



## **CSE 575: Statistical Machine Learning (2023 Spring)**

### *TEAM 16 Project Report*

### *The Power of Collaboration: Machine Learning for Team Performance Optimization*

#### Group Members

NAME	ASU ID
Deekshith Reddy Yeruva	1225545620
Aasish Tammanna	1225545568
Sai Charan Raghupatruni	1226175225
Sangeetha Ramaswami	1224264691
Shilpitha Gandla	1224631798
Praveen Sama	1226337608

## Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
<b>2. Overview of papers.....</b>	<b>1</b>
2.1. Overview of paper “Replacing the Irreplaceable: Fast Algorithms for Team Member Recommendation.” .....	1
2.2. Overview of paper “Finding a Team of Experts in Social Networks.” .....	2
<b>3. Details of Machine Learning Techniques used .....</b>	<b>3</b>
3.1. ML Techniques in paper 1. ....	3
3.1.1. Graph Kernels .....	3
3.1.2. Random Walk-based Graph Kernels.....	3
3.1.3. TeamRep-Basic Method.....	3
3.1.4. TeamRep-Fast-Exact Approach .....	5
3.1.5. TeamRep-Fast-Approx Approach .....	7
3.2. ML Techniques in paper 2 .....	9
3.2.1. Diameter TF problem .....	9
3.2.2. MST TF Problem .....	10
<b>4. Findings of the paper .....</b>	<b>13</b>
4.1. Findings of paper 1. ....	13
4.1.1. Experimental Evaluations.....	13
4.1.2. Effectiveness Results.....	13
4.1.3. Efficiency Results .....	15
4.2. Findings of paper 2 .....	18
4.2.1. Dataset.....	18
4.2.2. Performance Evaluation .....	18
<b>5. Implementation.....</b>	<b>19</b>
5.1. Dataset .....	19
5.2. Methodology Used.....	20
5.3. Results.....	20
<b>6. Applications .....</b>	<b>21</b>
<b>7. Possible Extensions.....</b>	<b>22</b>
<b>8. References</b>	

## 1. Introduction

The success of a team is heavily reliant on the efficiency and efficacy of its members. However, the sudden absence of a crucial team member due to unforeseeable events like illness, resignation, or termination can have a significant impact on team performance. Finding a competent substitute who can execute at the same level as the departing team member is crucial in such circumstances to prevent project delays or disruptions. This task can be difficult and time-consuming for project managers and team leaders, leading to a need for an effective algorithm that can quickly and economically suggest viable replacements for important team members.

To address this issue, our study is inspired by the article "Replacing the Irreplaceable: Fast Algorithms for Team Member Recommendation" by Liangyue Li, Hanghang Tong, Nan Cao, Kate Ehrlich, Yu-Ru Lin and Norbou Buchler and "Finding a Team of Experts in Social Networks" by Theodoros Lappas, Kun Liu and Evimaria Terzi. Our goal is to create a recommendation system that offers the best team configurations by maximizing team strengths and minimizing shortcomings using machine learning approaches. This approach can assist coaches and team managers in making well-informed judgments about their team's make-up and tactics. Additionally, it can be applied to substitute injured or sick athletes and to select employees suited for specific jobs in business teams. To achieve this goal, we aim to develop a recommendation system for coaches and team managers to create the best team configurations based on strengths and weaknesses discovered using machine learning techniques on NBA statistics data.

## 2. Overview

Here is a brief overview of the papers we have surveyed.

### 2.1. Overview of paper "Replacing the Irreplaceable: Fast Algorithms for Team Member Recommendation."

The algorithm proposed by the authors incorporates three crucial elements: skill-based matching, social network analysis, and a machine learning-based model.

The first component, skill-based matching, involves comparing the skill vectors of possible replacements and the remaining team members using a similarity measure based on Voronoi diagrams. The Voronoi diagram is a geometric structure that partitions a space into regions based on the distance to a set of points. In this case, the space represents the set of possible replacements, and the points represent the remaining team members. The similarity measure is based on the idea that a replacement is a good match for the team if they have similar skills to the departing team member and complement the skills of the remaining team members.

