

A Simple Greedy Algorithm for the Profit-Aware Social Team Formation Problem

Shengxin Liu¹ and Chung Keung Poon²(✉)

¹ Department of Computer Science, City University of Hong Kong,
Hong Kong, China

shengxliu2-c@my.cityu.edu.hk

² School of Computing and Information Sciences, Caritas Institute of Higher
Education, Hong Kong, China

ckpoon@cihe.edu.hk

Abstract. Team formation in social networks has attracted much attention due to its many applications such as the online labour market. In this paper, we focus on the problem of forming multiple teams of experts with diverse skills in social network to accomplish complex tasks of required skills. The goal is to maximize the total profit of tasks that these teams can complete. We provide a simple and practical algorithm that improves upon previous results in many situations.

1 Introduction

Team formation in a networked community of experts is concerned with forming teams of experts to complete certain tasks. A team is qualified (or *feasible*) for a task if the team as a whole possesses all the skills required by the task and the team members are “socially compatible”, i.e., they can collaborate smoothly according to an underlying social network. This topic has gained much attention recently due to the many applications in social collaboration made possible by the World Wide Web. One specific example is the *online labour market*. In online platforms such as Freelancer (www.freelancer.com), Guru (www.guru.com) and Upwork (www.upwork.com), projects with various skill requirements are posted and freelancers who possess the required skills can bid for the projects [7]. As observed by Greenwald [8], more and more freelancers are willing to team up with others who have complementary skills in order to take up more complicated and profitable projects. In parallel to this phenomenon, many major platforms (such as Upwork) also provide team-hiring services for their enterprise customers.

In this paper, we study the following team formation problem. Imagine that there is a collection of tasks, each specified by the set of skills it requires and the profit gained when the task is completed. There is also a group of experts over a social network, each possessing a certain set of skills and having a capacity which limits the maximum number of tasks he/she can take up. Our goal is to form multiple (possibly overlapping) feasible teams of experts to maximize the total profit of tasks that can be solved subject to the capacity constraints.

Table 1. Comparison of the approximation ratios of Tang’s and our algorithms. Note that $\sqrt{\sum_{V \in \mathcal{V}} c(V)/c(\min)} \geq \sqrt{m}$. In many applications, $\Delta(I)$ is much smaller than \sqrt{m} .

Social compatibility	Tang’s algorithm [17, 18] and extension	This paper
General case	$\beta \min\{\Delta(I) + 1, 2(\sqrt{\frac{\sum_{V \in \mathcal{V}} c(V)}{c(\min)}} + 1)\}$	$\min\{\Delta(I) + 1, m\}$
Hereditary case	Same as above	$\min\{\Delta(I) + 1, m, k\}$

The above problem, which we called the *Profit-aware Social Team Formation Problem*, was first introduced by Tang [17, 18]. (His version is actually slightly different from ours but the essence is the same.) Tang’s algorithm is based on an LP-rounding approach (see, for example, [3, 9]) which runs in polynomial time (when the value of expert’s capacity is polynomial in the input size) and produces an approximate solution with a performance guarantee. First, he formulated a linear programming relaxation of the problem using a modified input instance and obtained a fractional solution by invoking an ellipsoid algorithm. Assuming the availability of a polynomial time oracle for the *Min-cost Team Selection Problem* (to be defined in Sect. 2), the ellipsoid algorithm runs in polynomial time (when the value of expert’s capacity is polynomial with respect to the input size). Then, a clever rounding is applied to convert the fractional solution into an integral solution in polynomial time. To describe the approximation ratio of Tang’s algorithm, let m be the number of experts and $\Delta(I)$ be the size of a largest *minimal feasible* team for a task in the input I . A team is said to be *minimal feasible* for a task if none of its members can be removed without making the team infeasible for that task. (Obviously, $\Delta(I)$ is at most the number of total experts, i.e., m , in the input. However, $\Delta(I)$ is usually much smaller than m in our applications.) We also denote by \mathcal{V} the set of experts, $c(V)$ the capacity of expert $V \in \mathcal{V}$ and let $c(\min) = \min_{V \in \mathcal{V}} c(V)$. (Formal definition will be given in Sect. 2.) Then Tang’s algorithm guarantees an approximation ratio of $\beta \min\{\Delta(I) + 1, 2\sqrt{m} + 1\}$, for the unit expert capacity case (i.e., $c(V) = 1$ for each $V \in \mathcal{V}$) where β is the approximation ratio of the oracle. Based on Tang’s algorithm [17, 18], a minor generalization achieves a polynomial time and an approximation ratio of $\beta \min\{\Delta(I) + 1, 2(\sqrt{\sum_{V \in \mathcal{V}} c(V)/c(\min)} + 1)\}$ for general $c(V)$ (see Table 1). This generalization is also based on an LP-rounding method where we first solve the LP using the original input instance and then utilize certain generalized rounding methods. Note that $\sqrt{\sum_{V \in \mathcal{V}} c(V)/c(\min)} \geq \sqrt{m}$ where equality holds when each expert $V \in \mathcal{V}$ has the same capacity $c(V)$. The performance of Tang’s algorithm relies on β , which, in turn, depends on the precise problem definition. We will discuss β for different variants in Sect. 2.1. We also remark that although Tang’s algorithm runs in polynomial time, it requires running an ellipsoid algorithm which could be a big overhead in practice.

