Cluster Hire in a Network of Experts

Meet Patel and Mehdi Kargar
University of Windsor
Windsor, ON, Canada
{patel1g1, mkargar}@uwindsor.ca

ABSTRACT

Finding a group of experts is a natural way to perform a collection of tasks that needs a set of diversified skills. This can be done by assigning skills to different experts with complementary expertise. This allows organizations and big institutes to efficiently hire a group of experts with different skill sets to deliver a series of required tasks in order to finish a set of projects. We are given a collection of projects, in which each of them needs a set of required skills. Performing each project brings a profit to the organization. We are also given a set of experts, each of them are equipped with a set of skills. In order to hire an expert, the organization should provide her monetary cost (i.e., salary). Furthermore, we are given a certain amount of budget to hire experts. The goal is to hire a group of experts within the given budget to perform a subset of projects that maximizes the total profit. This problem is called Cluster Hire and was introduced recently. In this paper, we extend this problem by making the realistic assumption that there exist an underlying network among experts. This network is built based on past collaboration among experts. If two experts have past collaboration, they form a more collaborative and efficient team in the future. In addition of maximizing the total profit, we are also interested to find the most collaborative group of experts by minimizing the communication cost between them. We propose two greedy algorithms with different strategies to solve this problem. Extensive experiments on real dataset show our proposed algorithms are able to find a group of experts that cover projects with high profit while experts are able to communicate with each other efficiently.

1 INTRODUCTION

Organizations and big institutes often need to hire a group of experts in order to perform a set of predefined tasks. In this setting, they are interested to hire the most cost-effective group in which the profit of performing the tasks is maximized and the monetary costs (i.e., salary of experts) are minimized. Furthermore, organizations are interested to hire a group of experts in which they can collaborate effectively. Effective collaboration improves the quality of the work and results in faster completion of required tasks. For each set of required tasks, a certain amount of budget is assigned to recruit a set experts that are able to perform the tasks and perform a series of profitable projects.

For example, consider a big consulting company like Deloitte or PwC. A consulting company receives many projects to complete. Each project needs a set of required skills (e.g., programming or database design). Furthermore, upon the successful completion of the project, consulting company makes a profit. For each of the required tasks in each of the projects, the company needs to hire an expert to cover the required task. Furthermore, a certain amount of budget is assigned by the finance department to hire the experts. The goal of the consulting company is to hire a group of experts (i.e., consultants) in which the sum of their consulting rate (i.e., salary) is under the given budget. The company wants to maximize the sum of the total profit that it makes by completing the projects. In addition to this, assume that we have data about previous collaboration among experts (e.g., data from LinkedIn). The past collaboration between experts creates an underling graph (i.e., network) among them. For example, if two experts were working in the same project before, their associated nodes are connected to each other in the underling graph. The consulting company is interested to hire a group of experts in which they have past collaboration. This results in faster completion of the projects with higher quality. Furthermore, faster completion means the company receives the profit sooner.

Formally, we are interested in the problem of selecting a set of projects to complete by hiring a group of experts while maximizing the profit. However, for hiring the group of experts, we should not exceed the given budget. This problem is called Cluster Hire and is recently introduced by Golshan et. al. [9]. In the original Cluster Hire problem, authors ignore the underlying graph structure among experts. However, as we discuss before and is shown in previous work, considering the previous collaboration among experts and optimizing the communication cost among experts is an important aspect of having a successful team. The communication cost between two experts is defined based on the application need. It is not only limited to the past collaboration among experts. It might be the geometric distance between two experts if face-to-face meetings are necessary. In the underlying graph that model the communication cost among experts, the weight of the edge between any two experts determine the communication cost between them. The higher this value, the more expensive/time consuming the communication between them. This graph might be obtained from social networks (such as LinkedIn), scientific collaboration networks (such as DBLP), or other sources (such as IMDb for movie/actor data).

In this work, we tackle the problem of Cluster Hire in a network of experts and in addition of maximizing the profit, we also minimize the communication cost among experts. This turns the problem into a bi-objective optimization problem. We also make

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00

Table 1: Symbols used in this paper

E	set of n experts $\{e_1, e_2, \ldots, e_n\}$
S	set of m skills $\{s_1, s_2, \ldots, s_m\}$
G	input graph G that models the social network
P	set of k project $\{p_1, p_2, \ldots, p_k\}$
$ES(e)$	set of skills possessed by expert e
$C(e)$	cost of hiring expert e
$PS(p)$	set of required skills by project p
$PF(p)$	profit of completing project p
$Cap(e)$	capacity of expert e to offer her expertise
$Dist(e_i, e_j)$	distance between experts e_i and e_j in G
$CC(\mathcal{E})$	communication cost among a group of experts $\mathcal{E}$
$Profit(\mathcal{P})$	profit of performing a group of projects $\mathcal{P}$
B	total budget for hiring experts
$ProfitCC(\mathcal{P}, \mathcal{E})$	combined objective of profit and communication
$Skill(e, p)$	number of skills in p covered by expert e

other refinements to the original Cluster Hire problem and the algorithms that are introduced in [9].

2 PROBLEM STATEMENT

Let $E = \{e_1, e_2, \ldots, e_n\}$ determines a set of n experts, and $S = \{s_1, s_2, \ldots, s_m\}$ determines a set of m skills (all symbols used in this paper are summarized in Table 1). Each expert e posses a set of skills, which is denoted as $ES(e)$. Clearly, $\forall\ e \in E, ES(e) \subseteq S$. Each expert e demands a monetary cost (i.e., salary), to participate in performing different tasks. This is shown by $C(e)$ and is measured by dollar value. We also have a set of given projects which is denoted by $P = \{p_1, p_2, \ldots, p_k\}$. Each project is also composed of a set of required skills that need to be covered by experts in order for the project to be completed. This set is shown by $PS(p_j)$ for project p. Again, $\forall\ p \in P, PS(p) \subseteq S$. We assume each expert e is able to offer her expertise at most $Cap(e)$ times. This is a reasonable assumption since we do not want to overload an expert by assigning her to many projects. Therefore, each expert has a maximum capacity to participate in a number of tasks. We take this constraint in consideration when designing the algorithms.

