University of Windsor

Scholarship at UWindsor

Electronic Theses and Dissertations

Theses, Dissertations, and Major Papers

9-18-2018

Cluster Hire in a Network of Experts

Meet Sanjaykumar Patel
University of Windsor

Cluster Hire in a Network of Experts

By

Meet Patel

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for the
Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada
2018

Cluster Hire in a Network of Experts

By

Meet Patel

APPROVED BY:

R. Razavi-Far

Department of Electrical and Computer Engineering

P. M. Zadeh

School of Computer Science

M. Kargar, Co-Advisor

School of Computer Science

J. Chen, Advisor

School of Computer Science

September 10, 2018

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyones copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Finding a group of experts is a natural way to perform a collection of tasks that need 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 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 is equipped with a set of skills. To hire an expert, the organization should provide her with 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 maximize the total profit. This problem is called Cluster Hire and was introduced recently. We extend this problem by making the realistic assumption that there exists 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 to 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 a real dataset show our proposed algorithms can find a group of experts that cover projects with high profit while experts can communicate with each other efficiently.

DEDICATION

Dedicated to my parents Sanjay Patel and Vibhuti Patel.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my advisor Dr. Jessica Chen and Dr. Mehdi Kargar for their continuous support in my research, for their patience, motivation, and immense knowledge. Their guidance helped me throughout my time of research and writing of this thesis. I could not have imagined having better mentors for my study.

I would also like to thank my thesis committee members Dr. Pooya Moradian Zadeh and Dr. Roozbeh Razavi Far for their valuable comments and suggestions for the completion of this thesis. I would like to sincerely thank Dhruvi Patel, who has been the fundamental support throughout my work.

Also, I would like to thank my fellow lab mates for the stimulating discussions during the development of this work. Last but not the least, my deepest gratitude to my family for their unconditional love, support, and belief.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Overview

Forming a team of experts during a recruitment process, the recruiter aims to seek the most cost-effective team that can accomplish the tasks within the deadline. The process includes allocating the basic salary for each of the experts from the candidate's list, to form a team that has all the required expertise to perform many profitable tasks. This problem is a CLUSTER HIRE problem in Network Science introduced by Golshan et al. [1]. The problem came into accountability from the domains of online labour markets such as Freelancer and Guru. However, the area is growing worldwide. People develop agencies for more profit margin and, in industries also, Network Science is more prevalent.

Generally, team formation is a well-known natural way to obtain a group of experts with complementary skills which came out to be an effective solution for the organizations and big institutes. This can be achieved by assigning skills to various experts with varying skills. In other words, given a set of experts, each with specific skills, the goal is to select the team of experts that can collectively cover the required skills of the project, while ensuring the efficient communication between the selected team members. On successful completion of the project, the organization makes a profit. For a set of experts, each of them possesses a set of diversified skills and each project requires a subset of specified skills in the expert. To recruit an expert for the project, the organization should provide an economic cost (i.e., the salary of the expert). Also, the organization is assigned a certain amount of budget for every project to hire experts. To maximize the profit of the organization, they need to hire experts within the given budget to perform the projects.

Also, communication among the expert allocated for the project plays a vital role for better results and faster completion of the projects. Assuming that there exists a communication channel among the experts. The communication cost among the experts can be calculated based on the previous collaboration between them. If any of the two experts have worked previously on the past projects, have a good frequency of communication, it would result in the quality of work and effectively meeting the deadlines of the projects. However, if the team of experts never worked simultaneously before, there will be a higher communication cost, increasing the expense of the project. How to determine the communication cost between the experts depends on the projects needs. The communication cost is not limited to the previous collaboration of the experts, although the geometric distance among the experts can be calculated if in-person meetings are necessary for projects. Eventually, effective communication results in improved performance of the organization and a series of profitable projects.

Moreover, for reaching an effective and efficient team, a capacity of each expert becomes a major aspect. Employee satisfaction goes in parallel with the profits and the reputation earned by an organization. Each expert has a maximum capacity to participate in several tasks simultaneously. Hence, hiring an expert with a higher capacity for work turns out to be a profit for the organization as we can use the same expert for various projects, whereas it's pretty useless to hire experts with low capacity. Ultimately, communication cost between the experts and capacity of each expert results in a bunch of realistic teams.

Collectively, we are addressing to the CLUSTER HIRE problem in a network of experts, for maximizing the profit, and we are minimizing the communication cost of the experts. The problem turns out to be a bi-objective optimization problem.

1.2 Motivation

In the real world, forming communities in a network based on the requirements of the tasks to be performed as a community is a fundamental problem in Network Science. Many researchers are working in this area to find an optimum solution for a team formation by extending it in multiple dimensions. For instance, there exists a work that focuses on different ways of defining the communication costs between the experts. Others take into account the capacity of the expert. For the online version of the problem, the goal is to create multiple teams that can work on multiple projects [1].

In the immersing world of technology, the startup companies are growing worldwide where the owner needs to select a team of experts with the collective expertise required to benefit from on the various opportunities that have been identified within the market that the company targets at [1].

Another relevant motivation is from the online labour markets, such as FreeLancer (`www.freelancer.com`), and Guru (`www.guru.com`). In the online portals, they receive ample projects to work remotely in various areas. In the early stages, the freelancers registered on the portals and worked independently. However, with the gradual increase in competition, experts started developing a consulting company. The consulting company receives the projects with the specific set of required skills [1]. On every successful completion of the project, the consulting company receives the profit. For each required task on the project, the company needs to hire an expert to fulfill the set of demanded tasks. A certain amount of budget is defined by the financial department to hire the experts. The goal of the consulting company is to recruit a team of experts in which the sum of the salary of each of the expert falls within the budget. The target of the company is to maximize the total profit that it gains by completing the projects.

To overcome this problem, in this thesis, we are proposing an approach to form a realistic team for the assigned projects with required skills by maximizing the profit of the organization.

1.3 Problem & Solution Outline

The team formation problem is being researched nearly a decade to come with a realistic team considering the communication cost between the experts and capacity of each expert, to make a maximum profit. In our approach, we are making the realistic assumption that there exists an underlying network among the experts. Suppose each expert possesses a set of skills, and each expert demands a salary for participating in performing different tasks. Furthermore, in the list of given projects, each project is composed of a set of required skills that need to be covered by some experts to complete the project. Assume that, each expert offers their expertise with a capacity. This is a realistic assumption as we do not want to overload any expert by assigning them more tasks than affordable. Therefore, each expert can participate in some tasks.

The experts are connected to a network which is modelled as an undirected graph. Each expert is associated with a node in it. We use the terms node and expert interchangeably in this work. Two experts are connected by an edge in the graph if they have prior collaboration in the past (e.g., participating in the same project). The graph might be weighted. In this case, the weight on edge represents the strength of the collaboration between them (the smaller the 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 the graph 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 might be introduced to each other via a middle expert where the two experts worked with the middle expert in the past. This is certainly preferred over another case in which the two experts have no relationship via another expert in common. The communication cost between two experts is the weight of the shortest path between two experts in the graph. Our objective is to choose a group of experts where the communication cost among them is minimized. We model the communication cost among a group of experts as the sum of distances between each pair of the experts in the group.

For completing a given set of projects, we are given a predetermined budget. This budget is spent on hiring experts. Our goal is to hire as many experts as possible while the sum of their hiring costs (i.e., salary) is within the predetermined budget. Since we are interested in maximizing the profit and minimizing the communication cost, our problem is a bi-objective optimization problem.

1.4 Scope of thesis

In the Network Science group, Golshan et al. [1] proposed a detailed analysis of the computational complexity and approximation of the problem, as well as heuristic solutions. Moreover, they demonstrated the efficacy of the approaches through experiments on real datasets of experts and demonstrated their advantage over intuitive baselines. Also, they explored additional variants of the fundamental problem formulation, to account for constraints and considerations that emerge in realistic cluster-hiring scenarios [1]. In our thesis, we proposed a network formation based on past collaboration among experts. If two experts have past collaboration, they form a more collaborative and efficient team in the future. Maximizing the total profit, we are also interested in finding the most collaborative group of experts by minimizing the communication cost among them. Furthermore, we propose two greedy algorithms with different strategies to solve the problem. Extensive experiments on a real dataset show our proposed algorithms can find a group of experts that cover projects with high profit while experts can communicate with each other efficiently.

1.5 Structure of thesis

The remaining thesis is organized as follows. In Chapter 2, we review the basic concepts of Data Mining, Team Formation, and CLUSTER HIRE. This section also includes the detailed explanation of the problem statement of the thesis and review to the greedy approach used for the solution. The CLUSTER HIRE problem was introduced by Golshan et al. [1]. We present our proposed approach in Chapter 3. To make this thesis self-contained, a detailed explanation of the algorithm is also given in Chapter 3. Chapter 4 contains the primary contribution of this thesis and describes the implementation details of our approach, and the simulation results with the comparison of the approaches. Chapter 5 provides the summary concluding the thesis along with the directions for possible future work.

Chapter 2

PROBLEM BACKGROUND

2.1 Overview of the Research

2.1.1 Data Mining

Data Mining is a concept of discovering patterns in large data sets. This process of extracting important data from a large dataset is also known as knowledge discovery from the database. A process of extracting interesting patterns has been going on for ages, but in todays era, every industry is moving to a paperless system resulting in a huge amount of digital data. To keep up with the pace of data generation, one must have a good data mining technique to discover interesting patterns to analyze the trend. Depending on these trends a company may decide its future techniques to attract a larger amount of business. It sounds very easy, but such good facilities come at a huge cost. They are not affordable for all types of business. People choose different techniques based on their capacity, and that is what determines the fate of these companies.

Knowledge Discovery in Database is done in step by step procedure: [2]

- Data Cleaning: In this step, the noise, and inconsistent data are removed.
- Data Integration: In this step, multiple data sources are combined.
- Data Selection: In this step, the data relevant to the analysis task are retrieved from the database.
- Data Transformation: In this step, data is transformed or consolidated into

forms appropriate for mining by performing summary or aggregation operations.

- Data Mining: In this step, intelligent methods are applied to extract data patterns.
- Pattern Evaluation: In this step, data patterns are evaluated.
- Knowledge Presentation: In this step, knowledge is represented.

The following figure shows the process of knowledge discovery: [2]

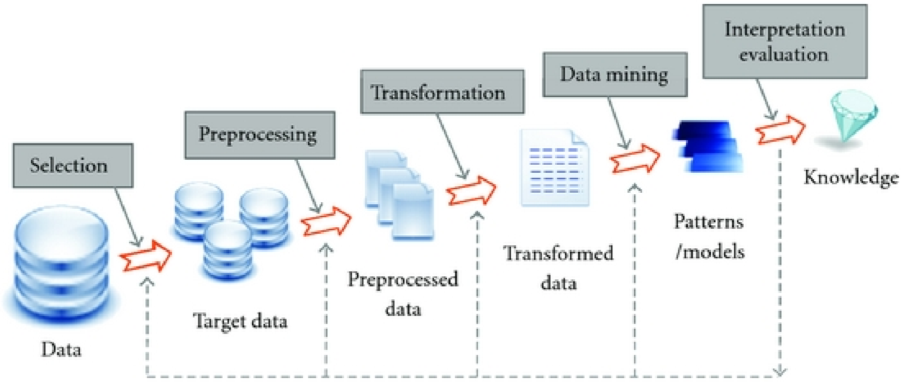


Figure 2.1: Process of Knowledge Discovery

2.1.2 Team Formation

Team Formation is a concept of hiring a set of individuals who possess some skills that are used to complete a project that will benefit an institution upon its completion. This helps an institution to complete projects undertaken, and it also helps experts to strengthen their position in the institute by performing well. The sole reason for this concept is to bring certain individuals together to get the best out of them. This can be achieved only when we have a group of individuals who are willing to work together. Selection of such individuals can be done in various forms using different constraints and factors. Such factors will determine how good a team can perform.

For example, we need a team for the three-leg race, so we look for individuals who can run fast. We have statistics of a 100 m race. We would like to go for 1st and 2nd candidate as they are the quickest, but it may be possible that the 3rd and 4th candidate from the 100m race performs better in 3 leg race because they run at a similar speed whereas the 1st and the 2nd candidate from that race performs lower than usual because of the difference in their speeds. So, depending on the type of task, we need to vary the factors to get the best results.

2.1.3 Cluster Hire

CLUSTER HIRE means hiring a group of individuals to accomplish a group of tasks in an institution within a given budget. For example, say we have five projects that need to be completed. These projects need a fixed set of skills, so we go to hire some experts that possess all the skills required to complete the five projects. This concept is termed as CLUSTER HIRE.

The selection process will be carried out by certain factors such as Budget, Profit, Communication Cost, Capacity, etc. This concept is growing its importance in the market as it covers various projects using the same set of individuals instead of hiring a fixed set of individuals for each project which is ultimately cutting down the Budget factor.

2.1.4 Difference between Team Formation and Cluster Hire

Given a set of n available experts, each associated with some skills, and given a set of m projects of an organization that requires some people with specific skills. Figure 2.2 demonstrates the CLUSTER HIRE problem and the team formation. Hiring an expert for a project by considering the factors such as required skills, the budget of the project, profit of

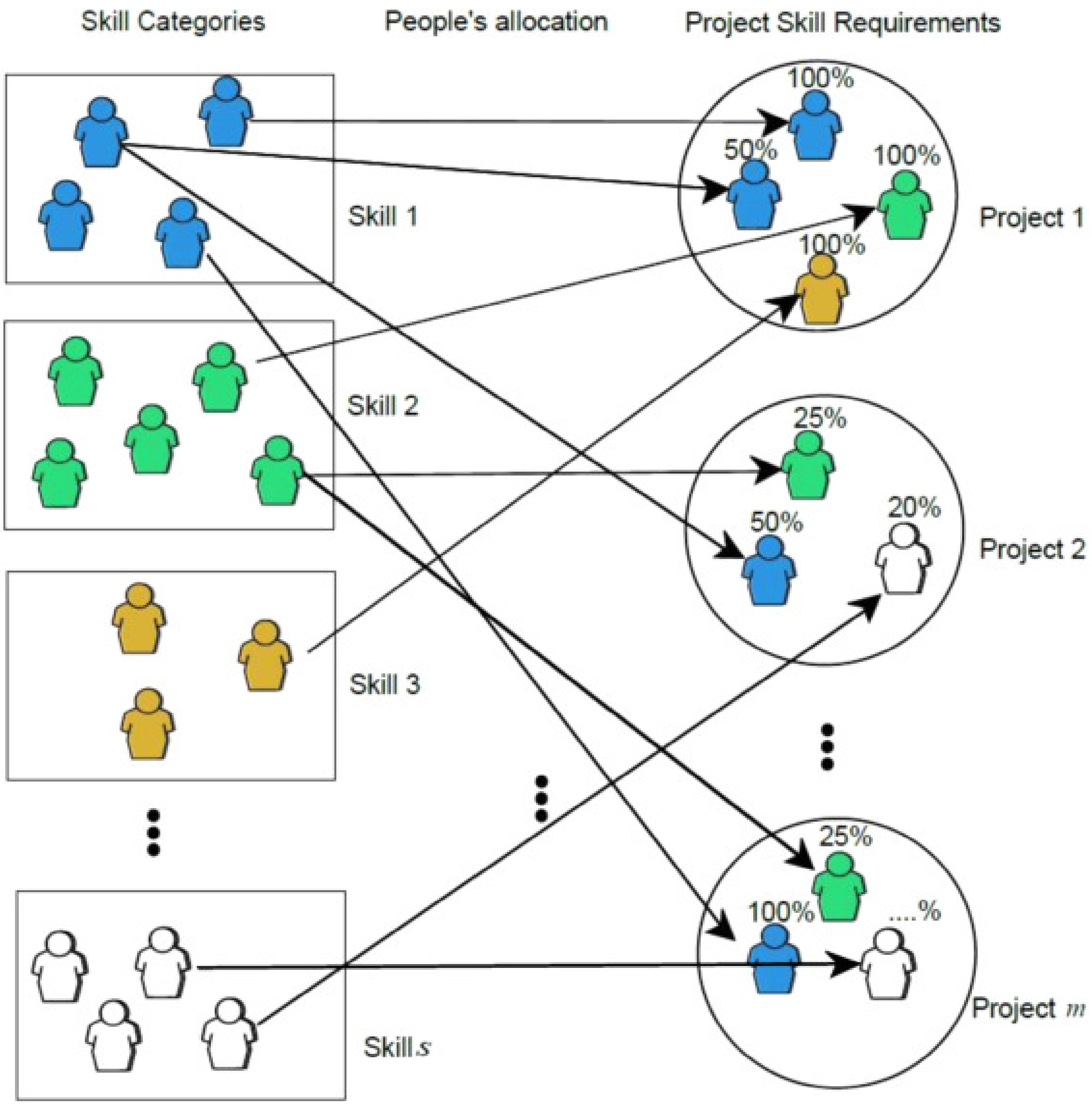


Figure 2.2: Multiple Team Formation problem Scheme

the project, communication cost between the experts, the capacity of the experts and many more aspects; collectively fall under the CLUSTER HIRE problem. A realistic and efficient team formed for a project regardless of the budget and the profit of the project is commonly known as team formation.

As per figure 2.2, an expert with Skill 1 is allocated to Project 1 while is another expert with similar skill works for Project 1 and Project 2 simultaneously which is considered to be a profit factor for the organization. Hence, hiring cost for *m* projects of the organization is reduced. This is a good example of Cluster Hire. However, forming a team for Project 1 to satisfy the required skills for the project shows team formation.

The figure 2.2 shows the main components of the problem and illustrates an example to the feasible solution to the problem.

2.2 Fundamental Concepts

2.2.1 Exact Algorithm

The exact algorithm guarantees an optimal solution to the problem if it exists but at the cost of high computational time. This approach should not be used when your search space is huge as it will take a huge amount of time to provide an exact solution. To search the solution in a huge search space, much of the computer power is needed which comes at a great price. Depending on the criticality of the solution, we should decide which approach should be taken to find the solution to the problem. If we need the optimal solution, we should follow this approach. Otherwise, we can get nearly optimal or optimal solution using heuristic approach. In figure 2.3, an exact algorithm calculates the solution for each possible combination and then it selects the optimal solution. It can be seen that the number of combinations and the computational time is very high. It does provide an optimal solution but it compromises with processing time.