In this paper, we design a simple and efficient greedy algorithm for the problem without using an ellipsoid algorithm. Our algorithm also makes use of an oracle for the Min-cost Team Selection Problem. However, it has an approximation ratio of $\min\{\Delta(I) + 1, m\}$ which is independent of the performance of the oracle, i.e., β . See Table 1 for a comparison of approximation ratios. When $\Delta(I) \leq 2\sqrt{\sum_{V \in \mathcal{V}} c(V)/c(\min)} + 1$, our algorithm has better approximation ratio than Tang’s algorithm by a factor of β . This case is common in practice, which can be seen from the experimental part of previous studies on different team formation algorithms (e.g., [6, 10, 13, 15, 18, 19]). For example, [18] considered a dataset from Upwork in which the number of experts is of the order $m = 10$ million while the maximum team size is about 500. Note that both Tang’s algorithm and ours are very general and work for any definition of social compatibility as long as there is an appropriate oracle for the corresponding Min-cost Team Selection Problem. We also consider the class of social compatibilities that are *hereditary*, i.e., any sub-team of a socially compatible team is also socially compatible. For this class of social compatibilities, our algorithm achieves an approximation ratio of $\min\{\Delta(I) + 1, m, k\}$ where k is the number of skills (see also Table 1). We will discuss variants of social compatibilities in Sect. 2.1.

In summary, our greedy algorithm has the following advantages. First, our algorithm is simple and efficient without utilizing the time-consuming ellipsoid algorithm. Second, the approximation ratio of our algorithm is better in our applications as verified by several studies on team formation problem. Lastly, we are the first to characterize and study the hereditary social compatibility and our algorithm can achieve an improved approximation ratio for this case of social compatibilities.

1.1 Other Related Works

A number of variants of team formation problems over social networks have been considered in the literature. In particular, there have been studies on the problems of selecting a single team for a single task while minimizing the *coordination cost* [10, 11, 14] or maximizing the *social compatibility* [6, 15] among the team members in the social network. Some other works generalized the *binary skill coverage model* (where a skill is covered by a team if at least one team member possesses that skill) to models where a skill may require more than one experts [6, 13, 15].

For the scenario of multiple tasks, Anagnostopoulos et al. [1, 2] considered the assignment of all tasks to the experts while balancing their workload. On the other hand, Golshan et al. [7] did not aim at covering all tasks but at maximizing the total profit of covered tasks by selecting a single team within a given budget. This is different from the problem of multiple teams solving multiple tasks studied in this paper.

For a discussion and comparison of various definitions of team formation in social networks, readers are referred to the surveys by Wang et al. [19, 20].

Paper Organization. The rest of this paper is organized as follows. In the next section, we provide the problem definitions as well as some notations and a discussion on social compatibilities, β and $\Delta(I)$. In Sect. 3, we present our greedy algorithm and its performance analysis. We discuss the adaption of our greedy algorithm to solve the problem considered by Tang [17, 18] in Sect. 4. Finally, Sect. 5 contains our conclusion and discusses future work.

2 Preliminaries

Throughout this paper, we denote by \mathcal{S} the set of skills, \mathcal{T} the set of tasks and \mathcal{V} the set of experts. Let k , n and m be their sizes respectively, i.e., $k = |\mathcal{S}|$, $n = |\mathcal{T}|$ and $m = |\mathcal{V}|$. For each task $T \in \mathcal{T}$, let $s(T) \subseteq \mathcal{S}$ be the set of skills required to complete task T and $p(T)$ be the profit gained when task T is completed. For each expert $V \in \mathcal{V}$, let $s(V) \subseteq \mathcal{S}$ be the set of skills that V possesses and $c(V)$ (the *capacity* of V) be the maximum number of tasks that V can take. Finally, the relationship among the experts in \mathcal{V} is captured by a social network \mathcal{G} , which is a graph over \mathcal{V} . The social compatibility of a team will be defined with respect to \mathcal{G} . In the literature, the social network \mathcal{G} is often a weighted undirected graph while different definitions of social compatibilities have been considered. In our problem definitions below, we leave the exact definition of the social network and social compatibility open so that our results are as generally applicable as possible. Nevertheless, we will give a discussion on variants of the social compatibilities after defining our problems.

We adopt the binary skill coverage model so that a task T is covered by a team \mathcal{V}' if every skill in $s(T)$ is possessed by at least one expert in \mathcal{V}' . A feasible team is one that is also socially compatible:

Definition 1 (feasibility). *Let T be a task in \mathcal{T} and $\mathcal{V}' \subseteq \mathcal{V}$ be a team. We say that \mathcal{V}' is feasible for T if and only if $s(T) \subseteq \cup_{V \in \mathcal{V}'} s(V)$ and \mathcal{V}' is a socially compatible team.*

Definition 2 (minimal feasibility). *Let T be a task in \mathcal{T} and $\mathcal{V}' \subseteq \mathcal{V}$ be a team. We say that \mathcal{V}' is minimal feasible for T if and only if \mathcal{V}' is feasible for T and no proper subset of \mathcal{V}' is feasible for T .*

Thus, a minimal feasible team has no obvious redundant members.

