BK20180369.

ABSTRACT With the development of the room rental market, many room rental websites have been created, e.g., SpareRoom and EasyRoommate. On these websites, people find not only rooms for rent but also suitable roommates. Inspired by the rental mode in practice, a benchmark room allocation model was introduced by Chan *et al.*, in which $2n$ agents must be allocated to n rooms that have the same capacity and each agent can be allocated to any room. However, in practice, rooms may differ in terms of capacity, e.g., college dorms or apartments may contain both two-bed rooms and four-bed rooms. Moreover, an agent can only be allocated to a room of which the rent does not exceed the agent's budget. In this scenario, we must consider not only the agents' preferences but also the capacity diversity of the rooms and the budget constraints while allocating the rooms. Therefore, this paper investigates the room allocation problem with capacity diversity and budget constraints. We mainly focus on finding an allocation that maximizes social welfare. First, this paper demonstrates that finding an allocation that maximizes the social welfare is NP-hard (i.e., non-deterministic polynomial-time hard), even if only one room's capacity is larger than 1 and the other rooms' capacities are all 1. Second, this paper presents a $(c^* + 2)/2 + \varepsilon$-factor approximation algorithm (with $\varepsilon > 0$) for the case in which the capacity of each room does not exceed a constant c^*. Third, this paper proposes a heuristic algorithm based on the local search for the general case in which the capacity of each room is not bounded by a constant. The experimental results demonstrate that the proposed algorithm can produce near-optimal solutions. Finally, this paper investigates how to find a roommate stable or room envy-free allocation with a social welfare guarantee.

INDEX TERMS Room allocation, capacity diversity, budget constraints, algorithm design.

I. INTRODUCTION

Room allocation is a frequently encountered problem in practice, e.g., allocating dorms among college students and allocating rooms among tenants [1], [2]. It has attracted the attention of economists and computer scientists for more than sixty years. Gale and Shapley [3] introduced the stable roommate problem, where the objective is to find a stable matching that partitions $2n$ agents into n pairs of roommates such that no two agents who are not roommates both prefer each other to their actual partners. They showed that an instance does not always admit a stable matching. Irving [4] presented an $O(n^2)$ algorithm that can determine whether an instance with strict preferences admits a stable matching and find such a matching if it exists. Since then, the stable roommate problem has been studied extensively [5]–[10]. Moreover, various types of solution notions have been proposed, for example, exchange stability [11]–[13] and popular matching [14]. In the stable roommate problem, only the preferences for the agents are considered and each agent has only one roommate.

With the development of the room rental market, many room rental websites have been created, e.g., Spareroom and Easyroommate. People find not only rooms for rent but also suitable roommates on these websites. Inspired by the rental

The associate editor coordinating the review of this manuscript and approving it for publication was Xudong Zhao.

mode in practice, Chan *et al.* [1] presented a room allocation model, in which $2n$ agents must be allocated to n rooms according to their preferences for both the rooms and the other agents. They assumed that each room can contain two agents and each agent can be allocated to any room. However, in practice, rooms may differ in terms of capacity, e.g., college dorms or apartments may have both two-bed rooms and four-bed rooms. Moreover, an agent can only be allocated to a room of which the rent does not exceed the agent's budget. In the scenario, not only the agents' preferences but also the capacity diversity of the rooms and budget constraints must be considered while allocating the rooms.

To solve this problem, this paper investigates the room allocation problem with capacity diversity and budget constraints. We mainly focus on finding an allocation that maximizes the social welfare that is defined as the utility sum of the agents, including the room owner (such as the university). Typically, this objective is discussed in the resource allocation literature [1], [2], [15]–[17]. This paper also investigates how to find a roommate stable allocation with a social welfare guarantee, in which no pair of agents who live in different rooms can increase both their utilities by swapping while ensuring that no agent's utility decreases. Moreover, this paper investigates the computational complexity of finding a room envy-free allocation with a social welfare guarantee, in which each pair of roommates would not like to switch rooms with any other pair of roommates. The main results of this paper are summarized as follows:

1. Firstly, this paper demonstrates that finding an allocation that maximizes the social welfare is NP-hard (i.e., non-deterministic polynomial-time hard), even if only one room's capacity is larger than 1 and the other rooms' capacities are all 1.

2. Secondly, this paper presents a polynomial-time $(c^* + 2)/2 + \varepsilon$-factor approximation algorithm (with $\varepsilon > 0$) for the case in which the capacity of each room does not exceed a constant $c^* > 0$. Because the capacity of a room in a residential rental market is typically not larger than 4, the proposed algorithm is usually a $3+\varepsilon$-factor approximation algorithm in real tenant allocation.

3. Thirdly, this paper demonstrates that there is no polynomial-time $c^*/2$-factor approximation algorithm for the social welfare maximization problem unless P=NP (i.e., each non-deterministic polynomial-time complete problem can be solved in polynomial time), where c^* is the capacity upper-bound of each room.

4. Fourthly, this paper proposes a heuristic algorithm based on local search for the general case in which the capacity of each room is not bounded by a constant. Experimental results demonstrate that the proposed algorithm can produce near-optimal solutions.

5. Fifthly, this paper demonstrates that if the capacity of each room does not exceed a constant $c^* > 0$, we can find a roommate stable allocation of which the social welfare is at least $1/((c^* + 2)/2 + \varepsilon)$ $(\varepsilon > 0)$ of the optimal value in polynomial time.

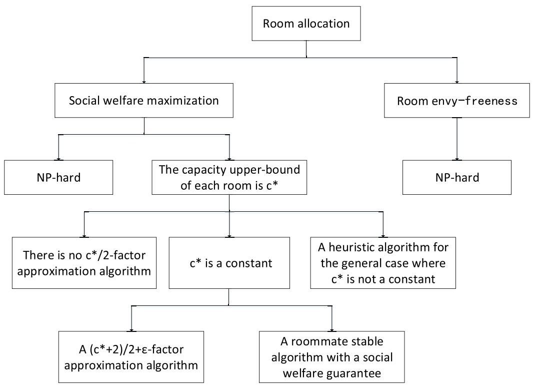

Room allocation
Social welfare maximization
Room envy-freeness
NP-hard
The capacity upper-bound of each room is c*
NP-hard
There is no c*/2-factor approximation algorithm
c* is a constant
A heuristic algorithm for the general case where c* is not a constant
A (c*+2)/2+ε-factor approximation algorithm
A roommate stable algorithm with a social welfare guarantee

FIGURE 1. The main results of this paper.

6. Lastly, this paper demonstrates that it is NP-hard to determine whether an instance with specified room prices admits a room envy-free allocation.

The remainder of this paper is organized as follows. In Section II, we review the related work. In Section III, we present the formal model of the room allocation problem. In Section IV, we investigate how to find an allocation that maximizes the social welfare. In Section V, we investigate how to find the allocation that maximizes the social welfare among all the allocations that have the maximum trade volume. The trade volume is defined as the sum of the agents' payments. In Section VI, we investigate how to find a roommate stable or room envy-free allocation with a social welfare guarantee. In Section VII, we present the experimental results. In Section VIII, we present the conclusions of this paper and discuss future work.

II. RELATED WORK

The stable roommate problem was introduced by Gale and Shapley [3]. They showed that an instance does not always admit a stable matching. Irving [4] proposed the first algorithm that can determine whether an instance with strict preferences admits a stable matching and find such a matching, if it exists, in $O(n^2)$ time. Bartholdi, III, and Trick [18] showed that the stable roommate problem with narcissistic and single-peaked preferences always admits a unique stable matching if the preferences are strict and complete and provided an $O(n)$ time algorithm for finding this matching. Ronn [10] showed that finding a stable matching is NP-hard if the agent preferences contain ties. Feder *et al.* [6] proposed an $O(n^3 logn)$ parallel algorithm for the stable roommate problem with strict preferences. Irving and Manlove [8] presented a line-time algorithm for finding a superstable matching if one exists, given a stable roommate instance with ties. Bredereck *et al.* [5] presented polynomial-time algorithms for the stable roommate problem with complete narcissistic, single-peaked and single-crossing preferences. Pęski [9] investigated the stable roommate problem with nontransferable random utility and analyzed the distribution of types of matched pairs in stable matchings. Pittel [19] investigated the stable roommate problem with unequal numbers of men and women. Lam and Plaxton [20] studied the problem of finding large weakly stable matchings when preference lists are incomplete and contain onesided ties and presented a $1 + 1/e$-factor approximation algorithm.

Moreover, various notions of stability have been proposed. For example, Alcalde [11] proposed "coalition exchange stability", in which there is no agent coalition that can increase the utility of each agent in the coalition via partner swapping. Cechlárová [12] defined a matching as exchange stable if there is no pair of agents that can increase both their utilities by swapping their partners. They showed that determining whether an instance admits an exchange-stable matching is NP-complete. Biró *et al.* [14] studied popular matching for the roommate allocation problem and showed that determining whether a popular matching exists in a roommate allocation instance is NP-hard if ties are permitted in the preferences. In the stable roommate problem, only the preferences for the agents are considered and each agent has only one roommate. However, in the problem of this paper, we also consider the preferences for the rooms and each agent may have multiple roommates.

Recently, Chan *et al.* [1] presented a room allocation model in which $2n$ agents must be allocated to n rooms that have the same capacity. They assumed that the valuation function of each agent is nonnegative and proposed a constant-factor approximation algorithm for the social welfare maximization problem in an offline situation. They also studied stable room allocation. Huzhang *et al.* [2] investigated the online situation and assumed that all the agents arrive online in uniformly random order. They presented an online algorithm with constant competitive ratio with respect to the optimal social welfare. In contrast, this paper investigates the more general offline case in which the rooms have diverse capacities and each agent has a limited budget.

Popular matching in the house allocation problem was first studied by Abraham *et al.* [21]. The objective is to find a popular matching M that allocates agents to one-capacity houses such that there is no matching M' with the property that the number of agents preferring M' to M exceeds the number of agents preferring M to M'. Abraham *et al.* [21] proposed an $O(\sqrt{n}m)$ algorithm for the problem, where n is the total number of agents and houses, and m is the total length of all the preference lists. Mahdian [22] studied the existence of popular matchings in a random instance of the house allocation problem. Mestre [23] studied a general case in which agents have different priority weights, and the objective is to find a weighted popular matching. They presented an $O(m+n)$ algorithm for instances that have strict preference lists.

Manlove and Sng [24] investigated the capacitated house allocation problem, in which the houses have diverse capacities. They presented an $O((\sqrt{C} + n_1)m)$ algorithm for the capacitated house allocation problem, where C is the total capacity of the houses and n_1 is the number of agents. Sng and Manlove [25] studied the weighted capacitated house allocation problem and proposed an $O(\sqrt{C}n_1 + m)$ algorithm for instances that have strict preference lists. Paluch [26] presented a faster (by a factor of $O(\sqrt{n})$) algorithm for the capacitated house allocation problem with ties. In the capacitated house allocation problem, only the preferences for the houses are considered. However, in the problem of this paper, we consider the preferences for both the rooms and agents.

Hedonic games are also closely related to the problem of this paper, in which n agents must be partitioned into disjoint coalitions according to their preferences for coalitions [27]–[31]. Various stability concepts, e.g., the core stability, Nash stability, individual stability and contractual individual stability (see e.g., [29]), have been proposed for analyzing these games. Variants of hedonic games have also been studied, e.g., additively separable hedonic games [32], fractional hedonic games [33]–[35], roommate games [36] and the group activity selection problem [37], [38]. In the games, the preference of an agent for a coalition is determined by the identities of the agent members of the coalition. In the group activity selection problem, the preference of an agent for an activity coalition depends on the number of agent members in the activity coalition. However, in the problem of this paper, the preference of an agent for a room allocation is determined by the preferences for both the roommates and the room, rather than only for the roommates or the number of roommates.

III. PRELIMINARIES

Let $A=\{1, 2, \ldots, n\}$ be the agent set and $\Re = \{r_1, r_2, \ldots, r_m\}$ be the room set. Each agent i has a budget b_i and a valuation function v_i: $\Re \cup A \rightarrow \mathbb{R}$. For a room $r \in \Re$, $v_i(r)$ represents the happiness of agent i when living in it and for an agent $j \in A$, $v_i(j)$ represents the happiness of agent i when agents i and j are allocated to the same room. If $v_i(r)$ (or $v_i(j)$) is smaller than 0, it means that agent i hates living in room r (or with agent j). Without loss of generality, we assume $v_i(\emptyset) = 0$. We use p_i ($0 \leq p_i \leq b_i$) to represent the payment of agent i and use p_r ($p_r \geq 0$) to represent the price (total rent) of room r. The payment p_i of agent i must not exceed its budget b_i. Let c_r ($c_r \geq 1$) be the capacity of room $r \in \Re$, namely, room r has c_r beds. We assume that the capacity sum of the rooms is n. Each agent can be allocated to at most one bed in the rooms and each bed can be allocated to at most one agent. Moreover, we assume that each bed in the same room has the same price [1] and that each agent only pays for the bed to which it is allocated. This scenario is very common in practice, e.g., dorm allocation at a university.

A feasible allocation π can be denoted as a vector $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$, where π_i is the room to which agent i is allocated and two constraints are satisfied: i) for each agent i, the bed rent of π_i does not exceed b_i (i.e., $p_{\pi_i}/c_{\pi_i} \leq b_i$) and ii) for any room r, at most c_r agents are allocated to it. The utility of agent i in the allocation π is defined as

$$u_i(\pi) = v_i(\pi_i) + \sum_{j \in A \setminus \{i\} \wedge \pi_j \neq \emptyset \wedge \pi_i = \pi_j} v_i(j) - p_i$$

Due to the budget constraints, an agent k may not be allocated to any room, namely, $\pi_k = \emptyset$ with $p_k = 0$. Inspired by [1], the social welfare of the allocation π is defined as

$$SW(\pi) = \sum_{i\in A} u_i(\pi) + \sum_{i\in A} p_i = \sum_{i\in A}(v_i(\pi_i) + \sum_{j\in A\setminus\{i\}\wedge\pi_j\neq\emptyset\wedge\pi_i=\pi_j} v_i(j))$$

In the definition, we consider both the utilities of the tenant agents and the utility of the room owner (such as the university). In this paper, we initially focus on how to find an allocation that maximizes the social welfare.