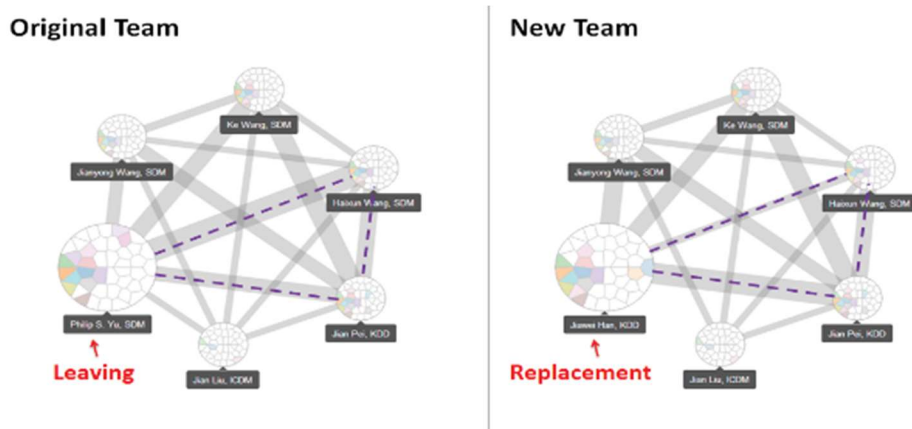


Figure 1: Team Graph before and after Replacement [1]

The second component, social network analysis, leverages the team's network structure to find candidates for replacements whose connections to the team members maximize their chances of integrating successfully. The authors propose using a modified version of the PageRank algorithm, a popular algorithm in network analysis, to compute a measure of each possible replacement's social centrality within the team.

The third component, machine learning-based modeling, involves using graph kernels to model the relationships between team members and potential replacements. Graph kernels are functions that take in graphs as input and output a similarity measure between the graphs. The authors propose several graph kernels that capture different aspects of the relationships between team members, such as their skills, positions, and interactions on the court. These kernels are used to compute a team context-aware similarity measure between the current team and potential replacements.

The authors evaluate their proposed algorithm on a real-world dataset of NBA players and teams. They compare their algorithm to several baseline methods and demonstrate that their algorithm outperforms the baselines in terms of precision and recall. They also perform a sensitivity analysis to show how the algorithm's performance varies with different parameter settings.

Overall, the paper presents a comprehensive approach to team member recommendation that incorporates multiple techniques from machine learning and network analysis. By leveraging the skills and social connections of existing team members and using machine learning-based modeling to capture the nuances of team dynamics, the proposed algorithm offers a promising solution to the problem of finding suitable replacements for critical team members.

## **2.2. Overview of paper “Finding a Team of Experts in Social Networks.”**

We addressed the topic of selecting a subset to accomplish the task given task  $T$ , a pool of individuals with diverse skills, and a social network that captures the compatibility among these individuals. The "Team Formation problem" refers to this. We need members who can not only meet the task's talent requirements but also work well as a team. We measured efficacy by computing the communication cost incurred by the only involved subgraph.

The paper proposes three algorithms to approach the solution by modeling the current “Team Formation problem” into optimizing for the diameter of the graph (Diameter-Tf problem) or the minimum spanning tree of the graph (MST-Tf problem). This is done through well-known NP-Complete problems:

1. Multiple Choice Cover (MCC) problem [3]
2. Group Steiner Tree (GST) problem [4]

For the Diameter-Tf problem, we apply the RarestFirst algorithm. The algorithm is based on the Multichoice algorithm. The CoverSteiner and EnhancedSteiner algorithms are used to solve the MST-Tf problem. Both approaches are based on their resemblance of MST-Tf to the Steiner tree problem.

Finally, the suggested algorithms are tested on the DBLP dataset. The purpose is to identify a team of writers from the dataset who have co-authored more than two publications and cover the requisite skill set based on their prior papers. In a rigorous experimental setup, we investigated the throughput of both the algorithms put them against appropriate baseline procedures. We finished with a great assessment, and generated teams given by our algorithms.

### 3. Details of Machine Learning Techniques

#### 3.1. ML Techniques in paper 1

##### 3.1.1. Graph Kernels:

The idea behind graph kernels is that the similarity of the sub-graphs between the two input graphs are compared and then aggregated to give the overall similarity between the two graphs.