2.2.2 Heuristic Algorithm

Heuristic algorithm comes into play when other methods take too long to find a solution. This approach finds a near to optimal solution or an optimal solution in some case. Depending on the heuristic function, it decides which branch should be followed. The decision is made depending upon the available information at each step. This method com-

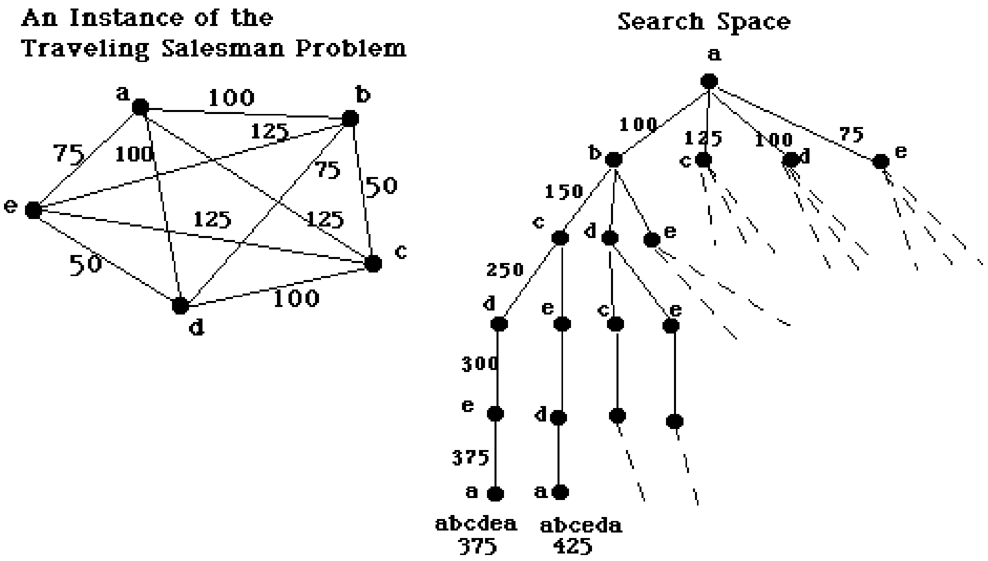
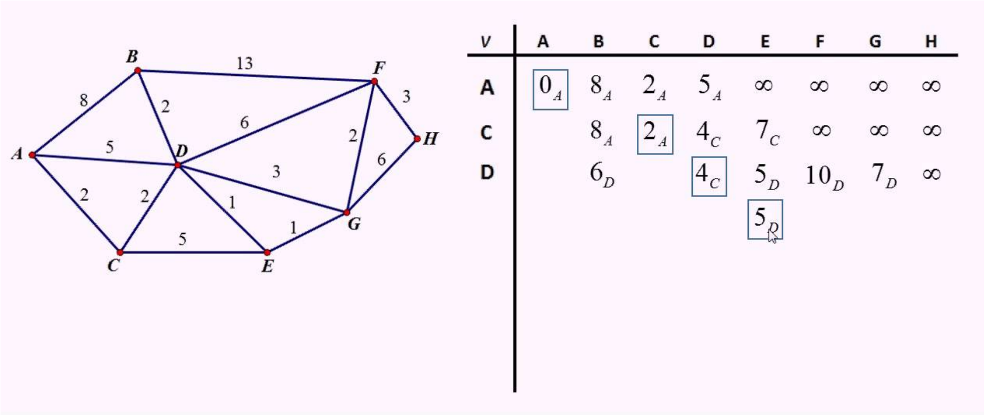


Figure 2.3: Exact Algorithm

promises with optimality, accuracy, completeness but provides quick results. In figure 2.4, the heuristic algorithm (Dijkstra Algorithm to find a shortest path between given nodes) randomly selects a vertex to start looking for a solution and from there on, it selects the next step depending on the information it has then. It provides us with a solution in a short period but it compromises with optimality. It may provide us with an optimal solution in some cases, but optimality cannot be guaranteed in this approach.



V	A	B	C	D	E	F	G	H
A	$\boxed{0_A}$	8_A	2_A	5_A	∞	∞	∞	∞
C		8_A	$\boxed{2_A}$	4_C	7_C	∞	∞	∞
D		6_D		$\boxed{4_C}$	5_D	10_D	7_D	∞
					$\boxed{5_D}$			

Figure 2.4: Heuristic Algorithm

2.2.3 Greedy Algorithm

A greedy algorithm is an algorithmic paradigm that follows the problem-solving heuristic of making the locally optimal choice at each stage with the intent of finding a global optimum. In many problems, a greedy strategy does not usually produce an optimal solution, but a greedy heuristic may yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.

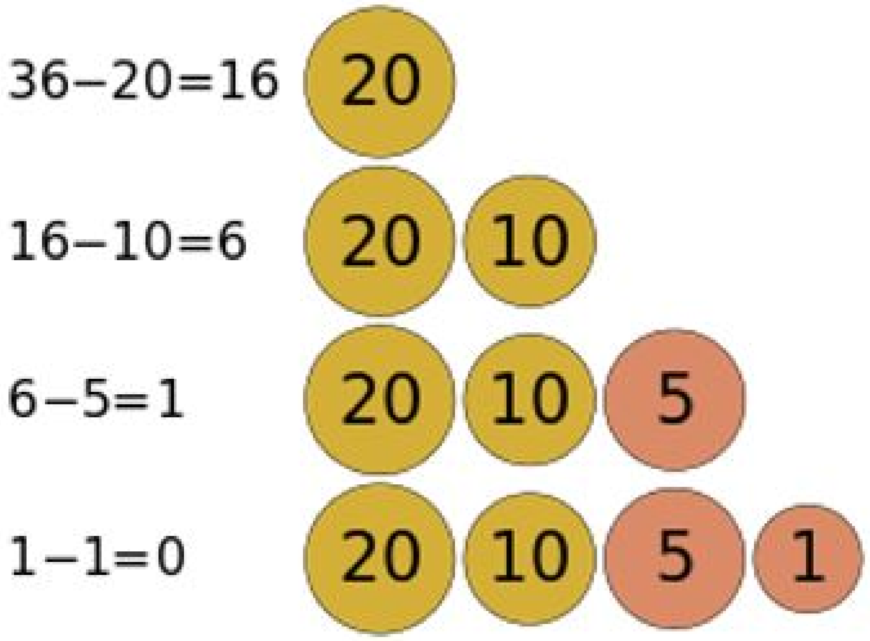


Figure 2.5: Greedy Algorithm

A greedy algorithm can be used to determine the minimum number of coins to give while making a change. These are the steps a human would take to emulate a greedy algorithm to represent 36 cents using only coins with values 1, 5, 10, 20 as shown in Figure 2.5. The coin of the highest value less than the remaining change owed is the local optimum. (In general, the change-making problem requires dynamic programming to find an optimal solution; however, most currency systems, including the Euro and US Dollar, are special cases where the greedy strategy does find an optimal solution.)

In general, greedy algorithms have five components:

- A candidate set, from which a solution is created
- A selection function, which chooses the best candidate to be added to the solution
- A feasibility function, that is used to determine if a candidate can be used to contribute to a solution
- An objective function, which assigns a value to a solution, or a partial solution, and
- A solution function, which will indicate when we have discovered a complete solution

2.2.4 Bi-objective Function

This concept is used when we need to optimize more than one function simultaneously. We use such functions when we need to take an optimal decision, and we have more than one deciding factors. This concept has been implemented in various fields like engineering, science, economics, etc. These type of functions are controlled by a trade-off parameter. The value of the trade-off parameter decides the ratio of deciding factors in the final result.

For instance, Score = A * (Factor 1) + (1-A) * (Factor 2) where A is in a range between 0 and 1.

Impact of Factor 1 and Factor 2 will depend on the value of A. More the value of A; higher will be the impact of Factor 1 on the score and vice versa.

2.3 Literature Review and Related Work

Finding a team of experts from a social network was introduced by Lappas et al.in

2009, [3]. Given a set of required skills to build a single team of experts, the authors considered the past collaboration among experts and proposed an algorithm to find the most collaborative team. The collaboration was computed using two functions, 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 associated each required skill with a specific number of experts [4]. Kargar and An proposed a new function to compute the collaboration cost among experts (i.e., the sum of the distance between every pair of expert holders) [5]. They argued that the new function is fairer towards all team members and is not biased towards only some of the team members. Authors of [6] proposed a new communication cost function based on the density of the induced sub-graph.

Kargar et al. proposed an approximation algorithm to find a team of experts that optimizes two objectives: the communication cost among team members and the personal cost of a team [7]. They assumed that every expert has a salary to be hired as a team member. Authors of [8] solved this problem using a different approach. They found a set of Pareto teams. They also proposed an approximation algorithm that receives a budget on the personnel cost and minimizes the communication cost under the given budget. Li et al. proposed an algorithm to find a replacement when a team member is not available anymore [9]. Authors of [10] studied team formation problem and optimized the authority of skill holders. Authors of [11] optimized three objectives to find the best team of experts and to find a set of Pareto teams. All of the above work assume that we are looking to find a single team of experts to perform a single project. The problem that we tackle in this thesis assumes that we want to cover a set of projects while optimizing the profit and communication cost.

The team discovery problem is well investigated by the operations research community [12, 13]. Different papers use genetic algorithms, branch and bound, and simulated annealing with the goal of finding a team for performing the given task [13–16]. None of these work considered the underlying social network among experts and ignored 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 [17]. The genetic-based algorithm uses the expertise score and trust score of experts as the fitness function. Again, this work assumes that the user is only interested in finding a single team of experts to perform a single project.

In [18], the author Xinyu et al. have considered various approaches to solve Team Formation problem. They were categorized based on communication cost functions:

- R-TF algorithms
- Steiner-TF algorithms
- SD-TF algorithms
- LD-TF algorithms

The first approach suggests that we can hire a team of well-known experts to review a paper. According to the second approach, the team of reviewers found can discover diversified results, as the reviewers are from a wide range of background. Their third approach suggests that they can hire experts that are seeking a rise in their position and are new to the industry. The final category concentrates on balancing work amongst the expert in a team and makes sure no one is overloaded. In spite of the above various approaches to solving this problem, they could not come to a solution that is realistic and feasible. Each approach considers only one factor at a time which makes it unrealistic.

The author Tang et al. [19] have studied profit aware team formation problem. They have proposed an LP-based approximation to solve this problem, but this solution is not feasible in the real world as it has not considered many factors such as the capacity of the expert, communication cost amongst team members, etc. which are very important while working on a real-world problem. Hence, this approach has its drawbacks, and an extension of this problem would resolve them.

The problem of CLUSTER HIRE was introduced by Golshan et al., [1]. As we discussed earlier, given a set of projects and a set of available experts, the authors found 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 [20] is close to the CLUSTER HIRE problem as the authors assumed that we are interested in selecting 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 no salary for the experts. The communication cost among experts is also ignored. Minimizing both load balance and communication cost was studied in [21] and [22]. In this work, we extended 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 the projects, we also minimized the communication cost among team members. We have made further refinement to the original problem by specifically assigning the expertise of each expert to specific projects.

In paper [23], provides a new approach for Team Formation in social networks, they focus on experts and projects. In a social network of experts, they need to find a group of experts who can perform the required task to complete a project. To find a cluster, they perform SCAN on the pool of experts that possesses the required skills. Then they select experts that are well connected which is determined by communication cost. This communication cost is decided based on their experience. They have implemented an advanced version of SCAN known as WSCAN (Weighted SCAN) to solve TFP with minimum communication cost. They claim to have a better run time than Generic, Cultural and Exact algorithms whereas their results are approximately the same as the Greedy algorithm and are marginally below than Generic and Cultural algorithm.

2.4 Problem Statement

Let $E = \{e_1, e_2, \ldots, e_n\}$ denote a set of n experts, and $S = \{s_1, s_2, \ldots, s_m\}$ denote a set of m skills (all symbols used in this thesis are summarized in Table 2.1). 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 for the project to be completed. This set is shown by $PS(p)$ for project p. Again, $\forall\ p \in P, PS(p) \subseteq S$. We assume each expert e can 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 too many projects. Therefore, each expert has a maximum capacity to participate in some tasks. We take this constraint into consideration when designing the algorithms.

Definition 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 $\mathscr{E} \subseteq E$ is able to complete a subset of projects $\mathscr{P} \subseteq P$ if the following holds:*

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

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

When two projects require a same skill, for example Artificial Intelligence, then two different experts are allocated different projects. Therefore, we requir two experts with expertise in Artificial Intelligence. In the following, without loss of generality, we assume that there is no overlap between the required skill sets of any two projects. That is,

PS: $P \rightarrow 2^S\ is\ a\ function,\ where\ PS(P_i)\ \cap\ PS(P_j)\ =\ \emptyset,\ \forall\ P_i,\ P_j \in P,\ P_i \neq P_j$

In this way, the assignment will be simply between the experts and skills. The experts are connected to a network which is modelled as an undirected graph G. Each expert e_i

Table 2.1: Symbols used in the Thesis

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
$CC(\mathscr{E})$	communication cost among a group of experts $\mathscr{E}$
$Profit(\mathscr{P})$	profit of performing a group of projects $\mathscr{P}$
B	total budget for hiring experts
$ProfitCC(\mathscr{P}, \mathscr{E})$	combined objective of profit and communication

is associated with a node in G. We use the terms node and expert interchangeably in this work. Two experts are connected by an edge in G if they have previous collaboration in the past (e.g., participating in the same project). Graph G may 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 who have 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. If an expert has no connection with any of the other expert in the network, then we consider the cost value between this expert and any other expert to be infinity. 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 where e_i and e_j worked with e_k in the past. This is certainly preferred over another case where the two experts e_i and e_j have not worked with any other 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 in choosing 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.

For instance, suppose the communication cost is given in the following graph.

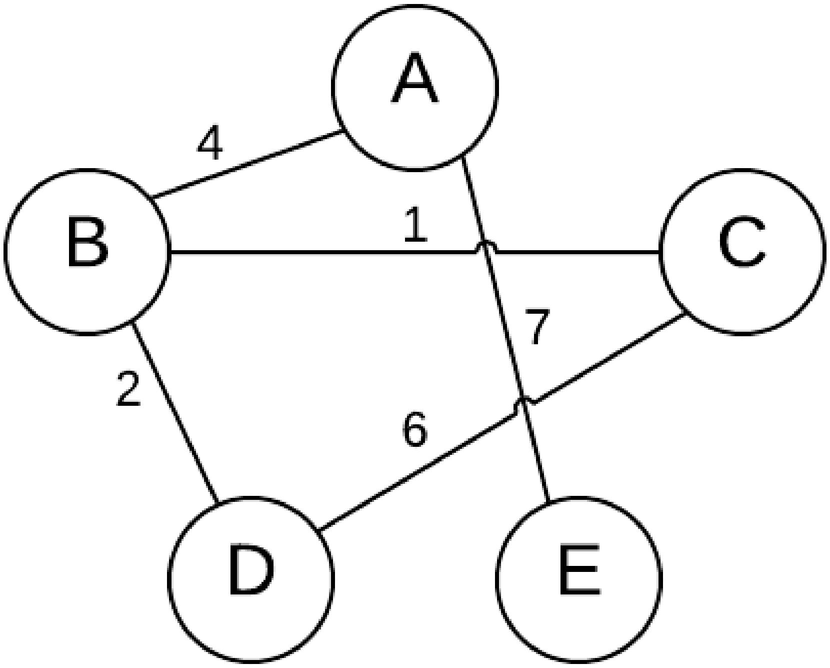


Figure 2.6: Example of Communication Cost

Here, expert A has a previous collaboration with experts B and E. Hence, the communication cost between experts A and B is 4, between A and E is 7. However, there is no direct edge between experts A and C, A and D.

Therefore, Dist(A,C) = Dist(A,B) + Dist(B,C) = 4+1 = 5 and Dist(A,D) = Dist(A,B) +

Dist(B,D) = 4+2 = 6.

In resultant, the communication cost between experts A and C is less compared to communication cost between A and D. So, the expert C will be appropriate for the team instead of D to minimize the communication cost.

Definition 2 ***Communication Cost:*** *Given a group of experts $\mathscr{E}$ and the weighted and connected graph G that determines the past collaboration among experts, the communication cost among the experts in $\mathscr{E}$ is defined as follows:*

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

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

Definition 3 ***Profit of Projects:*** *Given a set of projects $\mathscr{P}$, the profit of completing these projects is defined as follows:*

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

where Profit Function PF maps each project to an integer number denoting its profit.

$$PF : \mathscr{P} \rightarrow \mathbf{N}$$

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

A common approach to solving 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 be of the same type. Therefore, we maximize the reverse of the communication cost. Now we are ready to define the problem we tackle in this work formally.

Problem 1 *Given a set of n experts E, a set of m skills S, a set of k projects P, a trade-off λ between the profit and communication cost, and an underling graph G that determines the past collaboration among experts, we are interested in choosing a group of experts $\mathscr{E} \subseteq E$ (according to Definition 1) and a set of projects $\mathscr{P} \subseteq S$ so that the following objective is maximized:*

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

Furthermore, the following budget constraint must be satisfied:

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

Cost function: The cost function maps the set of experts $\mathscr{E}$ into set of Natural number **N**.

$$C : \mathscr{E} \to \mathbf{N}$$

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 by constant *a* so that they both fall into the same range (e.g., all of them fall between 0 and 1).

Theorem 1 *Problem 1 is NP-hard.*

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

Chapter 3

PROPOSED ALGORITHMS

In this chapter, 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 this expert 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, in each iteration, we greedily pick one expert to add to the pool of existing experts. We also check to see if this addition will cover any of the remaining projects. One important challenge is to make sure adding a new expert (with the corresponding salary) does not exceed the given budget B. In each iteration, we assign a score to each $(Expert, Project)$ pair and choose the expert with the highest score. The score is designed based on the following intuitions:

- We want to choose a *more affordable* 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.

As one might notice, these intuitions are not necessarily compatible. A less expensive expert may not be able to cover many skills from a profitable project. To take into account all these objectives, we design the following score function for each pair of projects and experts.

$$sc_e \leftarrow \sum_{p \in P} \frac{PF(p).Skill(e,p)}{C(e)} \tag{3.1}$$

Note that $Skill(e,p)$ determines the number of skills in p that could be covered by expert e. Our equation chooses an expert who covers high-profit project and 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.