When there are insufficiently many low-cost rooms, we cannot always ensure that every agent is allocated to a room. In this situation, the room owner or the rental intermediary typically hopes to maximize the trade volume, which equals the sum of the agents' payments. This is because the profit of the room owner or the rental intermediary is positively related to the trade volume. Therefore, this paper further investigates the social welfare maximization problem with the maximum trade volume constraint, namely, finding the allocation that maximizes the social welfare among the allocations that have the maximum trade volume.

This paper also investigates how to find a roommate stable or room envy-free allocation with a social welfare guarantee. Inspired by the "exchange stability" in the traditional stable roommate problem [11]–[13], Chan *et al.* [1] defined an allocation as 2-person stable if no pair of agents (i, j) who live in different rooms can increase both their utilities by swapping. In this work, they showed that determining whether a room allocation instance admits a 2-person stable allocation is NP-hard. However, in practice, a swap is typically infeasible if someone else opposes, e.g., the roommates of agent i or j. Then, inspired by [39], we propose the following notion of stability:

Definition 1: An allocation is 2-person weakly stable if no pair of agents (i, j) who live in different rooms can increase both their utilities by swapping while ensuring that no agent's utility decreases.

We refer to two agents who violate the person stability condition as "2-person weakly stable blocking pair". In this paper, we investigate how to find a 2-person weakly stable allocation with a social welfare guarantee.

Envy-freeness is a stronger solution concept than stability. A person envy-free allocation in which each agent would not like to switch rooms with any other agent must be a 2-person stable allocation. Finding a person envy-free allocation [1] is NP-hard. Then, Chan *et al.* [1] proposed a weaker concept – room envy-freeness. An allocation is room envy-free if no pair of roommates (i, j) envies any other pair of roommates (k, l). In this paper, we extend the definition of room envy-freeness to consider budget constraints:

Definition 2: An allocation is room envy-free if no pair of roommates (i, j) envies any other pair of roommates (k, l) under the specified budget constraints. That is, if (i, j) are allocated to room r, we cannot increase the utility sum of agents i and j by allocating them to another room s with $b_i \geq p_s/c_s$ and $b_j \geq p_s/c_s$.

Room envy-freeness implies that each pair of roommates would not like to switch rooms with any other pair of roommates. Chan *et al.* [1] showed that if each room can contain at most two agents and the room prices are adjustable, a room envy-free allocation can be found in polynomial time. However, this paper demonstrates that determining whether an instance with specified room prices admits a room envy-free allocation is NP-hard (see Section VI). In Table I, we present a summary of notations used in the paper.

TABLE 1. **Definitions of notations.**

Notation	Definition
A	The agent set
$\Re$	The room set
$v_i(r)$	The happiness of agent i when living in room r
$v_i(j)$	The happiness of agent i when agents i and j are allocated to the same room
b_i	The budget of agent i
p_r	The price (total rent) of room r
c_r	The capacity of room r
π_i	The room to which agent i is allocated in the allocation π
$u_i(\pi)$	The utility of agent i in the allocation π
$SW(\pi)$	The social welfare of the allocation π

IV. SOCIAL WELFARE MAXIMIZATION

In this section, first, we investigate the computational complexity of finding an allocation that maximizes the social welfare. Second, we propose a mixed-integer linear programming formulation for computing the optimal allocation. Third, we investigate the approximation algorithms for the problem. Finally, we investigate the dual problem of social welfare maximization.

A. COMPLEXITY ANALYSIS

Theorem 1: Finding an allocation that maximizes the social welfare is NP-hard, even if only one room's capacity is larger than 1 and the other rooms' capacities are all 1.

Proof: We prove the NP-hardness by a reduction from the clique problem. Given an undirected graph $G=(V, E)$ and a positive integer $k>1$, the clique problem is to determine whether G has a set of k mutually adjacent vertices. The clique problem is NP-complete [40]. Given an instance $<G, k>$ of the clique problem, we construct the corresponding instance of the room allocation problem as follows:

Let n be the number of vertices in graph G. First, we create n-k+1 rooms, where the capacity of room r_1 is k and the capacity of any other room is 1. We use $\Re$ to represent the room set. Second, we create an agent i for each vertex $u_i \in V$. If $(u_i, u_j)\in E$, we set $v_i(j) = v_j(i) = 1/2$; otherwise, we set $v_i(j) = v_j(i) = 0$. For each agent i and each room $r\in \Re$, we set $v_i(r) = 1/2$. Moreover, we set $b_i = 1$ for each agent i and set $p_r = 0$ for each room r. According to the settings, each agent can be allocated to any room and the utility of an agent who is allocated to a room $r \in \Re \backslash \{r_1\}$ is 1/2. Because at most k agents can be allocated to room r_1, the maximum utility that an agent can obtain is $(k\text{-}1)/2+1/2$. As a result, the maximum social welfare of the constructed instance of the room allocation problem does not exceed $k(k\text{-}1)/2+n/2$. Let π^* be the optimal allocation that maximizes the social welfare.

The $\Leftarrow$ direction: If $SW(\pi^*)=k(k\text{-}1)/2+n/2$, the utility of any agent i who is allocated to room r_1 must be $(k\text{-}1)/2+1/2$. We use A_1 to represent the agents who are allocated to room r_1. According to the preference setting mode, the corresponding vertices of the agents in A_1 are mutually adjacent. Then, the graph G has a clique of size k.

The $\Rightarrow$ direction: If G has a set of k mutually adjacent vertices, we can find an allocation that has a social welfare of $k(k\text{-}1)/2+n/2$ for the constructed instance of the room allocation problem. The strategy is to allocate the agents who correspond to the k mutually adjacent vertices to room r_1 and randomly allocate the remaining agents to the other rooms. Based on the preference setting mode, the utility of any agent who has been allocated to room r_1 is $(k\text{-}1)/2+1/2$ and the social welfare of the allocation is $k(k\text{-}1)/2+n/2$. Because the maximum social welfare of the constructed instance of the room allocation problem does not exceed $k(k\text{-}1)/2+n/2$, the proposed allocation has the maximum social welfare, i.e., $SW(\pi^*)=k(k\text{-}1)/2+n/2$.

Based on the above analyses, $SW(\pi^*)=k(k\text{-}1)/2+n/2$ if and only if G has a set of k mutually adjacent vertices. Then, we can determine whether G has a set of k mutually adjacent vertices by finding the optimal allocation π^* for the constructed instance and comparing its social welfare with $k(k\text{-}1)/2+n/2$. □

In this paper, the □ is used to indicate that the proof ends.

B. EXACT SOLUTION

Although the room allocation problem is NP-hard, it is feasible to compute the optimal allocation if the problem scale is small. In this subsection, a mixed-integer linear programming formulation is proposed for computing the optimal allocation, where $x_{ir} = 1$ represents that agent i is allocated to room r; otherwise $x_{ir} = 0$. The formulation is expressed as follows:

$$\max \sum_{i \in A} y_i$$

$$\text{s.t.} \sum_{r \in \Re} (x_{ir} \cdot v_i(r) + \sum_{j \in A \backslash \{i\}} z_{ij}^r \cdot v_i(j)) - y_i \geq 0 \quad \forall i \in A \quad \text{(a)}$$

$$\sum_{r \in \Re} x_{ir} \leq 1 \quad \forall i \in A \quad \text{(b)}$$

$$x_{ir} \cdot p_r / c_r \leq b_i \quad \forall i \in A, \forall r \in \Re \quad \text{(c)}$$

$$\sum_{i \in A} x_{ir} \leq c_r \quad \forall r \in \Re \quad \text{(d)}$$

$$z_{ij}^r \leq x_{ir} \quad \forall i, j \in A, \ \forall r \in \Re \quad \text{(e)}$$

$$z_{ij}^r \leq x_{jr} \quad \forall i, j \in A, \ \forall r \in \Re \quad \text{(f)}$$

$$x_{ir} + x_{jr} - z_{ij}^r \leq 1 \quad \forall i, j \in A, \forall r \in \Re \quad \text{(g)}$$

$$y_i \geq 0 \quad \forall i \in A \quad \text{(h)}$$

$$x_{ir}, z_{ij}^r \in \{0, 1\} \quad \forall i, j \in A, \forall r \in \Re \quad \text{(i)}$$

In the formulation, y_i is the lower bound of $u_i + p_i$ for any agent $i \in A$. Constraint (b) ensures that each agent is allocated to at most one room. Constraint (c) ensures that the budget constraints are satisfied. Constraint (d) ensures that the capacity constraints of the rooms are satisfied. Constraints (e)–(g) ensure that $z_{ij}^r = 1$ if and only if $x_{ir} = 1$ and $x_{jr} = 1$. These constraints can be viewed as the linearization of $z_{ij}^r = x_{ir} \cdot x_{jr}$ based on the McCormick inequalities [41].

Based on the mixed-integer linear programming formulation, we can compute the optimal allocation using a standard solver (e.g., Cplex) for mixed-integer linear programming. However, because the problem is NP-hard, exponential time is required for computing the optimal allocation in the worst case unless P=NP. Thus, the mixed-integer linear programming formulation is not applicable to large-scale instances. Therefore, we consider designing approximation algorithms for the social welfare maximization problem.

C. APPROXIMATION ALGORITHM

1) RESTRICTED CASE

The capacity of a room in a residential rental market typically does not exceed 4. Thus, we design an approximation algorithm for the restricted case in which the capacity of each room does not exceed a constant $c^* > 0$.

Theorem 2: If the capacity of each room does not exceed a constant $c^* > 0$, there is a polynomial-time $(c^* + 2)/2 + \varepsilon$-factor approximation algorithm (with $\varepsilon > 0$) for the social welfare maximization problem.

Proof: The algorithm is presented as Algorithm 1. In the algorithm, for each room $r \in \Re$, we enumerate all the feasible allocation states. For each feasible allocation state, we create a set that includes the related agents and room, and use the social welfare of the allocation state (i.e., the utility sum of the agents, including the room owner) as the set weight (Algorithm 1, lines 1–8). Because the capacity of each room does not exceed c^*, this process requires $O(c^* m n^{c^*})$ time. Because each room only has one allocation state and each agent is allocated to one room or is not allocated in a feasible allocation, every feasible allocation corresponds to a collection of disjoint sets. Then, the social welfare maximization problem is equivalent to finding a collection of disjoint sets that has the maximum weight sum among the candidate sets, which is an instance of the weighted (c^*+1)-set packing problem [43]. For the set packing instance, the optimal solution can be represented as a collection of disjoint sets that have positive weights. Then, we remove all the sets that have nonpositive weights from the candidate sets and obtain a new set packing instance (Algorithm 1, line 9). The maximum weight sum of disjoint sets of the new instance is the same as that of the previous instance.

We use weighted nodes to represent the weighted sets in the new instance and use edges to represent nonempty set intersections. The weight of each node equals the weight

Algorithm 1 Set-Packing Based Algorithm (*SPBA*))

```
Require: $A$, $\Re$
Ensure: the allocation $\pi$
1: $S \leftarrow \emptyset$
2: for each room $r \in \Re$ do
3:     for each feasible allocation state $AS$ of room $r$ do
4:         Create a set $s$ that includes the agents and room in $AS$
5:         Set the weight of $s$ as the social welfare of $AS$
6:         $S \leftarrow S \cup \{s\}$
7:     end for
8: end for
9: Remove all the sets with nonpositive weights from $S$
10: $V \leftarrow \emptyset$
11: $E \leftarrow \emptyset$
12: for each $s \in S$ do
13:     Create a node $n_s$ for $s$
14:     Set the weight of $n_s$ as the weight of $s$
15:     for each $n_{\bar{s}} \in V$ do
16:         if $s \cap \bar{s} \neq \emptyset$ then // $\bar{s}$ corresponds to $n_{\bar{s}}$
17:             $E \leftarrow E \cup \{(n_s, n_{\bar{s}})\}$ // Add edge $(n_s, n_{\bar{s}})$ to $E$
18:         end if
19:     end for
20:     $V \leftarrow V \cup \{n_s\}$
21: end for
22: Adopt the Berman's algorithm [42] to find an independent set $IS$ in the graph $G$=($V$, $E$)
23: for each $n_s \in IS$ do // $s$ corresponds to $n_s$
24:     Allocate each agent $i \in s$ to the room $r \in s$ in $\pi$
25: end for
26: for each unallocated agent $i$ do
27:     $\pi_i \leftarrow \emptyset$
28: end for
29: return $\pi$
```

of the corresponding set (Algorithm 1, lines 10–21). Then, we obtain an instance of the maximum-weight independent set problem [42]. In the problem, the objective is to find an independent set in which no two nodes are adjacent and the node weight sum is maximal. Because the element number of each set is less than $c^* + 2$, the obtained graph is $(c^* + 2)$-claw free [42]. Berman [42] proposed an algorithm that can find an independent set of which the node weight sum is at least $1/((c^* + 2)/2 + \varepsilon)$ of the optimal value in polynomial time. Then, based on the algorithm of Berman [42] and the corresponding relations between the nodes and sets, we can find a collection of disjoint sets of which the set weight sum is at least $1/((c^* + 2)/2 + \varepsilon)$ of the optimal value for the new set packing instance in time that is polynomial in $c^*mn^{c^*}$. If an agent belongs to one selected set, we allocate it to the room that belongs to the same set (Algorithm 1, lines 22–25). Because the weight of each set equals the utility sum of the agents who are related to the set, the utility sum of the agents who belong to the selected sets and the room owner is at least $1/((c^*+2)/2+\varepsilon)$ of the maximum social welfare. Finally, if an agent i is not included in any selected set, we set $\pi_i = \emptyset$ (Algorithm 1, lines 26–28). Then, the utility of each agent who is not included in any selected set is 0. Because the number of selected sets is at most m, the allocation process in the last phase requires $O(mn)$ time. Based on the above analyses, the proposed algorithm will terminate in time that is polynomial in $c^*mn^{c^*}$. Because c^* is a constant, the proposed algorithm is a polynomial-time algorithm for the restricted case. Moreover, the proposed algorithm can be regarded as a constant-factor approximation algorithm for the restricted case. □

The capacity of a room in a residential rental market typically does not exceed 4. Thus, Algorithm 1 is usually a $3+\varepsilon$-factor approximation algorithm in real tenant allocation.

Theorem 3: There is no polynomial-time $c^*/2$-factor approximation algorithm for the social welfare maximization problem unless P=NP, where c^* is the capacity upper-bound of each room.

