Efficient Bi-objective Team Formation in Social Networks

Mehdi Kargar, Aijun An, and Morteza Zihayat

Department of Computer Science and Engineering, York University
4700 Keele Street, Toronto, Ontario, Canada, M3J 1P3
{kargar,aan,zihayatm}@cse.yorku.ca

Abstract. We tackle the problem of finding a team of experts from a social network to complete a project that requires a set of skills. The social network is modeled as a graph. A node in the graph represents an expert and has a weight representing the monetary cost for using the expert service. Two nodes in the graph can be connected and the weight on the edge represents the communication cost between the two corresponding experts. Given a project, our objective is to find a team of experts that covers all the required skills and also minimizes the communication cost as well as the personnel cost of the project. To minimize both of the objectives, we define a new combined cost function which is based on the linear combination of the objectives (i.e. communication and personnel costs). We show that the problem of minimizing the combined cost function is an NP-hard problem. Thus, one approximation algorithm is proposed to solve the problem. The proposed approximation algorithm is bounded and the approximation ratio of the algorithm is proved in the paper. Three heuristic algorithms based on different intuitions are also proposed for solving the problem. Extensive experiments on real datasets demonstrate the effectiveness and scalability of the proposed algorithms.

1 Introduction

Team formation has been traditionally studied in operational research. Most of the traditional approaches do not consider the network structure behind the individuals. Nowadays, various forms of online social networks have been developed, making it possible to consider the relationships among individuals when forming a team to complete a task or project. Recently, some works have been devoted to find a team of experts from a social network that minimizes the cost of communication among the experts [11,10]. Such a team should possess a set of skills in order to complete the task or project. While effective communication is indeed important for the success of a project, there are some other factors that are also important but have not been considered in team formation from social networks. One of these factors is the personnel cost. In reality, people normally need to be paid for working on a project, and it is desirable to find a team of experts with a reasonable personnel cost.

In this paper, we tackle the problem of finding the best team of experts that possess a set of skills while minimizing both the communication cost and the personnel

P. Flach et al. (Eds.): ECML PKDD 2012, Part II, LNCS 7524, pp. 483–498, 2012.

cost. We use a graph to model a social network in which nodes represents experts, each having a set of skills and a cost for using his/her service. Two nodes in the graph can be connected and the weight on an edge represents the communication cost between the two connected experts. Generally, more previous collaborations between two experts, the lower the communication cost between them.

Clearly, our problem is a constrained bi-criteria optimization problem. A common approach to solving such a problem is to combine the two objectives into a single objective. In this paper, we take this approach and show that the problem of optimizing the combined objective function is NP-hard. Thus, we propose an approximation algorithm and three heuristic algorithms to efficiently solve the problem in polynomial time. The approximation algorithm is based on converting the input graph with both node and edge weights into a graph with weights on only edges. It has a performance guarantee with an approximation ratio of 2. The heuristic methods are based on either iteratively replacing cheapest experts with more expensive ones to improve the combined cost or incrementally adding experts with minimum cost contribution. We conduct extensive experiments on real data sets to show the effectiveness and efficiency of the proposed methods.

The paper is organized as follows. Related work is presented in Section 2. Problem statements and definitions are given in section 3. The algorithms for finding the best team of experts are presented in section 4. The experimental results are illustrated in section 5 and section 6 concludes the paper.

2 Related Work

Discovering a team of experts in a social network is introduced in [11]. The authors propose two functions for evaluating the communication cost among the members of a team. The communication cost functions are improved in [10]. The new function in [10] considers all the edges of the induced sub-graph, and is thus more stable to small or radical changes than the ones in [11]. The problem of finding a team of experts with a leader is also introduced in [10]. The problem is generalized by associating each required skill with a specific number of experts in [12]. The maximum load of the experts in the presence of several tasks is minimized in [1], but it does not consider finding teams with low communication cost. Recently, the problem of online team formation is studied in [2], which creates teams of experts with minimized work load and communication cost. The personnel cost of the experts is not considered in [2]. In this work, in addition to finding a team of experts with low communication cost, we also minimize the personnel cost of the team.

The problem of team formation has also been studied in the operation research community. Simulated annealing, branch and bound and genetic algorithms are used for solving the problem [4,13,14,7]. The main difference between this work and the works in operation research is that the experts are not connected through a social network in their work. The authors of [8], consider the effect of different graph structures among the members on the performance of the team. They performed their studies in an experimental setting and they do not study the problem from a computational point of view. Dynamics of group formation and its effect on the formation of groups in the network is considered in [3]. A game theoretic approach to this problem is discussed in [9]. Although these studies are not directly related to our setting, they might be considered as a complementary work for our problem.

The authors of [6] consider a bicriteria team formation problem. One objective is the communication cost and the other is the level of skills of the experts. It does not consider the personnel cost of the team. It produces a solution using a simulated annealing method and no approximation bound is given.