Algorithm 1 is our solution to Problem 1 to find a group of experts while maximizing the profit of the covered projects. This algorithm receives the set of n experts, set of m skills, set of k projects, a network of experts G, and the available budget B as input. The output of this algorithm is a subset of projects $\mathscr{P}$ and a group of experts $\mathscr{E}$ that covers all required skills in $\mathscr{P}$ while maximizing the objective of Problem 1. Furthermore, the sum of the salary of the experts in $\mathscr{E}$ is not more than the given budget B. Our work is motivated from the paper by Golshan et al. [1]. We are proposing two different algorithms (Algorithm 1 - without communication cost and expert capacity, and Algorithm 2 - with communication cost and expert capacity) as an extension and improvement of previous work.

In line 1, $\mathscr{E}$ and $\mathscr{P}$ 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 $\mathscr{E}$ as long as $b < B$. Line 2 starts the iteration of the greedy algorithm. As long as we have experts in E that have not been added to $\mathscr{E}$ and we have not overspent the budget, we can consider adding more experts to $\mathscr{E}$. This is the condition of the while loop in line 2. In line 3, we store all unassigned experts in E whose salary is within the remaining budget to $\mathscr{R}$. The for loop in line 4 checks to see if each expert in $\mathscr{R}$ possesses at least one required skill in an

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 $P = \{p_1, p_2, \ldots, p_k\}$, and budget B.
Output: subset of projects $\mathscr{P} \subseteq P$ and a group of experts $\mathscr{E} \subseteq E$ that maximize $Profit(\mathscr{P})$ under the given budget B.

```
 1: $\mathscr{E} \leftarrow \emptyset$, $\mathscr{P} \leftarrow \emptyset$, $b \leftarrow 0$
 2: while $b < B$ and $E/\mathscr{E} \neq \emptyset$ do
 3:     $\mathscr{R} \leftarrow \{e \mid e \in E$ and $e \notin \mathscr{E}$ and $C(e) + b \leq B\}$
 4:     for all $e \in \mathscr{R}$ do
 5:         if $e$ does not cover any required skills in $P/\mathscr{P}$ then
 6:             remove $e$ from $\mathscr{R}$
 7:     if $\mathscr{R} = \emptyset$ then
 8:         return $\mathscr{E}$ and $\mathscr{P}$
 9:     for all $p \in P/\mathscr{P}$ do
10:         for all $e \in \mathscr{R}$ do
11:             if $e$ covers at least one skill in $\mathscr{P}$ then
12:                 if $\mathscr{E} = \emptyset$ then
13:                     $sc_e \leftarrow \sum_{p \in P} \frac{PF(p).Skill(e,p)}{C(e)}$
14:             else
15:                 $sc_e \leftarrow 0$
16:     $(e, P) \leftarrow \arg\max_{e \in \mathscr{R}} sc_e$
17:     add $e$ to $\mathscr{E}$
18:     for all $p \in P/\mathscr{P}$ do
19:         assign skills of $e$ to $p$
20:         update $PS(p)$
21:         $b \leftarrow b + C(e)$
22:         if $|PS(p)| = 0$ then
23:             add $p$ to $\mathscr{P}$
24: return $\mathscr{E}$ and $\mathscr{P}$
```

uncovered project. If this is not the case, those experts are removed from $\mathscr{R}$ as these experts are useless for the remaining projects. In line 7, we check to see if $\mathscr{R}$ is empty or not. If it is empty, we return the current $\mathscr{E}$ and $\mathscr{P}$ and terminate the algorithm. The reason is that, if $\mathscr{R}$ is empty, we have no other experts to add to $\mathscr{E}$. In lines 9 to 17, we calculate a score for each expert in $\mathscr{R}$ for uncovered project p (projects in $P/\mathscr{P}$). Later, we choose the highest score and add the associated expert to $\mathscr{E}$. If e does not cover any of the required skills in p (line 11), the score is set to 0 (line 15) 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 according to Equation 3.1. In line 18, we choose an expert (e, P) in which the score sc_e is maximized among all experts. e is added to $\mathscr{E}$ in line 17. Then, the skills of e are assigned to p in line 20. We then update 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 22), p is added to $\mathscr{P}$ (line 23). 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. The worst-case running time of each iteration of Expert Pick Strategy is $O(kmn)$.

Table 3.1: Expert Details

Experts	**Cost**	**Skills**
A	40	AI, Java, DB
B	30	C, ML
C	20	C, AI, DB
D	40	Java, ML, DB

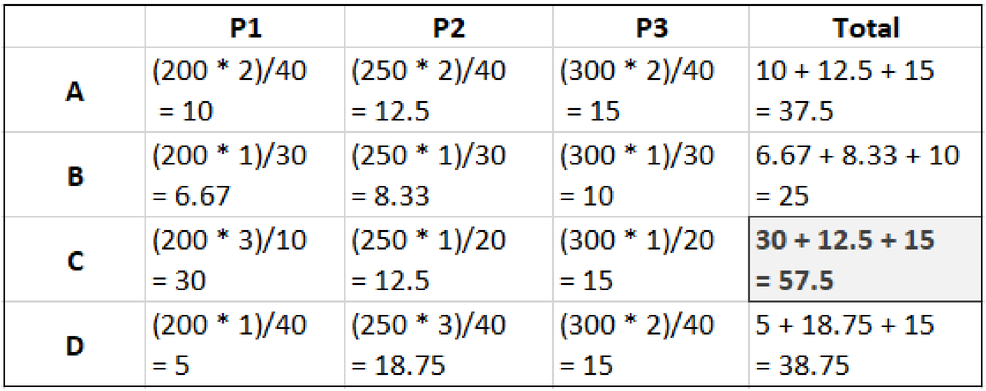
Table 3.2: Project Details

Projects	**Profit**	**Skills**
P1	200	AI, DB, C
P2	250	Java, DB, ML
P3	300	Java, AI, ML

Illustration 1: In the given set of projects with respective required skills and profit as represented in table 3.2, determine a group of experts to satisfy all skills required by those selected projects from table 3.1, within the given Budget of 100 by the Expert Pick Strategy.

Solution:

Iteration 1: First we calculate the scores for all pairs of experts and projects.

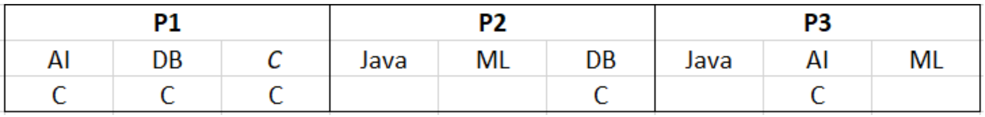


	P1	P2	P3	Total
A	(200 * 2)/40 = 10	(250 * 2)/40 = 12.5	(300 * 2)/40 = 15	10 + 12.5 + 15 = 37.5
B	(200 * 1)/30 = 6.67	(250 * 1)/30 = 8.33	(300 * 1)/30 = 10	6.67 + 8.33 + 10 = 25
C	(200 * 3)/10 = 30	(250 * 1)/20 = 12.5	(300 * 1)/20 = 15	**30 + 12.5 + 15 = 57.5**
D	(200 * 1)/40 = 5	(250 * 3)/40 = 18.75	(300 * 2)/40 = 15	5 + 18.75 + 15 = 38.75

Figure 3.1: Illustration 1: Iteration 1

Here, Expert C achieved the highest score. Therefore, in the first iteration Expert C is selected (Figure 3.1).

The situation after the first iteration 1 (Figure 3.2):

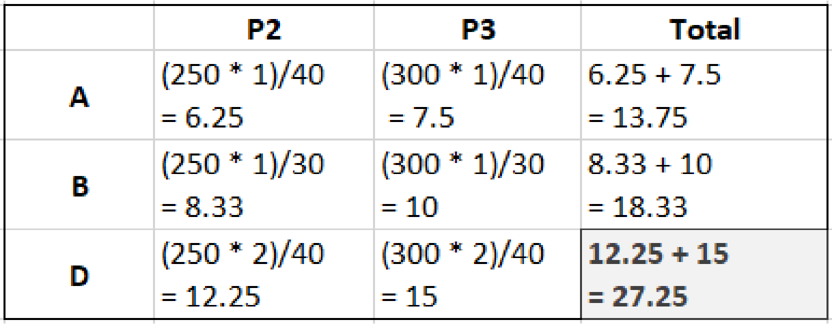


P1			P2			P3		
AI	DB	*C*	Java	ML	DB	Java	AI	ML
C	C	C			C		C	

Figure 3.2: Illustration 1: Team after Iteration 1

Remaining Budget = 100 - Cost of Expert C = 100 - 20 = 80

Iteration 2: We again calculate the scores for the remaining pairs of experts and projects.

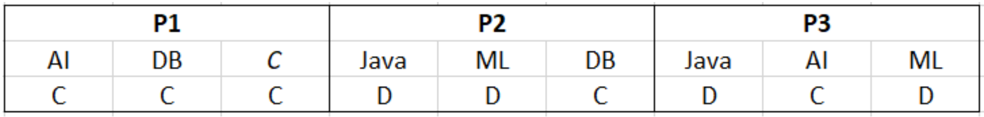


	P2	P3	Total
A	(250 * 1)/40 = 6.25	(300 * 1)/40 = 7.5	6.25 + 7.5 = 13.75
B	(250 * 1)/30 = 8.33	(300 * 1)/30 = 10	8.33 + 10 = 18.33
D	(250 * 2)/40 = 12.25	(300 * 2)/40 = 15	**12.25 + 15 = 27.25**

Figure 3.3: Illustration 1: Iteration 2

Here, Expert D achieved the highest score. Therefore, in the second iteration, expert D is selected (Figure 3.3).

Team after first iteration 2 (Figure 3.4):



P1			P2			P3		
AI	DB	C	Java	ML	DB	Java	AI	ML
C	C	C	D	D	C	D	C	D

Figure 3.4: Illustration 1: Team after Iteration 2

Remaining Budget = 80 - Cost of Expert D = 80 - 40 = 40

After iteration 2, all the project requirements are satisfied. Hence, the team after the second iteration will be the final team for this group of projects.

Final Team = {C, D}

3.1.1 Expert Pick Strategy with Communication Cost and Expert Capacity

In this section, we introduce a new version of our Algorithm 1 with the inclusion of Communication Cost and Expert Capacity with another institution. In each iteration, we assign a score to each pair of expert/projects and choose the expert with the highest score. The score is designed based on the following intuitions:

- We want to choose a *less expensive 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 can *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 much past collaboration with other experts. To take into account all these objectives, we design the following score function for each pair of projects and experts.

$$sc_e^p \leftarrow \lambda . \frac{PF(p). \min\{Skill(e,p),\ Cap(e)\}}{C(e)} + (1-\lambda). \frac{a}{\sum_{e' \in \mathscr{E}} Dist(e,e')} \quad (3.2)$$

Recall that λ is the trade-off parameter between profit and communication cost (see Definition 2). Note that a is normalization constant, $Skill(e,p)$ 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 an 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 an 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 can 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 of 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 2 is a similar approach to Algorithm 1 to find a group of collaborative experts while maximizing the profit of covered projects. This algorithm receives similar inputs as Algorithm 1 such as the set of n experts, set of m skills, set of k projects, network of experts G with the inclusion of Communication Cost and Expert Capacity, and the available budget B as input with the addition of the trade-off parameter λ. The output of this algorithm is a subset of projects $\mathscr{P}$ and a group of experts $\mathscr{E}$ that covers all required skills in $\mathscr{P}$ while maximizing the objective of Problem 1. Furthermore, the sum of the salary of the experts in $\mathscr{E}$ is not more than the given budget B.

Algorithm 2 Cluster Hire with Expert Pick Strategy with Communication Cost and Expert Capacity

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, trade-off parameter λ, capacity of each expert $Cap(e)$, normalization constant a, and budget B.
Output: subset of projects $\mathscr{P} \subseteq P$ and a group of experts $\mathscr{E} \subseteq E$ that maximize $ProfitCC(\mathscr{P}, \mathscr{E})$ under the given budget B.

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

In line 1, $\mathscr{E}$ and $\mathscr{P}$ 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 $\mathscr{E}$ as long as $b < B$. Line 2 starts the iteration of the greedy algorithm. As long as we have experts in E that have not been added to $\mathscr{E}$ and we have not overspent the budget, we can consider adding more experts to $\mathscr{E}$. This is the condition of the while loop in line 2. In line 3, we store all unassigned experts in E whose salary is within the remaining budget to $\mathscr{R}$. The for loop in line 4 checks to see if each expert in $\mathscr{R}$ possesses at least one required skill in an uncovered project. If this is not the case, those experts are removed from $\mathscr{R}$ as these experts are useless for the remaining projects. In line 7, we check to see if $\mathscr{R}$ is empty or not. If it is empty, we return the current $\mathscr{E}$ and $\mathscr{P}$ and terminate the algorithm. The reason is that, if $\mathscr{R}$ is empty, we have no other experts to add to $\mathscr{E}$. In lines 9 to 17, we assign a score to each pair of uncovered project p (projects in $P/\mathscr{P}$) and expert e in $\mathscr{R}$. Later, we choose the highest score and add the associated expert to $\mathscr{E}$. 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 3.2. Note that in the first iteration (line 12, $\mathscr{E} = \emptyset$), we do not consider the communication cost as set $\mathscr{E}$ is empty. In line 18, we choose pair (e, p) in which their score sc_e^p is maximized among all pairs. e is added to $\mathscr{E}$ 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 $\mathscr{P}$ (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 $\mathscr{E}$ in the 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), 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 $\mathscr{P}$.

Table 3.3: Expert Details with Capacity

Experts	**Cost**	**Skills**	**Capacity**
A	40	AI, Java, DB	3
B	30	C, ML	4
C	20	C, AI, DB	2
D	40	Java, ML, DB	5

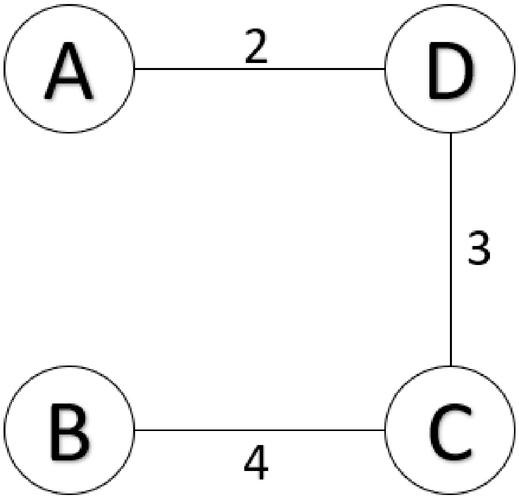
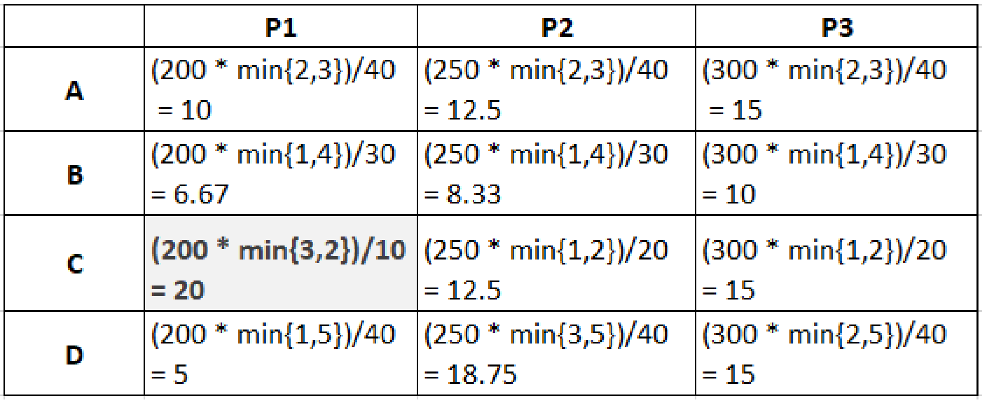


Figure 3.5: Network of Experts

Illustration 2: In the given set of projects with respective required skills and profit as represented in table 3.2, determine a group of experts to satisfy all skills required by the selected projects from table 3.3 within given Budget B of 100 by the Expert Pick Strategy with Communication Cost and Expert Capacity. (Note: Figure 3.5 indicates the communication cost between the experts by weighted edges. Consider Lambda = 0.5 and Constant a = 50.)

Solution:

Iteration 1: As there is no expert in the team, communication cost will not be considered. First, we calculate scores for all pairs of experts and projects.

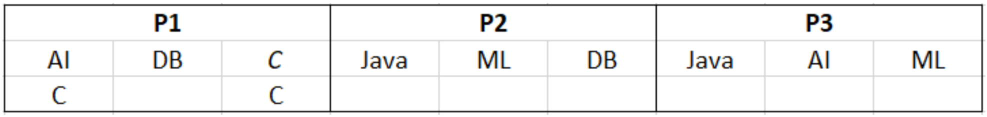


	P1	P2	P3
A	(200 * min{2,3})/40 = 10	(250 * min{2,3})/40 = 12.5	(300 * min{2,3})/40 = 15
B	(200 * min{1,4})/30 = 6.67	(250 * min{1,4})/30 = 8.33	(300 * min{1,4})/30 = 10
C	**(200 * min{3,2})/10 = 20**	(250 * min{1,2})/20 = 12.5	(300 * min{1,2})/20 = 15
D	(200 * min{1,5})/40 = 5	(250 * min{3,5})/40 = 18.75	(300 * min{2,5})/40 = 15

Figure 3.6: Illustration 2: Iteration 1

Here, Expert C achieved the highest score for P1. Therefore, in the first iteration, Expert C is selected (Figure 3.6). The capacity of expert C is 2 which is less than the possessed skills in P1. In that case, Expert C is assigned to rare skills first.