Proof: Inspired by [1], we prove the hardness of approximation by a reduction from the tripartite triangle partitioning problem. Let G=($X \cup Y \cup Z$, E) be a tripartite graph with $|X| = |Y| = |Z| = n$, where X, Y and Z are mutually disjoint. The tripartite triangle partitioning problem is to determine whether the graph can be partitioned into n vertex-disjoint triangles. The problem has been demonstrated to be NP-complete [44]. Given an instance G=($X \cup Y \cup Z$, E) of the tripartite triangle partitioning problem, we construct the corresponding instance of the room allocation problem as follows:

We create an agent i for each vertex $\overline{u}_i \in X \cup Y$, and create a room r_z for each vertex $\overline{u}_z \in Z$. We set the capacity of each room to 2 (i.e., $c^* = 2$). For two vertices $\overline{u}_i \in X \cup Y$ and $\overline{u}_z \in Z$, if $(\overline{u}_i, \overline{u}_z) \in E$, set $v_i(r_z) = 1$; otherwise, set $v_i(r_z) = 1/2$. For two vertices $\overline{u}_i$, $\overline{u}_j \in X \cup Y$, if $(\overline{u}_i, \overline{u}_j) \in E$, set $v_i(j) = v_j(i) = 1/2$; otherwise, set $v_i(j) = v_j(i) = 1/4$. We set $b_i = 1/2$ for each agent i and set $p_r = 1$ for each room r. According to the settings, each agent can be allocated to any room. Because the valuation function of each agent is positive, the optimal allocation must allocate each agent to one room and can be represented as n disjoint triples. Each triple (i, j, r) denotes that agents i and j are allocated to room r. For a triple (i, j, r), the social welfare is $u_i + u_j + p_r = v_i(r) + v_i(j) + v_j(i) + v_j(r) \leq 3$, where the equality holds if and only if the corresponding vertices of i, j and r can form a triangle in the graph G. We use *OPT* to represent the optimal allocation of the room allocation instance. Then, $SW(OPT)$ is $3n$ if and only if the graph G can be partitioned into n vertex-disjoint triangles. In the room allocation instance, the capacity of each room is 2 (i.e., $c^* = 2$). Thus, if we have a polynomial-time $c^*/2$-factor approximation algorithm for the social welfare maximization problem, the algorithm can find the allocation that maximizes the social welfare in polynomial time for the constructed instance of room allocation. Then, we can determine whether the graph G can be partitioned into n vertex-disjoint triangles in polynomial time by comparing the social welfare of the solution that is output by the approximation algorithm for the constructed instance with $3n$. □

According to Theorem 3, the approximation ratio of the best approximation algorithm is larger than $c^*/2$. Thus, the approximation ratio of the algorithm in Theorem 2 (Algorithm 1) is very close to that of the best approximation algorithm.

2) GENERAL CASE

In practice, the proposed room allocation model also applies to distributed application scenarios, e.g., file replica placements [45] in a geo-distributed system. In the scenario of file replica placements, we use rooms to represent the files and agents to represent the computers. Moreover, we use agent valuations to model the distances that satisfy the triangle inequality among the computers. Each file (room) has c replicas (capacity). Each computer (agent) manages one replica. We search for a placement such that the distance (agent valuation) sum among the computers that manage the replicas of the same file is maximum. The placement enables the access cost of each file to be moderate for each computer and makes the file system reliable under network attacks.

Algorithm 2 Local-Search Based Algorithm (*LSBA*)

```
Require: $A$, $\Re$
Ensure: the allocation $\pi$
1: for each room $r \in \Re$ do
2:     Randomly allocate $c_r$ unallocated agents to $r$ in $\pi$
3: end for
4: while there are two agents $i$ and $j$ that can increase $SW(\pi)$
   by swapping their room allocations do
5:     Swap the room allocations between agents $i$ and $j$
6: end while
7: for each agent $i \in A$ do
8:     if $b_i < p_{\pi_i}/c_{\pi_i}$ or $SW(\pi)$ is larger when $\pi_i = \emptyset$ then
9:         $\pi_i \leftarrow \emptyset$
10:    end if
11: end for
12: return $\pi$
```

Although the approximation ratio of the algorithm that is described in the proof of Theorem 2 is $(c^* + 2)/2 + \varepsilon$, the enumeration process is costly and not applicable to the file replica placements [45] in a geo-distributed system, where the number of deployed replicas of each file is relatively large and the distances among the computers satisfy the triangle inequality. Motivated by the application scenario of file replica placements, we propose a faster approximation algorithm for the general case in which the capacity of each room is not bounded by a constant. The algorithm is presented as Algorithm 2. In the algorithm, first, we randomly allocate each agent to a room and obtain an allocation π (Algorithm 2, lines 1–3). Then, we adopt local search to improve the allocation until we cannot increase $SW(\pi)$ by swapping the room allocations of two agents (Algorithm 2, lines 4–6). Third, if an agent i does not satisfy the budget constraint or $SW(\pi)$ is larger when $\pi_i = \emptyset$, we set π_i to $\emptyset$ (Algorithm 2, lines 7–11). The output allocation π of Algorithm 2 is a feasible allocation. In the algorithm, allocating agent i to a room r with $b_i < p_r/c_r$ is considered equivalent to not allocating agent i. Therefore, the operations in lines 7–11 of Algorithm 2 do not decrease the social welfare. For the analysis, we define the roommate valuation $w(i, j)$ between two agents i and j as follows:

$$w(i, j) = v_i(j) + v_j(i)$$

Theorem 4: If the valuation function of each agent is nonnegative and the agent valuations satisfy the triangle inequality that for any three agents i, j and k, $w(i, k) \leq w(i, j) + w(j, k)$, Algorithm 2 is a $2C$-factor approximation algorithm for the social welfare maximization problem in which each agent has sufficient budget and C is the capacity upper-bound of each room.

Proof: According to the definition, the social welfare of an allocation only depends on agent valuations for the allocated rooms and roommates. Because each agent has sufficient budget and the valuation function of each agent is nonnegative, the optimal allocation should allocate each agent to a room. We use OPT to represent the optimal allocation in which each agent is allocated to a room and use $V_r(OPT)$ (or $V_r(\pi)$) to represent the subset of the agents that are allocated to room r in OPT (or π). Let $V_1 = V_r(OPT) \cap V_r(\pi) = \{i_1, \ldots, i_l\}$ $(0 \leq l \leq c_r)$. Without loss of generality, we assume that $V_r(OPT) = V_1 \cup \{i_{l+1}, i_{l+2}, \ldots, i_k, \ldots, i_h\}$ $(1 \leq h = c_r \leq C)$ and $V_r(\pi) = V_1 \cup \{j_{l+1}, j_{l+2}, \ldots, j_h\}$. Moreover, we assume that $i_k \in V_r(OPT)\text{-}V_1$ is allocated to room r_j^k, i.e., $i_k \in V_j^k(\pi) = \{i_k, i'_2, \ldots, i'_L\}$ in π. We use $W_r(V')$ to represent the social welfare of the allocation state in which the agents that belong to V' are allocated to room r, namely, $W_r(V') = \sum_{i \in V'}(v_i(r) + \sum_{j \in V' \setminus \{i\}} v_i(j))$.

Based on the design of Algorithm 2, when the algorithm terminates, we conclude that no two agents i and j can increase $SW(\pi)$ by swapping their room allocations. Then, we have

$$W_r(V_r(\pi) - \{j_{l+1}\} \cup \{i_k\}) + W_{r_j^k}(V_j^k(\pi) - \{i_k\} \cup \{j_{l+1}\}) \leq W_r(V_r(\pi)) + W_{r_j^k}(V_j^k(\pi))$$

Because the agent valuations are nonnegative, we have

$$\begin{aligned} v_{i_k}(r) &+ \sum_{k'=1}^{l} w(i_k, i_{k'}) + \sum_{k'=l+2}^{h} w(i_k, j_{k'}) \\ &+ \sum_{k'=l+2}^{h-1} \sum_{u=k'+1}^{h} w(j_{k'}, j_u) \\ &\leq W_r(V_r(\pi) - \{j_{l+1}\} \cup \{i_k\}) \\ &\leq W_r(V_r(\pi)) + W_{r_j^k}(V_j^k(\pi)) \end{aligned}$$

Because i_k can be any agent in $V_r(OPT)$-V_1, we have the following inequalities:

$$v_{i_{l+1}}(r) + \sum_{k'=1}^{l} w(i_{l+1}, i_{k'}) + \sum_{k'=l+2}^{h} w(i_{l+1}, j_{k'}) + \sum_{k'=l+2}^{h-1}\sum_{u=k'+1}^{h} w(j_{k'}, j_u) \le W_r(V_r(\pi)) + W_{r_j^{l+1}}(V_j^{l+1}(\pi))$$

$$\vdots$$

$$v_{i_h}(r) + \sum_{k'=1}^{l} w(i_h, i_{k'}) + \sum_{k'=l+2}^{h} w(i_h, j_{k'}) + \sum_{k'=l+2}^{h-1}\sum_{u=k'+1}^{h} w(j_{k'}, j_u) \le W_r(V_r(\pi)) + W_{r_j^h}(V_j^h(\pi))$$

For each agent $i\in V_1$, we have

$$v_i(r) + \sum_{k'=1}^{l-1}\sum_{u=k'+1}^{l} w(i_{k'}, i_u) \le W_r(V_r(\pi))$$

Based on the above inequalities, we have

$$\sum_{k'=1}^{h} v_{i_{k'}}(r) + \sum_{k'=1}^{l}\sum_{u=k'+1}^{h} w(i_{k'}, i_u) + \sum_{k'=l+1}^{h}\sum_{u=l+2}^{h} w(i_{k'}, j_u) + (h-l)\sum_{k'=l+2}^{h-1}\sum_{u=k'+1}^{h} w(j_{k'}, j_u) \le hW_r(V_r(\pi)) + \sum_{k'=l+1}^{h} W_{r_j^{k'}}(V_j^{k'}(\pi)) \tag{1}$$

Next, we obtain the following inequality via mathematical induction.

$$\sum_{k'=l+1}^{h-1}\sum_{u=k'+1}^{h} w(i_{k'}, i_u) \le \sum_{k'=l+1}^{h}\sum_{u=l+2}^{h} w(i_{k'}, j_u) + (h-l)\sum_{k'=l+2}^{h-1}\sum_{u=k'+1}^{h} w(j_{k'}, j_u) \tag{2}$$

(i) $l\le h\le l+1$: Both sides of the inequality are equal to 0. Thus, the inequality is valid.

(ii) $h=l+2$: The inequality is $w(i_{l+1}, i_{l+2}) \le w(i_{l+1}, j_{l+2}) + w(j_{l+2}, i_{l+2})$. Because the agent valuations satisfy the triangle inequality, we conclude that Inequality (2) is valid for $h=l+2$.

(iii) We assume that Inequality (2) is valid for $h=k\ge l+2$, namely, we have

$$\sum_{k'=l+1}^{k-1}\sum_{u=k'+1}^{k} w(i_{k'}, i_u) \le \sum_{k'=l+1}^{k}\sum_{u=l+2}^{k} w(i_{k'}, j_u) + (k-l)\sum_{k'=l+2}^{k-1}\sum_{u=k'+1}^{k} w(j_{k'}, j_u) \tag{3}$$

Based on the triangle inequality relationships, we have

$$w(i_{l+1}, i_{k+1}) \le w(i_{l+1}, j_{k+1}) + w(i_{k+1}, j_{k+1}),$$

and for $\forall u\in[l+2, k]$,

$$w(i_u, i_{k+1}) \le w(i_u, j_{k+1}) + w(j_{k+1}, j_u) + w(j_u, i_{k+1})$$

Based on the above inequalities, we have

$$\sum_{u=l+1}^{k} w(i_{k+1}, i_u) \le \sum_{u=l+1}^{k+1} w(i_u, j_{k+1}) + \sum_{u=l+2}^{k} w(i_{k+1}, j_u) + \sum_{u=l+2}^{k} w(j_{k+1}, j_u) \tag{4}$$

Combining Inequality (3) and Inequality (4) yields

$$\sum_{k'=l+1}^{k}\sum_{u=k'+1}^{k+1} w(i_{k'}, i_u) \le \sum_{k'=l+1}^{k+1}\sum_{u=l+2}^{k+1} w(i_{k'}, j_u) + \sum_{u=l+2}^{k} w(j_{k+1}, j_u) + (k-l)\sum_{k'=l+2}^{k-1}\sum_{u=k'+1}^{k} w(j_{k'}, j_u)$$

Thus, we conclude that

$$\sum_{k'=l+1}^{k}\sum_{u=k'+1}^{k+1} w(i_{k'}, i_u) \le \sum_{k'=l+1}^{k+1}\sum_{u=l+2}^{k+1} w(i_{k'}, j_u) + (k+1-l)\sum_{k'=l+2}^{k}\sum_{u=k'+1}^{k+1} w(j_{k'}, j_u)$$

Namely, Inequality (2) is valid for $h=k+1$. Based on the above analyses, we conclude that Inequality (2) is valid for any $h\ge l$. Then, combining Inequality (1) and Inequality (2), we have that for $\forall l\in[0, h]$,

$$W_r(V_r(OPT)) = \sum_{k'=1}^{h} v_{i_{k'}}(r) + \sum_{k'=1}^{h-1}\sum_{u=k'+1}^{h} w(i_{k'}, i_u) \le hW_r(V_r(\pi)) + \sum_{k'=l+1}^{h} W_{r_j^{k'}}(V_j^{k'}(\pi))$$

Thus, we conclude that

$$SW(OPT) = \sum_{r=1}^{m} W_r(V_r(OPT)) \le \sum_{r=1}^{m}\Big(h_r W_r(V_r(\pi)) + \sum_{k'=l_r+1}^{h_r} W_{r_j^{k'}}(V_j^{k'}(\pi))\Big)$$

For an agent $i \in V_r(OPT)$, we refer to r as its target room. For an agent $i_k \in V_r(OPT)$-V_1, we refer to r_j^k as its stay room. In the above inequality, a room can be the target room of at most C agents and the stay room of at most C agents. Thus,

$$SW(OPT) \le 2C\sum_{r=1}^{m} W_r(V_r(\pi)) = 2CSW(\pi)$$

Then, we conclude that Algorithm 2 is a $2C$-factor approximation algorithm for the restricted case. □

Algorithm 2 is not guaranteed to terminate in polynomial time. Thus, we discuss how to design a polynomial-time approximation algorithm according to Algorithm 2. Inspired by [43], we consider only executing the swaps that can increase $SW(\pi)$ by at least W'/K ($K= \alpha Cm$ and $\alpha > 1$). The value of W' is computed based on Algorithm 3 and the running time is $O(n^3)$.