Reasons for choosing Graph Kernels for team context aware similarity over other methods like linear combination, multiplicative combination and sequential filtering are:

- Each subgraph in each team represents a certain skill among a subset of team members.
- By comparing the similarity between two subgraphs it's possible to measure the capability of the new team member to perform that sub-task
- By aggregating this similarity for each task, the overall capability of new team member to be chosen as replacement can be computed.
- In this paper mainly random walk-based [6][8] graph kernel approach is utilized

##### 3.1.2. Random Walk-based Graph Kernels:

Random Walks: A random walk is a sequence of nodes traversed in a graph, where each node is chosen uniformly at random from its neighbours. Random walks capture the local structure and connectivity of a graph.

This graph kernel incorporates the interactions between individual skills and team structure in a unified framework. We calculate the resemblance of two graphs by counting the number of common random walks they match.

These algorithms use effective pruning strategies to lower computational costs and investigate the consistency between existing and new team configurations for more rapid computations.

##### 3.1.3. TeamRep-Basic Method:

Symbols	Definition
$\mathbf{G} := \{\mathbf{A}, \mathbf{L}\}$	the entire social network
$\mathbf{A}_{n \times n}$	the adjacency matrix of $\mathbf{G}$
$\mathbf{L}_{n \times l}$	skill indicator matrix
$\mathcal{T}$	the team member index
$\mathbf{G}(\mathcal{T})$	the team network indexed by its members $\mathcal{T}$
$d_i$	the degree of the $i^{\text{th}}$ node in $\mathbf{A}$
$l$	the total number of skills
$t$	the team size, i.e., $t =  \mathcal{T} $
$n$	the total number of individuals in $\mathbf{A}$
$m$	the total number of connections in $\mathbf{A}$

Table 1: Table of symbols [1]

The above table describes the symbols and its definitions used in the below ML techniques and algorithms.

For each individual a score is computed and the individual with the highest score is recommended to the team. Random walk-based graph kernel is chosen due to its mathematical elegance also. The random walk-based graph kernel is computed using the following:

Given two labelled graphs where  $i=1,2$

$$G_i := \{A_i, L_i\},$$

Here  $G_i$  : represents the labelled graph,  $A_i$  : represents the  $n \times n$  adjacency matrix characterizing connectivity among different individuals,  $L_i$  : represents the  $n \times l$  skill indicator matrix. The  $i^{th}$  row vector in the  $L_i$  matrix represents the skill set of the individual.

As an example: Consider only three skills in total {data mining, databases, and information retrieval}. Then when the skill vector is [1,1,0] for an individual it means that they have skills only in data mining and databases. The value 0 for information retrieval means that there is no skill in information retrieval for that individual.

The random walk-based graph kernel between them can be computed using the following equation:

$$\text{Ker}(G_1, G_2) = y'(I - cA_{\times})^{-1}L_{\times}x$$

Equation (1) [6]

The above equation (1) is utilized for solving team member replacement problems and is called TeamRep-basic method.

In equation (1):

- The Kronecker product of the weight matrix of two graphs is given by:

$$A_{\times} = L_{\times}(A'_1 \otimes A'_2)$$

- $c$  is decay factor
- $y = y_1 \otimes y_2$  is a starting vector used to indicate weights of different nodes
- $x = x_1 \otimes x_2$  is a stopping vector used to indicate weights of different nodes
- $L_{\times}$  is a diagonal matrix.

**The computational challenges involved in TeamRep-Basic method:**

- Many random walk-based graph kernels must be computed.
- Computation of each graph kernel would be expensive especially when team size is large.
- The time complexity is  $O(nt^6)$  [6] as the random walk-based graph kernel has to be computed for each individual.

**Two methods are discussed to overcome the above computational challenges in TeamRep-Basic Method:**

- **Scale-Up: Candidate Filtering:**

In the scale-up method a pruning strategy is utilized. One main idea involved in the TeamRep-Basic method is that structural matching is taken into consideration while computing random walk-based graph kernels. In order to overcome the computational complexity involved in the TeamRep-Basic method to compute a random walk-based graph kernel each time, the candidates who do not have any connections with the current team are removed and filtered out. This method is known as pruning strategy. By utilizing pruning strategy, the number of graph kernel computations can be reduced from  $O(n)$  to  $O(\sum_{i \in \tau/p} d_i)$ .

- **Speedup Graph Kernel**

In the speed up algorithm two new methods are devised to address the problem of speed in the computation of above methods explained. The two methods discussed here to improve speed of computation are TeamRep-Fast-Exact and TeamRep-Fast-Approx.