Definition 2.1. **Group of Experts:** Given a set of n experts E, a set of m skills S, and a set of k projects P, a group of experts $\mathcal{E} \subseteq E$ is able to complete a subset of projects $\mathcal{P} \subseteq P$ if the following holds:

Coverage: $\forall\ p \in \mathcal{P}$ and $\forall\ s \in PS(p)$, an expert e in $\mathcal{E}$ is assigned to perform the required skill s.

Capacity: $\forall\ e \in \mathcal{E}$, e is not covering more that $Cap(e)$ skills.

The experts are connected together in a network which is modeled as an undirected graph G. Each experts e_i is associated to a node in G. We use terms node and expert interchangeably in this work. Two experts are connected by an edge in G if they have prior collaboration in the past (e.g., participating in the same project). Graph G might be weighted. In this case, the edge weight represents the strength of the collaboration between them (the smaller the edge weight, the stronger the prior collaboration). For example, if two experts participated in ten projects in the past, the edge weight between them is smaller than two experts that collaborated in only two projects in the past. When two experts are not directly connected, we use the weight of the shortest path between them in G to determine the communication cost between them. Again, the smaller the weight of the shortest path, the closer the two experts to each other. This is a reasonable assumption since two experts e_i and e_j might be introduced to each other via a middle expert e_k in which e_i and e_j worked with e_k in the past. This is certainly preferred over another case in which the two experts e_i and e_j has no expert in common. The communication cost between two experts e_i and e_j is shown as $Dist(e_i, e_j)$, i.e., the weight of the shortest path between e_i and e_j in G. We are interested to choose a group of experts in which the communication cost between them is minimized. We model the communication cost between a group of experts as the sum of distances between each pair of the experts in the group.

Definition 2.2. **Communication Cost:** Given a group of experts $\mathcal{E}$ and an underling graph G that determines the past collaboration among experts, the communication cost among experts in $\mathcal{E}$ is defined as follows:

$$CC(\mathcal{E}) = \sum_{i=1}^{|\mathcal{E}|} \sum_{j=i+1}^{|\mathcal{E}|} Dist(e_i, e_j)$$

Finishing each project brings a profit in dollar value which is shown by $PF(p)$ for project p. We are interested to choose a set of projects in which the sum of their profit is maximized.

Definition 2.3. **Profit of Projects:** Given a set of projects $\mathcal{P}$, the profit of completing these projects is defined as follows:

$$Profit(\mathcal{P}) = \sum_{p \in \mathcal{P}} PF(p)$$

For performing a subset of given projects, we are given a predetermined budget (also in dollar value) denoted as B. This budget is spent on hiring experts. Our goal is to hire as many experts as possible in which the sum of their hiring costs (i.e., salary) is under the given budget B. Since we are interested to maximize the profit and minimize the communication cost, our problem is a bi-objective optimization problem.

A common approach to solve a bi-objective optimization problem is to convert it into a single objective problem. This can be done by introducing a tradeoff parameter λ that varies between 0 and 1 and determines whether we want to put more weight towards profit or communication cost. Furthermore, since one of the objectives is a maximization problem (maximizing the profit) and the other one is a minimization problem (minimizing the communication cost), we have to modify one of them and make both of them to be of the same type. Therefore, we maximize the reverse of the communication cost. Now we are ready to formally define the problem we tackle in this work.

Problem 1. *Given a set of n experts E, a set of m skills S, a set of k projects P, a tradeoff λ between the profit and communication cost, and an underling graph G that determines the past collaboration among experts, we are interested to choose a group of experts $\mathcal{E} \subseteq E$*

(according to Definition 2.1) and a set of projects $\mathcal{P} \subseteq S$ in which the following objective is maximized:

$$ProfitCC(\mathcal{P}, \mathcal{E}) = (\lambda).Profit(\mathcal{P}) + (1-\lambda).\frac{1}{CC(\mathcal{E})}$$

Furthermore, the following budget constraint must be satisfied:

$$\sum_{e \in \mathcal{E}} C(e) \leq B$$

Note that since the dollar values of the project's profit and the distance between experts may have different scales, both of these values should be normalized before using the above objective so that they both fall into the same range (e.g., all of them fall between 0 and 1).

Theorem 2.4. *Problem 1 is NP-hard.*

Proof. Finding a group of experts to cover a set of projects $\mathcal{P} \subseteq P$ and maximizing $Profit(\mathcal{P})$ under the given budget B is proved to an NP-hard problem in [9]. Since the objective of Problem 1 is linearly related to $Profit(\mathcal{P})$, then optimizing Problem 1 is also an NP-hard problem. □

3 ALGORITHMS

In this section, we propose two different algorithms based on different strategies to find a group of experts while maximizing the profit and minimizing the communication cost. The first strategy picks an expert in each iteration and assigns her to some of the projects. The second strategy picks a project in each iteration and finds the best group of experts to finish that specific project.

3.1 Expert Pick Strategy

Since finding the best group of experts to cover a subset of projects while maximizing the profit and minimizing the communication cost is an NP-hard problem, here we propose the first greedy algorithm to find a group of experts while covering a subset of projects with high profit. In the first algorithm and in each iteration, we greedily pick one expert and add her to the pool of existing experts. We also check to see if adding her will cover any of the remaining projects. One important challenge is to make sure adding a new expert (with her salary) does not exceed the given budget B. In each iteration, we assign a score to each pair of expert/projects and choose the expert with highest score. The score is designed based on the following intuitions:

- We want to choose a *cheap expert* so that we do not overspend the budget on a single expensive expert.
- We want to choose an expert that covers many required skills for a *high profitable* project.
- We want to choose an expert that is able to *communicate effectively* with other experts that are already selected.