Algorithm 3 Compute the Value of W'

```
Require: $A, \Re$
Ensure: $W'$
1: $W' \leftarrow -1$
2: for each $i \in A$ do
3:     $V' = < j^\rho, \rho \geq 1 > \leftarrow$ sort $j \in A \backslash \{i\}$ by decreasing $v_i(j)$
4:     for each $r \in \Re$ do //Compute the maximum $u_i+p_i$
5:         if $b_i \geq p_r/c_r$ then
6:             $W_i \leftarrow v_i(r)$
7:             $n_1 \leftarrow 1$
8:             for $\rho$=1 to $n$-1 do
9:                 if $n_1 \geq c_r$ then
10:                    break
11:                end if
12:                if $b_{j^\rho} \geq p_r/c_r$ and $v_i(j^\rho) > 0$ then
13:                    $W_i \leftarrow W_i + v_i(j^\rho)$
14:                    $n_1 \leftarrow n_1+1$
15:                end if
16:            end for
17:            if $W_i > W'$ then $W' \leftarrow W_i$
18:        end if
19:    end for
20: end for
21: return $W'$
```

We use i^* to represent the agent that has the maximum u_i+p_i in OPT. Then, we have $u_{i^*}(OPT)+p_{i^*} \geq SW(OPT)/n$. In Algorithm 3, we let W' be the maximum u_i+p_i that an agent i can obtain in a feasible allocation. As a result, $W' \geq u_{i^*}(OPT) + p_{i^*} \geq SW(OPT)/n$. Because the agent valuations are nonnegative, we have $SW(\pi) \geq 0$ and $W' \leq SW(OPT)$ after running the loop of lines 1-3 in Algorithm 2. Now, we reset the loop condition of line 4 in Algorithm 2 to "there are two agents i and j that can increase $SW(\pi)$ by at least W'/K via swapping their room allocations". Then, each iteration increases $SW(\pi)$ by at least W'/K. Because $W' \geq SW(OPT)/n$ and $0 \leq SW(\pi) \leq SW(OPT)$, the modified algorithm executes at most Kn iterations. Because each iteration requires $O(n^3)$ time, the running time of the modified algorithm is $O(Kn^4)$.

Based on the new loop condition, when the modified algorithm terminates, we conclude that no two agents i and j can increase $SW(\pi)$ by at least W'/K via swapping their room allocations. Then, for an agent i_k that is allocated to room r in OPT but is allocated to room r_j^k in π, we have

$$W_r(V_r(\pi)-\{j_{l+1}\} \cup \{i_k\})+W_{r_j^k}(V_j^k(\pi)-\{i_k\} \cup \{j_{l+1}\}) < W_r(V_r(\pi)) + W_{r_j^k}(V_j^k(\pi)) + \frac{W'}{K}$$

Then, based on the analyses that are similar to the proof of Theorem 4 and $K= \alpha Cm$ ($\alpha > 1$), we conclude that

$$\begin{aligned} SW(OPT) &= \sum_{r=1}^{m} W_r(V_r(OPT)) \\ &< \sum_{r=1}^{m}(h_r W_r(V_r(\pi)) + \sum_{k'=l_r+1}^{h_r} W_{r_j^{k'}}(V_j^{k'}(\pi)) \\ &\quad + \frac{h_r W'}{K}) \\ &\leq 2C\sum_{r=1}^{m} W_r(V_r(\pi)) + \frac{CmW'}{K} \\ &= 2CSW(\pi) + \frac{W'}{\alpha} \end{aligned}$$

Because $W' \leq SW(OPT)$, we have $SW(OPT) \leq 2C(1+1/(\alpha\text{-}1))SW(\pi)$. Let $\alpha = 2C/\varepsilon + 1$. Then, we have the following conclusion:

Corollary 1: The modified algorithm is a $2C+\varepsilon$-factor approximation algorithm (with $\varepsilon > 0$) for the restricted case of the social welfare maximization problem in Theorem 4 and the time complexity is $O(mn^6/\varepsilon)$.

Based on the proof of Theorem 3, we know that there is no polynomial-time $C/2$-factor approximation algorithm for the social welfare maximization problem, even if the valuation function of each agent is nonnegative and the agent valuations satisfy the triangle inequality, unless P=NP. Then, the approximation ratio of the modified algorithm does not exceed $4+\varepsilon$ ($\varepsilon > 0$) times the best approximation ratio. Because the application scenario of file replica placements in geo-distributed systems satisfies the constraint conditions in Theorem 4, the approximation result of the modified algorithm applies to the application scenario of file replica placements in geo-distributed systems.

D. DUAL PROBLEM OF SOCIAL WELFARE MAXIMIZATION

If we use vertices to represent the agents and rooms and use weighted edges to represent the valuation relationships among the agents and rooms, we obtain an undirected graph with real-number edge weights. Then, finding the allocation that maximizes the social welfare can be regarded as finding the minimum multiway cut under the capacity constraints of the room vertices. Thus, when each agent has sufficient budget and is required to be allocated to one room, finding the allocation that maximizes the social welfare can also be modeled as the following problem, which is the dual problem of social welfare maximization:

Definition 3 (Multiway Cut With Size Constraints): Given an undirected graph $G=(V, E)$ ($|V| = n$) with real-number edge weights, and a set $T = \{t_1, t_2, \ldots, t_m\} \subseteq V$ of m capacitated terminals, the objective is to find a minimum weight edge subset whose removal can partition V into m disjoint subsets $V_1, V_2, \ldots, V_m$ (i.e., $\cup_{1\leq i\leq m} V_i = V$) with $t_i \in V_i$ and $|V_i| = c_i$ ($c_i \geq 1$). The weight of an edge subset is equal to the weight sum of the edges that belong to the subset. The capacity of terminal t_i is c_i-1.

We refer to an edge subset whose removal can partition V into feasible disjoint vertex subsets as a feasible multiway cut. The application scenarios of the proposed problem include assigning tasks to multiple capacitated computing terminals [46] while minimizing the communication cost.

The proposed problem is closely related to the classical multiway cut problem [47]–[54], where the objective is to find a minimum-weight edge subset whose removal can disconnect the k terminals of an undirected graph from each other. The main difference between the proposed problem and the multiway cut problem is that the proposed problem has definite constraints on the cardinality of each disjoint subset, whereas the multiway cut problem does not. It is well-known that the multiway cut problem has a 2–2/k-factor approximation algorithm [51]. However, we show that the multiway cut problem with size constraints is hard to approximate.

Theorem 5: For any constant $\rho > 0$, there is no polynomial-time ρ-factor approximation algorithm for the multiway cut problem with size constraints, even if the edge weights are nonnegative, unless P=NP.

Proof: Inspired by [55], we demonstrate the hardness of approximation by a reduction from the 3-partition problem. In the 3-partition problem, a multiset $S=\{I_1, I_2, \ldots, I_{3k}\}$ of $3k$ positive integers is specified, each of which is strictly between $B/4$ and $B/2$, namely, for any $I_j \in S$, $B/4 < I_j < B/2$ and the sum of the integers in S is kB. We are required to determine whether S can be partitioned into k triples $S_1, S_2, \ldots, S_k$ such that the sum of the integers in each triple is equal to B. The problem is strongly NP-complete [56]. This means that the problem remains NP-complete when all of its numerical parameters, including the integers in S, the k and B, are bounded by a polynomial in the length of the input.

Given an instance $<S, k, B>$ ($k>1$) of the 3-partition problem with polynomially bounded parameters, we create a circle that contains I_j nonterminal vertices for each integer $I_j \in S$ and set the weight of each edge in the circle as ρ. We refer to such circles as integer circles. After that, for each pair of non-terminal vertices that do not connect with each other, we add an edge between them and set the edge weight as 0. We refer to the edges that connect two nonterminal vertices as nonterminal edges. Finally, we create k terminal vertices and for each terminal vertex t_i, we set the corresponding c_i as B+1. We add an edge between each pair of terminal vertices and set the edge weight as 0. We refer to the edges that connect two terminal vertices as terminal edges. For each terminal vertex and each nonterminal vertex, we add an edge between them and set the edge weight as $1/(k(k\text{-}1)B)$. We refer to the edges of this type as mixed edges. Then, we obtain an undirected graph G. Because all the numerical parameters of the instance are polynomially bounded, the construction process of the graph G requires only polynomial time.

Suppose that we have a ρ-factor approximation algorithm for the multiway cut problem with size constraints. If the instance $<S, k, B>$ has a feasible partitioning $S_1, S_2, \ldots, S_k$, we allocate the circle vertices that correspond to the integers in S_i and terminal vertex t_i to the same subset for each triple S_i ($1\leq i\leq k$). Then, we obtain a feasible cut that does not include any nonterminal edge with positive weight. Because each nonterminal vertex has k mixed edges and is allocated to a vertex subset that has only one terminal vertex, it has k-1 mixed edges in a feasible cut. Because there are kB nonterminal vertices, a feasible cut has exactly $k(k\text{-}1)B$ mixed edges. Since the weights of terminal edges and mixed edges are 0 and $1/(k(k\text{-}1)B)$ respectively, the weight of the minimum-weight feasible multiway cut is 1. Then, the ρ-factor approximation algorithm will yield a feasible multiway cut with a weight of at most ρ.

Because $B/4 < I_j < B/2$ for any $I_j \in S$, an integer subset has exactly three integers if the integer sum of the subset is B. If the instance $<S, k, B>$ does not have a feasible partitioning, at least one integer circle is partitioned into multiple parts while dividing the vertex set. Then, any feasible multiway cut on the graph G includes at least one nonterminal edge with weight ρ and exactly $k(k\text{-}1)B$ mixed edges. Thus, the weight of the minimum-weight feasible multiway cut is at least $1+\rho$. Namely, the ρ-factor approximation algorithm will yield a feasible multiway cut with a weight of at least $1+\rho$ in this case. Then, we can determine whether a specified instance of the 3-partition problem with polynomially bounded parameters has a feasible partitioning in polynomial time by comparing the weight of the solution that is output by the approximation algorithm for the constructed graph with ρ. □

V. SOCIAL WELFARE MAXIMIZATION WITH THE MAXIMUM TRADE VOLUME CONSTRAINT

When there are insufficiently many low-cost rooms, we cannot always ensure that every agent is allocated to a room. In this scenario, the room owner or the rental intermediary typically hopes to maximize the trade volume, which equals the sum of the agents' payments. This is because the profit of the room owner or the rental intermediary is positively related to the trade volume. Although the trade volume is included in the calculation of social welfare, the allocation that maximizes the social welfare does not ensure the maximization of the trade volume (see Example 1). It is also desired to find an allocation with larger social welfare for the room owner or the rental intermediary. This is because an allocation with larger social welfare typically has a higher level of tenant satisfaction and contributes to the establishment of long-term leasehold relationships between the room owner and tenants. Therefore, we investigate how to find the allocation that maximizes the social welfare among the allocations that have the maximum trade volume. In this section, first, we discuss how to find an allocation that maximize the trade volume. Second, we design a heuristic algorithm for the social welfare

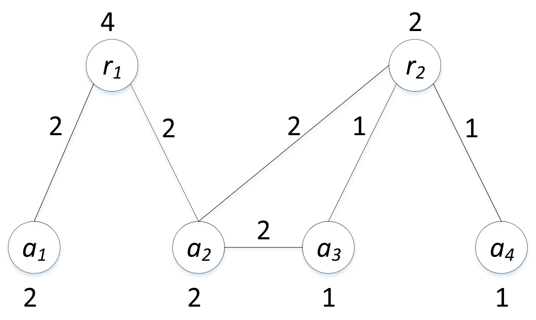


FIGURE 2. An example of room allocation.

maximization problem with the maximum trade volume constraint.

Example 1: Let $A= \{a_1, a_2, a_3, a_4\}$ and $\Re =\{r_1, r_2\}$ with $b_{a_1} = b_{a_2} = 2$, $b_{a_3} = b_{a_4} = 1$, $p_{r_1} = 4$ and $p_{r_2} = 2$. The capacity of each room is 2. The valuation relationships are described by the weighted edges of the graph in Fig. 2. If there is no edge between two vertices, the valuations between them are 0. The weight of edge (a_2, a_3) denotes $v_{a_2}(a_3)+ v_{a_3}(a_2)=2$. The allocation that corresponds to the maximum trade volume is (r_1, r_1, r_2, r_2), which allocates a_1, a_2 to r_1 and a_3, a_4 to r_2. The social welfare and trade volume are both 6. However, the allocation that maximizes the social welfare is $(r_1, r_2, r_2, \emptyset)$, which allocates a_1 to r_1, a_2, a_3 to r_2 and does not allocate a_4. The social welfare is 7, but the trade volume is $4<6$.

Theorem 6: The allocation that maximizes the trade volume can be found in $O(n^3)$ time.

Proof: Given an instance $<A, \Re>$ of the room allocation problem, we create one agent vertex $\overline{u}_i$ for each agent $i\in A$ and create c_r room vertices $\overline{u}_{r_1}, \overline{u}_{r_2}, \ldots, \overline{u}_{r_{c_r}}$ for each room $r\in\Re$. After that, we add an edge between each agent vertex and room vertex. If $b_i \geq p_r/c_r$, we set the weight of each edge between agent vertex $\overline{u}_i$ and each room vertex of room r to p_r/c_r; otherwise, we set the weight to 0. Then, we obtain a bipartite graph $G=(V_A \cup V_\Re, E)$ with $|V_A| = |V_\Re| =n$, where V_A is the set of agent vertices and $V_\Re$ is the set of room vertices.

For any feasible allocation $\pi =(\pi_1, \pi_2, \ldots, \pi_n)$ with maximum trade volume T, we can construct a perfect matching whose weight sum is T as follows: (i) for each agent i with $\pi_i \neq \emptyset$, match agent vertex $\overline{u}_i$ with one nonmatched room vertex of room π_i and (ii) for each agent j with $\pi_j = \emptyset$, match agent vertex $\overline{u}_j$ with one nonmatched room vertex. According to the weight setting strategy, the weight sum of the matching edges in step (i) is equal to T. Because π is a feasible allocation that maximizes the trade volume, the budget of any agent j with $\pi_j = \emptyset$ must be smaller than the bed rent of any room that has remaining space in step (ii); otherwise, we can find a feasible allocation with a trade volume that exceeds T. Then, the weight sum of the matching edges in step (ii) is equal to 0. Therefore, the weight sum of the constructed matching is T. Based on the above analyses, we conclude that the weight sum of the maximum-weight perfect matching of the graph G is not smaller than T.