Definition 3. *Given a set of skills \mathcal{S} , a set of tasks \mathcal{T} , a set of experts \mathcal{V} and the underlying social network \mathcal{G} on the experts, the **Profit-aware Social Team Formation Problem** is to form teams of experts $\mathcal{V}_1, \mathcal{V}_2, \dots$ such that \mathcal{V}_i is feasible for task T_{j_i} for all i and each expert V appears in at most $c(V)$ teams (i.e., $|\{\mathcal{V}_i | V \in \mathcal{V}_i\}| \leq c(V)$) while the total profit $\sum_i p(T_{j_i})$ is maximized.*

Both Tang's algorithm and ours for the above problem make use of an oracle for the Min-cost Team Formation Problem which we defined as follows:

Definition 4. Given a set of skills \mathcal{S} , a task T , a set of experts \mathcal{V} with a weight $w(V)$ on each expert $V \in \mathcal{V}$ and a social network \mathcal{G} on the experts, the **Min-cost Team Selection Problem** is to select a feasible team $\mathcal{V}' \subseteq \mathcal{V}$ for task T while the sum of weights for team members in \mathcal{V}' , i.e., $\sum_{V \in \mathcal{V}'} w(V)$, is minimized.

Specifically, both algorithms require the oracle to run in polynomial time and return a minimal feasible team whenever a feasible team exists.

2.1 Variants of Social Compatibility

Different definitions of social compatibility give rise to different variants of the above problems. A natural requirement on social compatibility is to require all team members to be connected. One can distinguish two models of connectivity, namely, the *Explicitly Connected Team* (ECT) model and the *Implicitly Connected Team* (ICT) model [2]. In the former model, we require that the team members are connected in the induced subgraph over the team members. On the other hand, the latter model just requires the team members to be connected within the original social network \mathcal{G} .

In the ECT model, Lappas et al. [11] were the first to consider two types of coordination cost, namely, the diameter (i.e., the maximum distance between a pair of team members) and the weight of a minimum spanning tree that connects all the team members in the social network. In the ICT model, Anagnostopoulos et al. [2] studied the problem using the diameter, weight of minimum Steiner tree as well as sum of pairwise distances in a team as coordination cost. Kargar and An [10] studied the problem in which each required skill of a task should be assigned an expert. (Thus a versatile expert may contribute to multiple skills in a task.) They then considered two definitions of coordination cost: (1) the sum of distances among the assigned expert of each skill and (2) the leader distance, i.e., the sum of distances between the leader and the assigned expert of each skill. Instead of minimizing a coordination cost, [6, 15] maximizes the team's compatibility measured by the density of the subgraph induced by the team. In general, any reasonable definition on community [5, 12] can be used here.

Recall that β is the approximation ratio of an oracle for the Min-cost Team Selection Problem. Clearly, β depends on the definition of the social compatibility. For example, when the social compatibility requirement is absent (i.e., any team is considered socially compatible), the corresponding Min-cost Team Selection Problem can be reduced to the Weighted Set Cover Problem. In this case, the oracle \mathcal{A} can be the classic greedy algorithm for the Weighted Set Cover Problem, which admits an approximation ratio of $\beta = O(\log k)$. As another example, suppose the social compatibility requires the selected team to form a connected subgraph in the ECT model. Then, we can apply algorithms for the directed Steiner tree problem as the oracle [4, 16] with $\beta = O(k^\epsilon)$ for any constant $\epsilon > 0$. We also point out that $\beta = \Omega(\log k)$ since the Min-cost Team Selection Problem generalizes the Weighted Set Cover Problem.

Besides β , the precise definition of social compatibility also affects $\Delta(I)$. Recall that $\Delta(I)$ is the size of a largest minimal feasible team for a task in the

input I . So, $\Delta(I)$ depends on the input I and obviously $\Delta(I) \leq m$. It can be proved that $\Delta(I) \leq k$ when the social compatibility is measured by the diameter, weight of minimum Steiner tree or sum of pairwise distances in the ICT model. In fact, these social compatibilities are examples of the class of hereditary social compatibilities which we defined below.

Definition 5 (hereditary social compatibility). *A social compatibility property is said to be hereditary if any team V' possessing this social compatibility property implies that every subset $V'' \subseteq V'$ also possesses this compatibility.*

We will prove that $\Delta(I) \leq k$ if the social compatibility property is hereditary in Sect. 4. Most previously studied social compatibilities are hereditary [2, 20]. On the other hand, any social compatibility on the ECT model does not have the hereditary property. This is because a subgraph of a connected graph is not necessarily connected. In this case, $\Delta(I)$ is not necessarily always less than k for the worst case instance I . For example, consider a social network which is a line of m vertices (i.e., experts) and a task that can only be completed by a team that includes the two end vertices of the social network due to the required skill set. Assume that $m > k$ and the social compatibility requires the selected team should form a connected subgraph on the ECT model. In order to connect these two end vertices, a feasible team should include all the m vertices which results in $\Delta(I) = m$. Thus, $\Delta(I) > k$ in this case.

3 Our Greedy Algorithm

Our algorithm, called GREEDY, is shown in Algorithm 1. The high-level idea of our algorithm is to process the tasks one by one in non-increasing order of their profit and for each task, select a suitable team using an oracle \mathcal{A} for the Min-cost Team Selection Problem. In each instance of the Min-cost Team Selection Problem, all experts will have the same cost. The oracle \mathcal{A} can return an exact or approximate solution. However, we require that the oracle runs in polynomial time (so that our algorithm also runs in polynomial time) and will return a minimal feasible team whenever a feasible team exists.

In Sect. 3.1, we present a simple charging scheme that proves the following result.