3 Problem Statement

Let $C = \{c_1, c_2, \ldots, c_m\}$ denote a set of m experts, and $S = \{s_1, s_2, \ldots, s_r\}$ denote a set of r skills. Each expert c_i has a set of skills, denoted as $Q(c_i)$, and $Q(c_i) \subseteq S$. If $s_j \in Q(c_i)$, expert c_i has skill s_j. In addition, a subset of experts $C' \subseteq C$ have skill s_j if at least one of them has s_j. For each skill s_j, the set of all experts having skill s_j is denoted as $C(s_j) = \{c_i | s_j \in Q(c_i)\}$. A project $P \subseteq S$ is defined as a set of skills which are required for the completion of the project. A subset of experts $C' \subseteq C$ is said to *cover* a project P if $\forall s_j \in P \; \exists \; c_i \in C', s_j \in Q(c_i)$.

The experts are connected together in a social network and it is modeled as an undirected and weighted graph (G). Each node in G represents an expert in C. Terms node and expert are used interchangeably through this paper. Two nodes are connected by an edge if the experts have collaborated before. The weight of an edge represents the communication cost between two experts. The lower the weight of the edge between two nodes, the more easily the two experts can collaborate or communicate, and the lower the communication cost between them. Each expert in the graph is also associated with a cost representing the monetary cost for using the expert service. The cost of an expert c_i is denoted as $t(c_i)$.

The **distance** between two nodes c_i and c_j, denoted as $d(c_i, c_j)$, is the sum of weights on the shortest path between them in G. It should be noted that the shortest distance function is a **metric** and satisfies the **triangle inequality**. If c_i and c_j are not connected in G (directly or indirectly), the distance between them is ∞.

Definition 1. ***(Team of Experts)*** *Given a set of experts C and a project P that requires a set of skills $\{s_1, s_2, \ldots, s_p\}$, a* team of experts *for P is a set of p skill-expert pairs: $\{\langle s_1, c_{s_1}\rangle, \langle s_2, c_{s_2}\rangle, \ldots, \langle s_p, c_{s_p}\rangle\}$, where c_{s_j} is an expert in C having skill s_j for $j = 1, \ldots, p$. A skill-expert pair $\langle s_i, c_{s_i}\rangle$ means that expert c_{s_i} is responsible for skill s_i in the project.*

Note that an expert may be responsible for more than one skill in a project. The same as our previous work [10], to evaluate the communication cost among the experts in a team T, we use the sum of distances among the experts of a team defined as follows.

Definition 2. ***(Sum of Distances)*** *Consider a graph G whose nodes represent experts and whose edges are weighted by the communication cost between two experts. Given a team T of experts from G for a project: $\{\langle s_1, c_{s_1}\rangle, \langle s_2, c_{s_2}\rangle, \ldots, \langle s_p, c_{s_p}\rangle\}$, the* sum of distances *of T with respect to G is defined as*

$$SD_G(T) = \sum_{i=1}^{p} \sum_{j=i+1}^{p} d(c_{s_i}, c_{s_j})$$

where $d(c_{s_i}, c_{s_j})$ is the distance between nodes c_{s_i} and c_{s_j} in G (as defined earlier).

To measure the personnel cost of a team, the following function is defined:

Definition 3. ***(Personnel Cost)*** *Consider a graph G whose nodes represent experts and each expert is associated with a cost. Given a team T of experts from G for a project: $\{\langle s_1, c_{s_1}\rangle, \langle s_2, c_{s_2}\rangle, \ldots, \langle s_p, c_{s_p}\rangle\}$, the* personnel cost *of T with respect to G is defined as*

$$PC_G(T) = \sum_{i=1}^{p} t(c_{s_i})$$

where $t(c_{s_i})$ is the cost of expert c_{s_i}.

In this cost model an expert x is paid $k \times t(x)$, where k is the number of skills the expert is responsible for in the project. This is reasonable because more skills the expert uses, more responsibility or tasks he/she has in the project.

We are interested in finding a team of experts that minimizes both the personnel and communication costs. Thus, our problem is a bi-objective optimization problem. One way to solve a bi-criteria optimization problem is to convert the problem into a single objective problem by combining the two objective functions into a single one. In this paper, we take this approach and define a single objective function that combines the communication and personnel costs with a tradeoff parameter (i.e., λ) as follows.

Definition 4. ***(Combined Cost Function)*** *Given a team T of experts from graph G for a project and a tradeoff λ between the communication and personnel costs, the combined cost of T with respect to G is defined as*

$$ComCost_G(T) = (p-1)(1-\lambda) \times PC_G(T) \ + \ 2\lambda \times SD_G(T)$$

where p is the number of required skills.

The parameter λ varies from 0 to 1 and indicates the tradeoff between the communication and personnel costs. Since the values of $PC_G(T)$ and $SD_G(T)$ may have different scales, $PC_G(T)$ and $SD_G(T)$ should be normalized before using the formula so that they both fall into the same range. The reason for having $(p-1)$ in the first term and 2 in the second is to scale up the terms because $PC_G(T)$ is the sum of p expert costs while $SD_G(T)$ is the sum of $\frac{p(p-1)}{2}$ pairwise communication costs. Given the combined cost function, we define the problem tackled in this paper as follows:

Problem 1. **(Team Formation by Minimizing the Combined Cost)** Given a project P, a graph G representing the social network of a set of experts and a tradeoff λ between the communication and personnel costs, the problem of team formation tackled in this paper is to find a team of experts T for P from G so that it covers all of the required skills and minimizes the combined cost function $ComCost_G(T)$ defined in 4.