As one might notice, these intuitions are not necessarily compatible. A cheap expert may not be able to cover many skills from a profitable project or she may not have many past collaboration with other experts. In order to take into account all these objectives, we design the following score function for each pair of projects and experts.

Algorithm 1 Cluster Hire with Expert Pick Strategy

Input: set of n experts $E = \{e_1, e_2, \ldots, e_n\}$, set of m skills $S = \{s_1, s_2, \ldots, s_m\}$, set of k projects $P = \{p_1, p_2, \ldots, p_k\}$, graph G that models the network of experts, tradeoff parameter λ, and budget B.
Output: subset of projects $\mathcal{P} \subseteq P$ and a group of experts $\mathcal{E} \subseteq E$ that maximize $ProfitCC(\mathcal{P}, \mathcal{E})$ under the given budget B.

$\mathcal{E} \leftarrow \emptyset, \mathcal{P} \leftarrow \emptyset, b \leftarrow 0$
while $b < B$ and $E/\mathcal{E} \neq \emptyset$ **do**
  $\mathcal{R} \leftarrow \{e \mid e \in E \text{ and } e \notin \mathcal{E} \text{ and } C(e) + b \leq B\}$
  for all $e \in \mathcal{R}$ **do**
    if e does not cover any required skills in $P/\mathcal{P}$ **then**
      remove e from $\mathcal{R}$
  if $\mathcal{R} = \emptyset$ **then**
    return $\mathcal{E}$ and $\mathcal{P}$
  for all $p \in P/\mathcal{P}$ **do**
    for all $e \in \mathcal{R}$ **do**
      if e covers at least one skill in p **then**
        if $\mathcal{E} = \emptyset$ **then**
          $sc_e^p \leftarrow \frac{PF(p).\min\{Skill(e,p),\ Cap(e)\}}{C(e)}$
        else
          $sc_e^p \leftarrow \lambda.\frac{PF(p).\min\{Skill(e,p),\ Cap(e)\}}{C(e)} + (1-\lambda).\frac{1}{\sum_{e' \in \mathcal{E}} Dist(e,e')}$
      else
        $sc_e^p \leftarrow 0$
  $(e,p) \leftarrow \arg\max_{e \in \mathcal{R},\ p \in P/\mathcal{P}}\ sc_e^p$
  add e to $\mathcal{E}$
  assign skills of e to p based on rarest skill strategy
  update $Cap(e)$
  update $PS(p)$
  $b \leftarrow b + C(e)$
  if $|PS(p)| = 0$ **then**
    add p to $\mathcal{P}$
  while $Cap(e) > 0$ **do**
    $p' \leftarrow \arg\max_{p \in P/\mathcal{P}}\ score_e^p$
    assign skills of e to p' based on rarest skill strategy
    update $PS(p')$ according to $ES(e)$
    update $Cap(e)$
    if $|PS(p')| = 0$ **then**
      add p' to $\mathcal{P}$
return $\mathcal{E}$ and $\mathcal{P}$

$$sc_e^p \leftarrow \lambda.\frac{PF(p).\min\{Skill(e,p),\ Cap(e)\}}{C(e)} + (1-\lambda).\frac{1}{\sum_{e' \in \mathcal{E}} Dist(e,e')} \quad (1)$$

Recall that λ is the tradeoff parameter between profit and communication cost (see Definition 2.2). Note that $Skill(e,p)$ determines the number of skills in p that could be covered by expert e. The first part of the above equation chooses a pair of expert/project in which the project p has high profit and the expert e covers as many skills as possible in p. This number is divided by the cost of expert e to ensure we take into account the salary of expert e. Between the

number of skills that expert e can cover in p and the capacity of e, we choose the minimum value. This is because we do not want to violate the capacity of expert e and overload her with many tasks. For example, if an expert is able to cover 5 skills in a project, but her capacity is only 3, we use 3 in the above equation to make sure if she is the selected expert with that project, she is only assigned to 3 skills and not 5. The next part of this equation maximizes the reverse communication cost between expert e and other experts already in the group. Note that in the first iteration, we do not take into account the communication cost between experts since the group is empty.

Algorithm 1 is our solution to Problem 1 to find a group of collaborate experts while maximizing the profit of covered projects. This algorithm receives the set of n experts, set of m skills, set of k projects, network of experts G, the tradeoff parameter λ, and the available budget B as input. The output of this algorithm is a subset of projects $\mathcal{P}$ and a group of experts $\mathcal{E}$ that covers all required skills in $\mathcal{P}$ while maximizing the objective of Problem 1. Furthermore, the sum of the salary of the experts in $\mathcal{E}$ is not more than the given budget B.