For any perfect matching M of the graph G that has weight sum T', we can construct a feasible allocation that has trade volume T' as follows: for each edge $(\overline{u}_i, \overline{u}_{r_k})\in M$, if the weight is equal to p_r/c_r, allocate agent i to room r; otherwise, do not allocate agent i to any room. According to the weight setting strategy, we conclude that the trade volume of the constructed allocation is T'. Based on this property, we conclude that the weight sum of the maximum-weight perfect matching of the graph G is not larger than T. Combining the previous analyses, we conclude that the weight sum of the maximum-weight perfect matching of the graph G equals T. Then, we find the maximum-weight prefect matching of the graph G. After that, via the aforementioned construction method, we can find a feasible allocation that has trade volume T.

The constructed process of the graph G requires $O(n^2)$ time. The maximum weight perfect matching can be found via the Hungarian method [57] in $O(n^3)$ time. Based on the maximum-weight perfect matching, we can construct a feasible allocation that has trade volume T in $O(n)$ time. Therefore, we conclude that the allocation that maximizes the trade volume can be found in $O(n^3)$ time. □

Theorem 7: It is NP-hard to find the allocation that maximizes the social welfare among the allocations that have the maximum trade volume.

The proof is similar to that of Theorem 1. Next, we investigate the inapproximability of the problem.

Theorem 8: For any constant $\rho >0$, there is no polynomial-time ρ-factor approximation algorithm for the social welfare maximization problem with the maximum trade volume constraint unless P=NP.

Proof: Let $<G=(V, E), k>$ $(k>1)$ be an instance of the clique problem, where the graph G has n vertices. First, we create n-k+1 rooms, where the capacity of room r_1 is k and the capacity of any other room is 1. Let $\Re$ denote the room set. Second, we create an agent i for each vertex $u_i \in V$. We set $v_i(j) = v_j(i) = 1/2$ if $(u_i, u_j)\in E$; otherwise, we set $v_i(j) = v_j(i) = -k^2$-$n/2$. For each agent i and each room $r\in \Re$, we set $v_i(r) = 1/2$. Moreover, we set $b_i = 1$ for each agent i and set $p_r = 1/2$ for each room r. Then, an allocation that maximizes the trade volume must allocate each agent to one room. If the graph G has a set of k mutually adjacent vertices, the maximum social welfare of an allocation that has the maximum trade volume is $k(k$-$1)/2+n/2>0$. Conversely, if the graph G does not have a set of k mutually adjacent vertices, at least two agents i and j with $v_i(j) = v_j(i) = -k^2$-$n/2$ are allocated to room r_1. Thus, the maximum social welfare of an allocation that has the maximum trade volume must be smaller than 0. Then, if we have a polynomial-time ρ-factor approximation algorithm (with $\rho>0$) for the social welfare maximization problem with the maximum trade volume constraint, we can determine whether G has a set of k mutually adjacent vertices in polynomial time by comparing the social welfare of the solution that is output by the approximation algorithm for the constructed instance with 0. □

Corollary 2: If each agent has sufficient budget and is required to be allocated to one room, there is no polynomial-time ρ-factor approximation algorithm (with $\rho > 0$) for the social welfare maximization problem unless P=NP.

According to Theorem 8, it is impossible to design a polynomial-time approximation algorithm that has a positive approximation ratio for the general case of the social welfare maximization problem with the maximum trade volume constraint unless P=NP. Therefore, we study the approximation algorithm for a restricted case that is often encountered in real applications.

In practice, each room for rent has a use value and can satisfy the residential demand of a tenant agent. Therefore, we assume that for each agent i and each room r, $v_i(r) > 0$ (termed as hypothesis **H1**). In an area of a city, the bed rent of a room is typically positively related to the comfortable and convenient levels of the room. Based on this observation, we assume that for each agent i, $v_i(r) \geq v_i(\bar{r})$ if $p_r/c_r > p_{\bar{r}}/c_{\bar{r}}$ (termed as hypothesis **H2**). In addition, we assume that the mutual valuations among the agents are nonnegative [1], [2] and the capacity of each room does not exceed a constant $c^* > 0$ (termed as hypothesis **H3**). Aiming at this restricted case, we propose a $2c^*(1+1/\beta)$-factor approximation algorithm that is presented as Algorithm 4, where β is the largest positive number that can ensure $v_i(r) \geq \beta c^* v_i(j)$ for any agent $i, j \in A$ and any room $r \in \Re$. According to the proof of Theorem 3, the restricted case of the social welfare maximization problem with the maximum trade volume constraint is still NP-hard.

Algorithm 4 Local-Search Based Algorithm for Social Welfare Maximization Problem With the Maximum Trade Volume Constraint (*LSBAMC*)

```
Require: $A$, $\Re$
Ensure: the allocation $\pi$
 1: Find an allocation $\pi$ that maximizes the trade volume
    using the algorithm in the proof of Theorem 6
 2: for each agent $i \in A$ do
 3:     if $\pi_i = \emptyset$ then
 4:         Allocate it to one room with residual space
 5:     end if
 6: end for
 7: while there is an allocation swap $(i, j)$ in $\pi$ that can
    increase $SW(\pi)$ without decreasing the trade volume do
 8:     Swap the room allocations between agents $i$ and $j$
 9: end while
10: for each agent $i \in A$ do
11:     if $b_i < p_{\pi_i}/c_{\pi_i}$ then
12:         $\pi_i \leftarrow \emptyset$
13:     end if
14: end for
15: return $\pi$
```

In a real rental market, the tenancy periods of tenant agents are typically indeterminate and diverse, leading to dynamic roommate relationships. By contrast, the room leasehold relationship is relatively stable for a tenant agent and a tenant agent can determine the tenancy period. Therefore, a tenant agent typically assigns higher valuations for rooms with respect to other tenant agents. Since the capacity of a room in a residential rental market is typically small, β is usually a relatively large positive number.

Theorem 9: Algorithm 4 is a $2c^*(1+1/\beta)$-factor approximation algorithm for the restricted case of the social welfare maximization problem with the maximum trade volume constraint in which hypotheses H1, H2 and H3 are satisfied.

Proof: In Algorithm 4, first, we find an allocation π that maximizes the trade volume using the algorithm in the proof of Theorem 6 (Algorithm 4, line 1). Second, if an agent i has not been allocated to a room, we randomly allocate it to a room with residual space (Algorithm 4, lines 2–6). Third, we adopt local search to improve the allocation until we cannot increase $SW(\pi)$ by swapping the room allocations of two agents while ensuring the trade volume does not decrease (Algorithm 4, lines 7-9). Finally, if an agent i does not satisfy the budget constraint, we set π_i to $\emptyset$ (Algorithm 4, lines 10-14). The final allocation π of Algorithm 4 has the maximum trade volume. In the algorithm, allocating agent i to a room r with $b_i < p_r/c_r$ is considered equivalent to not allocating agent i. Then, the operations in lines 2-6 and lines 10-14 of Algorithm 4 do not change the social welfare and the trade volume. Moreover, any feasible allocation that maximizes the trade volume can be represented as m disjoint sets, which include all the agents and rooms. Each set $(i, i_2, \ldots, i_{c_r}, r)$ denotes that agents $i, i_2, \ldots, i_{c_r}$ are allocated to room r.

Let OPT be the m disjoint sets that correspond to the optimal allocation and π' be the m disjoint sets that correspond to allocation π. Without loss of generality, we assume that agents $i, i_2, \ldots, i_{c_r}$ are allocated to room r in OPT. In π', agents $i, j_2, \ldots, j_{c_{\bar{r}}}$ are allocated to room $\bar{r}$ and agents $i', i'_2, \ldots, i'_{c_r}$ are allocated to room r. We use $i_{-\bar{r}}$ to represent the set of agents that are allocated to room $\bar{r}$ and do not include agent i in π'. If $b_i < p_{\bar{r}}/c_{\bar{r}}$, the social welfare $SW(i, i_{-\bar{r}}, \bar{r})$ of set $(i, i_{-\bar{r}}, \bar{r})$ is equal to that of $(\emptyset, i_{-\bar{r}}, \bar{r})$, i.e., $u_i + p_i = 0$. According to the previous assumption, $SW(i, i_{-\bar{r}}, \bar{r}) \geq 0$ for any set $(i, i_{-\bar{r}}, \bar{r})$ and $SW(\pi) \geq 0$ at any time. We assume that agent i' has the maximum budget among the agents in room r. If $b_i < p_r/c_r$, we have $u_i + p_i = 0$ in OPT, which does not exceed $SW(i, i_{-\bar{r}}, \bar{r})$. If $b_i \geq p_r/c_r$, we have $u_i + p_i > 0$ in OPT. If agent i is allocated to r in π' (i.e., $\bar{r} = r$) and $b_i \geq p_r/c_r$, $v_i(r) \leq SW(i, i_{-\bar{r}}, \bar{r})$) because the valuation function of each agent is nonnegative. Next, we mainly analyze the scenario in which $b_i \geq p_r/c_r$ and $\bar{r} \neq r$.

Case 1: $b_{i'} < p_{\bar{r}}/c_{\bar{r}}$ and $b_{i'} \geq p_r/c_r$: If $b_i < p_{\bar{r}}/c_{\bar{r}}$, swapping the room allocations between agent i and i' does not decrease the trade volume. However, Algorithm 4 does not swap the room allocations between agents i and i' prior to its termination. We conclude that $SW(i, i'_{-r}, r) + SW(i', i_{-\bar{r}}, \bar{r}) \leq SW(i', i'_{-r}, r) + SW(i, i_{-\bar{r}}, \bar{r})$. Because the valuation function of each agent is nonnegative, we have

$v_i(r) \leq SW(i, i'_{-r}, r) \leq SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})$. If $b_i \geq p_{\bar{r}}/c_{\bar{r}}$, then $b_i \geq p_{\bar{r}}/c_{\bar{r}} > b_{i'} \geq p_r/c_r$. Because the room valuation of each agent i is positively related to the bed rent of a room and the valuation function of each agent is nonnegative, we have $v_i(r) \leq v_i(\bar{r}) \leq SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})$.

Case 2: $b_{i'} < p_{\bar{r}}/c_{\bar{r}}$ and $b_{i'} < p_r/c_r$: We must have $b_i \geq p_{\bar{r}}/c_{\bar{r}}$ and $p_{\bar{r}}/c_{\bar{r}} \geq p_r/c_r$. This is because if $b_i < p_{\bar{r}}/c_{\bar{r}}$ or $p_{\bar{r}}/c_{\bar{r}} < p_r/c_r$, we can increase the trade volume by swapping the room allocations between agents i and i', which is in contradiction with that allocation π has the maximum trade volume. Because the room valuation of each agent i is positively related to the bed rent of a room and the valuation function of each agent is nonnegative, we have $v_i(r) \leq v_i(\bar{r}) \leq SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})$ if $p_{\bar{r}}/c_{\bar{r}} > p_r/c_r$. If $p_{\bar{r}}/c_{\bar{r}} = p_r/c_r$, swapping the room allocations between agent i and i' does not decrease the trade volume. Because Algorithm 4 does not swap the room allocations between agents i and i' prior to its termination and the valuation function of each agent is nonnegative, we conclude that $v_i(r) \leq SW(i, i'_{-r}, r) \leq SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})$.

Case 3: $b_{i'} \geq p_{\bar{r}}/c_{\bar{r}}$: We must have $b_i \geq p_{\bar{r}}/c_{\bar{r}}$ and $b_{i'} \geq p_r/c_r$. This is because if $b_i < p_{\bar{r}}/c_{\bar{r}}$ or $b_{i'} < p_r/c_r$, we can increase the trade volume by swapping the room allocations between agents i and i', which is in contradiction with that allocation π has the maximum trade volume. Then, swapping the room allocations between agent i and i' does not decrease the trade volume. Because Algorithm 4 does not swap the room allocations between agents i and i' prior to its termination and the valuation function of each agent is nonnegative, we conclude that $v_i(r) \leq SW(i, i'_{-r}, r) \leq SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})$.

In *OPT*, if $b_i \geq p_r/c_r$, we have $u_i+p_i \leq v_i(r)+\sum_{k=2}^{c_r} v_i(i_k) \leq (1+c_r/\beta c^*)\cdot v_i(r) \leq (1+c_r/\beta c^*)\cdot(SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r}))$. If $b_i < p_r/c_r$, we have $u_i + p_i = 0 \leq (1+c_r/\beta c^*)\cdot(SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r}))$. Because the capacity of each room does not exceed c^*, we have

$$\begin{aligned} SW(OPT) &= \sum_{i \in A}(u_i + p_i) \\ &\leq (1 + 1/\beta)\sum_{i \in A}(SW(i', i'_{-r}, r) + SW(i, i_{-\bar{r}}, \bar{r})) \end{aligned}$$

We refer to room r as agent i's target room and room $\bar{r}$ as agent i's stay room. In the inequality relationships, a room will be the target room of at most c^* agents and the stay room of at most c^* agents. Then, we have

$$\begin{aligned} SW(OPT) &\leq (1 + 1/\beta)\sum_{i \in A}(SW(i', i'_{-r}, r) + SW(i, i_{-\bar{r}}, \bar{r})) \\ &\leq 2c^*(1 + 1/\beta)\sum_{r \in \Re} SW(i', i'_{-r}, r) \\ &= 2c^*(1 + 1/\beta)SW(\pi) \end{aligned}$$

Therefore, we conclude that Algorithm 4 is a $2c^*(1+1/\beta)$-factor approximation algorithm for the restricted case of the social welfare maximization problem with the maximum trade volume constraint. □

Algorithm 4 is not guaranteed to terminate in polynomial time. Thus, we discuss how to obtain a polynomial-time approximation algorithm based on Algorithm 4. Inspired by [43], we discretize the social welfare values of the feasible allocations. First, for each possible set (i, i_{-r}, r), we determine whether it can be included in a disjoint-set collection that is equivalent to a feasible allocation with maximum trade volume T. The method is invoking the algorithm that is described in the proof of Theorem 6 to compute the maximum trade volume T' for the instance $< A \setminus \{i, i_{-r}\}, \Re \setminus \{r\} >$ and determining whether the trade volume of set (i, i_{-r}, r) is equal to T-T'. If the trade volume of set (i, i_{-r}, r) is equal to T-T', we refer to it as a "qualified set". Let $(i^*, i^*_{-r^*}, r^*)$ be the set that has the maximum social welfare among all the qualified sets. Because the trade volume of *OPT* is T, any set in *OPT* is a qualified set. Let $(i', i'_{-\bar{r}}, \bar{r})$ be the set that has the maximum social welfare among the sets in *OPT*. Then, $SW(i^*, i^*_{-r^*}, r^*) \geq SW(i', i'_{-\bar{r}}, \bar{r}) \geq SW(OPT)/m$. Because the valuation function of each agent is nonnegative, $SW(OPT) \geq SW(i^*, i^*_{-r^*}, r^*)$. According to Theorem 6, the set $(i^*, i^*_{-r^*}, r^*)$ can be found in $O(mn^{c^*+3})$ time.