Theorem 1. *Problem 1 is an NP-hard problem.*

Proof. Finding a team of experts from graph G while minimizing the sum of distances ($SD_G(T)$) is proved to be an NP-hard problem in [10]. Since $SD_G(T)$ is linearly related to $ComCost_G(T)$ (the objective function of Problem 1), then minimizing $ComCost_G(T)$ is also an NP-hard problem.

Since Problem 1 is an NP-hard problem, we have to rely on approximation or heuristic algorithms. Therefore, in the next section, we propose a 2-approximation algorithm and three heuristic algorithms for solving the problem.

4 Algorithms

4.1 Approximation Algorithm

In this section, we propose an approximation algorithm for solving Problem 1 with an approximation ratio of 2. The algorithm is based on converting the input graph (where the costs are associated with both nodes and edges) into a graph with costs/weights on only edges. The new graph G' has the same sets of nodes (experts) as the original graph G, but the node costs in G are moved onto edges in G'. In G', the edge weight between nodes u and v is defined as follows:

$$d'(u,v) = (1-\lambda)(t(u)+t(v)) \; + \; 2\lambda d(u,v) \tag{1}$$

where $t(x)$ is the cost of node/expert x in the original graph G, $d(u,v)$ is the shortest distance between experts u and v in G, and λ is the tradeoff between the communication and personnel costs.

Below we show that the *combined cost* of a team T of experts with respect to graph G is the same as the *Sum of Distances* of T with respect to the converted graph G'.

Lemma 1. *For any team T of experts from graph G for a project, the following holds:*

$$ComCost_G(T) = SD_{G'}(T)$$

where G' is converted from G by moving the node weights in G onto edges using Equation (1).

Proof. Let $T = \{\langle s_1, c_{s_1}\rangle, \langle s_2, c_{s_2}\rangle, \ldots, \langle s_p, c_{s_p}\rangle\}$. According to Definition 2,

$$SD_{G'}(T) = \sum_{i=1}^{p} \sum_{j=i+1}^{p} d'(c_{s_i}, c_{s_j})$$

where p is the number of required skills in the project. According to the definition of d', we have:

$$\begin{aligned} SD_{G'}(T) &= \sum_{i=1}^{p} \sum_{j=i+1}^{p} ((1-\lambda)(t(c_{s_i}) + t(c_{s_j})) + 2\lambda d(c_{s_i}, c_{s_j})) \\ &= (1-\lambda) \sum_{i=1}^{p} \sum_{j=i+1}^{p} (t(c_{s_i}) + t(c_{s_j})) + 2\lambda \sum_{i=1}^{p} \sum_{j=i+1}^{p} d(c_{s_i}, c_{s_j}) \\ &= (p-1)(1-\lambda) \sum_{i=1}^{p} t(c_{s_i}) + 2\lambda SD_G(T) \\ &= (p-1)(1-\lambda) PC(T) + 2\lambda SD_G(T) \\ &= ComCost_G(T) \end{aligned}$$

Based on the above lemma, finding a team of experts from graph G that minimizes the combined cost function (defined in Definition 4) is equivalent to finding a team of experts from graph G' that minimizes the *Sum of Distances* function (defined in Definition 2) based on d'. In [10], we proved that finding a team of experts while minimizing sum of distances is an NP-hard problem and proposed an approximation algorithm that finds a team of experts that minimizes the *Sum of Distances*. The approximation ratio of the algorithm is 2 as long as the pairwise distance function used in the *Sum of Distances* definition satisfies the triangle inequality. Below we show that function d' satisfies the triangle inequality.

Lemma 2. *The distance function $d'(u, v)$ defined in Equation (1) satisfies the triangle inequality.*

Proof. Function $d(u, v)$ in Equation (1) is the shortest distance between two nodes u and v in graph G. Since the shortest distance satisfies the triangle inequality, function d satisfies the triangle inequality. Thus, we have $d(a, b) \leq d(a, c) + d(c, b)$, where $\{a, b, c\}$ is an arbitrary triplet of nodes. Then, the following inequality holds since $0 \leq \lambda \leq 1$:

$$2\lambda d(a, b) \leq 2\lambda d(a, c) + 2\lambda d(c, b)$$

Thus,

$$\begin{aligned} (1-\lambda)(t(a) + t(b)) + 2\lambda d(a, b) &\leq (1-\lambda)(t(a) + t(b)) + 2\lambda d(a, c) + 2\lambda d(c, b) \\ &\leq 2(1-\lambda)t(c) + (1-\lambda)(t(a) + t(b)) + \\ &\quad 2\lambda d(a, c) + 2\lambda d(c, b) \end{aligned}$$

Based on the definition of d', the last inequality is equivalent to $d'(a, b) \leq d'(a, c) + d'(c, b)$. Since $\{a, b, c\}$ are chosen arbitrarily, d' satisfies the triangle inequality.