### 3.1.4. TeamRep-Fast-Exact Approach

TeamRep-Fast-Exact leverages effective pruning strategies to decrease the computational cost of computing the kernel, such as pruning irrelevant edges and nodes in the graph and avoids redundant computations.

The computation of exact graph kernel approach is as below:

$$\begin{aligned} \text{Ker}(\mathbf{G}(\mathcal{T}), \mathbf{G}(\mathcal{T}_{p \rightarrow q})) &= \mathbf{y}'(\mathbf{Z} - c\mathbf{X}\mathbf{Y})^{-1}(\sum_{j=1}^l \mathbf{L}_1^{(j)} \otimes \mathbf{L}_2^{(j)})\mathbf{x} \\ &= \mathbf{y}'(\mathbf{Z}^{-1} + c\mathbf{Z}^{-1}\mathbf{X}(\mathbf{I} - c\mathbf{Y}\mathbf{Z}^{-1}\mathbf{X})^{-1}\mathbf{Y}\mathbf{Z}^{-1}) \\ &\quad ((\sum_{j=1}^l \mathbf{L}_1^{(j)} \otimes \mathbf{L}_c^{(j)})\mathbf{x} + (\sum_{j=1}^l (\mathbf{L}_1^{(j)} \otimes \mathbf{e}^{(j)})(\mathbf{I} \otimes \mathbf{f}^{(j)}))\mathbf{x}) \end{aligned}$$

The above is implemented using the following steps:

---

**Algorithm 1:** TEAMREP-FAST-EXACT
 

---

**Input:** (1) The entire social network  $\mathbf{G} := \{\mathbf{A}, \mathbf{L}\}$ , (2) original team members  $\mathcal{T}$ , (3) person  $p$  who will leave the team, (4) starting and ending probability  $\mathbf{x}$  and  $\mathbf{y}$  (be uniform by default), and (5) an integer  $k$  (the budget)

**Output:** Top  $k$  candidates to replace person  $p$

```

1 Initialize  $\mathbf{A}_c, \mathbf{L}_1^{(j)}, \mathbf{L}_2^{(j)}, j = 1, \dots, l$ ;
2 Pre-compute
    $\mathbf{Z}^{-1} \leftarrow (\mathbf{I} - c(\sum_{j=1}^l \mathbf{L}_1^{(j)} \otimes \mathbf{L}_c^{(j)})(\mathbf{A}_1 \otimes \mathbf{A}_c))^{-1}$ ;
3 Set  $\mathbf{R} \leftarrow (\sum_{j=1}^l \mathbf{L}_1^{(j)} \otimes \mathbf{L}_c^{(j)})\mathbf{x}$ ;  $\mathbf{b} \leftarrow \mathbf{y}^T \mathbf{Z}^{-1} \mathbf{R}$ ;  $\mathbf{l} \leftarrow c\mathbf{y}^T \mathbf{Z}^{-1}$ ;
4 for each candidate  $q$  in  $\mathbf{G}$  after pruning do
5   Initialize  $\mathbf{s} \leftarrow$  a zero vector of length  $t$  except the
   last element is 1;
6   Initialize  $\mathbf{w} \leftarrow$  weight vector from  $q$  to the new
   team members;
7   Set  $\mathbf{E} \leftarrow [\mathbf{w}, \mathbf{s}]$ ;  $\mathbf{F} \leftarrow [\mathbf{s}', \mathbf{w}']$ ;
8   Set  $\mathbf{e}^{(j)} \leftarrow$  a  $t$  by 1 zero vector except the last
   element is 1, for  $j = 1, \dots, d_n$ ;
9   Set  $\mathbf{f}^{(j)} \leftarrow$  a  $1 \times t$  zero vector except the last
   element which is label  $j$  assignment for  $q$ ;
10  Set  $\mathbf{P} \leftarrow [\mathbf{L}_1^{(1)} \otimes \mathbf{e}^{(1)}, \dots, \mathbf{L}_1^{(l)} \otimes \mathbf{e}^{(l)}]$ ;
11  Set  $\mathbf{Q} \leftarrow [\mathbf{I} \otimes \mathbf{f}^{(1)}; \dots; \mathbf{I} \otimes \mathbf{f}^{(l)}]$ ;
12  Compute  $\mathbf{X}_1 \leftarrow (\sum_{j=1}^l \mathbf{L}_1^{(j)} \mathbf{A}_1 \otimes \mathbf{L}_c^{(j)} \mathbf{E})$ ;
13  Compute  $\mathbf{X}_2 \leftarrow (\sum_{j=1}^l \mathbf{L}_1^{(j)} \mathbf{A}_1 \otimes \mathbf{e}^{(j)} \mathbf{f}^{(j)} \mathbf{E})$ ;
14  Compute  $\mathbf{Y}_1 \leftarrow \mathbf{Q}(\mathbf{A}_1 \otimes \mathbf{A}_c)$ ;
15  Compute  $\mathbf{Y}_2 \leftarrow (\mathbf{I} \otimes \mathbf{F})$ ;
16  Set  $\mathbf{X} \leftarrow [\mathbf{P}, \mathbf{X}_1, \mathbf{X}_2]$ ,  $\mathbf{Y} \leftarrow [\mathbf{Y}_1; \mathbf{Y}_2; \mathbf{Y}_2]$ ;
17  Update  $\mathbf{M} \leftarrow (\mathbf{I} - c\mathbf{Y}\mathbf{Z}^{-1}\mathbf{X})^{-1}$ ;
18  Compute  $\mathbf{r}' \leftarrow \mathbf{Z}^{-1}\mathbf{P}\mathbf{Q}\mathbf{x}$ ;
19  Compute  $\text{score}(q) = \mathbf{b} + \mathbf{y}^T \mathbf{r}' + \mathbf{l}\mathbf{X}\mathbf{M}\mathbf{Y}(\mathbf{Z}^{-1}\mathbf{R} + \mathbf{r}')$ ;
20 end
21 Return the top  $k$  candidates with the highest scores.