Team after iteration 1 (Figure 3.7):



P1			P2			P3		
AI	DB	*C*	Java	ML	DB	Java	AI	ML
C		C						

Figure 3.7: Illustration 2: Team after Iteration 1

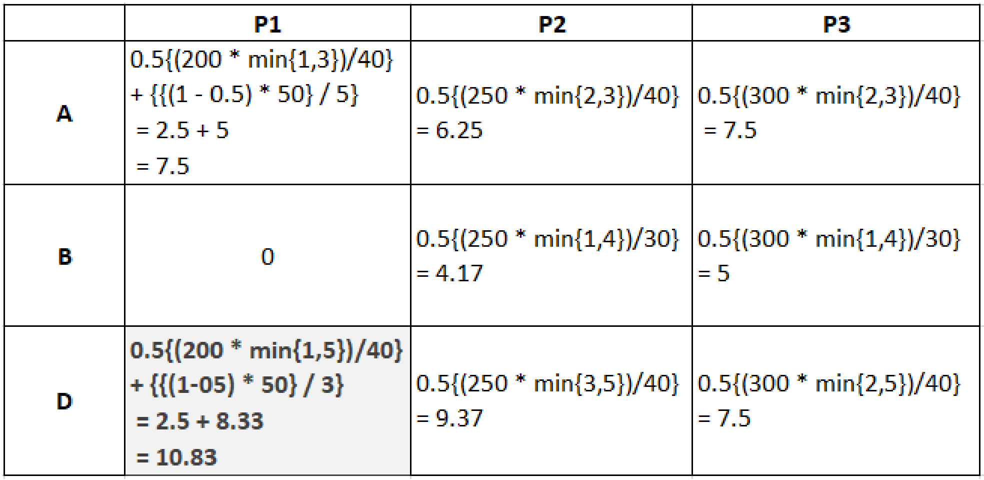
Remaining Budget = 100 - Cost of Expert C = 100 - 20 = 80

Remaining Capacity of Experts after Iteration 1:

- Expert A = 3
- Expert B = 4

- Expert D = 5

Iteration 2: We again calculate scores for remaining pairs of experts and projects.



	P1	P2	P3

A	0.5{(200 * min{1,3})/40}
	+ {{(1 - 0.5) * 50} / 5}
	= 2.5 + 5
	= 7.5
		0.5{(250 * min{2,3})/40}
		= 6.25
			0.5{(300 * min{2,3})/40}
			= 7.5

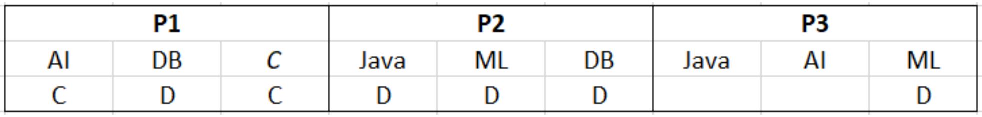
B	0	0.5{(250 * min{1,4})/30} = 4.17	0.5{(300 * min{1,4})/30} = 5

D	**0.5{(200 * min{1,5})/40}
	+ {{(1-05) * 50} / 3}
	= 2.5 + 8.33
	= 10.83**
		0.5{(250 * min{3,5})/40}
		= 9.37
			0.5{(300 * min{2,5})/40}
			= 7.5

Figure 3.8: Illustration 2: Iteration 2

Here, Expert D achieved the highest score for P1. Therefore, in the second iteration, Expert D is selected (Figure 3.8). The capacity of expert D is 5; so expert D is utilized in other projects too after the ones in P1.

Team after iteration 2 (Figure 3.9):



P1			P2			P3		
AI	DB	C	Java	ML	DB	Java	AI	ML
C	D	C	D	D	D			D

Figure 3.9: Illustration 2: Team after Iteration 2

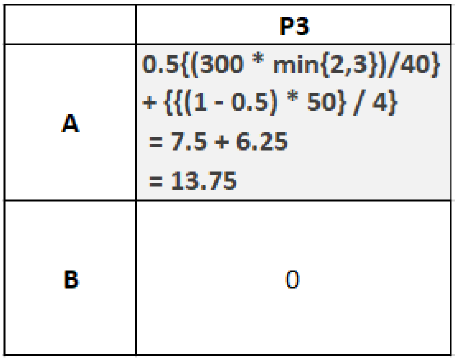
Remaining Budget = 80 - Cost of Expert D = 80 - 40 = 40

Remaining Capacity of Experts after Iteration 2:

- Expert A = 3

- Expert B = 4

Iteration 3: We again calculate scores for remaining pairs of experts and projects.



	P3

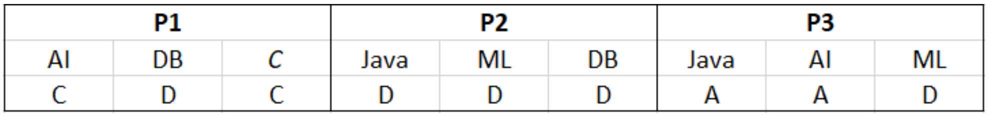
A	0.5{(300 * min{2,3})/40}
	+ {{(1 - 0.5) * 50} / 4}
	= 7.5 + 6.25
	= 13.75

B	0

Figure 3.10: Illustration 2: Iteration 3

Here, Expert A achieved the highest score for P3. Therefore, in the third iteration, Expert A is selected (Figure 3.10).

Team after iteration 3 (Figure 3.11):



P1			P2			P3		
AI	DB	C	Java	ML	DB	Java	AI	ML
C	D	C	D	D	D	A	A	D

Figure 3.11: Illustration 2: Team after Iteration 3

Remaining Budget = 40 - Cost of Expert A = 40 - 40 = 0

After iteration 3, all the project requirements are satisfied. Hence, the team after the third iteration will be the final team for this group of projects.

Final Team = {A, C, D}

3.2 Project Pick Strategy

The second approach to find a group of collaborative experts by covering 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 for covering the required skills in the project should be *less expensive*.

Same as in the first strategy, these intuitions are not necessarily compatible. A high profitable project might need an 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. To do that, we use a modified version of the greedy weighted set cover algorithm. Recall that in the weighted set covering problem, 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 set (corresponding to the set of skills required for a given project in our problem). In the 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. In each iteration, and for any remaining project, we find a set of experts that can cover that project. To find this set for project p, we start with an empty set E_p. Then, we select an expert to be added to E_p that maximizes the following formula:

$$sc_e \leftarrow \frac{Skill(e,p)}{C(e)} \tag{3.3}$$

Here, $Skill(e,p)$ indicates the set of skills possessed by an expert e in project p and $C(e)$ indicates cost of an expert e. 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. To do that, we select a project that maximizes the following equation.

$$\frac{PF(p)}{\sum_{e \in E_p} C(e)} \tag{3.4}$$

Algorithm 3 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 $P = \{p_1, p_2, \ldots, p_k\}$, and budget B.
Output: subset of projects $\mathscr{P} \subseteq P$ and a group of experts $\mathscr{E} \subseteq E$ that maximize $Profit(\mathscr{P})$ under the given budget B.

```
$\mathscr{E} \leftarrow \emptyset$, $\mathscr{P} \leftarrow \emptyset$, $b \leftarrow 0$
while $b < B$ and $P/\mathscr{P} \neq \emptyset$ do
    $P' \leftarrow P/\mathscr{P}$, $\mathscr{R} \leftarrow \{e \mid e \in E$ and $e \notin \mathscr{E}$ and $C(e) + b \leq B\}$
    if $\mathscr{R} = \emptyset$ then
        return $\mathscr{E}$ and $\mathscr{P}$
    for all $p \in P'$ do
        $E_p \leftarrow \emptyset$, $S_p \leftarrow PS(p)$, $\mathscr{R}' \leftarrow \mathscr{R}$
        while $S_p \neq \emptyset$ do
            for all $e \in \mathscr{R}'$ do
                if $e$ covers at least one skill in $p$ then
                    $sc_e \leftarrow \frac{Skill(e,p)}{C(e)}$
                else
                    $sc_e \leftarrow 0$
            $e \leftarrow \arg\max_{e \in \mathscr{R}'} sc_e$
            add $e$ to $E_p$, update $S_p$
    for all $p \in P'$ do
        if $(\sum_{e \in E_p} C(e)) + b > B$ then
            remove $p$ from $P'$
    if $P' = \emptyset$ then
        return $\mathscr{E}$ and $\mathscr{P}$
    $(p, E_p) \leftarrow \arg\max_{p \in P'} \frac{PF(p)}{\sum_{e \in E_p} C(e)}$
    add $p$ to $\mathscr{P}$, assign skills of experts in $E_p$ to $p$
    for all $e \in E_p$ do
        add $e$ to $\mathscr{E}$, $b \leftarrow b + C(e)$
        assign skills of $e$ to $p$ in $P/\mathscr{P}$
return $\mathscr{E}$ and $\mathscr{P}$
```

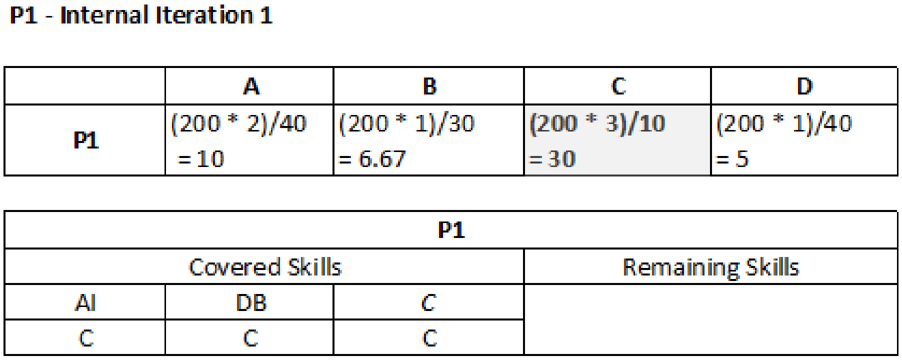
The intuition is the same. We are interested in choosing the project that has a high profit, needs experts with a low salary. Now, we are ready to present Algorithm 3 that returns a group of experts for performing a set of profitable projects with project pick strategy. The input and output of this algorithm are similar to Algorithm 1.

In the first line, we initialize three variables $\mathscr{E}$, $\mathscr{P}$, and b, which are responsible for storing 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 $\mathscr{E}$ will not violate the budget to set $\mathscr{R}$. If $\mathscr{R}$ 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 assigning a score to each project p in P'. Sets E_p, S_p, and $\mathscr{R}'$ 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. $\mathscr{R}'$ is a duplicate of $\mathscr{R}$. 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 $\mathscr{R}'$ and each iteration chooses the one that maximizes the modified weighted greedy set cover score. In lines 9 to 14, we assign a score to each pair of uncovered project p (projects in $P/\mathscr{P}$) and expert e in $\mathscr{R}$. Later, we choose the highest score and add the associated expert to E_p. If e does not cover any of the required skills in p (line 12), the score is set to 0 (line 13) 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 3.4. In line 14, we choose pair (e,p) in which their score sc_e is maximized among all other pairs. This expert is selected in line 17 and is added to E_p in line 15. We also update S_p in line 15. After finding the set of experts for all projects, in lines 16 to 18, we remove the projects in which adding their associated expert set to $\mathscr{E}$ violate the budget constraint. If set P' becomes empty after this operation, we terminate the algorithm in line 20. If P' is not empty, we select the best project p in P' in line 21 according to equation 3.4. In line 22, we add the best project to $\mathscr{P}$ and cover the skills of p. The for loop of line 23 iterates through all experts in E_p. These are the set of experts that are responsible for covering the best-selected project in line 21. In line 24, each expert in E_p is added to $\mathscr{E}$. We utilize all selected experts for all remaining projects in line 25. The worst-case running time of each iteration of Project Pick Strategy is $O(kmn)$.

Illustration 3: In the given set of projects with respective required skills and profit as represented in table 3.2, determine the group of experts to satisfy all skills required by the selected projects from table 3.1 within the given Budget B of 100 by the Project Pick Strategy.

Solution: In this method, we make temporary teams for all remaining projects in each iteration and proceed with the team who achieves the highest score.

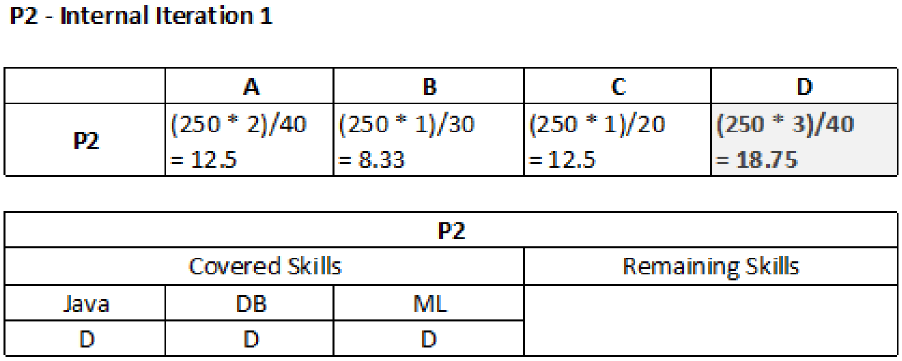
Iteration 1:



P1 - Internal Iteration 1

	A	B	C	D
P1	(200 * 2)/40 = 10	(200 * 1)/30 = 6.67	(200 * 3)/10 = 30	(200 * 1)/40 = 5

P1			
Covered Skills			Remaining Skills
AI	DB	C	
C	C	C	

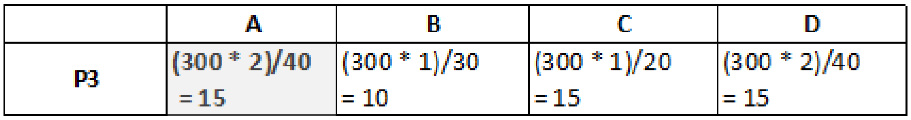


P2 - Internal Iteration 1

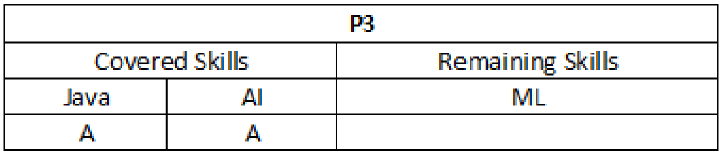
	A	B	C	D
P2	(250 * 2)/40 = 12.5	(250 * 1)/30 = 8.33	(250 * 1)/20 = 12.5	(250 * 3)/40 = 18.75

P2			
Covered Skills			Remaining Skills
Java	DB	ML	
D	D	D	

P3 - Internal Iteration 1

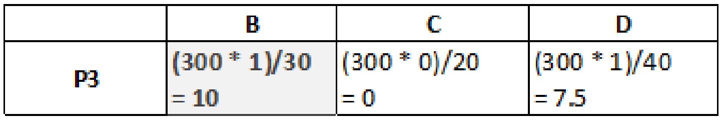


	A	B	C	D
P3	(300 * 2)/40 = 15	(300 * 1)/30 = 10	(300 * 1)/20 = 15	(300 * 2)/40 = 15

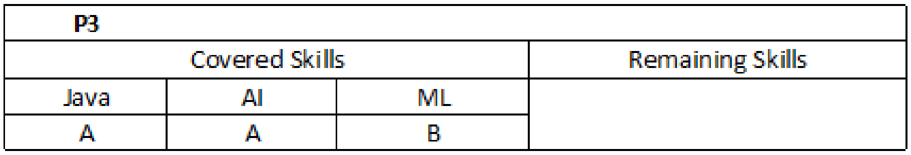


P3		
Covered Skills		Remaining Skills
Java	AI	ML
A	A	

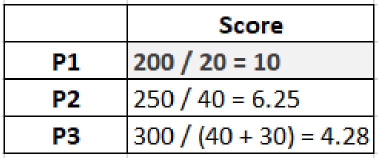
P3 - Internal Iteration 2



	B	C	D
P3	(300 * 1)/30 = 10	(300 * 0)/20 = 0	(300 * 1)/40 = 7.5



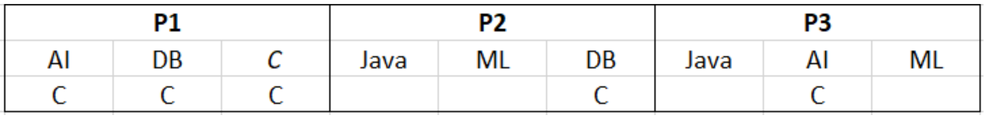
P3			
Covered Skills			Remaining Skills
Java	AI	ML	
A	A	B	



	Score
P1	200 / 20 = 10
P2	250 / 40 = 6.25
P3	300 / (40 + 30) = 4.28

As P1 achieved the highest score in Iteration 1, Expert C is selected for the team and will be utilized for the other projects.

Team after iteration 1 (Figure 3.12):

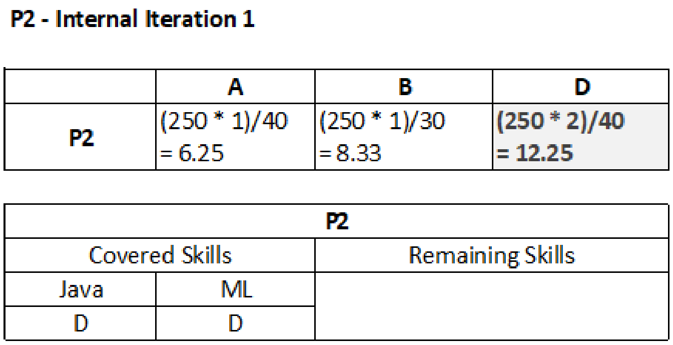


P1			P2			P3		
AI	DB	C	Java	ML	DB	Java	AI	ML
C	C	C			C		C	

Figure 3.12: Illustration 3: Team after Iteration 1

Remaining Budget = 100 - Cost of an Expert C = 100 - 20 = 80

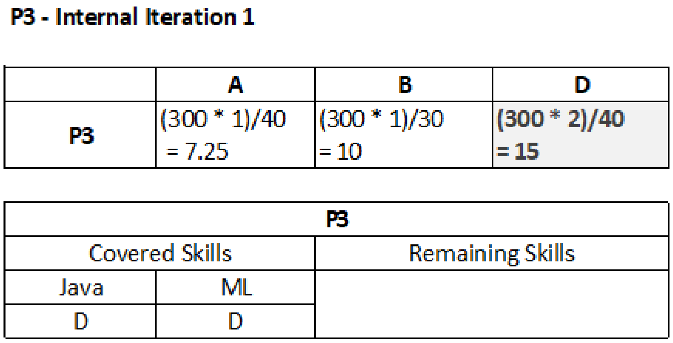
Iteration 2:



P2 - Internal Iteration 1

	A	B	D
P2	(250 * 1)/40 = 6.25	(250 * 1)/30 = 8.33	**(250 * 2)/40 = 12.25**

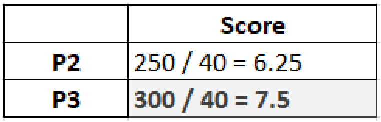
P2		
Covered Skills		Remaining Skills
Java	ML	
D	D	



P3 - Internal Iteration 1

	A	B	D
P3	(300 * 1)/40 = 7.25	(300 * 1)/30 = 10	**(300 * 2)/40 = 15**

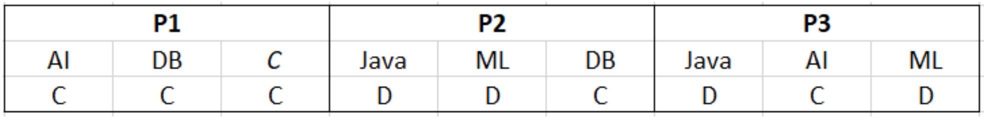
P3		
Covered Skills		Remaining Skills
Java	ML	
D	D	



	Score
P2	250 / 40 = 6.25
P3	**300 / 40 = 7.5**

As P3 achieved the highest score in Iteration 2, Expert D is selected for the team and will be utilized for the other projects.