In line 1, $\mathcal{E}$ and $\mathcal{P}$ are initialized to $\emptyset$. Also b is set to 0. We store the amount of money that we have spent so far in b. We can keep adding experts to $\mathcal{E}$ as long as $b < B$. Line 2 starts the iteration of the greedy algorithm. As long as we have experts in E that has not been added to $\mathcal{E}$ and we have not over spent the budget, we can consider adding more experts to $\mathcal{E}$. This is the condition of the while loop in line 2. In line 3, we store all unassigned experts in E in which their salary is within the remaining budget to $\mathcal{R}$. The for loop in line 4 checks to see each expert in $\mathcal{R}$ satisfy at least one required skill in an uncovered project. If this is not the case, the expert is removed from $\mathcal{R}$ as these experts are useless for the remaining projects. In line 7, we check to see if $\mathcal{R}$ is empty or not. If it is empty, we return the current $\mathcal{E}$ and $\mathcal{P}$ and terminate the algorithm. The reason is, if $\mathcal{R}$ is empty, we have no other experts to add to $\mathcal{E}$. In lines 9 to 17, we assign a score to each pair of uncovered project p (projects in $P/\mathcal{P}$) and expert e in $\mathcal{R}$. Later, we choose the highest score and add the associated expert to $\mathcal{E}$. If e does not cover any of the required skills in p (line 11), the score is set to 0 (line 17) as expert e is useless for project p. If e covers at least one of the required skills in p, then we calculate the score of e and p according to Equation 1. Note that in the first iteration (line 12, $\mathcal{E} = \emptyset$), we do not consider the communication cost as set $\mathcal{E}$ is empty. In line 18, we choose pair (e, p) in which their score sc_e^p is maximized among all other pairs. e is added to $\mathcal{E}$ in line 19. Then, the skills of e are assigned to p in line 20. Note that if the capacity of e is smaller than the required number of skills in p, we assign the rarest skills in e first. We then update the capacity of e, the required skills in p (i.e., $PS(p)$), and the value of b. If all of the required skills in p are covered (line 24), p is added to $\mathcal{P}$ (line 25).

One advantage of our proposed algorithm is that if an expert e is added to the group of experts, we try to use her maximum capacity as she will get paid the same amount of salary regardless of the number of skills she covers in different projects. Based on this strategy, after expert e is selected to be added to $\mathcal{E}$ in current iteration, we assign her remaining capacity to other projects in lines 26 to 32. As long as her capacity is larger than zero (line 26),

Algorithm 2 Cluster Hire with Project Pick Strategy

Input: set of n experts $E = \{e_1, e_2, \ldots, e_n\}$, set of m skills $S = \{s_1, s_2, \ldots, s_m\}$, set of k projects $P = \{p_1, p_2, \ldots, p_k\}$, graph G that models the network of experts, tradeoff parameter λ, and budget B.
Output: subset of projects $\mathcal{P} \subseteq P$ and a group of experts $\mathcal{E} \subseteq E$ that maximize $ProfitCC(\mathcal{P}, \mathcal{E})$ under the given budget B.

1: $\mathcal{E} \leftarrow \emptyset, \mathcal{P} \leftarrow \emptyset, b \leftarrow 0$
2: **while** $b < B$ and $P/\mathcal{P} \neq \emptyset$ **do**
3: $P' \leftarrow P/\mathcal{P}, \mathcal{R} \leftarrow \{e \mid e \in E$ and $e \notin \mathcal{E}$ and $C(e) + b \leq B\}$
4: **if** $\mathcal{R} = \emptyset$ **then**
5: **return** $\mathcal{E}$ and $\mathcal{P}$
6: **for all** $p \in P'$ **do**
7: $E_p \leftarrow \emptyset, S_p \leftarrow PS(p), \mathcal{R}' \leftarrow \mathcal{R}$
8: **while** $S_p \neq \emptyset$ **do**
9: **for all** $e \in \mathcal{R}'$ **do**
10: **if** e covers at least one skill in p **then**
11: **if** $E_p = \emptyset$ **then**
12: $sc_e \leftarrow \frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)}$
13: **else**
14: $sc_e \leftarrow \lambda . \frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)} + (1-\lambda) . \frac{1}{\sum_{e' \in E_p} Dist(e,e')}$
15: **else**
16: $sc_e \leftarrow 0$
17: $e \leftarrow \arg\max_{e \in \mathcal{R}'}\ sc_e$
18: **add** e to E_p, **update** S_p
19: **for all** $p \in P'$ **do**
20: **if** $(\sum_{e \in E_p} C(e)) + b > B$ **then**
21: **remove** p from P'
22: **if** $P' = \emptyset$ **then**
23: **return** $\mathcal{E}$ and $\mathcal{P}$
24: $(p, E_p) \leftarrow \arg\max_{p \in P'}\ \lambda . \frac{PF(p)}{\sum_{e \in E_p} C(e)} + (1-\lambda) . \frac{1}{\sum_{e \in E_p} \sum_{e' \in \mathcal{E}} Dist(e,e')}$
25: **add** p to $\mathcal{P}$, **assign** skills of experts in E_p to p
26: **for all** $e \in E_p$ **do**
27: **add** e to $\mathcal{E}$, **update** $Cap(e)$, $b \leftarrow b + C(e)$
28: **while** $Cap(e) > 0$ **do**
29: $s \leftarrow$ rarest skill in e which is required by a p in $P/\mathcal{P}$
30: **assign** skill s to the most expensive p in $P/\mathcal{P}$
31: **update** $Cap(e)$
32: **return** $\mathcal{E}$ and $\mathcal{P}$

we find a project p' that maximizes the expert/project score when the expert e is fixed (line 27). We then assign the skills of e to p' and update $PS(p')$ and the capacity of e. If all required skills of p' are covered, it will be added to $\mathcal{P}$.

3.2 Project Pick Strategy

The second algorithm to find a group of collaborative experts to cover the most profitable set of projects is designed based on the idea of selecting a project in each iteration. In each iteration, we assign a score to each uncovered project and choose the one with

the highest score to be added to the pool of projects. The score of each project is designed based on the following intuitions:

- We want to choose a project with *high profit*.
- The set of experts responsible to cover the required skills in the project should be *cheap*.
- The set of experts that cover the skills of the project should be able to *communicate effectively* with each other and with existing group of experts.