```

---

Figure 2: TeamRep-Fast-Exact algorithm [1]



The algorithm starts by initializing:

- set of active nodes  $A_c$
- adjacency matrices  $L_1^{(j)}$  and  $L_2^{(j)}$  for  $j = 1, \dots, l$ , where  $l$  is the number of layers in the network.

Then it precomputes  $Z^{-1}$  which is the inverse of the matrix. In step 3:  $c$  is a decay factor,  $A_1$  is the adjacency matrix for the first layer, and  $\otimes$  denotes the Kronecker product.

$Z^{-1}$ ,  $R$ ,  $b$ , and  $l$  are used in steps 2 and 3 to compute each candidate's score later.

The most important step in the loop is to update  $M(I - cYZ^{-1}X)$  which involves matrix inverse of size  $(l+4) \times (l+4)$ . This matrix is of smaller size and helps to accelerate computation without losing the accuracy of graph kernel.

**Accuracy:** Accuracy of TeamRep-Fast-Exact is similar to TeamRep-Basic method discussed earlier.

**Time Complexity:** The TeamRep-Fast-Exact algorithm takes  $O((\sum_{i \in \tau/p} d_i)(lt^5 + l^3t^3))$

### 3.1.5. TeamRep-Fast-Approx Approach

TeamRep-Fast-Approx works at the cost of a small approximation error. It uses a sampling-based approach to approximate the kernel computation, by randomly selecting a subset of random walks instead of considering all possible walks.