Since d' satisfies the triangle inequality, we can use the 2-approximation algorithm that we proposed in [10] to find a team T of experts in graph G' that minimizes the *Sum of Distances* function, $SD_{G'}(T)$. Based on Lemma 1, such a team also minimizes the combined cost, $ComCost_G(T)$, with respect to graph G. The pseudo-code of this algorithm is presented in Algorithm 1. The major difference between Algorithm 1 and the one in [10] is that Algorithm 1 takes two more inputs, namely, the expert costs t and the tradeoff λ between the communication and personnel costs, and uses distance function d' to compute the sum of distances of a team. The algorithm finds an approximate solution by using an expert ($e1$ in the pseudo-code) with a required skill as a seed in a team and adding its nearest expert with each of other required skills into the team. After checking all such teams, the team with the smallest sum of distances to the seed (calculated based on d') is returned as the best team. More explanation about the algorithm can be found in [10]. Note that two pre-built indexes are used as inputs to the algorithm. One is an inverted index for accessing $C(s_i)$, the set of experts in G having skill s_i. The other is a hash index that stores the shortest distances d of all pairs of experts in G. Both indexes can be pre-built because they are independent of the input project. The time complexity of this algorithm is $O(p^2 \times (C_{max})^2)$, where p is the number of required skills, and $C_{max} = \max_{1 \leq i \leq p} |C(s_i)|$ in which $|C(s_i)|$ is the cardinality of $C(s_i)$.

Theorem 2. *Algorithm 1 finds the team T of experts that minimizes $ComCost_G(T)$ with 2-approximation.*

Proof. In [10], we proved that Algorithm 1 is a 2-approximation algorithm for finding a team that minimizes the Sum of Distances. Since $SD_{G'}(T)$ based on d' is equivalent to $ComCost_G(T)$ according to Lemma 1, Algorithm 1 is a 2-approximation algorithm for finding a team of experts that minimizes the $ComCost_G(T)$.

4.2 Iterative Replace Algorithm

In this section, a heuristic algorithm is proposed for finding the best team of experts. The basic idea is as follows. Again, we use $C(s_i)$ to denote the set of experts that hold skill s_i. The algorithm consists of two phases. In the first phase, for each required skill s_i, the experts in $C(s_i)$ are sorted based on their cost in ascending order, and a team T is initialized by selecting the first expert in each $C(s_i)$ (i.e. the cheapest expert with the required skill s_i). This is the cheapest feasible team without consideration of the communication cost. In the second phase, each remaining expert in each $C(s_i)$ is tested to replace an expert in T which is currently assigned to skill s_i. If the replacement decreases the value of $ComCost_G(T)$, then the replace operation is applied permanently on T. The pseudo-code of the above procedure is presented in Algorithm 2. Note that $C(s_i)_j$ means the j-th expert in the sorted list of $C(s_i)$.

Since the algorithm needs to sort each $C(s_i)$, its run time is $O(C_{max}\log(C_{max}) + C_{max} \times p)$, where p is the number of required skills, and $C_{max} = \max_{1 \leq i \leq p} |C(s_i)|$. Note that if the experts in the initial team formed in the first phase are

Algorithm 1. The Approximation Algorithm for Finding Best Team

Input: project $P = \{s_1, s_2, \dots, s_p\}$, set $C(s_i)$ of experts in graph G with skill s_i for $i = 1, \dots, p$, distance d between each pair of nodes in G, cost t of each expert in G, and tradeoff λ between communication and personnel costs.

Output: the best team

```
$leastSumDistance \leftarrow \infty$
$bestTeam \leftarrow \emptyset$
for $i \leftarrow 1$ to $p$ do
    for each expert $e1 \in C(s_i)$ do
        $sumDistance \leftarrow 0$
        $T \leftarrow \{\langle s_i, e1 \rangle\}$
        for $j \leftarrow 1$ to $p$ and $j \neq i$ do
            $closestDistance = \infty$
            for each expert $e2 \in C(s_j)$ do
                $d'(e1, e2) \leftarrow (1 - \lambda)(t(e1) + t(e2)) + 2\lambda d(e1, e2)$
                if $d'(e1, e2) < closestDistance$ then
                    $closestDistance = d'(e1, e2)$
                    $closestNeighbor = e2$
            add $\langle s_j, closestNeighbor \rangle$ to $T$
            $sumDistance \leftarrow sumDistance + closestDistance$
        if $sumDistance < leastSumDistance$ then
            $leastSumDistance \leftarrow sumDistance$
            $bestTeam \leftarrow T$
return $bestTeam$
```

Algorithm 2. The Iterative Replace Algorithm for Finding Best Team

Input: project $P = \{s_1, s_2, \dots, s_p\}$, set $C(s_i)$ of experts in graph G with skill s_i for $i = 1, \dots, p$, distance d between each pair of nodes in G, cost t of each expert in G, and tradeoff λ between communication and personnel costs.

Output: the best team

```
for $i \leftarrow 1$ to $p$ do
    sort the experts in $C(s_i)$ based on their cost (i.e. $t$ function) in ascending order.
$T \leftarrow \emptyset$
for $i \leftarrow 1$ to $p$ do
    add $\langle s_i, C(s_i)_1 \rangle$ to the $T$
    $index_i \leftarrow 1$
while at least one $index_i$ has not reached $|C(s_i)|$ do
    for $i \leftarrow 1$ to $p$ do
        if $index_i < |C(s_i)|$ then
            $index_i \leftarrow index_i + 1$
            $T_{new} \leftarrow T$
            replace the holder of $s_i$ in $T_{new}$ with $C(s_i)_{index_i}$.
            if $ComCost_G(T_{new}) < ComCost_G(T)$ then
                $T \leftarrow T_{new}$
return $T$
```

disconnected from all the other experts with a required skill, no expert in the initial team will be replaced in the second phase, and thus the cheapest team is returned as the best team. However, the communication cost of the cheapest team can be very high (e.g., when any two members are disconnected from each other). Thus, when the graph is disconnected, this algorithm may return a poor team in terms of the combined cost. This problem can be alleviated by using the largest connected subgraph (i.e., the core of the graph) as the input data to the algorithm. Therefore, in our experiments, if the team produced by the algorithm is disconnected, we run the algorithm one more time on the core of the graph.