The same as the first strategy, these intuitions are not necessarily compatible. A high profitable project might need expensive set of experts and/or non-collaborative set. We design the scoring function that takes into account a combination of all these objectives. In each iteration and for each uncovered project p, we find a set of experts E_p to cover the required skills of p. In order to do that, we use a modified version of the greedy weighted set cover algorithm. Recall that in greedy set cover, we are given a collection of sets (corresponding to the set of skills of each expert) in which each set is associated with a cost (corresponding to the salary of the expert in our problem). The goal is to choose a subset of sets to cover a given union set (corresponding to the set of skills required for a given project in our problem). In greedy weighted set cover algorithm, in each iteration, a set that maximizes the number of covered elements divided by the cost of the set, is selected. In other words, the algorithm selects a set, in which the price of covering a single element is minimized. We also add the communication cost to the price per skill when selecting the next expert to cover a given project. Formally, in each iteration, and for any remaining project, we find a set of experts that are able to cover that project. To find this set for project p, we start with a an empty set E_p. Then, we select an expert to be added to E_p that maximizes the following equation:

$$sc_e \leftarrow \lambda.\frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)} + (1-\lambda).\frac{1}{\sum_{e' \in E_p} Dist(e,e')} \quad (2)$$

Recall that λ is the tradeoff parameter. The first part of this equation is taken from the greedy set cover algorithm with a slight modification that takes the capacity of the expert into account. The second part of it evaluates the communication cost of adding a new expert to E_p. If this is the first expert to be chosen (i.e., $E_p = \emptyset$), we do not consider the communication cost. After finding the set of experts E_p for all uncovered projects, we select one of the projects as the winner and add it to the pool of already selected projects. In order to to that, we select a project that maximizes the following equation.

$$\lambda.\frac{PF(p)}{\sum_{e \in E_p} C(e)} + (1-\lambda).\frac{1}{\sum_{e \in E_p}\sum_{e' \in \mathcal{E}} Dist(e,e')} \quad (3)$$

The intuition is the same, we are interested to choose the project that has a high profit, needs experts with low salary, and the set of experts responsible to perform the project's tasks has small communication cost with existing experts. Now, we are ready to present Algorithm 2 that returns a group of experts for performing a set of profitable projects with project pick strategy. The input and output of this algorithm is similar to Algorithm 1. In the first line, we initialize three variables $\mathcal{E}$, $\mathcal{P}$, and b, which are responsible to store the final group of experts, the selected projects, and amount of budget spent so far, respectively. The while loop of line 2 iterates until we run out of budget or no project is left to be covered. In line 3, we first assign all uncovered projects to set P'. We then put all of the experts in which adding them to $\mathcal{E}$ will not violate the budget to set $\mathcal{R}$. If $\mathcal{R}$ is empty, we terminate the algorithm in line 5 as we cannot proceed further and cover any more projects. The for loop of line 6 starts the process of assigns a score to each project p in P'. As we discuss before, the score function is a modification of the weighted greedy set cover algorithm that also takes into account the capacity of experts and communication cost. Sets E_p, S_p, and $\mathcal{R}'$ are initialized in line 7. E_p stores the set of experts to perform p. S_p is a duplicate of the set of required skills in p, in which we try to cover them by adding experts to E_p. $\mathcal{R}'$ is a duplicate of $\mathcal{R}$. The while loop of line 8 is executed until no more skill is required by p (i.e., S_p becomes $\emptyset$). Line 9 iterates over all experts in $\mathcal{R}'$ and each iteration chooses the one that maximizes the modified weighted greedy set cover score. This expert is selected in line 17 and is added to E_p in line 18. We also update S_p in line 18. After finding the set of experts for all projects, in lines 19 to 21, we remove the projects in which adding their associated expert set to $\mathcal{E}$ violate the budget constraint. If set P' becomes empty after this operation, we terminate the algorithm in line 23. If P' is not empty, we selects the best project p in P' in line 24 according to equation 3. In line 25, we add the best project to $\mathcal{P}$, and cover the skills of p. The for loop of line 26 iterates through all experts in E_p. These are the set of experts that are responsible to cover the best selected project in line 24. In line 27, each expert in E_p is added to $\mathcal{E}$ and its capacity is updated. If the expert has some unassigned capacity, we assign her rarest skill to the most profitable project in lines 28 to 31 until her capacity is full. The motivation for doing it the same as the last part of Algorithm 1, as soon as we hire an expert, we prefer to use her maximum capacity.

4 EXPERIMENTS

In this section, we evaluate the performance of our proposed algorithms to find the best group of experts for the problem of cluster hire in a network of experts. We create the input graph (i.e., network of experts) from the DBLP[1] XML dataset in the same way as [12, 14]. The dataset contains information about a set of papers and their authors. The same as [12, 14], we only consider papers that are published in major conferences in databases and data mining: {SIGMOD, VLDB, ICDE, ICDT, EDBT, PODS, KDD, WWW , SDM, PKDD, CIKM, ICDM}. The dataset contaisn information about 44K experts. All edges have the same weight. Note that edge weights, experts' costs, and projects' profits are all normalized to have the same scale. If two authors publish at least two papers together, there will be an edge between them in the graph. If two experts are not directly connected, we use the value of the shortest path between them as the communication cost value. The experts are authors of the papers. The expertise (i.e., skill) of an expert (i.e., author), is extracted from the titles of her papers. We randomly create a collection of projects in which each of them requires 4 to 9 skills