For labelled graphs, the approximated graph kernel can be calculated as

$$\begin{aligned}
 \hat{Ker}(\mathbf{G}(\mathcal{T}), \mathbf{G}(\mathcal{T}_{p \rightarrow q})) &= \mathbf{y}^T (\mathbf{I} - c\mathbf{L}_X (\hat{\mathbf{A}}'_1 \otimes \hat{\mathbf{A}}'_2))^{-1} \mathbf{L}_X \mathbf{x} \\
 &= \mathbf{y}' (\mathbf{I} - c\mathbf{L}_X (\mathbf{X}_1 \mathbf{Y}_1) \otimes (\mathbf{X}_2 \mathbf{Y}_2))^{-1} \mathbf{L}_X \mathbf{x} \\
 &= \mathbf{y}' (\mathbf{I} - c\mathbf{L}_X (\mathbf{X}_1 \otimes \mathbf{X}_2) (\mathbf{Y}_1 \otimes \mathbf{Y}_2))^{-1} \mathbf{L}_X \mathbf{x} \\
 &= \mathbf{y}' (\mathbf{I} + c\mathbf{L}_X (\mathbf{X}_1 \otimes \mathbf{X}_2) \mathbf{M}(\mathbf{Y}_1 \otimes \mathbf{Y}_2)) \mathbf{L}_X \mathbf{x} \\
 &= \mathbf{y}' \mathbf{L}_X \mathbf{x} + c\mathbf{y}' (\sum_{j=1}^l \mathbf{L}_1^{(j)} \mathbf{X}_1 \otimes \mathbf{L}_2^{(j)} \mathbf{X}_2) \mathbf{M}(\mathbf{Y}_1 \otimes \mathbf{Y}_2) \mathbf{L}_X \mathbf{x} \\
 &= (\sum_{j=1}^l (\mathbf{y}'_1 \mathbf{L}_1^{(j)} \mathbf{x}_1) (\mathbf{y}'_2 \mathbf{L}_2^{(j)} \mathbf{x}_2)) + c \\
 &\quad (\sum_{j=1}^l \mathbf{y}'_1 \mathbf{L}_1^{(j)} \mathbf{X}_1 \otimes \mathbf{y}'_2 \mathbf{L}_2^{(j)} \mathbf{X}_2) \mathbf{M}(\sum_{j=1}^l \mathbf{Y}_1 \mathbf{L}_1^{(j)} \mathbf{x}_1 \otimes \mathbf{Y}_2 \mathbf{L}_2^{(j)} \mathbf{x}_2)
 \end{aligned}$$

The above approximated graph kernel is an even faster algorithm. The steps to implement this approach is as follows:

---

**Algorithm 2: TEAMREP-FAST-APPROX**


---

**Input:** (1) The entire social network  $\mathbf{G} := \{\mathbf{A}, \mathbf{L}\}$ , (2) original team members  $\mathcal{T}$ , (3) person  $p$  who will leave the team, (4) starting and ending probability  $\mathbf{x}$  and  $\mathbf{y}$  (be uniform by default), and (5) an integer  $k$  (the budget)

**Output:** Top  $k$  candidates to replace person  $p$

```

1 Initialize  $\mathbf{A}_c, \mathbf{L}_1^{(j)}, \mathbf{L}_2^{(j)}, j = 1, \dots, l$ ;
2 Compute top  $r$  eigen-decomposition for  $\mathbf{A}_c$ :
    $\mathbf{U}\mathbf{\Lambda}\mathbf{U}' \leftarrow \mathbf{A}_c$ ;
3 Set  $\mathbf{V} \leftarrow \mathbf{\Lambda}\mathbf{U}'$ ;
4 Initialize  $\mathbf{s} \leftarrow$  a zero vector of length  $t$  except the last
   element is 1;
5 Initialize  $\mathbf{w}_1 \leftarrow$  weight vector from  $p$  to  $\mathcal{T}$ ;
6 Set  $\mathbf{E}_1 \leftarrow [\mathbf{w}_1, \mathbf{s}], \mathbf{F}_1 \leftarrow [\mathbf{s}', \mathbf{w}_1']$ ;
7 Set  $\mathbf{X}_1 \leftarrow [\mathbf{U}, \mathbf{E}_1], \mathbf{Y}_1 \leftarrow [\mathbf{V}, \mathbf{F}_1]$ ;
8 for each candidate  $q$  in  $\mathbf{G}$  after pruning do
9   Initialize  $\mathbf{w}_2 \leftarrow$  weight vector from  $q$  to the new
   team members;
10  Set  $\mathbf{E}_2 \leftarrow [\mathbf{w}_2, \mathbf{s}], \mathbf{F}_2 \leftarrow [\mathbf{s}', \mathbf{w}_2']$ ;
11  Set  $\mathbf{X}_2 \leftarrow [\mathbf{U}, \mathbf{E}_2], \mathbf{Y}_2 \leftarrow [\mathbf{V}, \mathbf{F}_2]$ ;
12  Compute  $\mathbf{S} \leftarrow \sum_{j=1}^l \mathbf{y}_1' \mathbf{L}_1^{(j)} \mathbf{X}_1 \otimes \mathbf{y}_2' \mathbf{L}_2^{(j)} \mathbf{X}_2$ ;
13  Compute  $\mathbf{T} \leftarrow \sum_{j=1}^l \mathbf{Y}_1 \mathbf{L}_1^{(j)} \mathbf{x}_1 \otimes \mathbf{Y}_2 \mathbf{L}_2^{(j)} \mathbf{x}_2$ ;
14  Update  $\mathbf{M} \leftarrow (\mathbf{I} - c(\sum_{j=1}^l \mathbf{Y}_1 \mathbf{L}_1^{(j)} \mathbf{X}_1 \otimes \mathbf{Y}_2 \mathbf{L}_2^{(j)} \mathbf{X}_2))^{-1}$ ;
15  Set  $\text{score}(q) = (\sum_{j=1}^l (\mathbf{y}_1' \mathbf{L}_1^{(j)} \mathbf{x}_1)(\mathbf{y}_2' \mathbf{L}_2^{(j)} \mathbf{x}_2)) + c\mathbf{S}\mathbf{M}\mathbf{T}$ ;
16 end
17 Return the top  $k$  candidates with the highest scores.