Team after iteration 2 (Figure 3.13):



P1			**P2**			**P3**		
AI	DB	C	Java	ML	DB	Java	AI	ML
C	C	C	D	D	C	D	C	D

Figure 3.13: Illustration 3: Team after Iteration 2

Remaining Budget = 80 - Cost of an Expert D = 80 - 40 = 40

After iteration 2, all the project requirements are satisfied. Hence, the team after the second iteration will be the final team for this group of projects.

Final Team = $\{$C, D$\}$

3.2.1 Project Pick Strategy with Communication Cost and Expert Capacity

In this section, we introduce an improved version of our Algorithm 3 with the inclusion of Communication Cost and Expert Capacity with the additional institution. In each iteration, we define new teams for all remaining projects based on existing teams and remaining skills and choose the set of experts with the highest score. The score is designed based on the following intuitions:

- We want to choose a project with *high profit*.
- The set of experts responsible for covering 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 an existing group of experts.

Same as in the first strategy, these intuitions are not necessarily compatible. A high profitable project might need an 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. To do that, we use a modified version of the weighted set covering problem. Recall that in the weighted set covering problem, 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 set (corresponding to the set of skills required for a given project in our problem). In the 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 can cover that project. To find this set for project p, we start with an empty set E_p. Then, we select an expert to be added to E_p that maximizes the following equation:

$$sc_e \leftarrow \lambda . \frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)} + \\ (1-\lambda) . \frac{a}{\sum_{e' \in E_p} Dist(e,e')} \tag{3.5}$$

Here, $Skill(e,p)$ indicates the set of skills possessed by an expert e in project p. $C(e)$ indicates cost of an expert e, λ is the trade-off parameter, and a & b (formula 3.6) are normalization constants. 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. To do that, we select a project that maximizes the following formula.

$$\lambda . \frac{PF(p)}{\sum_{e \in E_p} C(e)} + (1-\lambda) . \frac{b}{\sum_{e \in E_p} \sum_{e' \in \mathscr{E}} Dist(e,e')} \tag{3.6}$$

The intuition is the same. We are interested in choosing the project that has a high profit, needs experts with low salary, and the set of experts responsible for performing the project's tasks has small communication cost with existing experts. Now, we are ready to present Algorithm 4 which returns a group of experts for performing a set of profitable projects with project pick strategy with the inclusion of communication cost and expert capacity. The input and output of this algorithm are similar to Algorithm 2.

Algorithm 4 Cluster Hire with Project Pick Strategy with Communication Cost and Expert Capacity

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, trade-off parameter λ, capacity of each expert $Cap(e)$, normalization constants a & b, and budget B.
Output: subset of projects $\mathscr{P} \subseteq P$ and a group of experts $\mathscr{E} \subseteq E$ that maximize $ProfitCC(\mathscr{P}, \mathscr{E})$ under the given budget B.

```
$\mathscr{E} \leftarrow \emptyset$, $\mathscr{P} \leftarrow \emptyset$, $b \leftarrow 0$
while $b < B$ and $P/\mathscr{P} \neq \emptyset$ do
    $P' \leftarrow P/\mathscr{P}$, $\mathscr{R} \leftarrow \{e \mid e \in E$ and $e \notin \mathscr{E}$ and $C(e) + b \leq B\}$
    if $\mathscr{R} = \emptyset$ then
        return $\mathscr{E}$ and $\mathscr{P}$
    for all $p \in P'$ do
        $E_p \leftarrow \emptyset$, $S_p \leftarrow PS(p)$, $\mathscr{R}' \leftarrow \mathscr{R}$
        while $S_p \neq \emptyset$ do
            for all $e \in \mathscr{R}'$ do
                if $e$ covers at least one skill in $p$ then
                    if $E_p = \emptyset$ then
                        $sc_e \leftarrow \frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)}$
                    else
                        $sc_e \leftarrow \lambda . \frac{\min\{Skill(e,p),\ Cap(e)\}}{C(e)} +$
                            $(1-\lambda) . \frac{a}{\sum_{e' \in E_p} Dist(e,e')}$
                else
                    $sc_e \leftarrow 0$
            $e \leftarrow \arg\max_{e \in \mathscr{R}'}\ sc_e$
            add $e$ to $E_p$, update $S_p$
    for all $p \in P'$ do
        if $(\sum_{e \in E_p} C(e)) + b > B$ then
            remove $p$ from $P'$
    if $P' = \emptyset$ then
        return $\mathscr{E}$ and $\mathscr{P}$
    $(p, E_p) \leftarrow \arg\max_{p \in P'}\ \lambda . \frac{PF(p)}{\sum_{e \in E_p} C(e)} +$
        $(1-\lambda) . \frac{b}{\sum_{e \in E_p} \sum_{e' \in \mathscr{E}} Dist(e,e')}$
    add $p$ to $\mathscr{P}$, assign skills of experts in $E_p$ to $p$
    for all $e \in E_p$ do
        add $e$ to $\mathscr{E}$, update $Cap(e)$, $b \leftarrow b + C(e)$
        while $Cap(e) > 0$ do
            $s \leftarrow$ rarest skill in $e$ which is required by a $p$ in $P/\mathscr{P}$
            assign skill $s$ to the most expensive $p$ in $P/\mathscr{P}$
            update $Cap(e)$
return $\mathscr{E}$ and $\mathscr{P}$
```

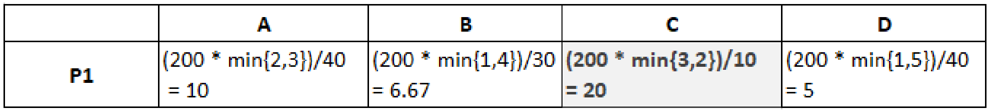
In the first line, we initialize three variables $\mathscr{E}$, $\mathscr{P}$, and b, which are responsible for storing the final group of experts, the selected projects, and amount of budget spent so far, respectively. The while loop of line 2 iterates until we run out of budget or no project is left 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 $\mathscr{E}$ will not violate the budget to set $\mathscr{R}$. If $\mathscr{R}$ 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 discussed 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 $\mathscr{R}'$ 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. $\mathscr{R}'$ is a duplicate of $\mathscr{R}$. 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 $\mathscr{R}'$ and each iteration chooses the one that maximizes the modified weighted greedy set cover score. In lines 9 to 14, we assign a score to each pair of uncovered project p (projects in $P/\mathscr{P}$) and expert e in $\mathscr{R}$. Later, we choose the highest score and add the associated expert to E_p. If e does not cover any of the required skills in p (line 12), the score is set to 0 (line 13) 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 3.4. In line 14, we choose pair (e,p) in which their score sc_e is maximized among all other pairs. 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 $\mathscr{E}$ 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 3.4. In line 25, we add the best project to $\mathscr{P}$ 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 for covering the best-selected project in line 24. In line 27, each expert in E_p is added to $\mathscr{E}$ 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 of doing so is the same as the last part of Algorithm 2: once we hired an expert, we prefer to use her maximum capacity.

Illustration 4: In the given set of projects with respective required skills and profit as represented in table 3.2, determine a group of experts to satisfy all skills required by selected projects from table 3.3 within given Budget B of 100 by the Project Pick Strategy with Communication Cost and Expert Capacity. (Note: Figure 3.5 indicates the communication cost between the experts by weighted edges. Lambda = 0.5, and Constant a = 50 & b = 10.)

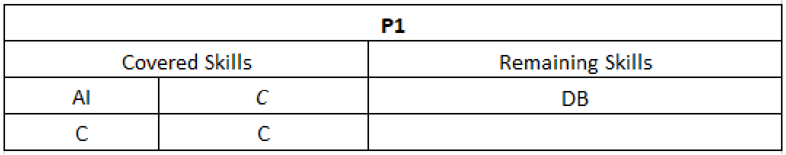
Solution: In this method, we make temporary teams for all remaining projects in each iteration and proceed with the team who achieves the highest score.

Iteration 1: As there is no expert in the team, communication cost will not be considered. First we calculate scores for all pairs of experts and projects.

P1 - Internal Iteration 1

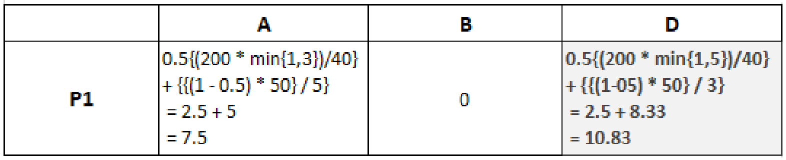


	A	B	C	D
P1	(200 * min{2,3})/40 = 10	(200 * min{1,4})/30 = 6.67	**(200 * min{3,2})/10 = 20**	(200 * min{1,5})/40 = 5



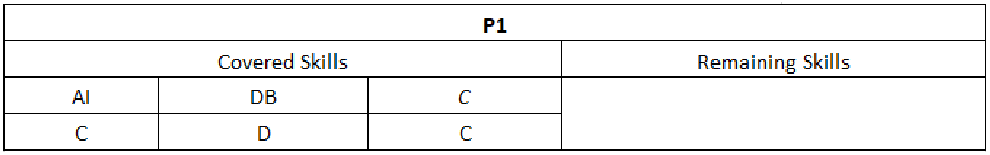
P1		
Covered Skills		Remaining Skills
AI	*C*	DB
C	C	

P1 - Internal Iteration 2



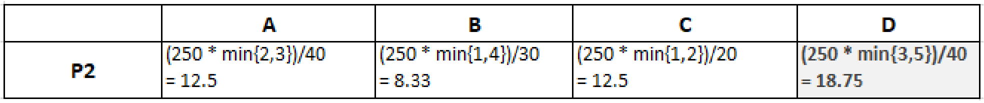
	A	B	D

P1	0.5{(200 * min{1,3})/40}
	+ {{(1 - 0.5) * 50} / 5}
	= 2.5 + 5
	= 7.5
		0	**0.5{(200 * min{1,5})/40}
			+ {{(1-05) * 50} / 3}
			= 2.5 + 8.33
			= 10.83**

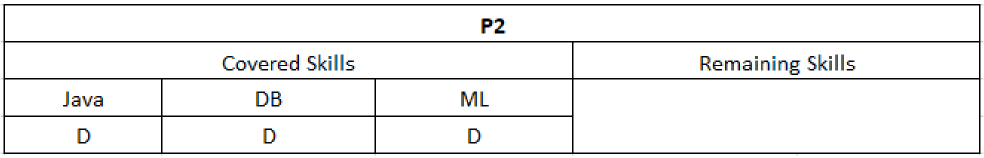


P1			
Covered Skills			Remaining Skills
AI	DB	*C*	
C	D	C	

P2 - Internal Iteration 1

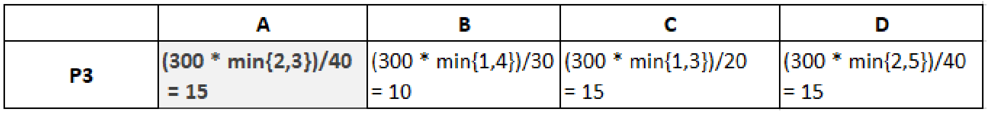


	A	B	C	D
P2	(250 * min{2,3})/40 = 12.5	(250 * min{1,4})/30 = 8.33	(250 * min{1,2})/20 = 12.5	**(250 * min{3,5})/40 = 18.75**

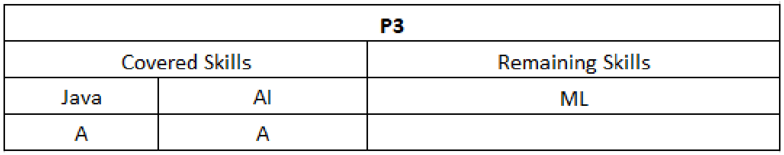


P2			
Covered Skills			Remaining Skills
Java	DB	ML	
D	D	D	

P3 - Internal Iteration 1

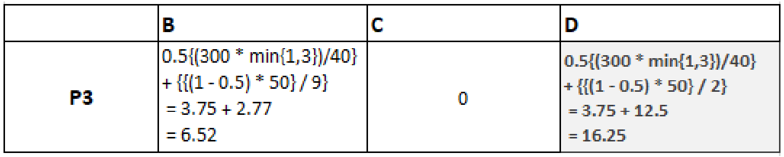


	A	B	C	D
P3	(300 * min{2,3})/40 = 15	(300 * min{1,4})/30 = 10	(300 * min{1,3})/20 = 15	(300 * min{2,5})/40 = 15



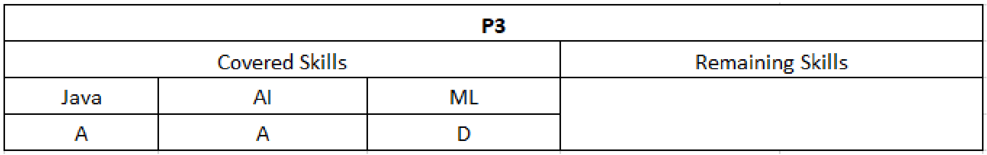
P3		
Covered Skills		Remaining Skills
Java	AI	ML
A	A	

P3 - Internal Iteration 2

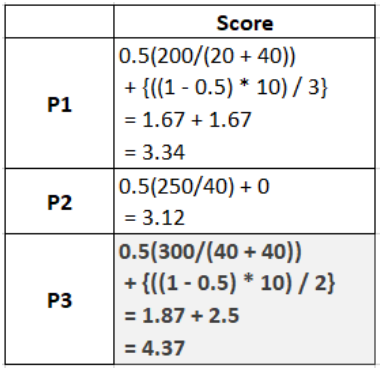


	B	C	D

P3	0.5{(300 * min{1,3})/40}
	+ {{(1 - 0.5) * 50} / 9}
	= 3.75 + 2.77
	= 6.52
		0	0.5{(300 * min{1,3})/40}
			+ {{(1 - 0.5) * 50} / 2}
			= 3.75 + 12.5
			= 16.25



P3			
Covered Skills			Remaining Skills
Java	AI	ML	
A	A	D	



	Score

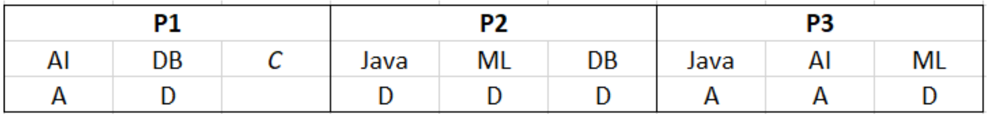
P1	0.5(200/(20 + 40))
	+ {((1 - 0.5) * 10) / 3}
	= 1.67 + 1.67
	= 3.34

P2	0.5(250/40) + 0 = 3.12

P3	0.5(300/(40 + 40))
	+ {((1 - 0.5) * 10) / 2}
	= 1.87 + 2.5
	= 4.37

As P3 achieved the highest score in Iteration 1, Experts A & D are selected for the team and will be utilized for the other projects.

Team after iteration 1 (Figure 3.14):



P1			P2			P3		
AI	DB	C	Java	ML	DB	Java	AI	ML
A	D		D	D	D	A	A	D

Figure 3.14: Illustration 4: Team after Iteration 1

Remaining Budget = 100 - Cost of an Expert A - Cost of an Expert D = 100 - 40 - 40 = 20

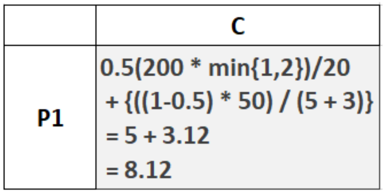
Remaining Capacity of Experts after Iteration 1:

- Expert B = 4
- Expert C = 2

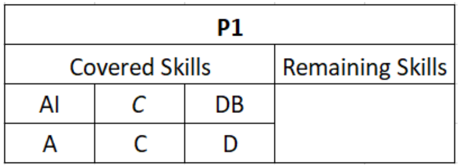
Here, the cost of an Expert B is higher than the remaining budget. Therefore Expert B will not be considered in the further iterations.

Iteration 2:

P1 - Internal Iteration 1



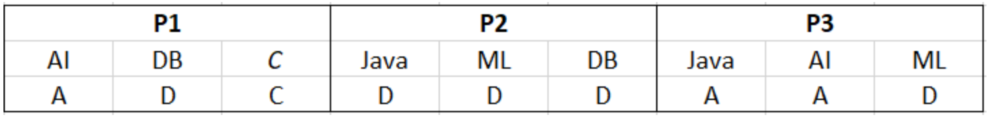
	C
P1	0.5(200 * min{1,2})/20 + {((1-0.5) * 50) / (5 + 3)} = 5 + 3.12 = 8.12



P1			
Covered Skills			Remaining Skills
AI	*C*	DB	
A	C	D	

As Expert C satisfies the requirements of P1, Expert C is selected.

Team after iteration 2 (Figure 3.14):



P1			**P2**			**P3**		
AI	DB	*C*	Java	ML	DB	Java	AI	ML
A	D	C	D	D	D	A	A	D

Figure 3.15: Illustration 4: Team after Iteration 2

Remaining Budget = 20 - 20 = 0

After iteration 2, all the project requirements are satisfied. Hence, the team after the second iteration will be the final team for this group of projects.