After that, we truncate the social welfare of any feasible allocation to an integer multiple of $SW(i^*, i^*_{-r^*}, r^*)/L$ ($L=2K(1+1/\beta)\cdot n$ and $K>1$). Let $ISW(\pi)= \lfloor L\cdot SW(\pi)/SW(i^*, i^*_{-r^*}, r^*)\rfloor$ be the modified social welfare of allocation π. We replace the $SW(\pi)$ in Algorithm 4 with $ISW(\pi)$. Then, each iteration increases $ISW(\pi)$ by at least 1. Because $SW(i^*, i^*_{-r^*}, r^*) \geq SW(OPT)/m$ and $0 \leq SW(\pi) \leq SW(OPT)$, the modified algorithm will execute at most Lm iterations. Because each iteration requires $O(n^2)$ time, the running of the while loop in the modified algorithm requires $O(Lmn^2)$ time. Thus, running the modified algorithm requires $O(Lmn^{c^*+3})$ time. Because c^* is a constant, the modified algorithm is a polynomial-time algorithm.

If the modified algorithm does not execute a swap (i, i') that does not decrease the trade volume prior to its termination, we have that $v_i(r) \leq SW(i, i'_{-r}, r)+SW(i', i_{-\bar{r}}, \bar{r}) < SW(i', i'_{-r}, r)+SW(i, i_{-\bar{r}}, \bar{r})+SW(i^*, i^*_{-r^*}, r^*)/L$. Based on the analyses that are similar to the proof of Theorem 9, we have $SW(OPT) \leq 2c^*(1+1/\beta)SW(\pi)+2(1+1/\beta)\cdot n\cdot SW(i^*, i^*_{-r^*}, r^*)/L$. Because $L=2K(1+1/\beta)\cdot n$ ($K>1$) and $SW(i^*, i^*_{-r^*}, r^*) \leq SW(OPT)$, we have $SW(OPT) \leq 2c^*(1+1/\beta)(1+1/(K-1))SW(\pi)$. Let $K=2c^*(1+1/\beta)/\varepsilon + 1$ ($0< \varepsilon <1$). Then, we have the following conclusion:

Corollary 3: The modified algorithm is a polynomial-time $2c^*(1+1/\beta)+\varepsilon$-factor approximation algorithm for the restricted case.

When the capacity of each room does not exceed 2 (i.e., $c^* = 2$) and the agents prefer rooms, i.e., for any agent $i, j \in A$ and any room $r \in \Re$, $v_i(r) \geq v_i(j)$ with $\beta = 1/2$, the modified algorithm is a $12+\varepsilon$-factor approximation algorithm for the restricted case.

VI. ROOMMATE STABILITY AND ROOM ENVY-FREENESS

In this section, we investigate how to find a 2-person weakly stable or room envy-free allocation with a social welfare guarantee. The proposed notion of stability is a variant of "exchange stability" in the traditional stable roommate problem [11], [12], in which a matching is exchange stable if no pair of agents who live in different rooms desire to switch their rooms. Based on the "exchange stability", Chan *et al.* [1] defined an allocation as 2-person stable if no pair of agents (i, j) who live in different rooms can increase both their utilities via swapping. They showed that determining whether a specified room allocation instance admits a 2-person stable allocation is NP-hard. They also proposed 4-person stability: an allocation is 4-person stable if no pair of agents (i, j) who live in different rooms can increase the utilities of all 4 agents in the two rooms via swapping. They assumed that the capacity of each room is two. Comparing with 4-person stability, the proposed 2-person weak stability is stronger. Namely, a 2-person weakly stable allocation must be a 4-person stable allocation, but a 4-person stable allocation is typically not a 2-person weakly stable allocation.

Inspired by [1], we propose a room allocation algorithm that is presented as Algorithm 5. In the algorithm, first, we use Algorithm 1 to find an allocation (Algorithm 5, line 1). Second, if there is a 2-person weakly stable blocking pair (i, j) in the current allocation, we swap the room allocations between agents i and j. The check and adjustment are repeated until there is not a 2-person weakly stable blocking pair in the allocation (Algorithm 5, lines 2–4). Finally, if an agent i does not satisfy the budget constraint, we set π_i to $\emptyset$ (Algorithm 5, lines 5–9).

Algorithm 5 Compute the Roommate Stable Allocation

```
Require: $A$, $\Re$
Ensure: the roommate stable allocation $\pi'$
1: Find an allocation $\pi'$ based on Algorithm 1
2: while there is a 2-person weakly stable blocking pair $(i, j)$
   in current allocation $\pi'$ do
3:     Swap the room allocations between agents $i$ and $j$
4: end while
5: for each agent $i \in A$ do
6:     if $b_i < p_{\pi_i}/c_{\pi_i}$ then
7:         $\pi_i \leftarrow \emptyset$
8:     end if
9: end for
10: return $\pi'$
```

Theorem 10: If the capacity of each room does not exceed a constant $c^* > 0$, Algorithm 5 can find a 2-person weakly stable allocation with a social welfare that is at least $1/((c^*+2)/2 + \varepsilon)$ of the optimal value in polynomial time.

Proof: According to Theorem 2, allocation π' has a social welfare that is at least $1/((c^* + 2)/2 + \varepsilon)$ of the optimal value after running line 1 of Algorithm 5. According to the definition of a 2-person weakly stable blocking pair, the social welfare does not decrease in the allocation adjustment process. Therefore, the social welfare of the final allocation of Algorithm 5 is at least $1/((c^* + 2)/2 + \varepsilon)$ of the optimal value. In the algorithm, allocating agent i to a room r with $b_i < p_r/c_r$ is considered equivalent to not allocating agent i. Thus, the operations in lines 5-9 of Algorithm 5 do not decrease the social welfare. Because the capacity of each room does not exceed c^*, each agent has at most $c^*(m+1)\cdot \max_{1\le c_r \le c^*} \binom{n-1}{c_r-1} < c^*(m+1)n^{c^*-1}$ possible allocation states. When we swap the room allocations of a 2-person weakly stable blocking pair (i, j), the utilities of agents i and j increase and the other agents' utilities do not decrease. Therefore, the utility of each agent $i \in A$ increases at most $c^*(m+1)n^{c^*-1}$ times. Moreover, the allocation adjustment process requires at most $c^*(m+1)n^{c^*}$ iterations. Because each iteration requires $O(n^2)$ time, the allocation adjustment process requires $O(c^*(m+1)n^{c^*+2})$ time. Since the running of Algorithm 1 requires time that is polynomial in $c^*mn^{c^*}$, we conclude that Algorithm 5 will terminate in time that is polynomial in $c^*mn^{c^*}$. Because c^* is a constant, the proposed algorithm is a polynomial-time algorithm for the restricted case. Moreover, the proposed algorithm can be regarded as having a constant approximation ratio with respect to the maximum social welfare for the restricted case. $\square$

Theorem 11: If the capacity of each room does not exceed a constant $c^* > 0$, we can find a 2-person weakly stable allocation that has the maximum trade volume in polynomial time.

Proof: The proposed algorithm is similar to Algorithm 5, except that we initially find an allocation that maximizes the trade volume via the algorithm in the proof of Theorem 6 in $O(n^3)$ time. According to the definition of a 2-person weakly stable blocking pair, swapping the room allocations between the blocking pair agents does not decrease the utility of any agent, including the room owner. Thus, swapping the room allocations between the blocking pair agents does not decrease the trade volume. Based on the analyses in the proof of Theorem 10, the algorithm will terminate in $O(c^*(m+1)n^{c^*+2})$ time. Thus, the algorithm will output a 2-person weakly stable allocation that maximizes the trade volume in $O(c^*(m+1)n^{c^*+2})$ time. Because c^* is a constant, the proposed algorithm is a polynomial-time algorithm for the restricted case. $\square$

Envy-freeness is a stronger solution concept than stability. A person envy-free allocation in which each agent would not like to switch rooms with any other agent must be a 2-person stable allocation. Finding a person envy-free allocation [1] is NP-hard. Chan *et al.* [1] introduced a weaker concept – room envy-freeness. They showed that if each room can contain at most two agents and the room prices are adjustable, a room envy-free allocation can be found in polynomial time. However, in real rental market, the room prices are typically determined by the room owner in advance, according to the current market situation. In other words, we typically need to face the room allocation scenario in which the room prices are specified in advance. Therefore, we investigate the computational complexity of determining whether an instance with specified room prices admits a room envy-free allocation.

Theorem 12: Determining whether an instance with specified room prices admits a room envy-free allocation is NP-hard.

Proof: Given an instance $G=(X\cup Y\cup Z, E)$ of the tripartite triangle partitioning problem with $|X| = |Y| = |Z| = n$, we create an agent i for each vertex $\bar{u}_i \in X\cup Y$ and create a vertex room r_z for each vertex $\bar{u}_z \in Z$. For two vertices $\bar{u}_i \in X\cup Y$ and $\bar{u}_z \in Z$, if $(\bar{u}_i, \bar{u}_z)\in E$, set $v_i(r_z) = c>0$; otherwise, set $v_i(r_z) = c/2$. For two vertices $\bar{u}_i, \bar{u}_j \in X\cup Y$, if $(\bar{u}_i, \bar{u}_j)\in E$, set $v_i(j) = v_j(i) = c/2$; otherwise, set $v_i(j) = v_j(i) = 0$. We set $b_i = c/2$ for each agent i and set $p_r = c$ for each room r. We use A to represent the agent set and $\Re$ to represent the vertex room set. After that, for any two vertices $\bar{u}_i, \bar{u}_j \in X\cup Y$, if $(\bar{u}_i, \bar{u}_j)\notin E$, we create a room r_{ij} with price $c/4$ and set $v_i(r_{ij}) = v_j(r_{ij}) = c$. For any agent $i' \in A\backslash\{i, j\}$, set $v_{i'}(r_{ij}) = 0$. We refer to these rooms as edge rooms. In addition, we create a common room r_c with price $c/4$. For each agent $i\in A$, set $v_i(r_c) = 5c/8$. Let $\Re_1$ ($|\Re_1| = m$) be the room set that includes all the edge rooms and the common room. Finally, we create $2m$ agents with budget $c/4$. We use A_1 to represent the agent set. For each agent $k\in A_1$, set $v_k(k') = 0$ for each agent $k' \neq k$, $v_k(\bar{r}) = 5c/8$ for each $\bar{r} \in \Re_1$ and $v_k(r) = 0$ for each $r\in \Re$. Let $A' = A \cup A_1$ and $\Re' = \Re \cup \Re_1$. We set the capacity of each room in $\Re'$ as two. The reduction requires $O(n^4)$ time. We regard not allocating agent i as allocating roommate pair $(i, \emptyset)$ to room $\emptyset$. If no agents are allocated to room r, we regard this as allocating roommate pair $(\emptyset, \emptyset)$ to room r.

The $\Leftarrow$ direction: For instance $< A', \Re' >$, a room envy-free allocation must ensure that each agent is allocated to one room. This is because if an agent i is not allocated to any room, the utility of roommate pair $(i, \emptyset)$ is 0. However, if roommate pair $(i, \emptyset)$ is allocated to room r_c, its utility will become $5c/8\text{-}c/8=c/2>0$. Thus, roommate pair $(i, \emptyset)$ will envy the roommate pair that is allocated to room r_c. Because of the budget constraints, each agent $k\in A_1$ is allocated to one room $\bar{r} \in \Re_1$ and each agent $i\in A$ is allocated to one room $r\in \Re$ in a room envy-free allocation. Because all the agents are allocated, a room envy-free allocation can be represented as $n+m$ triples. For a triple (i, j, r) $(r\in \Re)$, if there is no edge between vertices $\bar{u}_i$ and $\bar{u}_j$ that correspond to agents i and j respectively, the utility of roommate pair (i, j) does not exceed $2c\text{-}c=c$. However, if roommate pair (i, j) is allocated to the edge room r_{ij}, the utility will become $2c\text{-}c/4=7c/4>c$. Then, roommate pair (i, j) envies the roommate pair that is allocated to room r_{ij}. For a triple (i, j, r) $(r\in \Re)$, if there is no edge between vertices $\bar{u}_i$ (or $\bar{u}_j$) and $\bar{u}_r$ that correspond to agent i (or j) and room r respectively, the utility of roommate pair (i, j) does not exceed $v_i(j) + v_j(i)+c/2+c\text{-}c= v_i(j) + v_j(i)+c/2$. In contrast, if roommate pair (i, j) is allocated to room r_c, the utility will become $v_i(j)+v_j(i)+c> v_i(j)+v_j(i)+c/2$. Then, roommate pair (i, j) envies the roommate pair that is allocated to room r_c. Based on the above analyses, we conclude that if agents i and j $(i, j\in A)$ are allocated to room $r\in \Re$ in a room envy-free allocation, the corresponding vertices $\bar{u}_i$, $\bar{u}_j$ and $\bar{u}_r$ must be able to construct a triangle. Namely, if the instance $< A', \Re' >$ admits a room envy-free allocation, $G=(X\cup Y\cup Z, E)$ can be partitioned into n vertex-disjoint triangles.