Theorem 1. *GREEDY is $(\Delta(I) + 1)$ -competitive.*

Since $\Delta(I) \leq m$, it follows from Theorem 1 that GREEDY is $(m + 1)$ -competitive. In Sect. 3.2, by adjusting the charging scheme carefully, we will show that:

Theorem 2. *GREEDY is m -competitive.*

Theorems 1 and 2 will complete our result stated for the general case in Table 1. We also study the class of hereditary social compatibilities in Sect. 3.4 and prove the following result below:

Algorithm 1. The GREEDY Algorithm

```

1 Sort the set of tasks  $\mathcal{T}$  in non-increasing order of profit. Without loss of
  generality, we assume that  $p(T_1) \geq p(T_2) \geq \dots \geq p(T_n)$ .
2 for each task  $T_i$  from  $i = 1$  to  $n$  do
3   Apply the polynomial-time  $\beta$ -approximation oracle  $\mathcal{A}$  for the Min-cost
   Team Selection Problem to find a team  $\mathcal{V}' \subseteq \mathcal{V}$ .
4   if there exists such a team  $\mathcal{V}'$  then
5     Complete  $T_i$  by using  $\mathcal{V}'$ .
6     Decrease the capacity  $c(V_j)$  of each expert  $V_j \in \mathcal{V}'$  by 1.
7     Remove the experts with zero capacity from  $\mathcal{V}$ .
8   else // The current set of available experts  $\mathcal{V}$  cannot cover  $T_i$ .
9     Leave  $T_i$  uncompleted.

```

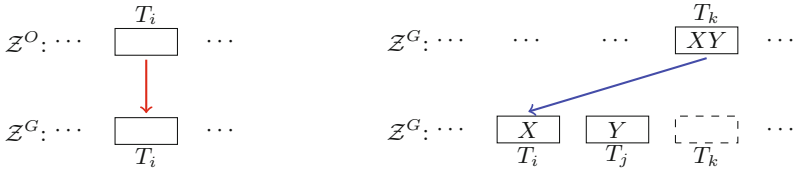
Theorem 3. GREEDY is k -competitive when the social compatibility property is hereditary.

We now introduce some more definitions and notations for the analysis. Fix an arbitrary input I . Let \mathcal{Z}^O and \mathcal{Z}^G be the set of tasks completed by OPT and GREEDY respectively. An expert is called a *common* expert if he/she is used by both OPT and GREEDY (but not necessarily for the same task).

To simplify the analysis, we will first modify the set of experts in the solutions by GREEDY and OPT. Consider the sequence of tasks T_1, T_2, \dots, T_n sorted in non-increasing order of profit. For each GREEDY’s team (and OPT’s team) and for each expert V , we create the i -th copy V^i of V with identical skill set to replace V if expert V is involved in GREEDY’s teams (and OPT’s teams) the i -th time. Clearly, each expert V has at most $c(V)$ copies of V . However, the social network \mathcal{G} remains unchanged. (So, the different copies of V are represented by the same vertex in \mathcal{G} .) In the analysis, we treat each copy as an expert with unit capacity while the number of experts may be increased to m' . Observe that any two copies of the same expert will not join the same minimal feasible team. Hence we still have $\Delta(I) \leq m$ (not m'). With this modification, we can describe the following charging schemes more easily.

3.1 A Simple Charging Scheme

The charging scheme maps each task in \mathcal{Z}^O to some task in \mathcal{Z}^G . For each task $T \in \mathcal{Z}^O$, if T is also present in \mathcal{Z}^G , we construct a pointer from T in \mathcal{Z}^O to T in \mathcal{Z}^G , meaning that we charge the profit of T in \mathcal{Z}^O to T in \mathcal{Z}^G . We call this a “task” pointer (or T -pointer for short). Otherwise T is not in \mathcal{Z}^G . This happens only if some expert in OPT’s team for T has been used by GREEDY for some task before T . In that case, let $T_a \in \mathcal{Z}^G$ be the earliest task in which GREEDY’s team shares some common expert with OPT’s team for T . (Thus T_a also has the largest profit among all tasks by GREEDY that shares common expert with OPT’s team for T .) We construct a pointer from $T \in \mathcal{Z}^O$ to $T_a \in \mathcal{Z}^G$. (So, we



(a) A T-pointer from $T_i \in \mathcal{Z}^O$ to $T_i \in \mathcal{Z}^G$. (b) An E-pointer from $T_k \in \mathcal{Z}^O$ to $T_i \in \mathcal{Z}^G$. T_i, T_j and T_k are in non-increasing order of their profit which implies $i < j < k$. The GREEDY's teams for T_i and T_j share the experts X and Y with the OPT's team for T_k , respectively. Note that $T_k \notin \mathcal{Z}^G$.

Fig. 1. Examples for T-pointer and E-pointer in the simple charging scheme.

charge the profit of T in \mathcal{Z}^O to T_a in \mathcal{Z}^G .) We call this an “expert” pointer (*E-pointer*). For convenience, we denote by (T_a, T_b) the pointer from task T_a to task T_b . Notice that each pointer (T_a, T_b) , whether a T-pointer or E-pointer, always points from a task T_a in \mathcal{Z}^O to a task T_b in \mathcal{Z}^G . See Fig. 1 for an example.