Final Team = $\{$A, C, D$\}$

Chapter 4

EXPERIMENTS

In this chapter, we evaluate the performance of our proposed algorithms to find the best group of experts for the problem of a 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 [3, 7]. The dataset contains information about a set of papers and their authors. Same as in [3, 7], 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 contains 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 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. Same as in [8], 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 stated.

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

4.1 Comparison with Previous Approach

We compare the results with the previous approach, which selects the group among the various random groups that tend to complete the given set of projects within the required budget. We compare our results with previous approach implemented in Java and executed on an Intel Core i7 2.8GHz computer with 16GB of RAM.

4.1.1 Total Profit vs. Budget

In this section, we are evaluating the effect of the budget on the total profit of projects. Figure 4.1 shows the values of total profit for increasing values of the budget for the different number of projects. Here, for comparison with previous approach we will consider k projects, where $k = \{10, 25\}$. We then calculate the total profit for different values of the budget. The results suggest that our Expert Pick approach achieves a higher total profit than previous approach when we have a limited budget. For the higher values of the budget, Expert Pick, Project Pick and previous Project-Greedy approach return the same profit. Moreover, as observed in figure 4.1 and 4.2, 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.

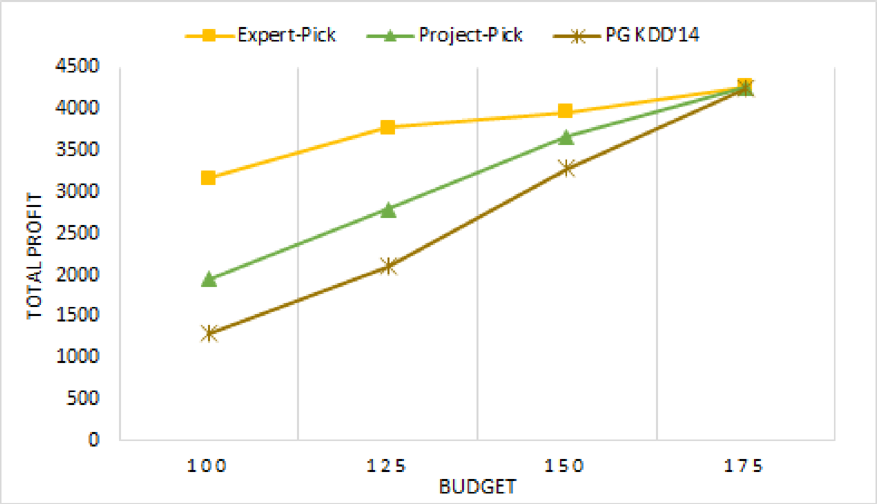


Figure 4.1: Total Profit vs Budget (No. of Projects = 10)

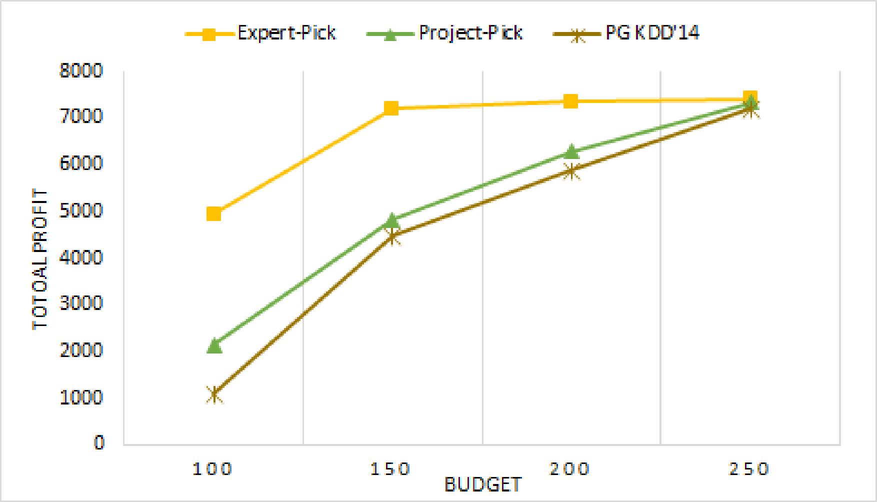


Figure 4.2: Total Profit vs Budget (No. of Projects = 25)

4.1.2 Number of Completed Projects vs. Budget

Figure 4.3 and 4.4 shows the number of completed projects for increasing value of the budget for a different number of projects. The results suggest that our Expert Pick approach completes more projects than our Project Pick and previous Project-Greedy approach when we have a limited budget. For higher values of the budget, they complete the same number of projects (almost all the projects are completed).

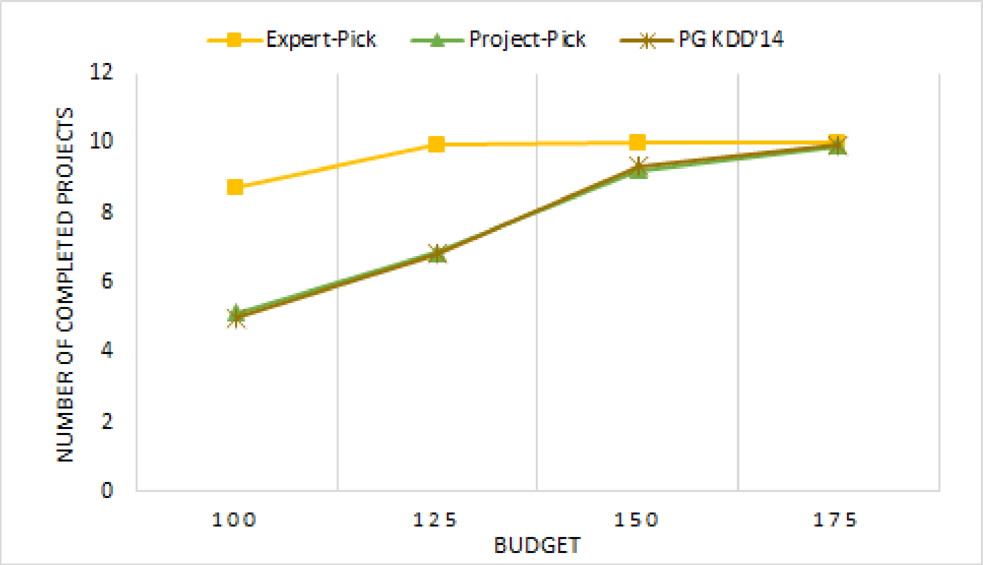


Figure 4.3: Number of Completed Projects vs Budget (No. of Projects = 10)

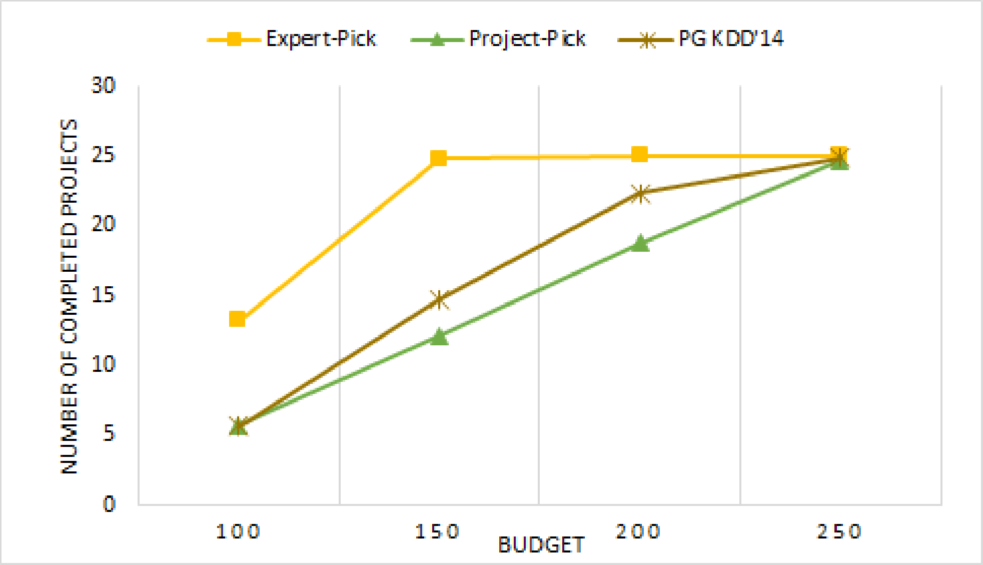


Figure 4.4: Number of Completed Projects vs Budget (No. of Projects = 25)

4.1.3 Discussion

As observed in the above figure 4.1, 4.2, 4.3 and 4.4, previous project-greedy approach completes the projects with higher total profit. To do so, the approach allocates the experts with a higher salary to the project, resulting in only a few projects covered in the approved budget. In contrast, our approach concentrates on picking less expensive experts, so that we can cover some projects in less budget.

4.2 Experiments with our Improvement

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 maximize 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.

4.2.1 Total Profit vs. Budget

We start by evaluating the effect of the budget on the total profit of the projects. Figure 4.5, 4.6, 4.7 and 4.8 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 the total profit for different values of the budget. Recall that λ is set to 0.5. The results suggest that Project-Pick achieves a higher total profit than Expert-Pick when we have the limited budget. For higher values of the budget, they return the same profit. As 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.

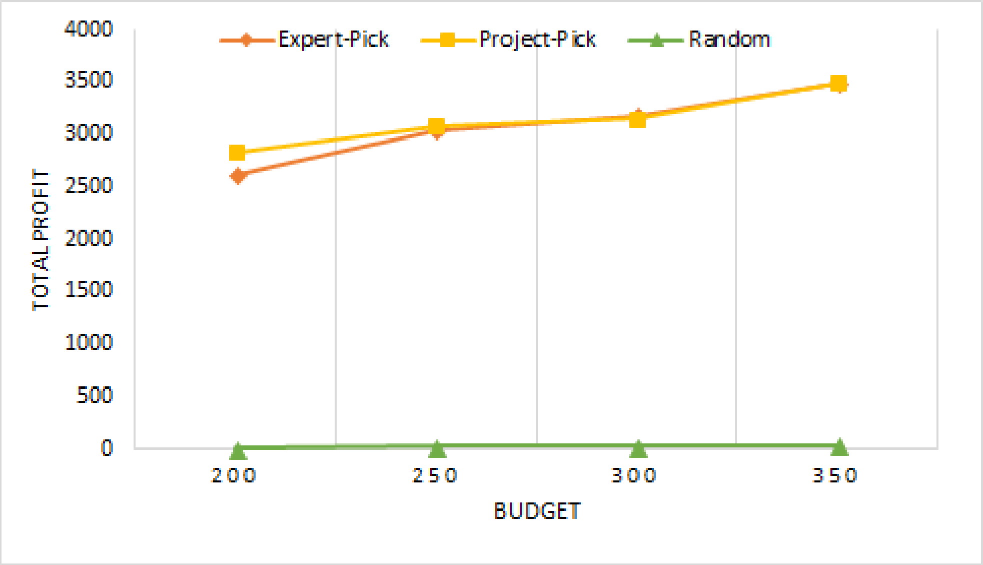


Figure 4.5: Total Profit vs Budget (No. of Projects = 10)

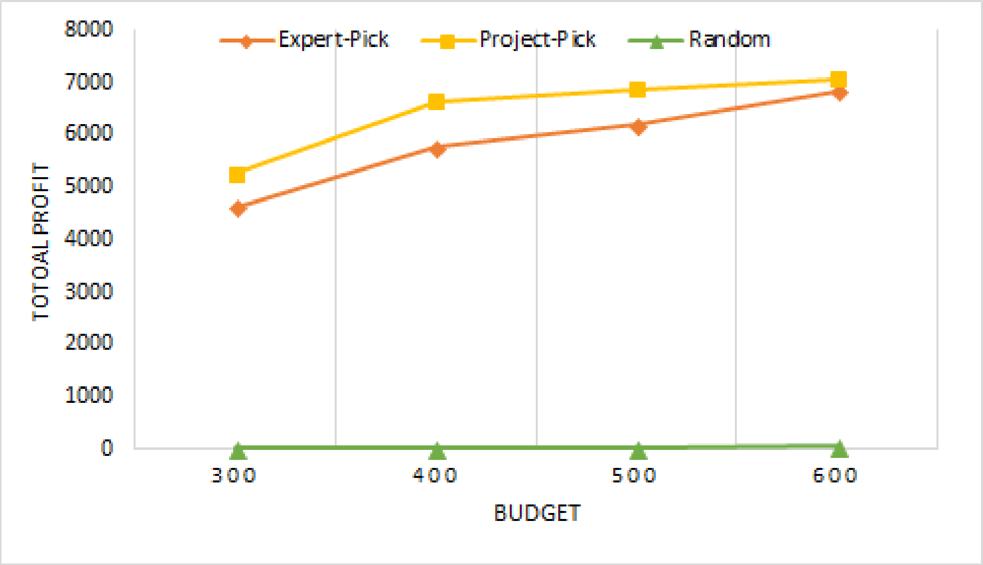


Figure 4.6: Total Profit vs Budget (No. of Projects = 25)

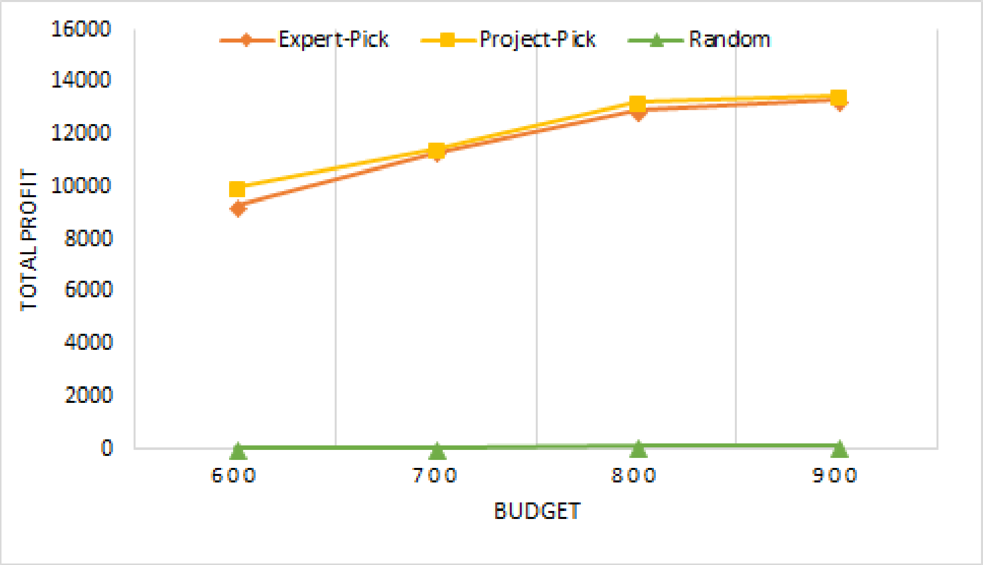


Figure 4.7: Total Profit vs Budget (No. of Projects = 40)

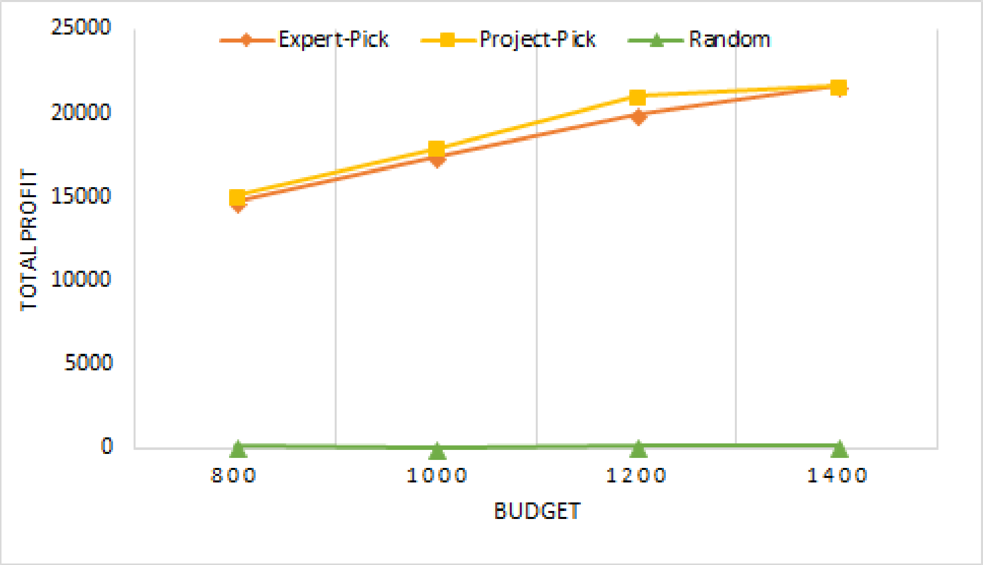


Figure 4.8: Total Profit vs Budget (No. of Projects = 60)

4.2.2 Communication Cost vs. Budget

Now, we study the effect of the budget on communication cost. Figure 4.9, 4.10, 4.11

and 4.12 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 the value, the more collaborative the group of experts are. The results suggest that Expert-Pick has significant better communication cost. Note that the communication cost of the Random is lower than others due to very small number of selected experts.