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

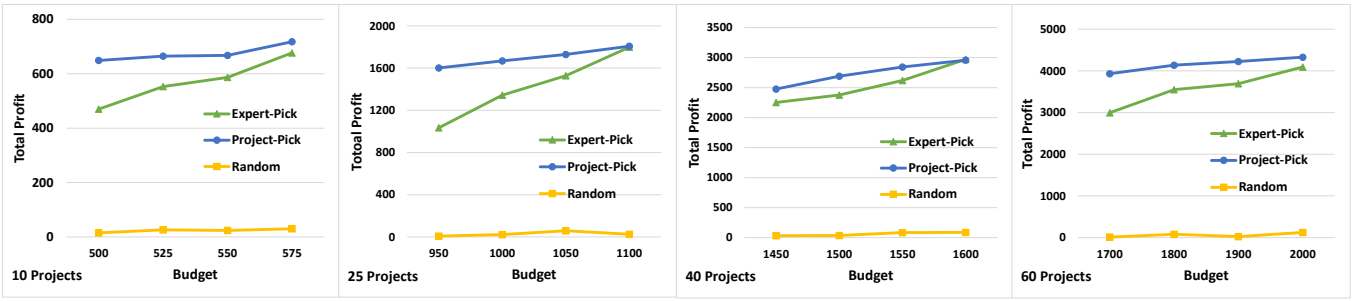

Figure 1: Total profit vs. budget.

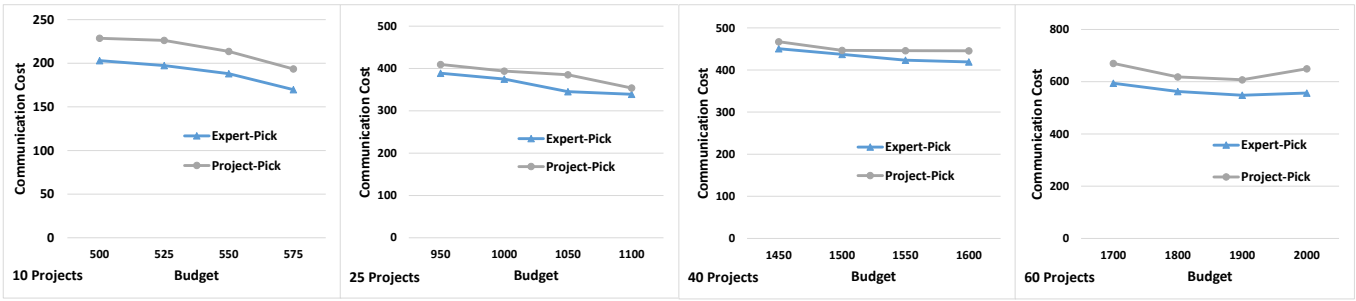

Figure 2: Communication cost vs. budget. Communication cost of Random was higher than others in orders of magnitude . Thus, they are omitted to better show the difference between the results of Expert-Pick and Project-Pick.

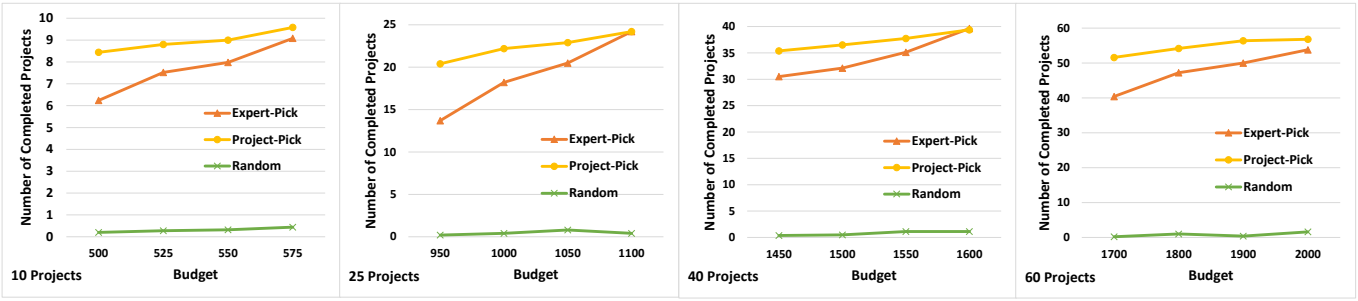

Figure 3: Number of completed projects vs. budget.

for completion. Each collection contains 10 to 60 projects. For each collection, we run the experiments 100 times and report the average of the values. The same as [13], the cost of an expert is set based on the number of publications of the expert, assuming that the more publications an expert has, the more expensive she is. The capacity of an expert is randomly set between 5 to 15. The profit of each project is set randomly between 50 to 100. In our experiments, we use different values for the budget to see the total profit returned by each algorithm. The value of λ is set to 0.5 by default unless otherwise is stated.

As a baseline for the comparisons, we compare the results with the random algorithm, which selects the best group among 10,000 random groups that maximizes the objective within the given budget. Since we are the first one to study the problem of Cluster Hire in a network of experts, there does not exist a prior work to compare our results with. Our algorithms are implemented in Java and executed on an Intel Core i7 2.8 GHz computer with 16 GB of RAM.

4.1 Total Profit vs. Budget

We start by evaluating the effect of the budget on total profit of the projects. Figure 1 shows the values of total profit for increasing values of the budget for different number of projects. In each experiment, we initially create k projects, in which $k = \{10, 25, 40, 60\}$. We then report total profit for different values of the budget. Recall that λ is set to 0.5. The results suggest that Project-Pick achieves higher total profit than Expert-Pick when we have limited budget. For higher values of the budget, they return the same profit. As

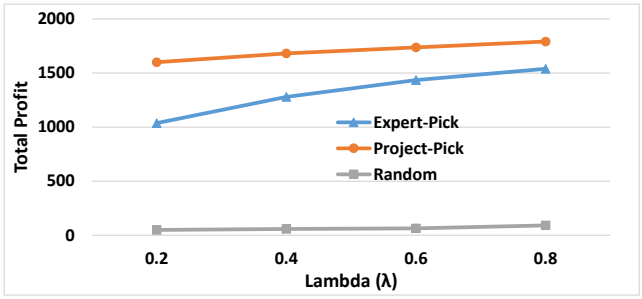

Figure 4: Total profit for different values of tradeoff parameter λ (25 projects and \$1050 budget).