The $\Rightarrow$ direction: If $G=(X\cup Y\cup Z, E)$ can be partitioned into n vertex-disjoint triangles $\{< \bar{u}_i, \bar{u}_j, \bar{u}_r >\}$, for each triangle $< \bar{u}_i, \bar{u}_j, \bar{u}_r >$, we allocate the corresponding agents i and j to the corresponding vertex room r. Then, the utility of roommate pair (i, j) is $2c$, which is the maximum utility that it can obtain in a room $r\in \Re\cup\{r_c\}$. Moreover, for each edge room $\bar{r} \in \Re_1\backslash\{r_c\}$, $v_i(\bar{r})+v_j(\bar{r})$ is at most c because there is an edge between vertices $\bar{u}_i$ and $\bar{u}_j$. Then, the utility of roommate pair (i, j) is at most $7c/4<2c$ when they are allocated to an edge room. Thus, roommate pair (i, j) does not envy any other roommate pair. After that, we randomly allocate all the agents in A_1 to the rooms in $\Re_1$. Because each agent $k\in A_1$ has the same valuation function and $v_k(\bar{r}) = 5c/8> v_k(r) = 0$ for $\forall\bar{r} \in \Re_1$ and $\forall r\in \Re$, no matter how they are allocated to rooms in $\Re_1$, they do not envy any roommate pair in a room $r\in \Re$ and do not envy each other. Therefore, the final allocation is room envy-free. □

VII. EXPERIMENTAL EVALUATION

In the section, we mainly evaluate the solution qualities of the proposed algorithms based on simulation experiments. In the experiments, we randomly generate the agent valuations, agent budgets, room prices and room capacities. More precisely, the agent valuations obey the uniform distribution in [-10, 10]. The agent budgets and room prices obey the uniform distribution in [0, 10] and [0, 50] respectively. The room capacities obey the uniform distribution in $[1, C]$, where C $(C>1)$ is a constant.

To evaluate the performance of the proposed algorithms (i.e., the exact algorithm, SPBA, LSBA, LSBAMC), we compare them with the following approaches.

- **Double-matching approach** (**DMA**): in this approach, first, we transform the problem instance to an instance where the capacity of each room is 2. More precisely, we check the capacity of each room. If the capacity of a room is odd, we add a virtual bed to the room and add a virtual agent to the instance. For a virtual agent i, any agent j and any room r, we set $v_i(j)=v_j(i)=v_i(r)=0$. After that, we divide each room into several sub-rooms, each of which can contain 2 agents. Let room $\bar{r}$ be a sub-room of room r. For an agent i, we set $v_i(\bar{r})=v_i(r)$. Second, we adopt the algorithm of Chan *et al.* [1] to find an allocation π' for the agents. Finally, if a real agent $i \in A$ is allocated to a real bed that belongs to a sub-room of room r with $b_i \geq p_r/c_r$ in π', we allocate agent i to room r. Otherwise, we do not allocate agent i to any room.
- **Popular-matching approach** (**PMA**): in this approach, first, we adopt the algorithm of Paluch [26] to find an allocation π' for the agents. Second, if an agent $i \in A$ is allocated to a room r with $b_i \geq p_r/c_r$ in π', we allocate agent i to room r. Otherwise, we do not allocate agent i to any room.
- **Greedy approach** (**Greedy**): in this approach, we allocate each agent i to the room that can maximize u_i+p_i.

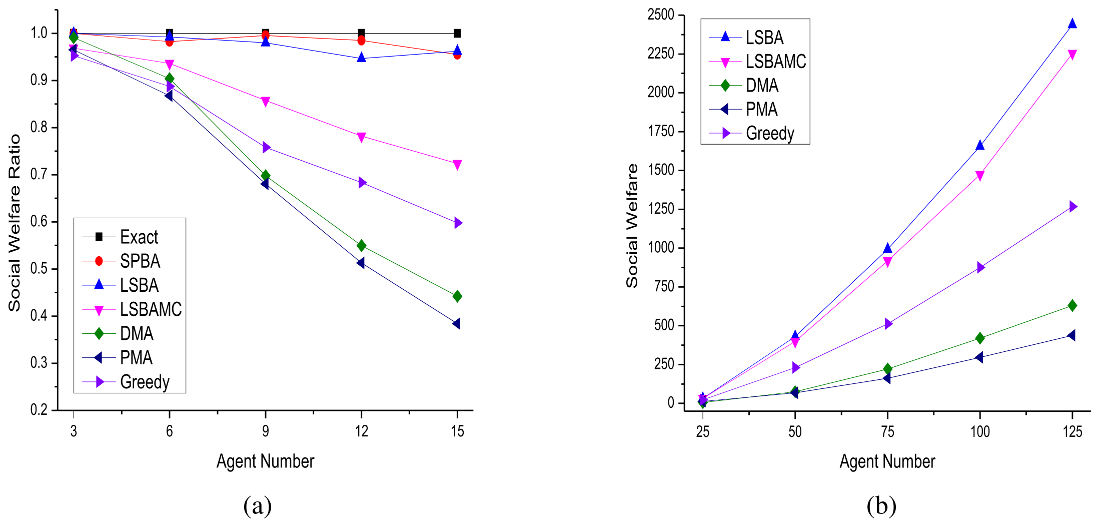


FIGURE 3. The comparisons with the change of agent number.

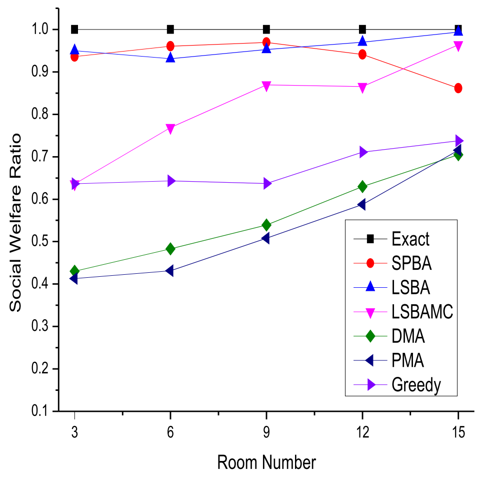
In the LSBA (or LSBAMC) algorithm, we only execute the swaps that can increase $SW(\pi)$ by at least $W'/1000$ (or the swaps that can increase $SW(\pi)$ by at least $W'/1000$ and do not decrease the trade volume), where W' is computed based on Algorithm 3. We conduct the experiments at a computer with Intel E5-2640 v4 CPU (2.40GHz) and 32GB RAM. In the experiments, the mixed-integer linear programming is solved using CPLEX 12.5. Each experiment comprises 100 runs to obtain the average results.

Fig. 3 shows the comparisons of the approaches with the change of agent number. In Fig. 3(a), the room number is 5 and the C is 5. In Fig. 3(a), the ordinate is the ratio between the social welfare $SW(\pi)$ of the allocation π that is output by an approach and the value of the optimal solution, i.e., $SW(OPT)$. For convenience, we refer to the ratio as social welfare ratio. In Fig. 3(a), we compare all the approaches. From Fig. 3(a), we find that the social welfare ratios of both the SPBA algorithm and the LSBA algorithm are larger than 0.9 all the time. Moreover, the SPBA algorithm and the LSBA algorithm outperform the other non-optimal approaches. Comparing with the LSBA algorithm, the LSBAMC algorithm has a smaller social welfare ratio. This is because the LSBAMC algorithm only executes the swaps that do not decrease the trade volume in the local search process. However, the LSBA algorithm does not consider the constraint of trade volume in the local search process. In spite of this, the LSBAMC algorithm outperforms the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy) when the agent number is larger 6.

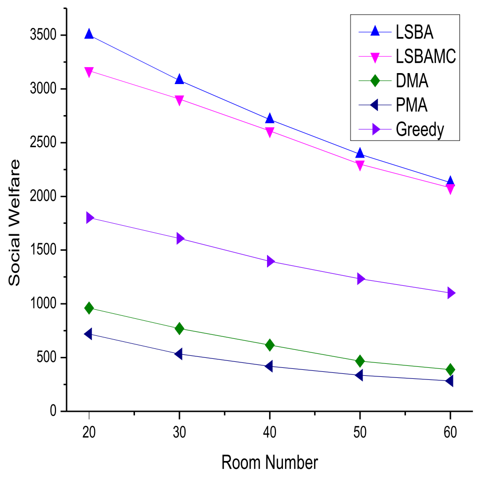
In Fig. 3(b), the room number is 25 and the C is 10. Because the social welfare maximization problem is NP-hard, the exact (optimal) algorithm is not applicable to large-scale instances. Moreover, because the SPBA algorithm will enumerate each possible allocation state of each room, the SPBA algorithm is not applicable to the instances that include rooms with large capacities. Thus, we only compare the LSBA algorithm, the LSBAMC algorithm, the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy) in Fig. 3(b). In Fig. 3(b), the ordinate is the social welfare $SW(\pi)$ of the allocation π that is output by an approach. From Fig. 3(b), we find that both the LSBA algorithm and the LSBAMC algorithm outperform the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy). Moreover, we find that when the instance scale is relatively large, the performance of the LSBAMC algorithm is close to that of the LSBA algorithm.

Fig. 4 shows the comparisons of the approaches with the change of room number. In Fig. 4(a), the agent number is 15 and the C is 5. In Fig. 4(a), the ordinate is social welfare ratio and we compare all the approaches. From Fig. 4(a), we find that the SPBA algorithm, the LSBA algorithm and the LSBAMC algorithm outperform the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy). Moreover, the social welfare ratio of the LSBA algorithm is larger than 0.9 all the time. Not only that, we also find that the social welfare ratio of the LSBAMC algorithm increases in the increasing process of the room number.

In Fig. 4(b), the agent number is 150 and the C is 10. Because the exact (optimal) algorithm is not applicable to large-scale instances and the SPBA algorithm is not applicable to the instances with large room capacities, we only compare the LSBA algorithm, the LSBAMC algorithm, the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy) in Fig. 4(b). In Fig. 4(b), the ordinate is the social welfare $SW(\pi)$ of the allocation π that is output by an approach. When the room number becomes larger, we usually need to divide the agents into more roommate groups.

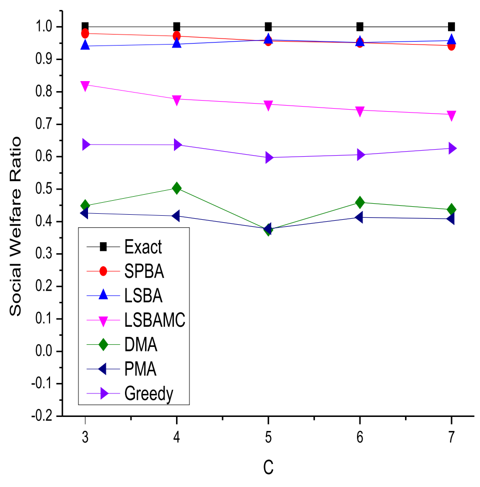


(a)

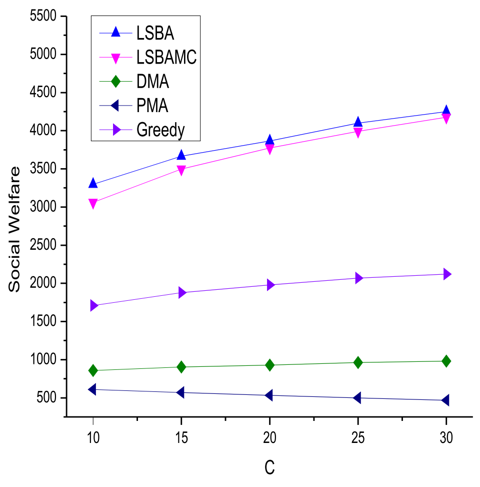


(b)

FIGURE 4. The comparisons with the change of room number.



(a)



(b)

FIGURE 5. The comparisons with the change of the capacity upper-bound of each room.

This usually makes each agent have fewer roommates. Therefore, the social welfare of each algorithm in Fig. 4(b) decreases with the increase of the room number. In spite of this, both the LSBA algorithm and the LSBAMC algorithm outperform the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy). Moreover, we can also find that the performance of the LSBAMC algorithm becomes closer to that of the LSBA algorithm with the increase of the room number.

Fig. 5 shows the comparisons of the approaches with the change of the capacity upper-bound of each room. In Fig. 5(a), the agent number is 15 and the room number is 5. In Fig. 5(a), the ordinate is social welfare ratio and we compare all the approaches. From Fig. 5(a), we find that the social welfare ratios of both the SPBA algorithm and the LSBA algorithm are larger than 0.9 all the time. Although the LSBAMC algorithm has a smaller social welfare ratio than the LSBA algorithm, it still outperforms the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy).

In Fig. 5(b), the agent number is 150 and the room number is 25. Because the exact (optimal) algorithm is not applicable to large-scale instances and the SPBA algorithm is not applicable to the instances with large room capacities, we only compare the LSBA algorithm, the LSBAMC algorithm, the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy) in Fig. 5(b). In Fig. 5(b), the ordinate is the social welfare $SW(\pi)$ of the allocation π that is output by an approach. From Fig. 5(b), we find that both the LSBA algorithm and the LSBAMC algorithm outperform the Double-matching approach (DMA), the Popular-matching approach (PMA) and the Greedy approach (Greedy). Moreover, we can also find that the performance of the LSBAMC algorithm becomes closer to that of the LSBA algorithm with the increase of the C.

According to the experimental results, we find that the LSBA algorithm can produce near-optimal solutions. Moreover, the performance of the LSBAMC algorithm is close to that of the LSBA algorithm when the instance scale is relatively large. This illustrates that the LSBAMC algorithm can have good performance when the instance scale is relatively large, although it considers the constraint of trade volume.

VIII. CONCLUSIONS AND FUTURE WORK

This paper investigates the room allocation problem with capacity diversity and budget constraints. We mainly focus on finding an allocation that maximizes the social welfare. First, this paper demonstrates that finding an allocation that maximizes the social welfare is NP-hard, even if only one room's capacity is larger than 1 and the other rooms' capacities are all 1. Second, this paper presents a polynomial-time $(c^* + 2)/2+\varepsilon$-factor approximation algorithm for the case in which the capacity of each room is bounded by a constant c^*. Third, this paper demonstrates that there is no polynomial-time $c^*/2$-factor approximation algorithm for the social welfare maximization problem unless P=NP, where c^* is the capacity upper-bound of each room. Fourth, this paper proposes a heuristic algorithm based on local search for the general case in which the capacity of each room is not bounded by a constant. The experimental results demonstrate that the proposed algorithm can produce near-optimal solutions. Fifth, this paper shows that a 2-person weakly stable allocation with a provable social welfare guarantee can be found in polynomial time if the capacity of each room is bounded by a constant c^*. Finally, this paper proves that it is NP-hard to determine whether an instance with specified room prices admits a room envy-free allocation. In future work, we will investigate the room allocation with couples, in which a couple must be allocated to the same room.

REFERENCES

[1] P. H. Chan, X. Huang, Z. Liu, C. Zhang, and S. Zhang, "Assignment and pricing in roommate market," in *Proc. 13th AAAI Conf. Artif. Intell. (AAAI)*, Phoenix, AZ, USA, Feb. 2016, pp. 446–452.