4.3 Minimal Cost Contribution Algorithm

In this section, another heuristic algorithm is proposed. The general structure of the algorithm is similar to Algorithm 1 in that they both seed a team with an expert with a required skill and add new members to the team to cover all the other required skills. The teams with different seeds are compared to select the best team. The difference is in how to expand the team with other experts. In Algorithm 1, the team was expanded by adding the nearest neighbor (according to the d' function) from each $C(s_j)$ where s_j is a required skill not covered by the seed expert. In that team expansion process, only the seed expert affects the addition of other members and the relationships among other experts in the team are not considered.

Algorithm 3. The Minimal Cost Contribution Algorithm for Finding Best Team Using Each of the Experts with a Required Skill as Initial Team Member

Input: project $P = \{s_1, s_2, \ldots, s_p\}$, set $C(s_i)$ of experts in graph G with skill s_i for $i = 1, \ldots, p$, distance d between each pair of nodes in G, cost t of each expert in G, and tradeoff λ between communication and personnel costs.

Output: the best team

```
$bestTeam \leftarrow$ NULL
$leastComCost \leftarrow \infty$
for $i \leftarrow 1$ to $p$ do
    for each expert $e_{initial} \in C(s_i)$ do
        $T \leftarrow \emptyset$
        add $\langle s_i, e_{initial} \rangle$ to $T$
        for $j \leftarrow 1$ to $p$ and $i \neq j$ do
            $leastMCC \leftarrow \infty$
            for each expert $e \in C(s_j)$ do
                Compute $MCC(e, T)$ using Formula (2)
                if $MCC(e, T) < leastMCC$ then
                    $leastMCC \leftarrow MCC(e, T)$
                    $expertWithLeastMCC \leftarrow e$
            add $\langle s_j, expertWithLeastMCC \rangle$ to $T$
        if $ComCost_G(T) < leastComCost$ then
            $leastComCost \leftarrow ComCost_G(T)$
            $bestTeam \leftarrow T$
return $bestTeam$
```

In the new algorithm proposed in this section, new members of a team are added incrementally and each new member is chosen by considering its communication costs with all the current members (not only the seed member) of the team in addition to the personnel cost of the new member. Assume that T is the current team of experts with some (but not all) of the required skills. To add into T an expert with an uncovered skill s_k, our new algorithm selects an expert e from the set $C(s_k)$ of experts with skill s_k that minimizes the following function:

$$MCC(e,T) = (1-\lambda)t(e) + \lambda\frac{\sum_{i=1}^{|T|} d(e,e_i)}{|T|} \tag{2}$$

where e_i is the expert responsible for the i-th skill in T and λ is the trade-off between communication and personnel costs. The first term of this function considers the personnel cost of expert e and the second considers the average communication cost between e and each of the current members of T. We refer to this function as Minimal Cost Contribution (MCC) function because the selected expert makes the minimal contribution to the total *ComCost* of the expanded new team compared to other experts being considered in $C(s_k)$.

The pseudo-code of the new algorithm, called the Minimal Cost Contribution (MCC) algorithm, is presented in Algorithm 3. The algorithm iterates through each expert with a required skill and uses the expert as the initial member of a candidate team T. For each candidate team T and each skill s_j uncovered by T, the algorithm incrementally selects and adds to T an expert e from $C(S_j)$ that minimizes the $MCC(e,T)$ function. The candidate team T is expanded in this way until all the required skills are covered by it. If the combined cost of T, $ComCost(T)$, is less than the least combined cost among all the previously generated candidates, T becomes the current best team (held in $bestTeam$). After all the candidate teams are generated (each with a different expert as the initial member), the team in $bestTeam$ is returned[1]. The time complexity of the MCC algorithm is $O(p^3 \times (C_{max})^2)$, where p is the number of required skills, and $C_{max} = \max_{1\leq i\leq p} |C(s_i)|$ in which $|C(s_i)|$ is the cardinality of $C(s_i)$.

[1] Note that the MCC algorithm is inspired by the Maximal Marginal Relevance (MMR) method [5] for increasing diversity in document retrieval results while maintaining high query relevance in the retrieved documents. The MMR method re-ranks the search results by using the most relevant document as the the first returned document and incrementally adding a new document to the result list by selecting a document that maximizes the MMR function which combines the relevance of the document and the dissimilarity of the document to the documents in the current result list. However, our MCC method is different from the MMR retrieval method in the following aspects. First, MMR generates the result with only one seed (i.e., the most relevant document) while MCC uses each expert with a required skill to initialize a candidate team and the final team is the one with the least *ComCost* among all the candidate teams. Second, the second term in the MMR function measures the maximal dissimilarity between the new document and the documents on the current result list, while MCC uses the average distance between the new team member and the current team members. We believe that the average distance better reflects the cost contribution of a new member to the team.