```

---

Figure 3: TeamRep-Fast-Approx algorithm [1]

The algorithm starts by initializing:

- Adjacency matrix  $A_c$
- matrices  $L_1^{(j)}$  and  $L_2^{(j)}$  for  $j = 1, \dots, l$ , where  $l$  is the number of layers in the network. Laplacian matrices capture the structure of the network and can be used to calculate various metrics related to the network's connectivity.

In step 2 top  $r$  eigen-decomposition for  $A_c$  is computed. In this method it has to be only computed once. For every new team this value computed in step 2 is used for updating low rank approximation.

In this case for computation for  $\mathbf{M}$  in step 14 is even more less complex compared to original graph kernel. Here the  $\mathbf{M}$  matrix inverse is of size  $(r+2)^2 * (r+2)^2$  [7]. By updating this  $\mathbf{M}$  the time is not dependent of team size in this method.

The TeamRep-Fast-Approx is faster compared to TeamRep-Basic and TeamRep-Fast-Exact.

**Accuracy:** If the top  $r$  eigen decomposition  $A_c$  computed in step 2 of algorithm holds then it outputs same set of candidates as TeamRep-Basic.

**Time Complexity:** The TeamRep-Fast-Approx takes  $O((\sum_{i \in \tau/p} d_i)(lt^2r + r^6))$

### 3.2. ML Techniques in paper 2

From the paper [2], the authors define a problem named a team formation problem where the project's main goal is to identify a group of people who may be used to solve complicated problems or provide tough answers and who have distinct and varied skills in a certain topic. By utilizing the extensive social network data readily available on internet platforms, the authors of the research hope to overcome the difficulty of finding such a team of specialists. So in the paper authors suggest we can solve this problem using graph theory. The problem is described as a weighted, undirected graph  $G(X, E)$ , where nodes represent candidates and  $X$  represents a pool of candidates. The cost of communication between candidates at an edge's endpoints is represented by the weight on the edge. Finding a subgraph in  $G$  that only includes  $X'$  (a subset of  $X$ ) and meets the necessary skill set for a particular task  $T$  with the lowest possible communication cost is the objective.

Here are the preliminaries defined to approach the above TEAM-FORMATION problem.

$X = \{1, 2, 3, \dots, n\}$ , where  $X$  is the pool of candidates

$X_i$  = set of skills of candidate  $i$ .

$T$  = set of skills required to complete the task.

$S(a)$  = set of individuals in  $X$  that have the skill "a"

As the problem is np-complete [3], it is formulated into two problems, where it is reduced to known np-complete problems, which are the Diameter- TF problem which consists of a Multiple-Choice Cover (MCC) problem [3], and the MST-TF problem which consists of Group Steiner problem [4].

#### 3.2.1. Diameter TF problem:

Diameter (R) is defined as:

If we take a graph  $G(X, E)$  and also certain candidates  $X' \subseteq X$ , The Cc-R ( $X'$ ) (Diameter communication cost) is defined as the subgraph  $G(X')$ 's diameter. [2]

Now we define Multiple choice cover problem as follows:

We define the instance here which contains a universe  $A = \{1, \dots, N\}$  an  $N \times N$  symmetric real matrix  $D$  with nonnegative entries, and  $S = \{S_1, \dots, S_k\}$  such that each  $S_i \subseteq A$ . The problem is to find a value  $K$  such that there exists  $A' \subseteq A$  such that for every  $i \in \{1, \dots, k\}$ ,  $|A' \cap S_i| > 0$  and  $\max_{(u,v) \in A' \times A'} D(u, v) \leq K$ . [2]

To solve this problem, we use the RarestFirst algorithm:

**RarestFirst algorithm:** According to a predetermined area of knowledge, the algorithm starts by identifying the set of all prospective experts in the social network. The number of other possible experts who have the same competence is then used to determine how rare each potential expert is. The rarest expert is then chosen as the team's initial member, and that expert's relationships are disregarded in the process. Until the desired team size is reached, this process is repeated, selecting the next rarest expert who has not already been chosen or eliminated. The algorithm's objective is to choose a group of experts with the most expertise overall while reducing redundancies. The algorithm is more likely to find special knowledge that is absent from other team members by prioritizing the rarest specialists.