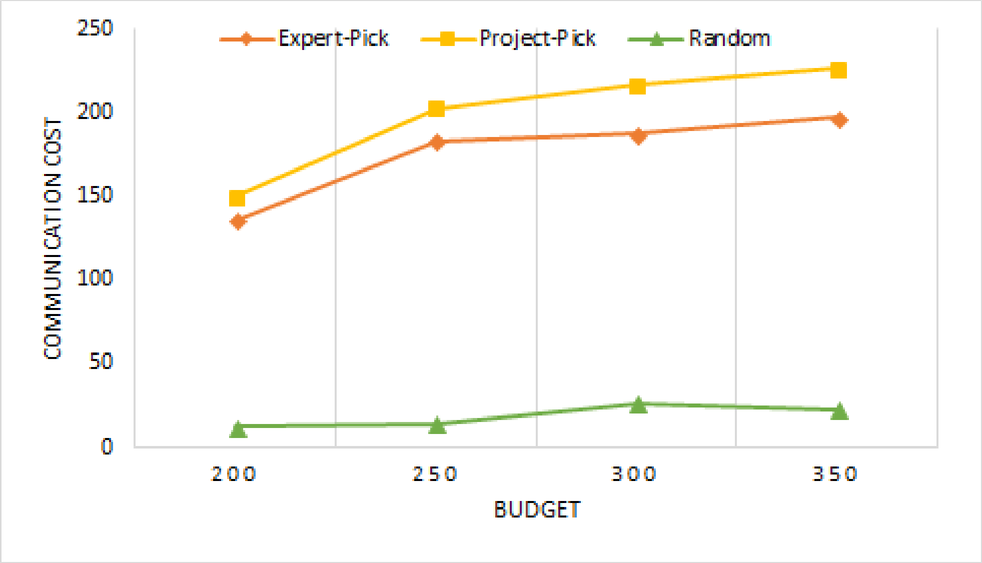


Figure 4.9: Communication Cost vs Budget (No. of Projects = 10)

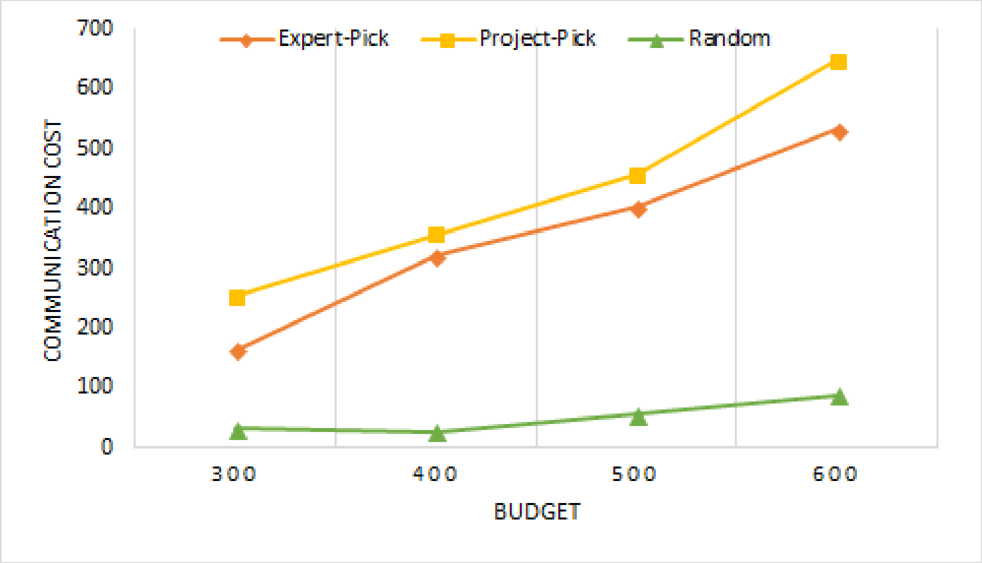


Figure 4.10: Communication Cost vs Budget (No. of Projects = 25)

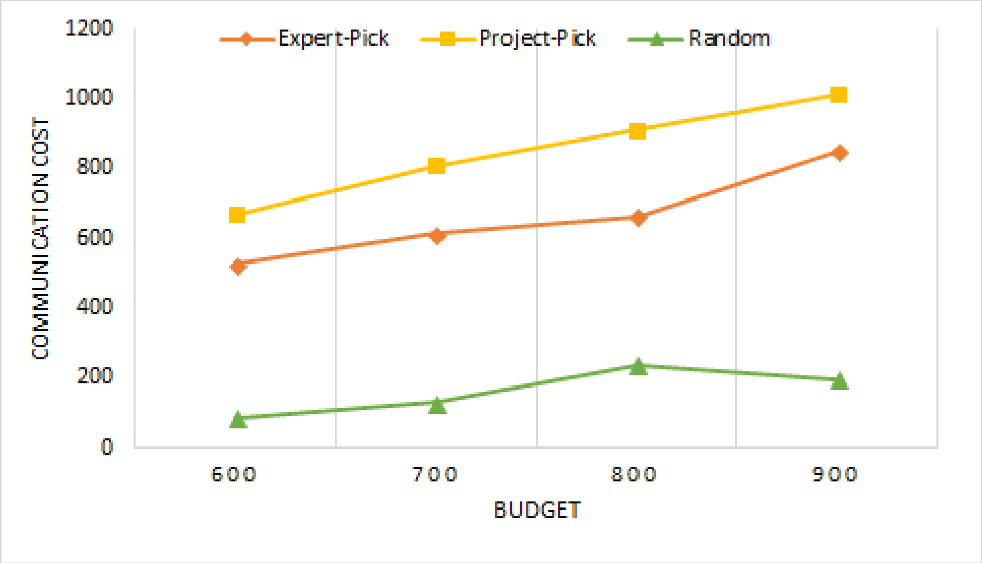


Figure 4.11: Communication Cost vs Budget (No. of Projects = 40)

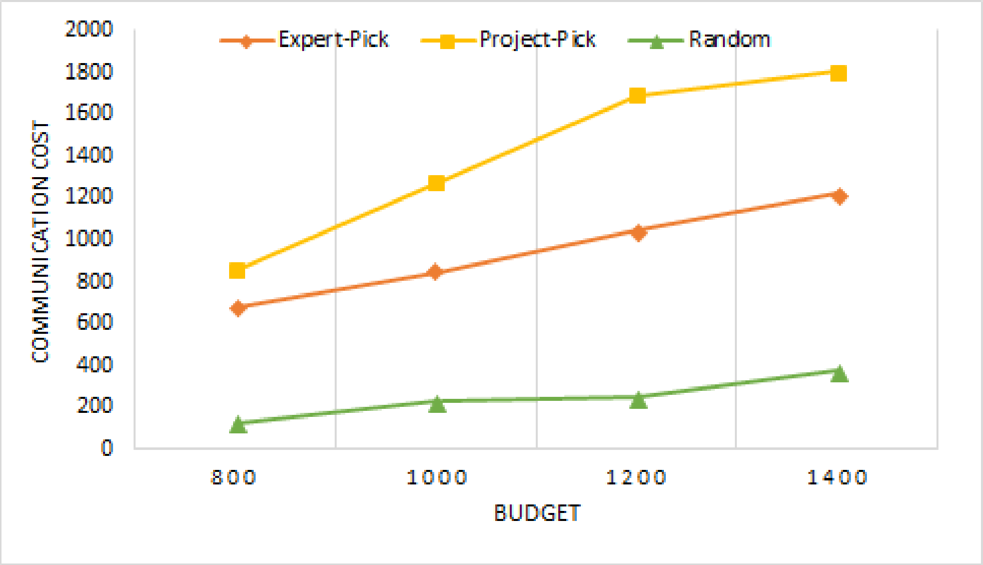


Figure 4.12: Communication Cost vs Budget (No. of Projects = 60)

4.2.3 Number of Completed Projects vs. Budget

Figure 4.13, 4.14, 4.15 and 4.16 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 the 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 the 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.

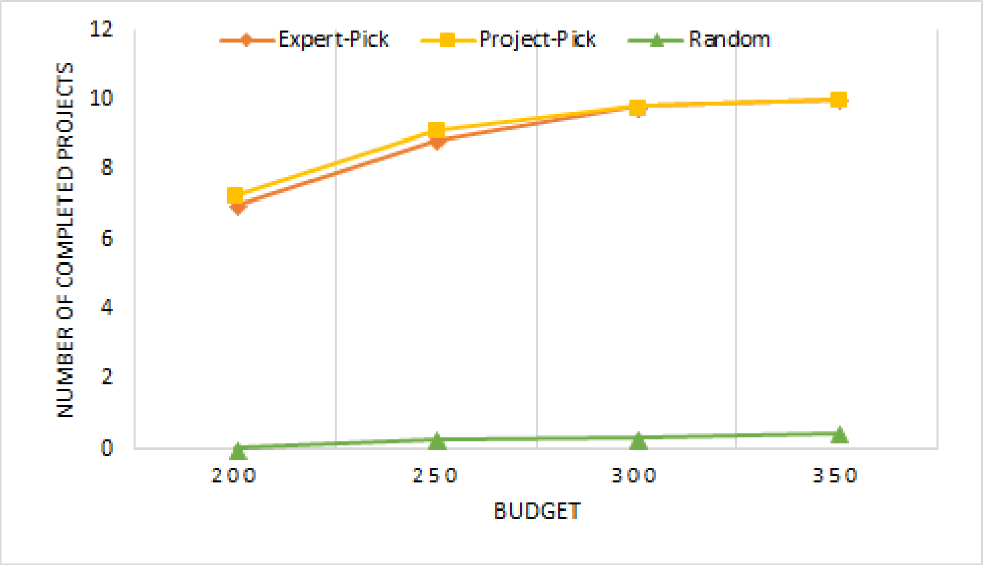


Figure 4.13: Number of Completed Projects vs Budget (No. of Projects = 10)

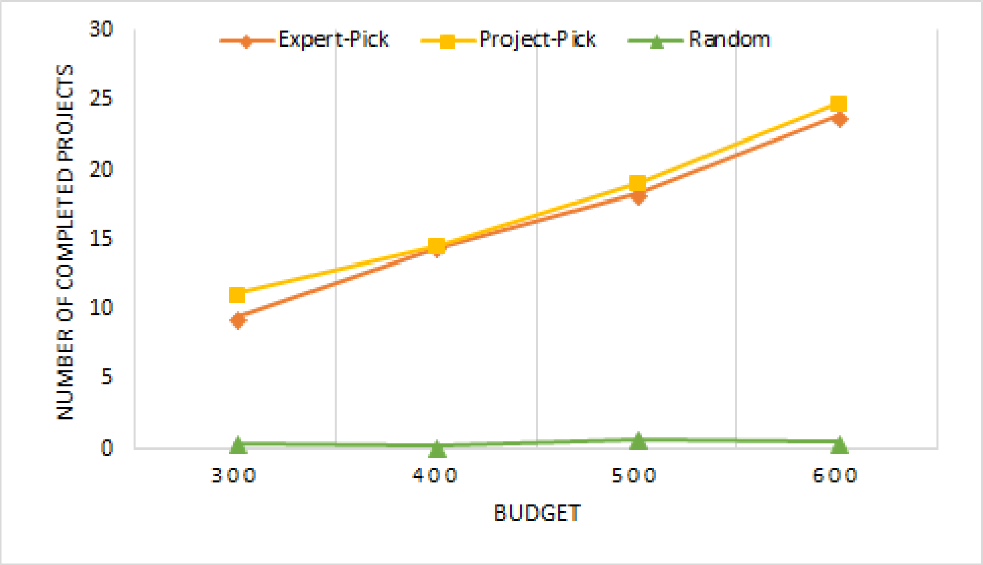


Figure 4.14: Number of Completed Projects vs Budget (No. of Projects = 25)

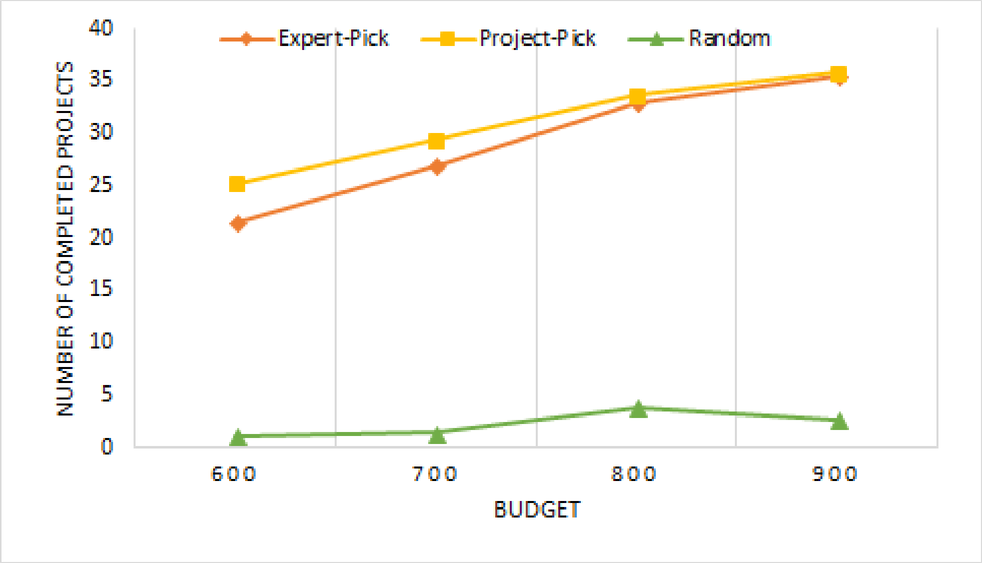


Figure 4.15: Number of Completed Projects vs Budget (No. of Projects = 40)

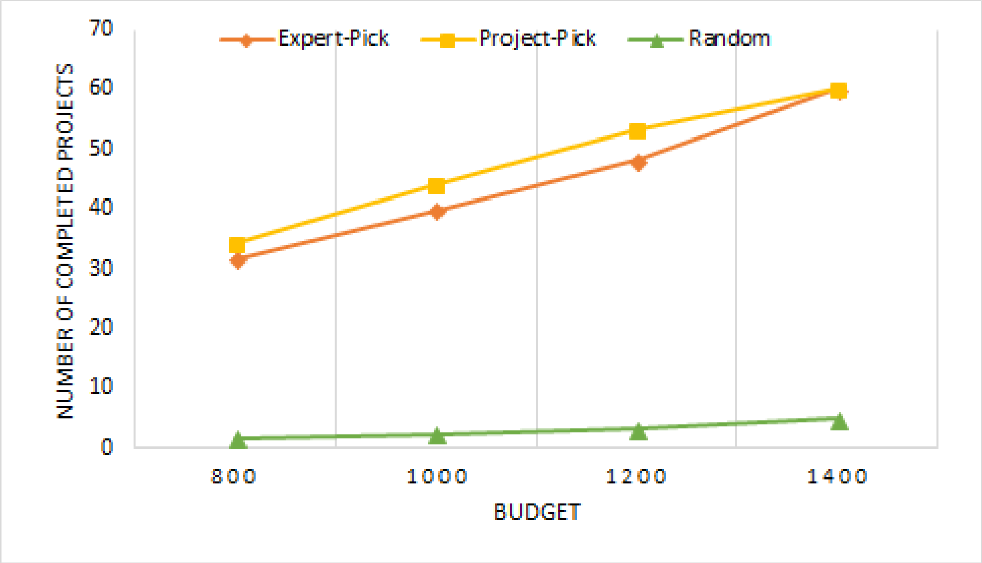


Figure 4.16: Number of Completed Projects vs Budget (No. of Projects = 60)

4.2.4 Sensitivity of the Trade-off Parameter λ

Figures 4.17 and 4.18 show the total profit for different values of trade-off 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 4.19 and 4.20 show the number of completed projects for different values of trade-off parameter λ. The results suggest that Project-Pick completes more projects than Expert-Pick when we have smaller λ. Figures 4.21 and 4.22 show communication cost for different values of trade-off 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 weights on the communication cost and therefore produces a more collaborative group of experts rather than generating high profit.

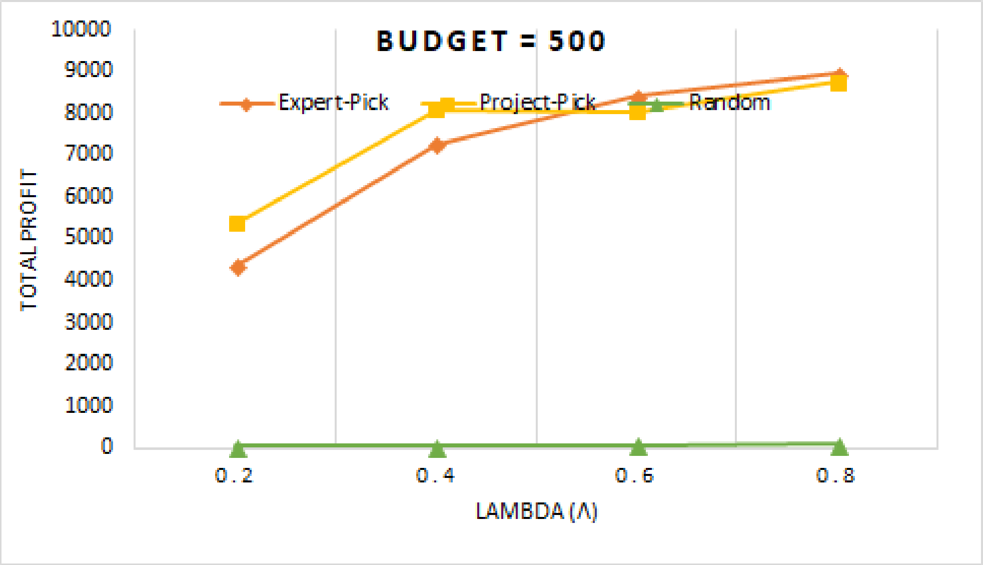


Figure 4.17: Total Profit vs Lambda (λ) (No. of Projects = 25)

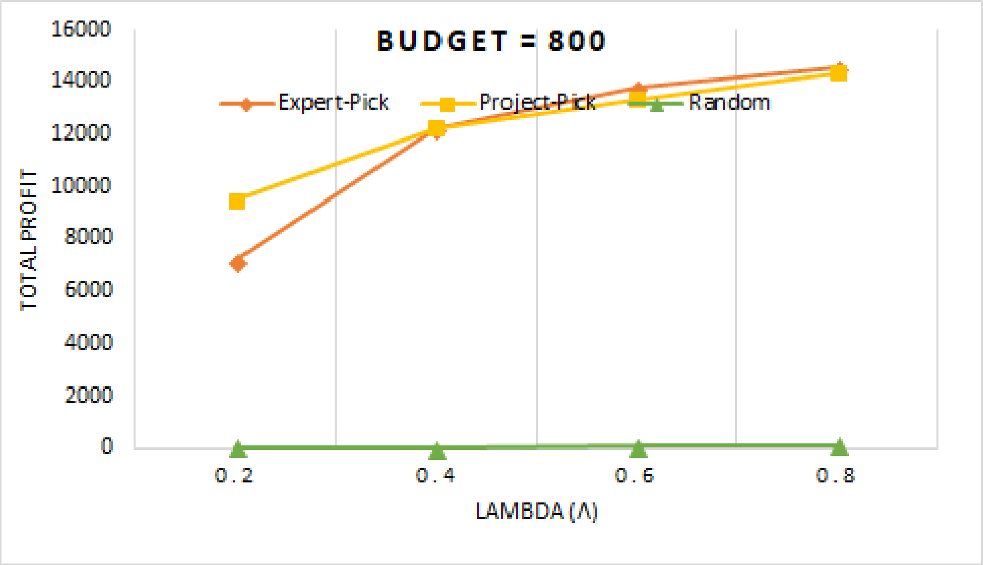


Figure 4.18: Total Profit vs Lambda (λ) (No. of Projects = 40)