To improve the speed of the algorithm, we propose a variation of the MCC algorithm by using only an expert with the rarest required skill (i.e., the skill s whose $|C(s)|$ is the smallest among all the required skills) as the initial member of a candidate team. Thus, the number of candidate teams is reduced to the number of the skill holders of the the rarest required skill. We call this variation of the MCC algorithm MCC-*Rare*. Its time complexity is $O(p^2 \times C_{min} \times C_{max})$, where $C_{min} = \min_{1 \leq i \leq p} |C(s_i)|$.

5 Experimental Results

In this section, the proposed algorithms for finding a team of experts from a graph G which minimizes $ComCost_G(T)$ are evaluated. All the algorithms are implemented in Java. The experiments are conducted on an Intel(R) Core(TM) i7 2.80 GHz computer with 4 GB of RAM.

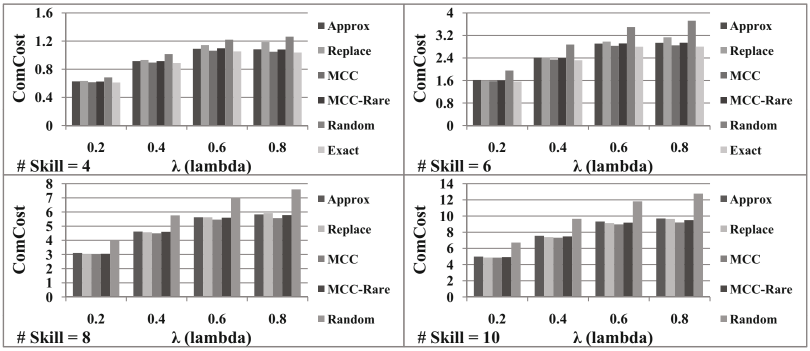


Fig. 1. The combined cost values of different methods on the DBLP dataset

5.1 The Datasets and Experimental Setup

To the best of our knowledge, real datasets that completely match our problem are not publicly available. Therefore, our proposed algorithms are evaluated on the datasets that were previously used in this domain [11,10]. The DBLP and IMDb data sets are used in the experiments. The DBLP XML data[2] is used for producing the DBLP graph. The dataset contains information about a set of papers and their authors. For each paper, the paper title, the author names, and the conference where it was published are specified. The same as in [11,10], only the papers of some major conferences in computer science are used for building the data graph, which include: SIGMOD, VLDB, ICDE, ICDT, EDBT, PODS, KDD, WWW , SDM, PKDD, ICDM, ICML, ECML, COLT, UAI, SODA, FOCS, STOC, and STACS. The set of experts and their skills are generated in the same way as in [11,10].

[2] http://dblp.uni-trier.de/xml/

The experts are the authors that have at least three papers in the DBLP. The skills of each expert is the set of keywords (terms) that appear in the titles of at least two papers of the expert. Two experts are connected together if they have at least two publications together. The weight of the edge between two experts n_i and n_j is equal to $1 - |\frac{p_{n_i} \cap p_{n_j}}{p_{n_i} \cup p_{n_j}}|$ where p_{n_i} is the set of papers of author n_i. The cost of an expert is set to be the number of publications of the expert. This is based on the assumption that the more publications an expert has, the more expertise the expert possesses, and thus the more expensive he/she is. The final graph has 5,658 nodes (experts) and 8,588 edges.

The same as [10], we use the part of the IMDb dataset which contains information about the actors and the list of movies that each actor played in[3]. It is preprocessed exactly the same as [10]. The communication cost between each pair of experts are also calculated the same as the DBLP dataset. Due to the space limit, more details are omitted. The cost of an expert is defined as the number of movies the actor plays in. The graph has 6,784 nodes and 35,875 edges. Due to the space limit, most of the results are only presented for the DBLP dataset. However, the results of the IMDb dataset show similar trend.

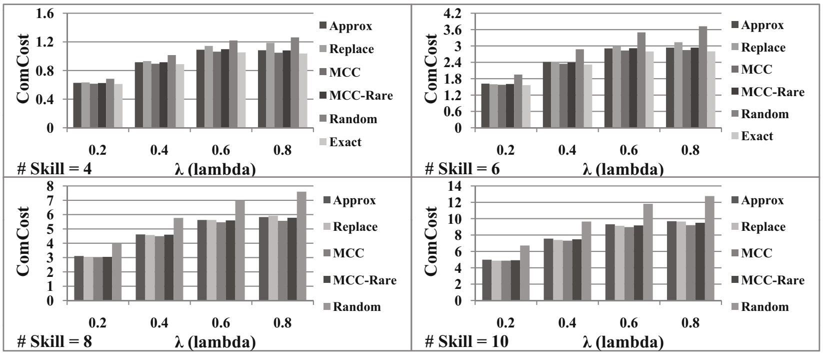
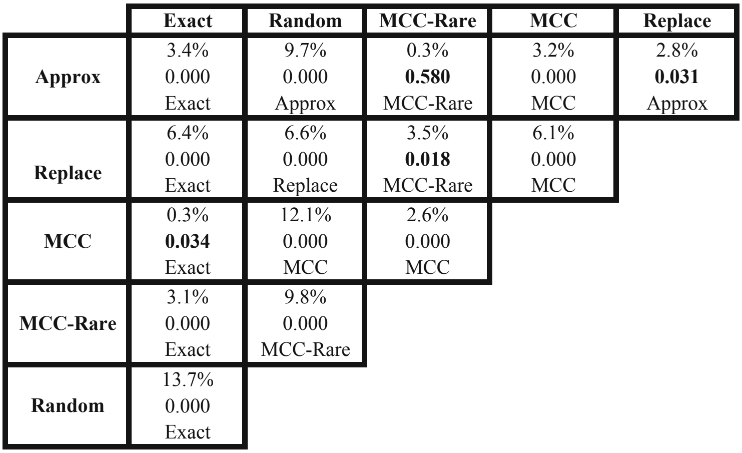