[2] G. Huzhang, X. Huang, S. Zhang, and X. Bei, "Online roommate allocation problem," in *Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI)*, Melbourne, Australia, Aug. 2017, pp. 235–241.

[3] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 69, no. 1, pp. 9–15, Jan. 1962.

[4] R. W. Irving, "An efficient algorithm for the 'stable roommates' problem," *J. Algorithms*, vol. 6, no. 4, pp. 577–595, 1985.

[5] R. Bredereck, J. Chen, U. P. Finnendahl, and R. Niedermeier, "Stable roommate with narcissistic, single-peaked, and single-crossing preferences," in *Proc. 5th Int. Conf. Algorithmic Decision Theory*. Springer, Luxembourg, U.K., Oct. 2017, pp. 315–330.

[6] T. Feder, N. Megiddo, and S. A. Plotkin, "A sublinear parallel algorithm for stable matching," *Theor. Comput. Sci.*, vol. 233, nos. 1–2, pp. 297–308, 2000.

[7] D. Gusfield and R. W. Irving, *The Stable Marriage Problem: Structure and Algorithms*. Cambridge, MA, USA: MIT Press, 1989.

[8] R. W. Irving and D. F. Manlove, "The stable roommates problem with ties," *J. Algorithms*, vol. 43, no. 1, pp. 85–105, 2002.

[9] M. Pęski, "Large roommate problem with non-transferable random utility," *J. Econ. Theory*, vol. 168, pp. 432–471, Mar. 2017.

[10] E. Ronn, "NP-complete stable matching problems," *J. Algorithms*, vol. 11, no. 2, pp. 285–304, 1990.

[11] J. Alcalde, "Exchange-proofness or divorce-proofness? Stability in one-sided matching markets," *Econ. Des.*, vol. 1, no. 1, pp. 275–287, 1994.

[12] K. Cechlárová, "On the complexity of exchange-stable roommates," *Discrete Appl. Math.*, vol. 116, no. 3, pp. 279–287, 2002.

[13] K. Cechlárová and D. F. Manlove, "The exchange-stable marriage problem," *Discrete Appl. Math.*, vol. 152, nos. 1–3, pp. 109–122, 2005.

[14] P. Biró, R. W. Irving, and D. F. Manlove, "Popular matchings in the marriage and roommates problems," in *Proc. 7th Int. Conf. Algorithms Complex.*, Rome, Italy: Springer, May 2010, pp. 97–108.

[15] L. Liu and Q. Fan, "Resource allocation optimization based on mixed integer linear programming in the multi-cloudlet environment," *IEEE Access*, vol. 6, pp. 24533–24542, 2018.

[16] M. B. Qureshi, M. A. Alqahtani, and N. Min-Allah, "Grid resource allocation for real-time data-intensive tasks," *IEEE Access*, vol. 5, pp. 22724–22734, 2017.

[17] R. Liu, M. Sheng, and W. Wu, "Energy-efficient resource allocation for heterogeneous wireless network with multi-homed user equipments," *IEEE Access*, vol. 6, pp. 14591–14601, 2018.

[18] J. Bartholdi, III, and M. A. Trick, "Stable matching with preferences derived from a psychological model," *Oper. Res. Lett.*, vol. 5, no. 4, pp. 165–169, 1986.

[19] B. Pittel, "On likely solutions of the stable matching problem with unequal numbers of men and women," *Math. Oper. Res.*, vol. 44, no. 1, pp. 1–25, 2018.

[20] C.-K. Lam and C. G. Plaxton, "A (1+1/e)-approximation algorithm for maximum stable matching with one-sided ties and incomplete lists," in *Proc. 13th Annu. ACM-SIAM Symp. Discrete Algorithms*, San Diego, CA, USA, Jan. 2019, pp. 2823–2840.

[21] D. J. Abraham, R. W. Irving, T. Kavitha, and K. Mehlhorn, "Popular matchings," in *Proc. 16th Annu. ACM-SIAM Symp. Discrete Algorithms*, Vancouver, BC, Canada, Jan. 2005, pp. 424–432.

[22] M. Mahdian, "Random popular matchings," in *Proc. 7th ACM Conf. Electron. Commerce*, Ann Arbor, MI, USA, Jun. 2006, pp. 238–242.

[23] J. Mestre, "Weighted popular matchings," in *Proc. 33rd Int. Colloq. Automata, Lang., Program.* Venice, Italy: Springer, Jul. 2006, pp. 715–726.

[24] D. F. Manlove and C. T. Sng, "Popular matchings in the capacitated house allocation problem," in *Proc. Eur. Symp. Algorithms*. Zürich, Switzerland: Springer, Sep. 2006, pp. 492–503.

[25] C. T. S. Sng and D. F. Manlove, "Popular matchings in the weighted capacitated house allocation problem," *J. Discrete Algorithms*, vol. 8, no. 2, pp. 102–116, 2010.

[26] K. Paluch, "Capacitated rank-maximal matchings," in *Proc. 8th Int. Conf. Algorithms Complex.* Barcelona, Spain: Springer, May 2013, pp. 324–335.

[27] H. Aziz, F. Brandt, and P. Harrenstein, "Pareto optimality in coalition formation," *Games Econ. Behav.*, vol. 82, pp. 562–581, Nov. 2013.

[28] S. Banerjee, H. Konishi, and T. Sönmez, "Core in a simple coalition formation game," *Social Choice Welfare*, vol. 18, no. 1, pp. 135–153, 2001.

[29] A. Bogomolnaia and M. O. Jackson, "The stability of hedonic coalition structures," *Games Econ. Behav.*, vol. 38, no. 2, pp. 201–230, 2002.

[30] K. Cechlárová and A. Romero-Medina, "Stability in coalition formation games," *Int. J. Game Theory*, vol. 29, no. 4, pp. 487–494, 2001.

[31] J. H. Dreze and J. Greenberg, "Hedonic coalitions: Optimality and stability," *Econometrica, J. Econ. Soc.*, vol. 48, no. 4, pp. 987–1003, 1980.

[32] H. Aziz, F. Brandt, and H. G. Seedig, "Computing desirable partitions in additively separable hedonic games," *Artif. Intell.*, vol. 195, pp. 316–334, Feb. 2013.

[33] H. Aziz, F. Brandt, and P. Harrenstein, "Fractional hedonic games," in *Proc. Int. Conf. Auton. Agents Multi-Agent Syst. (IFAAMAS)*, Paris, France, May 2014, pp. 5–12.

[34] H. Aziz, S. Gaspers, J. Gudmundsson, J. Mestre, and H. Täubig, "Welfare maximization in fractional hedonic games," in *Proc. 24th Int. Joint Conf. Artif. Intell. (IJCAI)*, Buenos Aires, Argentina, Jul. 2015, pp. 461–467.

[35] V. Bilò, A. Fanelli, M. Flammini, G. Monaco, and L. Moscardelli, "Nash stable outcomes in fractional hedonic games: Existence, efficiency and computation," *J. Artif. Intell. Res.*, vol. 62, pp. 315–371, Jun. 2018.

[36] H. Aziz, "Stable marriage and roommate problems with individual-based stability," in *Proc. 2013 Int. Conf. Auto. agents multi-Agent Syst.*, Saint Paul, MN, USA, May 2013, pp. 287–294.

[37] A. Darmann, E. Elkind, S. Kurz, J. Lang, J. Schauer, and G. Woeginger, "Group activity selection problem," in *Proc. 8th Int. Workshop Internet Netw. Econ.*, Montreal, Canada: Springer, Dec. 2012, pp. 156–169.

[38] A. Igarashi, D. Peters, and E. Elkind, "Group activity selection on social networks," in *Proc. 31st AAAI Conf. Artif. Intell. (AAAI)*, San Francisco, CA, USA, Feb. 2017, pp. 565–571.
[39] H. Aziz, F. Brandt, and H. G. Seedig, "Stable partitions in additively separable hedonic games," in *Proc. 10th Int. Conf. Auto. Agents Multiagent Syst. (IFAAMAS)*, Taipei, Taiwan, May 2011, pp. 183–190.
[40] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum, 1972, pp. 85–103.
[41] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems," *Math. Program.*, vol. 10, no. 1, pp. 147–175, 1976.
[42] P. Berman, "A d/2 approximation for maximum weight independent set in d-claw free graphs," in *Proc. Scand. Workshop Algorithm Theory*. Bergen, Norway: Springer, Jul. 2000, pp. 214–219.
[43] B. Chandra and M. M. Halldórsson, "Greedy local improvement and weighted set packing approximation," *J. Algorithms*, vol. 39, no. 2, pp. 223–240, 2001.
[44] C. D. Complessitá and R. Rizzi, "Np-complete problem: Partition into triangles," Univ. Trento, Trento, Italy, Tech. Rep. 112360, 2004, pp. 1–4.
[45] J. R. Douceur and R. P. Wattenhofer, "Optimizing file availability in a secure serverless distributed file system," in *Proc. 20th IEEE Symp. Reliable Distrib. Syst.*, New Orleans, LA, USA, Oct. 2001, pp. 4–13.
[46] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. 29th IEEE Int. Conf. Comput. Commun.*, San Diego, CA, USA, Mar. 2010, pp. 1–9.
[47] H. Angelidakis, Y. Makarychev, and P. Manurangsi, "An improved integrality gap for the Călinescu-Karloff-Rabani relaxation for multiway cut," in *Proc. 19th Int. Conf. Integer Program. Combinat. Optim.* Waterloo, ON, Canada: Springer, Jun. 2017, pp. 39–50.
[48] N. Buchbinder, J. S. Naor, and R. Schwartz, "Simplex partitioning via exponential clocks and the multiway cut problem," in *Proc. 44th Annu. ACM Symp. Theory Comput.*, Palo Alto, CA, USA, Jun. 2013, pp. 535–544.
[49] N. Buchbinder, R. Schwartz, and B. Weizman, "Simplex transformations and the multiway cut problem," in *Proc. 28th Annu. ACM-SIAM Symp. Discrete Algorithms*, Barcelona, Spain, Jan. 2017, pp. 2400–2410.
[50] G. Călinescu, H. Karloff, and Y. Rabani, "An improved approximation algorithm for MULTIWAY CUT," in *Proc. 13th Annu. ACM Symp. Theory Comput.*, Dallas, Texas, USA, May 1998, pp. 48–52.
[51] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis, "The complexity of multiterminal cuts," *SIAM J. Comput.*, vol. 23, no. 4, pp. 864–894, 1994.
[52] A. Freund and H. Karloff, "A lower bound of 8/(7+ 1k−1) on the integrality ratio of the Călinescu–Karloff–Rabani relaxation for multiway cut," *Inf. Process. Lett.*, vol. 75, nos. 1–2, pp. 43–50, 2000.
[53] D. R. Karger, P. Klein, C. Stein, M. Thorup, and N. E. Young, "Rounding algorithms for a geometric embedding of minimum multiway cut," *Math. Oper. Res.*, vol. 29, no. 3, pp. 436–461, 2004.
[54] A. Sharma and J. Vondrák, "Multiway cut, pairwise realizable distributions, and descending thresholds," in *Proc. 46th Annu. ACM Symp. Theory Comput.*, New York, NY, USA, May/Jun. 2014, pp. 724–733.
[55] K. Andreev and H. Racke, "Balanced graph partitioning," *Theory Comput. Syst.*, vol. 39, no. 6, pp. 929–939, 2006.
[56] M. R. Garey and D. S. Johnson, "Computers and intractability: A guide to the theory of np-completeness," in *Computer Intractability*, vol. 340. New York, NY, USA: Freeman 1979.
[57] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics*, vol. 52, no. 1, pp. 7–21, 2010.

YUNPENG LI received the B.S. degree from the School of Computer Science and Engineering, Southeast University, Nanjing, China, in 2013, where he is currently pursuing the Ph.D. degree with the Intelligent Systems and Social Computing Laboratory. He has published several scientific articles in refereed journals and conference proceedings, such as the *ACM Transactions on Autonomous and Adaptive Systems*, *Expert Systems With Applications*, the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), the IEEE International Conference on Tools With Artificial Intelligence (ICTAI), and the IEEE International Conference on Web Services (ICWS). His current research interests include multiagent systems and services computing.

YICHUAN JIANG received the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2005. He is currently a Full Professor with the School of Computer Science and Engineering, Southeast University, Nanjing, China. He has published more than 100 scientific articles in refereed journals and conference proceedings, such as the IEEE Transactions on Mobile Computing, the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, the IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, the IEEE Transactions on Systems, Man, and Cybernetics: Systems, the IEEE Transactions on Cybernetics, the *ACM Transactions on Autonomous and Adaptive Systems*, the *Journal of Autonomous Agents and Multi-Agent Systems*, the *Journal of Parallel and Distributed Computing*, the International Joint Conference on Artificial Intelligence (IJCAI), the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), and the AAAI Conference on Artificial Intelligence (AAAI). His main research interests include multiagent systems, social computing, and social networks. He received the Best Paper Award from the PRIMA 2006 and two best student paper awards from the ICTAI 2013 and the ICTAI 2014.

WEIWEI WU received the B.Sc. degree from the South China University of Technology and the Ph.D. degree from the Department of Computer Science, City University of Hong Kong (CityU) and the University of Science and Technology of China (USTC), in 2011. He went to the Mathematical Division, Nanyang Technological University (NTU), Singapore, for his Ph.D. research, in 2012. He is currently an Associate Professor with Southeast University, China. His research interests include optimizations and algorithm analysis, wireless communications, crowdsourcing, cloud computing, game theory, and network economics.

JIUCHUAN JIANG is currently with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. He has published several scientific articles in refereed journals and conference proceedings, such as the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Systems, Man, and Cybernetics: Systems, the *ACM Transactions on Autonomous and Adaptive Systems*, *Information Processing and Management*, *Information Processing Letters*, and the International Conference on Autonomous Agents and Multiagent Systems (AAMAS). His main research interests include crowdsourcing, multiagent systems, and social networks.

HUI FAN received the B.S. degree in computer science from Shandong University, Jinan, China, in 1984, and the Ph.D. degree in computer science from the Taiyuan University of Technology, Taiyuan, China, in 2007. From 1984 to 2001, he was a Professor with the Computer Department, Taiyuan University of Technology. He is currently a Professor with Shandong Technology and Business University. His research interests include computer-aided geometric design, computer graphics, information visualization, virtual reality, image processing, and multiagent systems.