Lemma 1. *The above charging scheme has the following properties:*

- M1. *Every task T in \mathcal{Z}^G receives at most $\Delta(I) + 1$ pointers.*
- M2. *For every pointer from T_a to T_b , we have $p(T_a) \leq p(T_b)$.*

Proof. The first property M1 can be seen as follows. Clearly a task T in \mathcal{Z}^G can receive at most one T-pointer. By definition, any team chosen by GREEDY has at most $\Delta(I)$ experts. Hence, any task T in \mathcal{Z}^G has at most $\Delta(I)$ E-pointers.

The second property M2 is also obvious. It clearly holds for T-pointers. For an E-pointer (T_a, T_b) , notice that T_a is not completed by GREEDY. It must be the case that some expert in OPT's team for T_a has been used by GREEDY for some task before T_a . Therefore, $p(T_b) \geq p(T_a)$.

Theorem 1 directly follows from the above lemma.

3.2 A Modified Charging Scheme

We next show a modified charging scheme by adjusting the pointers carefully. This modified charging scheme shows that GREEDY is m -competitive. To explain in detail, we need more definitions:

Definition 6 (overloaded task). *A task $T \in \mathcal{Z}^G$ is overloaded if it has $m + 1$ pointers. Otherwise, it is said to be normal.*

We note that any task can have at most $m + 1$ pointers. It is obvious that each task can have at most one T-pointer. For E-pointer, we observe that each team is composed of at most m (not m') experts since any two copies of the same expert will not join the same minimal feasible team.

Definition 7 (universal task). A task $T \in \mathcal{Z}^O$ is universal if OPT's team for T possesses all possible skills, i.e., the union of all experts' skill sets.

Then we observe the relationship between an overloaded task and a universal task as follows:

Lemma 2. *If a task T is overloaded, T must be universal.*

Proof. In this proof, an “original expert” refers to an expert in the original input. Since T is overloaded, it must have one T-pointer and m E-pointers. This implies that $T \in \mathcal{Z}^O$ and GREEDY's team for T consists of m experts where the latter is due to that each E-pointer corresponds to an expert. Note that each expert in GREEDY's team for T corresponds to a distinct original expert. That is, GREEDY's team for T corresponds to the set of all m original experts. Recall that the oracle \mathcal{A} will return a minimal feasible team whenever a feasible team exists. By Definition 2, OPT's team for T must not correspond to a proper subset of the m original experts. In other words, OPT's team for T possesses all possible skills. This completes the proof.

We will repeatedly apply a procedure, which we called the Chaining Procedure, to convert all the overloaded tasks into normal ones. Some of its useful properties are stated in the lemma below. Details of the procedure and the proof of the lemma will be given in the next subsection.

Lemma 3. *When given an overloaded task T_a as input, the Chaining Procedure will locate another task T_b in \mathcal{Z}^G and replace one of the E-pointers to T_a (say (T_c, T_a)) by a pointer (T_c, T_b) to T_b such that*

- R1. T_b in \mathcal{Z}^G has at most m pointers after the change, and
- R2. $p(T_b) \geq p(T_c)$.

Moreover, no other pointers are affected.

By Lemma 3, it is clear that each application of the Chaining Procedure will reduce the number of overloaded tasks by one. Moreover, due to property R2 (Lemma 3) and property M2 (Lemma 1), we know that every pointer points from a task to another task with higher profit after each application of the Chaining Procedure. Thus by applying it sufficiently many times, each task $T \in \mathcal{Z}^G$ will have at most m pointers from tasks in \mathcal{Z}^O with profit no more than $p(T)$. Hence GREEDY is m -competitive and Theorem 2 follows.

3.3 The Chaining Procedure

Given an overloaded task T_a , the Chaining Procedure will take one or more iterations to locate a task T_b that can receive an extra pointer without becoming overloaded. Let $T_0 = T_a$. In the i -th iteration ($i \geq 1$), we begin with a task T_{i-1} , which has at least m pointers and thus has no room to receive an extra pointer. We try to locate a candidate task T_i in \mathcal{Z}^G as follows. We choose T_i to be the

earliest task in \mathcal{Z}^G that shares some common expert(s) with OPT's team for T_{i-1} . If T_i has at least m pointers, then T_i cannot receive any extra pointer. So we increase i by one and go to the next iteration. Otherwise, T_i has at most $m - 1$ pointers. In this case, T_i can serve as the required T_b . So, we replace an E-pointer (say, (T_c, T_a)) by a new pointer (T_c, T_i) and the Chaining Procedure ends. We call the newly created pointer an N-pointer. Note that only one N-pointer is installed in each application of the Chaining Procedure. We refer to the sequence of tasks, T_0, T_1, \dots, T_b , the *chain* involved in this application of the Chaining Procedure and say that each task T_{i-1} *links* to the next one T_i via one or more common experts between $T_{i-1} \in \mathcal{Z}^O$ and $T_i \in \mathcal{Z}^G$.