Fig. 2. The combined cost values of different methods on the IMDb dataset

The projects used in the experiments are generated as follows. We set the number of skills in a project to 4, 6, 8 or 10. For each number of skills, 50 sets of skills are generated randomly, corresponding to 50 random projects. The average result over the 50 projects for each number of skills is computed for each algorithm. As a baseline for the comparisons, we also include the results of a random method, which simply selects, among 10,000 random teams, the team with the lowest combined cost for the required set of skills. We also include the results of the *Exact* algorithm. It simply uses an exhaustive search to find the best answer among all possible teams.

[3] http://www.imdb.com/interfaces



	Exact	**Random**	**MCC-Rare**	**MCC**	**Replace**
Approx	3.4% 0.000 Exact	9.7% 0.000 Approx	0.3% **0.580** MCC-Rare	3.2% 0.000 MCC	2.8% **0.031** Approx
Replace	6.4% 0.000 Exact	6.6% 0.000 Replace	3.5% **0.018** MCC-Rare	6.1% 0.000 MCC	
MCC	0.3% **0.034** Exact	12.1% 0.000 MCC	2.6% 0.000 MCC		
MCC-Rare	3.1% 0.000 Exact	9.8% 0.000 MCC-Rare			
Random	13.7% 0.000 Exact				

Fig. 3. Results of t-test to show the level of difference between the algorithms on the DBLP dataset. The number of required skills is set to 4 and λ is set to 0.5.

5.2 Evaluation on Combined Cost

Figures 1 and 2 show the average combined cost values of teams for different algorithms for DBLP and IMDb datasets receptively. Please note that the results of the *Exact* algorithm is only provided for four and six skills. It is because by increasing the number of required skills, the number of possible teams grows exponentially. Thus, the *Exact* algorithm does not terminate in reasonable time for eight or more skills. The results show that all of the algorithms outperform the *Random* method. The results also suggest that the *MCC* method has the lowest cost values among non-exact methods. The results of *MCC-Rare* and *Approx* are very similar. In most of the cases, the *Replace* method has higher combined cost value than other proposed methods. *MCC-Rare* uses only an expert with the rarest required skill as the initial member of a candidate team. Therefore, *MCC-Rare* considers less number of candidate teams than *MCC* and thus its performance in terms of the combined cost is worse than the *MCC* algorithm.

To see whether the results of different algorithms are significantly different from each other, we run a t-test on each pair of methods. The results are shown in Figure 3. In each cell, the first number shows the percentage difference between the two methods (i.e., the absolute difference between the two values divided by the average of the two values). The second number is the p value from the t-test and the third row indicates which method has lower combined cost value (e.g. the *Exact* method always has the lowest combined cost.). In terms of percentage difference, the closest method to the *Exact* method is *MCC*. Their percentage difference is only 0.3%. Also, the p-values indicate that all pairs of methods are significantly different from each other except for *MCC-Rare* and *Approx*, which are not significantly different although *MCC-Rare* is slightly better.

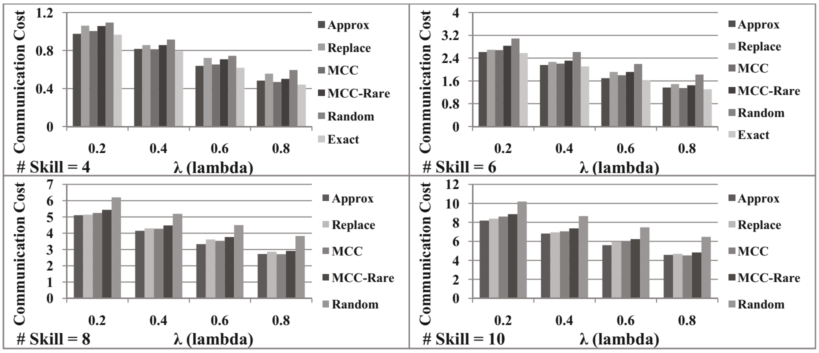


Fig. 4. The communication cost values of different methods on the DBLP dataset

5.3 Evaluation on Communication Cost

Figure 4 shows the average communication cost values of teams for different algorithms. The same as previous section, the results of the *Exact* algorithm is only provided for four and six skills. Please note that none of the algorithms explicitly minimize the communication cost. However, all of them implicitly minimize it by minimizing the combined cost function. The results suggest that the *Approx* algorithm has the lowest communication cost than among non-exact methods. In addition, for four skills, its results are very close to the one for the *Exact* method. Please note that by increasing the value of λ, the communication cost decreases. This is an expected result based on Definition 4.