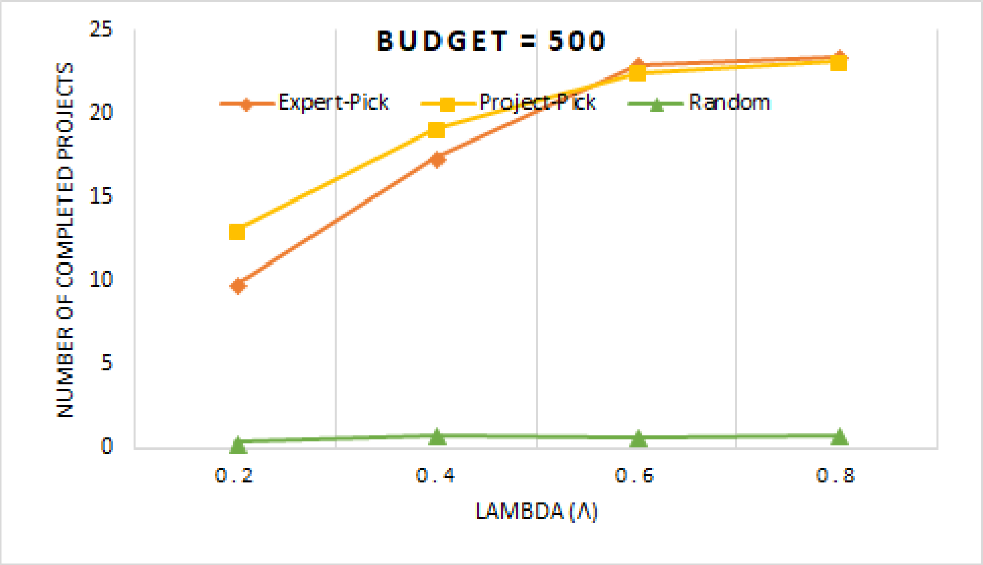


Figure 4.19: Number of Completed Projects vs Lambda (λ) (No. of Projects = 25)

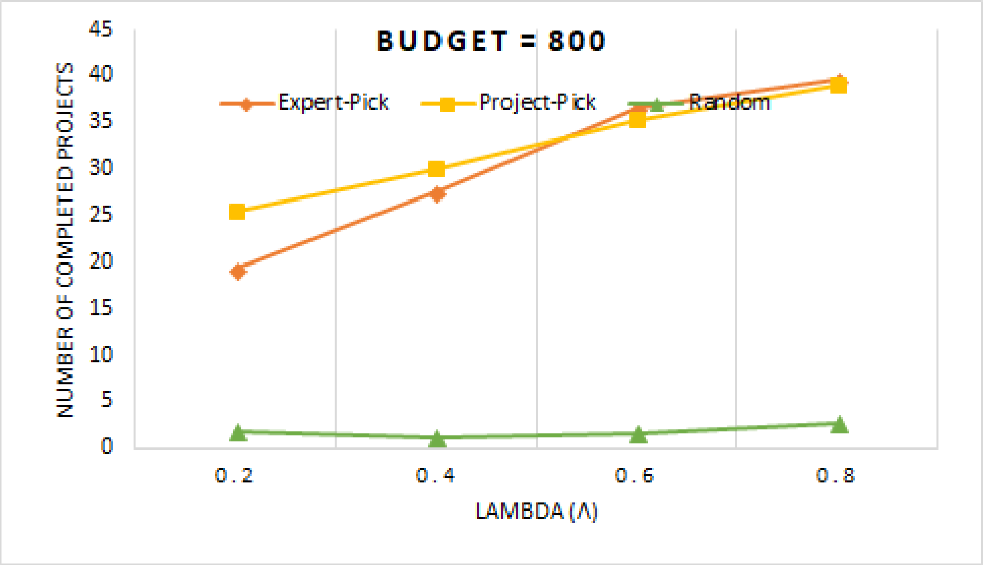


Figure 4.20: Number of Completed Projects vs Lambda (λ) (No. of Projects = 40)

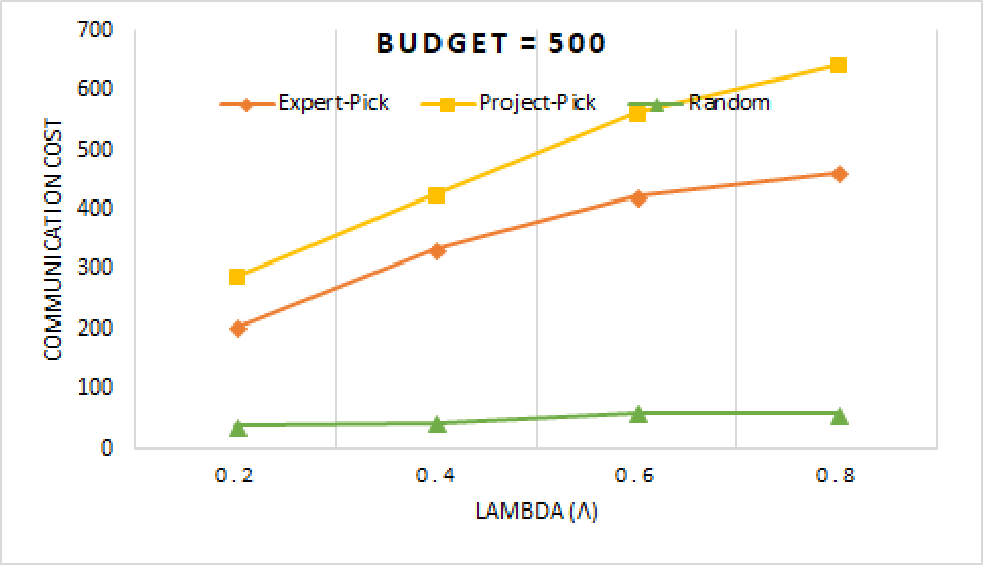


Figure 4.21: Communication Cost vs Lambda (λ) (No. of Projects = 25)

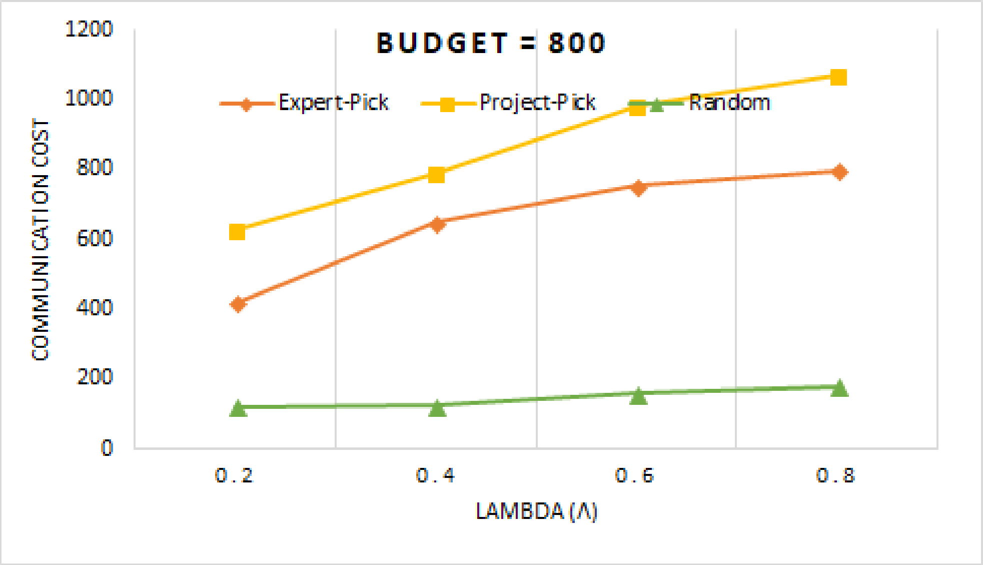


Figure 4.22: Communication Cost vs Lambda (λ) (No. of Projects = 40)

4.2.5 Exact Algorithm

The problem is NP-Hard problem therefore to optimize the problem using Exact Algorithm generates an infinite combination of teams which leads into infinite time. Hence, we are considering a limited combination for comparison purpose with our approach. For this experiment, we consider k projects where each of them requires s skills, so unique set of required skills varies from $s \dots k * s$. The number of experts e, let $k = 4$, $s = 4$, so the unique set of required skills varies from 4 … 16. Let $e = 100$. Therefore, the number of possible combinations will be e^{k*s} which is 100^{16} in our example. The results in figure 4.23 and 4.24 suggests that the Exact Algorithm achieves higher total profit than Expert-Pick and Greedy-Pick algorithm. There is an increase in total profit with increasing the budget where we achieve an almost optimal solution with our approach compared to the exact algorithm.

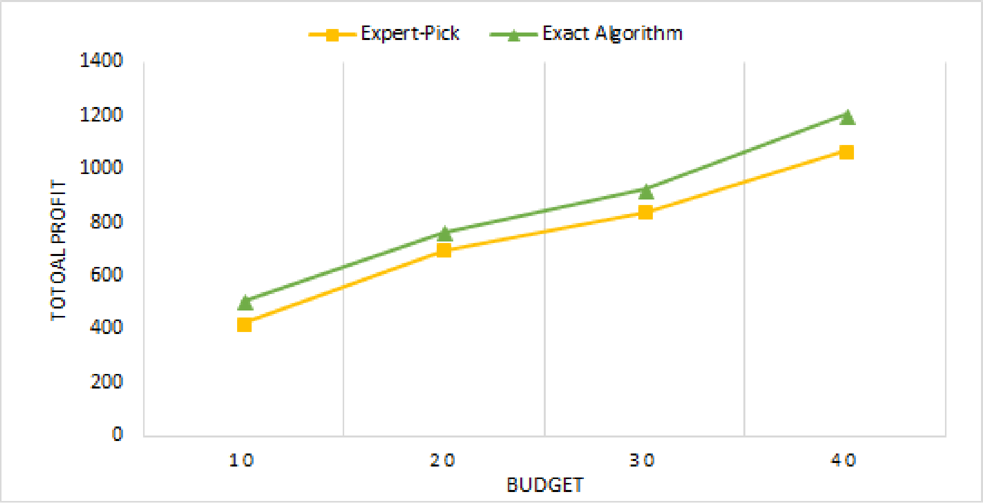


Figure 4.23: Exact Algorithm vs Expert Pick

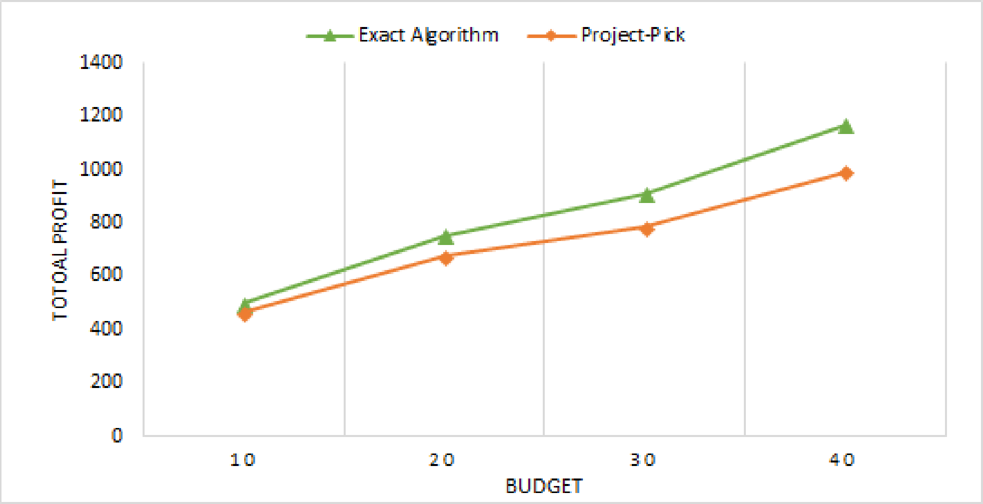


Figure 4.24: Exact Algorithm vs Project Pick

4.2.6 Discussion

Both Expert-Pick and Project-Pick have the polynomial runtime. The results suggest that Project-Pick requires a higher running time than Expert-Pick (4.25). Both our approaches take a bit higher running time than previous project greedy approach due to inclusion of capacity and communication cost which causes high number of iterations. Regarding the results, when we have enough budget, they both generate the groups of experts with high profit and low communication cost. However, when the budget is limited, Project-Pick produces better results, whereas Expert-pick generates a little better results for the higher budget and the higher Lambda (λ) values.

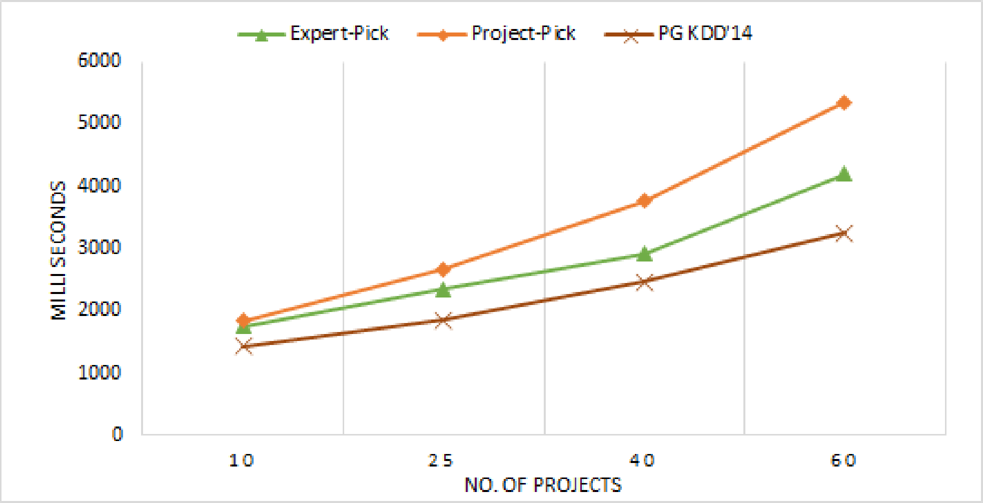


Figure 4.25: Running Time Comparison

Chapter 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

In this thesis, we extended the problem of CLUSTER HIRE to the context of 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 networks with communication cost among experts and capacity of each expert. 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 trade-off parameter λ that determines which optimization objective is more important. We then propose two greedy algorithms to find the best group:

- Expert-Pick selects an expert in each iteration as long as we don't exceed the budget.
- 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 mostly able to select a group of experts to optimize both of the objectives.

5.2 Future work

In our work, we introduced parameter such as communication cost among the experts which is an undirected edge associated between the two experts in a network. In the future, we can incorporate a directed edge among the experts by considering a communication cost from Expert A to Expert B and vice-a-versa which improves the quality of work.

Moreover, we have used a graph with the network of experts in our work, and possible improvements for the future include the levels of the expert (i.e., manager of the team who coordinates a group of individual) by using a tree structure. For instance, the manager is considered as a parent node and a group of individuals as child nodes in a tree. This feature leads to the hiring of the appropriate experts at each level of the project.

BIBLIOGRAPHY

[1] B. Golshan, T. Lappas, and E. Terzi, “Profit-maximizing Cluster Hires,” in *KDD*, 2014, pp. 1196–1205.

[2] T. Point, “Data mining - knowledge discovery.” [Online]. Available: https://www.tutorialspoint.com/data_mining/dm_knowledge_discovery.htm

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

[4] C. Li and M. Shan, “Team formation for generalized tasks in expertise social networks,” in *Proc. of IEEE International Conference on Social Computing*, 2010.

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

[6] A. Gajewar and A. D. Sarma, “Multi skill collaborative teams based on densest subgraphs,” in *Proc. of the SDM’12*, 2012.

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

[8] M. Kargar, M. Zihayat, and A. An, “Finding Affordable and Collaborative Teams from a Network of Experts,” in *SDM*, 2013, pp. 587–595.

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

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

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

[12] Z. Ani, A. Yasin, M. Husin, and Z. Hamid, "A Method for Group Formation Using Genetic Algorithm," *Int. Journal of Advanced Trends in Computer Sci. & Eng.*, vol. 2, no. 9, pp. 3060–3064, 2010.

[13] H. Wi, S. Oha, J. Muna, and M. Jung, "A Team Formation Model based on Knowledge and Collaboration," *Expert Syst Appl*, vol. 36, no. 5, pp. 9121–9134, 2009.

[14] E. Fitzpatrick and R. Askin, "Forming effective worker teams with multi functional skill requirements," *Comput. Ind. Eng.*, vol. 48, no. 3, pp. 593–608, 2005.

[15] A. Baykasoglu, T. Dereli, and S. Das, "Project team selection using fuzzy optimization approach," *Cybern. Syst.*, vol. 38, no. 2, pp. 155–185, 2007.

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

[17] G. K. Awal and K. K. Bharadwaj, "Team Formation in Social Networks based on Collective Intelligence - an Evolutionary Approach," *Appl Intell*, vol. 41, no. 2, pp. 627–648, 2014.

[18] X. Wang, Z. Zhao, and W. Ng, "USTF: A Unified System of Team Formation," *IEEE Transactions on Big Data*, vol. 2, no. 1, 2016.

[19] S. Tang, "Profit-aware team grouping in social networks: A generalized cover decomposition approach," 2016.

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

[21] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi, “Online team formation in social networks,” in *Proc. of the WWW’12*, 2012.

[22] S. Datta, A. Majumder, and K. Naidu, “Capacitated team formation problem on social networks,” in *Proc. of KDD’12*, 2012.

[23] Z. K. M. K. Kalyani Selvarajah, Amangel Bhullar, “Wscan-tfp: Weighted scan clustering algorithm for team formation problem in social networks,” in *The Thirty-First International Florida Artificial Intelligence Research Society Conference (FLAIRS-31)*, 2018, pp. 209–212.

VITA AUCTORIS

NAME:	Meet Patel
PLACE OF BIRTH:	Gujarat, India
YEAR OF BIRTH:	1994
EDUCATION:	Alembic Vidyalaya
	Gujarat, India, 2012
	Charusat University
	Gujarat, India, 2016
	University of Windsor, M.Sc
	Windsor, ON, 2018