We now proceed to prove Lemma 3. To provide some intuition, consider the overloaded task T_0 . Lemma 2 shows that T_0 is also in \mathcal{Z}^O and universal. From this, we will be able to show that OPT's team for T_0 must share some experts with GREEDY's team for some task. Hence the Chaining Procedure must be able to locate T_1 . Now, if T_1 has m pointers, we will show that T_1 is also in \mathcal{Z}^O and universal. Hence we can locate T_2 , etc. The proof of universality uses a similar argument as in Lemma 2, which involves proving that GREEDY's team for T_1 consists of m experts. This is relatively straightforward when we only have E- and T-pointers. For subsequent applications of the Chaining Procedure, we need to deal with N-pointers as well and the argument becomes more complicated. To handle the complication, we introduce the following definitions:

Definition 8 (experts associated with E-pointers). *For each E-pointer (T_a, T_b) , we define the set of experts associated with this E-pointer to be the set of common experts between OPT's team for T_a and GREEDY's team for T_b .*

Definition 9 (linking experts). *For every application of the Chaining Procedure and for every $i \geq 1$, we define the set of linking experts between the two tasks T_{i-1} and T_i to be the set of common experts between them.*

Lemma 4. *For every application of the Chaining Procedure and every integer $i \geq 1$, the following properties hold at the beginning of the i -th iteration:*

- C1. *For any E-pointer (T_c, T_a) , we have $p(T_c) \leq p(T_{i-1})$.*
- C2. *The experts associated with the E-pointers and the linking experts are all distinct.*
- C3. *T_{i-1} is universal and has a T-pointer from T_{i-1} in \mathcal{Z}^O .*

We will prove Lemma 4 by induction on the total number of iterations accumulated over all applications of the Chaining Procedure.

(Base Case). At the beginning of the first iteration of the first application of the Chaining Procedure, properties C1 and C3 are true since $T_0 = T_a$ is overloaded. Property C2 is also trivially true.

(Induction Step). Assume C1, C2 and C3 are true at the beginning of the i -th iteration of the j -th application of the Chaining Procedure. The following lemmas show that C1, C2 and C3 remain true at the beginning of the next iteration, if exist.

Lemma 5. T_i as described in the Chaining Procedure is well-defined and for any E-pointer, (T_c, T_a) , we have $p(T_c) \leq p(T_i)$.

Proof. By property C3, T_{i-1} is universal. If OPT's team for T_{i-1} does not share any expert with GREEDY's teams for any other task T in \mathcal{Z}^G , then OPT's team for T_{i-1} , which can complete any task, is always available for GREEDY. Then GREEDY must have completed all input tasks. This contradicts the fact that T_a has an E-pointer (T_c, T_a) for some T_c in \mathcal{Z}^O (which implies that T_c is not completed by GREEDY). Hence the first part of the lemma follows.

To prove the second part of the lemma, we consider the following two cases.

Case (1): T_i comes before T_{i-1} . Then $p(T_{i-1}) \leq p(T_i)$. By C1, $p(T_c) \leq p(T_{i-1})$. Thus $p(T_c) \leq p(T_i)$.

Case (2): T_i comes after T_{i-1} . Note that T_{i-1} is universal and OPT's team for T_{i-1} is available for GREEDY until the first expert is used in T_i . Therefore, T_c must arrive no earlier than T_i (or else GREEDY would have completed T_c and there would have been a T-pointer (T_c, T_c) instead of an E-pointer (T_c, T_a)). Hence $p(T_c) \leq p(T_i)$.

Thus, if the j -th application of the Chaining Procedure goes to the $(i + 1)$ -st iteration, C1 continues to hold. Otherwise, if the j -th application ends in the i -th iteration, C1 is also true at the beginning of the first iteration of the $(j + 1)$ -st application of the Chaining Procedure due to the same reasoning mentioned in the base case.

Lemma 6. Any linking expert between $T_{i-1} \in \mathcal{Z}^O$ and $T_i \in \mathcal{Z}^G$ is distinct from any expert associated with the current E-pointers and any linking expert found so far in the current and previous applications of the Chaining Procedure.

Proof. Let V be a linking expert between $T_{i-1} \in \mathcal{Z}^O$ and $T_i \in \mathcal{Z}^G$. Clearly, V is distinct from any E-pointers because T_{i-1} has a T-pointer (which implies that T_{i-1} is completed by both OPT and GREEDY) and there cannot be an E-pointer from T_{i-1} to T_i .

Suppose to the contrary that expert V is also a previous linking expert between $T'_{j-1} \in \mathcal{Z}^O$ and $T'_j \in \mathcal{Z}^G$. (T'_{j-1} and T'_j can be in the chain of tasks, T'_0, T'_1, \dots , involved in a previous application of the Chaining Procedure or an earlier part of the chain, T_0, T_1, \dots , involved in the current application of the Chaining Procedure.) Note that V appears in only one task in \mathcal{Z}^G and only one task in \mathcal{Z}^O . Therefore, $T_{i-1} = T'_{j-1}$ and $T_i = T'_j$. We will show some contradictions in all possible cases.

Case (1): Both T_{i-1} and T'_{j-1} have preceding tasks T_{i-2} and T'_{j-2} in their respective chains. Then T_{i-1} has at least $m - 1$ E/N-pointers and a linking expert U between T_{i-2} and T_{i-1} . Note that each N-pointer to T_{i-1} is due to a previous chain that terminates at T_{i-1} . By C2, each such previous chain located T_{i-1} via one or more linking experts distinct from any previous linking experts. Note also that T'_{j-1} has the same set of E/N-pointers as T_{i-1} has. This is because T'_{j-1} has a preceding T'_{j-2} , T'_{j-1} has m pointers and hence no change is made on the pointers of T'_{j-1} by that (and any subsequent) application of the Chaining

Procedure. Since each task is completed by a team with at most m experts, we can deduce that U is both the linking expert between T_{i-2} and T_{i-1} and between T'_{j-2} and T'_{j-1} , contradicting C2.