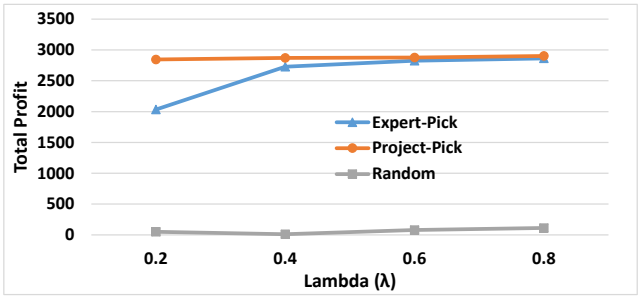

Figure 5: Total profit for different values of tradeoff parameter λ (40 projects and \$1550 budget).

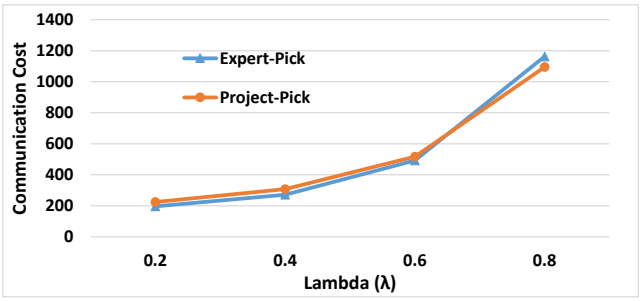

Figure 6: Communication cost for different values of tradeoff parameter λ (25 projects and \$1050 budget). Result of Random are omitted to better show the difference between Expert-Pick and Project-Pick.

expected, both Project-Pick and Expert-Pick outperform the Random algorithm. Furthermore, and as expected, by increasing the budget, total profit increases too. This is because we have more money to hire more experts, and therefore more profitable projects are covered.

4.2 Communication Cost vs. Budget

Now, we study the effect of the budget on communication cost. Figure 2 shows the values of communication cost for increasing values of the budget for different number of projects ($\lambda = 0.5$). Note that we want to minimize the communication cost, so the smaller

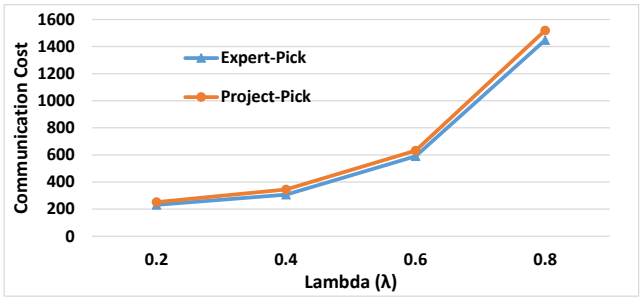

Figure 7: Communication cost for different values of tradeoff parameter λ (40 projects and \$1550 budget). Result of Random are omitted to better show the difference between Expert-Pick and Project-Pick.

the value, the more collaborative the group of experts are. The results suggest that Expert-Pick has slightly better communication cost. Note that the communication cost of the Random is larger than others in order to magnitude and is omitted from our charts.

4.3 Number of Completed Projects vs. Budget

Figure 3 shows the number of completed projects for increasing values of the budget for different number of projects ($\lambda = 0.5$). The results suggest that Project-Pick completes more projects than Expert-Pick when we have limited budget. For higher values of the budget, they complete the same number of projects (almost all projects are completed). As expected, both Project-Pick and Expert-Pick outperform the Random algorithm. Even with higher budget, Random is not able to complete many projects. This is because Random does not hire appropriate experts to be assigned to uncompleted projects. Furthermore, and as expected, by increasing the budget, more projects are completed.

4.4 Sensitivity of the Tradeoff Parameter λ

Figures 4 and 5 show the total profit for different values of tradeoff parameter λ. As expected, by increasing the value of λ, total profit increases as we put more weight on profit compared to communication cost. Figures 6 and 7 show communication cost for different values of tradeoff parameter λ. By increasing the value of λ, communication cost increases too. Note that the smaller the communication cost, the more collaborative group of experts we have. Thus, smaller values of λ put more more weights on communication cost and therefore produces more collaborative group of experts.

4.5 Discussion

Both Expert-Pick and Project-Pick have polynomial runtime. In our experiments over DBLP dataset, Expert-Pick has an average runtime of 2,316ms while Project-Pick has the runtime of 2,614ms. In terms of the results, when we have enough budget, they both generate the groups of experts with high profit and low communication cost. However, when budget is limited, Project-Pick produces better results.

5 RELATED WORK

Discovering a team of experts from a social network was introduced by Lappas et al., [14]. Given a set of required skills to build a single team of experts, authors considered the past collaboration among experts and propose an algorithm to find the most collaborative team. The collaboration was computed using two function, the diameter of the team (i.e., selected sub-graph) and the weight of the Steiner tree among team members. Li and Shan generalized this problem and associate each required skill with a specific number of experts [15]. Kargar and An proposed a new function to compute the collaboration cost among experts (i.e., the sum of distance between evrey pair of expert holders) [11]. They argued that the new function is fairer towards all team members and is not biased towards only some of team members. Authors of [8] propose a new communication cost function based on the density of the induced sub-graph.

Kargar et al., proposed approximation algorithms to find a team of experts that optimizes two objectives: the communication cost among team members and the personnel cost of a team [12]. They assumed every expert has a salary to be hired as a team member. Authors of [13] solved this problem using a different approach, they find a set of Pareto teams. They also propose an approximation algorithm that receives a budget on the personnel cost and minimize the communication cost under the given budget. Li et al., proposed an algorithm to find a replacement when a team member is not avaiable anyore [16]. Authors of [18] study team formation and optimize the authority of skill holders. Authors of [19] optimize three objectives to find the best team of experts and find a set of Pareto teams. All of the above works assume we are looking to find a single team of experts to perform a single project. The problem that we tackle in this paper assumes that we want to cover a set of projects while optimizing the profit and communication cost.