5.4 Evaluation on Personnel Cost

Figure 5 shows the average personnel cost values of teams for different algorithms. As can be seen, *MCC-Rare* has the lowest personnel cost on all skill

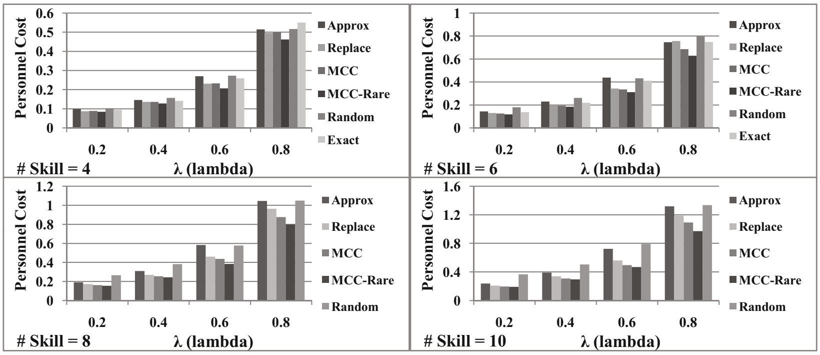


Fig. 5. The personnel cost values of different methods on the DBLP dataset

Table 1. Run time in milliseconds of different algorithms on DBLP. λ is set to 0.5.

# Req. Skills	Approx	Replace	MCC	MCC-Rare	Random	Exact
3	2	1	3	1	23	1,331
4	3	1	9	2	37	4,537
5	7	1	15	3	46	86,115
6	28	2	63	5	98	1,415,856

numbers and for all λ values, even lower than the exact algorithm. Note that the exact algorithm always finds the team with the lowest combined cost, which may not have the lowest personnel cost. The results also show that the *Random* method has the highest personnel cost,and the *Approx* algorithm has the second highest personnel cost. Please note that by increasing the value of λ, the personnel cost increases. This is also an expected result based on Definition 4.

5.5 The Run Time

Table 1 provides the run time of each method. It shows that the *Replace* algorithm is the fastest among others. *MCC-Rare* and *Approx* are the second and third respectively. *MCC* is the slowest among the 4 proposed methods, but still much faster than the *Random* method. The results also show the unreasonable run time of the *Exact* method and its inapplicability in practice.

6 Conclusions

We have proposed four algorithms for finding a team of experts in a social network that minimizes both the communication cost and the personnel cost of the team. The first algorithm is an approximation algorithm with a provable performance bound. The other three algorithms use heuristics to find sub-optimal solutions. Our experiments show that the *MCC* method has the lowest combined cost among the non-exact methods, but its run time is higher than other proposed heuristic or approximation algorithms. *MCC-Rare* reduces the run time of *MCC* and has the second lowest combined cost. The *Approx* algorithm has similar combined cost to *MCC-Rare* with a bit higher run time. The *Replace* method is the fastest but with the highest combined cost among the proposed methods. All the proposed methods are much faster than the *Random* and *Exact* methods. The *Random* method has the highest cost. The results indicate that the proposed methods are both effective and efficient.

References

1. Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., Leonardi., S.: Power in unity: Forming teams in large-scale community systems. In: Proc. of CIKM 2010 (2010)

2. Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., Leonardi, S.: Online team formation in social networks. In: Proc. of the WWW 2012 (2012)
3. Backstrom, L., Huttenlocher, D., Kleinberg, J., Lan, X.: Group formation in large social networks: membership, growth, and evolution. In: Proc. of KDD 2006 (2006)
4. Baykasoglu, A., Dereli, T., Das, S.: Project team selection using fuzzy optimization approach. Cybern. Syst. 38(2), 155–185 (2007)
5. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proc. of SIGIR 1998 (1998)
6. Dorn, C., Dustdar, S.: Composing near-optimal expert teams: A trade-off between skills and connectivity. In: Proc. of the International Conference on Cooperative Information Systems (2010)
7. Fitzpatrick, E.L., Askin, R.G.: Forming effective worker teams with multi functional skill requirements. Comput. Ind. Eng. 48(3), 593–608 (2005)
8. Gaston, M., Simmons, J., des Jardins, M.: Adapting network structures for efficient team formation. In: Proc. of the AAAI Fall Symposium on Artificial Multi-agent Learning (2004)
9. Jackson, M.: Network formation. The New Palgrave Dictionary of Economics and the Law (2008)
10. Kargar, M., An, A.: Discovering top-k teams of experts with/without a leader in social networks. In: Proc. of CIKM 2011 (2011)
11. Lappas, T., Liu, L., Terzi, E.: Finding a team of experts in social networks. In: Proc. of KDD 2009 (2009)
12. Li, C., Shan, M.: Team formation for generalized tasks in expertise social networks. In: Proc. of IEEE International Conference on Social Computing (2010)
13. Wi, H., Oh, S., Mun, J., Jung, M.: A team formation model based on knowledge and collaboration. Expert Syst. Appl. 36(5), 9121–9134 (2009)
14. Zakarian, A., Kusiak, A.: Forming teams: an analytical approach. IIE Transactions 31, 85–97 (2004)