Case (2): T_{i-1} is the beginning of its chain (i.e., $T_{i-1} = T_0$) and T'_{j-1} has preceding T'_{j-2} . Then T_{i-1} has m E-pointers. By C2, it has at least m distinct experts associated with these E-pointers. Hence it cannot have any linking expert and T'_{j-2} cannot link to T'_{j-1} (i.e., T_{i-1}) via a linking expert.

Case (3): T'_{j-1} is the beginning of its chain (i.e., $T'_{j-1} = T'_0$) and T_{i-1} has preceding T_{i-2} . If T'_{j-1} and T_{i-1} belong to the same chain, T'_{j-1} has exactly m E-pointers. It is obvious that T_{i-2} cannot link to T_{i-1} (i.e., T'_{j-1}) via a linking expert. Now consider the case that T'_{j-1} and T_{i-1} belong to different chains. So, T'_{j-1} is the beginning of a previous application of the Chaining Procedure. After completion of that Chaining Procedure, T'_{j-1} has $m - 1$ E-pointers and exactly one expert not associated with any of these E-pointers. This expert, say, U , was originally associated with the E-pointer that was replaced by an N-pointer. Thus T_{i-2} cannot link to T_{i-1} (i.e., T'_{j-1}).

Case (4): Both T_{i-1} and T'_{j-1} are the beginning. This case is impossible since after the previous application of the Chaining Procedure, T'_{j-1} (i.e., T_{i-1}) is no longer overloaded.

Therefore, V cannot be a previous linking expert and the lemma follows.

Note that Lemma 6 together with C2 implies that C2 is maintained in the next iteration (or the first iteration of the next application of the Chaining Procedure).

Lemma 7. *For integer $i \geq 1$, if $T_i \in \mathcal{Z}^G$ has at least m pointers, then T_i has a T-pointer and is universal.*

Proof. Note that T_i has at least $m - 1$ E/N-pointers since each task can have at most one T-pointer. Each E-pointer is associated with at least one expert. Each N-pointer is due to an application of the Chaining Procedure that terminates at T_i and hence can be associated with the corresponding set of linking experts. There is also at least one linking expert between $T_{i-1} \in \mathcal{Z}^O$ and $T_i \in \mathcal{Z}^G$ in the current application of the Chaining Procedure. By C2 and Lemma 6, all these experts are distinct. In other words, there are at least m experts in GREEDY's team. On the other hand, each task is completed by a minimal feasible team with at most m experts. Therefore, we conclude that GREEDY's team for T_i has exactly m experts; and that T_i has exactly $m - 1$ E/N-pointers and a T-pointer. Similar to the proof of Lemma 2, OPT's team for T_i also has m experts, each corresponding to a distinct expert in the original input. This implies that T_i is universal.

By Lemma 7, property C3 remains true in the next iteration. Again, if the current application of the Chaining Procedure ends in the i -th iteration, property C3 is true in the first iteration of the $(j + 1)$ -st application of the Chaining Procedure due to the same reasoning mentioned in the base case.

Notice that property C2 implies that in each new iteration, some experts are designated as linking experts. However, there can be at most m' linking experts. Hence each application of the Chaining Procedure will always terminate. Upon termination, T_a becomes a normal task while the last task in the chain still has at most m pointers. Thus, property R1 is guaranteed. By C1, property R2 is also guaranteed. This completes the proof of Lemma 4.

3.4 Hereditary Social Compatibility

We consider the class of hereditary social compatibilities. Our greedy algorithm can achieve a better approximation ratio (Theorem 3) by the critical observation of the following lemma:

Lemma 8. *For any hereditary social compatibility and for any input I , $\Delta(I) \leq k$.*

Proof. Recall that $\Delta(I)$ is the size of a largest minimal feasible team on input I . Suppose to the contrary that $\Delta(I) > k$ for some input I . We assume that a task T can be completed by a minimal feasible team \mathcal{V}' of size $\Delta(I)$. For each skill required in T , we pick a representative expert in team \mathcal{V}' that possesses this skill. Since T requires at most k skills, we can form a team $\mathcal{V}'' \subseteq \mathcal{V}'$ of size at most k that covers T . At the same time, team \mathcal{V}'' satisfies the social compatibility requirement due to the hereditary property. Thus \mathcal{V}'' is a feasible team for task T , contradicting the assumption that \mathcal{V}' is a minimal feasible team for task T . This completes the proof of Lemma 8.

Now we need to define the overloaded task a bit differently for the hereditary social compatibility.

Definition 10 (overloaded task for hereditary case). *Consider the hereditary social compatibility. A task $T \in \mathcal{Z}^G$ is overloaded if it has $k + 1$ pointers. Otherwise, it is said to be normal.*

Then we can obtain the following lemma, which is similar to Lemma 2.

Lemma 9. *Consider the hereditary social compatibility. If a task T is overloaded, T must be universal.*

Proof. Since T is overloaded for the hereditary social compatibility, it must have one T-pointer and k E-pointers by Lemma 8. This implies that $T \in \mathcal{Z}^O$ and GREEDY's team for T consists of k experts. Recall that the oracle \mathcal{A} will return a minimal feasible team whenever a feasible team exists. By Definition 2, each expert in GREEDY's team for T is a representative for a skill. Thus task T requires all possible skills and OPT's team for T possesses all possible skills.

Based on Lemmas 8 and 9, and an analogous proof of Lemma 4, Theorem 3 is thus complete.