The team discovery problem is well investigaed by the operation research community [3, 17]. Different papers use genetic algorithms, branch and bound, and simulated annealing with the goal of finding a team for performing the given task [5, 7, 10, 17]. None of these work consider the underlying social network among experts and ignore the communication cost of the selected team. Awal et al., proposed an algorithm to find a team of experts in social networks based on a collective intelligence index [4]. The genetic-based algorithm uses the expertise score and trust score of experts as the fitness function. Again, this work assumes user is only interested to find a single team of experts to perform a single project.

The problem of Cluster Hire was introduce by Golshan et al., [9]. As we discuss earlier, given a set of projects and a set of available experts, the authors find a subset of projects along with a subset of experts to perform the projects while maximizing the profit of projects and not violating the given budget constraint. The work of [1] is close to the Cluster Hire problem as the authors assume we are interested to select a set of projects while minimizing the maximum load of participating experts. However, in their setting, projects do not come with a profit and there is not salary for experts. The also ignore the communication cost among experts. Minimizing both load balance and communication cost was studied in [2] and [6]. In this work, we extend the original Cluster Hire problem by taking into account the underlying social network and graph structure among experts and on top of maximizing the profit of projects, we also minimize the communication cost among team members. We make further refinement to the original problem by specifically assigning expertise of each expert to specific projects.

6 CONCLUSION

In this paper, we extended the problem of Cluster Hire in a network of experts. Given a set of projects and a set of experts, the goal is to find the most collaborative group of experts that maximizes the total profit under the given budget to hire experts. We are the first to study this problem in the context of social networks. Optimizing the communication cost as well as the total profit turns the problem into a bi-objective optimization problem. We first combine the two objectives using a tradeoff parameter λ that determines which optimization objective is more important. We then propose two greedy algorithms to find the best group. The first algorithm, Expert-Pick, selects an expert in each iteration as long as we don't exceed the budget. The second algorithm, Project-Pick, selects a project to be covered in each iteration. For each of these algorithms, we design proper scoring functions to rank the experts and projects to be selected. Our experiments on DBLP shows that both of these algorithms are able to select a group of experts to optimize both of the objectives.

REFERENCES

[1] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. 2010. Power in unity: Forming teams in large-scale community systems. In *Proc. of CIKM'10*.

[2] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. 2012. Online Team Formation in Social Networks. In *Proc. of the WWW'12*.

[3] Z. Ani, A. Yasin, M. Husin, and Z. Hamid. 2010. A Method for Group Formation Using Genetic Algorithm. *Int. Journal of Advanced Trends in Computer Sci. & Eng.* 2, 9 (2010), 3060–3064.

[4] G. K. Awal and K. K. Bharadwaj. 2014. Team Formation in Social Networks based on Collective Intelligence - an Evolutionary Approach. *Appl Intell* 41, 2 (2014), 627–648.

[5] A. Baykasoglu, T. Dereli, and S. Das. 2007. Project team selection using fuzzy optimization approach. *Cybern. Syst.* 38, 2 (2007), 155–185.

[6] S. Datta, A. Majumder, and K. Naidu. 2012. Capacitated Team Formation Problem on Social Networks. In *Proc. of KDD'12*.

[7] E.L. Fitzpatrick and R.G. Askin. 2005. Forming effective worker teams with multi functional skill requirements. *Comput. Ind. Eng.* 48, 3 (2005), 593–608.

[8] A. Gajewar and A. Das Sarma. 2012. Multi skill Collaborative Teams based on Densest Subgraphs. In *Proc. of the SDM'12*.

[9] B. Golshan, T. Lappas, and E. Terzi. 2014. Profit-maximizing Cluster Hires. In *KDD*. 1196–1205.

[10] M. Jackson. 2008. *Network formation.* The New Palgrave Dictionary of Economics and the Law.

[11] M. Kargar and A. An. 2011. Discovering top-k Teams of Experts with/without a Leader in Social Networks. In *CIKM*. 985–994.

[12] M. Kargar, A. An, and M. Zihayat. 2012. Efficient Bi-objective Team Formation in Social Networks. In *ECML/PKDD*. 483–498.

[13] M. Kargar, M. Zihayat, and A. An. 2013. Finding Affordable and Collaborative Teams from a Network of Experts. In *SDM*. 587–595.

[14] T. Lappas, L. Liu, and E. Terzi. 2009. Finding a Team of Experts in Social Networks. In *KDD*. 467–476.

[15] C. Li and M. Shan. 2010. Team Formation for Generalized Tasks in Expertise Social Networks. In *Proc. of IEEE International Conference on Social Computing*.

[16] L Li, H. Tong, N. Cao, K. Ehrlich, Y. Lin, and N. Buchler. 2015. Replacing the Irreplaceable: Fast Algorithms for Team Member Recommendation. In *WWW*. 636–646.

[17] H. Wi, S. Oha, J. Muna, and M. Jung. 2009. A Team Formation Model based on Knowledge and Collaboration. *Expert Syst Appl* 36, 5 (2009), 9121–9134.

[18] M. Zihayat, A. An, L. Golab, M. Kargar, and J. Szlichta. 2017. Authority-based Team Discovery in Social Networks. In *EDBT*. 498–501.

[19] M. Zihayat, M. Kargar, and A. An. 2014. Two-Phase Pareto Set Discovery for Team Formation in Social Networks. In *WI*. 304–311.