4 Handling Task Multiplicity

We consider Tang's problem where each task T is also associated with a multiplicity $g(T)$, i.e., the maximum number that T can be completed. Our greedy algorithm can be easily adapted to solve Tang's problem in polynomial time. Specifically, for each task T in an instance of Tang's problem, we repeatedly find a minimal feasible team of experts \mathcal{V}' to complete task T $\min\{\min_{V \in \mathcal{V}'} c(V), g(T)\}$ times. Notice that for each task T , the greedy algorithm invokes the oracle at most $\min\{m, n\}$ times since each application of the oracle will result in either an expert's capacity being used up or T being completed $g(T)$ times. This also implies that in our algorithm, each task can be completed by at most m different minimal feasible teams. Thus the adapted algorithm runs in polynomial time. In the performance analysis, we create $g(T)$ copies of the same task T in the corresponding instance for our problem. This only increases the number of tasks and our greedy algorithm will achieve the same approximation ratio.

For the special case where $g(T)$ is infinite, i.e., task T can be completed infinitely many times, Tang's algorithm achieves an approximation ratio of $\beta \min\{\Delta(I), 2\sqrt{\sum_{V \in \mathcal{V}} c(V)/c(\min)}\}$. For our algorithm adaption, we observe that we only need to create m copies of the same task since each task can be completed by at most m different minimal feasible teams.

Tang [17, 18] also gave a worst-case lower bound of $\Omega(\log k)$ when the task multiplicity is infinite (and $m = k$). Here, we point out that a similar lower bound holds when the task multiplicity is bounded (instead of infinite). To see this, observe that a γ -approximation algorithm for the bounded case can be used to solve the unbounded case with the same ratio γ by creating m copies for each task. This is because each task can be completed by at most m different minimal feasible teams even for the unbound task multiplicity case. With this polynomial adaption, the lower bound for the unbounded case carries over to the bounded case.

5 Conclusion

In this paper, we study the Profit-aware Social Team Formation Problem and provide a simple greedy algorithm that improves upon previous results in many situations. There are a number of interesting open problems related to this problem, including the design of improved algorithms for this problem and the characterization of different definitions of social compatibility. Another direction is to consider the online version of the problem.

References

1. Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., Leonardi, S.: Power in unity: Forming teams in large-scale community systems. In: Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), pp. 599–608 (2010)

2. Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., Leonardi, S.: Online team formation in social networks. In: Proceedings of the International Conference on World Wide Web (WWW), pp. 839–848 (2012)
3. Carr, R., Vempala, S.: Randomized metarounding. *Random Struct. Algorithms* **20**(3), 343–352 (2002)
4. Charikar, M., Chekuri, C., Cheung, T.-Y., Dai, Z., Goel, A., Guha, S., Li, M.: Approximation algorithms for directed steiner problems. *J. Algorithms* **33**(1), 73–91 (1999)
5. Fortunato, S.: Community detection in graphs. *Phys. Rep.* **486**, 75–174 (2010)
6. Gajewar, A., Sarma, A.D.: Multi-skill collaborative teams based on densest subgraphs. In: Proceedings of the SIAM International Conference on Data Mining (SDM), pp. 165–176 (2012)
7. Golshan, B., Lappas, T., Terzi, E.: Profit-maximizing cluster hires. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 1196–1205 (2014)
8. Greenwald, R.: Freelancers find it pays to team up. *Wall Street J.* 3 February 2014. <https://www.wsj.com/articles/freelancers-find-it-pays-to-team-up-1389267711>
9. Jain, K., Mahdian, M., Salavatipour, M.R.: Packing steiner trees. In: Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 266–274 (2003)
10. Kargar, M., An, A.: Discovering top-k teams of experts with/without a leader in social networks. In: Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), pp. 985–994 (2011)
11. Lappas, T., Liu, K., Terzi, E.: Finding a team of experts in social networks. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 467–476 (2009)
12. Lee, V.E., Ruan, N., Jin, R., Aggarwal, C.: A survey of algorithms for dense subgraph discovery. In: Aggarwal, C., Wang, H. (eds.) *Managing and Mining Graph Data*. *Advances in Database Systems*, vol. 40, pp. 303–336. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-6045-0_10
13. Li, C.-T., Shan, M.-K., Lin, S.-D.: On team formation with expertise query in collaborative social networks. *Knowl. Inf. Syst.* **42**(2), 441–463 (2015)
14. Majumder, A., Datta, S., Naidu, K.V.M.: Capacitated team formation problem on social networks. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 1005–1013 (2012)
15. Rangapuram, S.S., Bühler, T., Hein, M.: Towards realistic team formation in social networks based on densest subgraphs. In: Proceedings of the International Conference on World Wide Web (WWW), pp. 1077–1088 (2013)
16. Rothvoß, T.: Directed steiner tree and the lasserre hierarchy. *CoRR*, abs/1111.5473 (2011)
17. Tang, S.: Profit-aware team grouping in social networks: a generalized cover decomposition approach. *CoRR*, abs/1605.03205 (2016)
18. Tang, S.: Profit-driven team grouping in social networks. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pp. 45–51 (2017)
19. Wang, X., Zhao, Z., Ng, W.: A comparative study of team formation in social networks. In: Renz, M., Shahabi, C., Zhou, X., Cheema, M.A. (eds.) *DASFAA 2015*. *LNCS*, vol. 9049, pp. 389–404. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18120-2_23
20. Wang, X., Zhao, Z., Ng, W.: USTF: a unified system of team formation. *IEEE Trans. Big Data* **2**(1), 70–84 (2016)