---

**Algorithm 1** The RarestFirst algorithm for the DIAMETER-Tf problem.

---

**Input:** Graph  $G(\mathcal{X}, E)$ ; individuals' skill vectors  $\{X_1, \dots, X_n\}$  and task  $T$ .  
**Output:** Team  $\mathcal{X}' \subseteq \mathcal{X}$  and subgraph  $G[\mathcal{X}']$ .

- 1: **for** every  $a \in T$  **do**
- 2:    $S(a) = \{i \mid a \in X_i\}$
- 3:  $a_{rare} \leftarrow \arg \min_{a \in T} |S(a)|$
- 4: **for** every  $i \in S(a_{rare})$  **do**
- 5:   **for**  $a \in T$  and  $a \neq a_{rare}$  **do**
- 6:      $R_{ia} \leftarrow d(i, S(a))$
- 7:    $R_i \leftarrow \max_a R_{ia}$
- 8:  $i^* \leftarrow \arg \min R_i$
- 9:  $\mathcal{X}' = i^* \cup \{Path(i^*, S(a)) \mid a \in T\}$

---

Figure 4: RarestFirst Algorithm. [2]

From the above figure [2], we can see the input here is taken as Graph  $G(X, E)$  where each individual in  $X$  has a skill vector  $X_i$ , where  $i$  is the  $i^{th}$  candidate from the pool of candidates. In steps 1 and 2, for every skill 'a' the algorithm finds the number of people from the pool that have the specific skill 'a'. In step 3, it only takes the rarest skill and set it to  $a_{rare}$ . From steps 4 to 7, the algorithm picks a candidate with the skill  $a_{rare}$  and tries to find the subgraph that satisfies the required skill vector 'a'. The iteration is continued for all the members in the  $a_{rare}$  vector. Finally, in step 8, we find the minimum diameter subgraph from the generated sub graphs in the iterations. We finally append the rarest skilled expert with the minimum diameter sub-graph given in step 8.

Assuming that the shortest path between all pairs of candidates in a social network has already been calculated and hash tables are used to store the attributes of each candidate, as well as which individuals possess a specific attribute. With these data structures in place, the algorithm's runtime is  $O(|S(a_{rare})| \times n)$ , where  $|S(a_{rare})|$  is the size of the set of rarest individuals. A worst-case analysis indicates that  $|S(a_{rare})|$  is  $O(n)$ , meaning that the worst-case runtime of the algorithm is  $O(n^2)$ .

### 3.2.2. MST TF Problem:

Minimum Spanning Tree (MST):

If we take a graph  $G(X, E)$  and also certain candidates  $X' \subseteq X$ , The Cc-Mst ( $X'$ ) (MST Communication cost) is defined as the subgraph  $G(X')$ 's minimum spanning tree cost. [2]

GST problem consists of an undirected graph  $G(A, E)$ , cost function  $c: E \rightarrow \mathbf{R}$  and  $k$  subsets of vertices (called groups)  $\{g_1, \dots, g_k\}$  with  $g_i \subseteq A, i \in \{1, \dots, k\}$ . The problem is to find  $K$  such that there exists a subtree  $T(A', E')$  of  $G(A, E)$  (i.e.,  $A' \subseteq A$  and  $E' \subseteq E$ ) such that  $|A' \cap g_i| > 0$  for every  $i \in \{1, \dots, k\}$  and  $\text{cost} \sum_{e \in E} c(e) \leq K$ . [2]

To solve this problem, the paper proposed 2 algorithms:

1. CoverSteiner Algorithm
2. EnhancedSteiner Algorithm

#### CoverSteiner Algorithm:

Firstly, we propose for the MST-TF problem is divided into two steps. Network is disregarded in the first stage, and the algorithm concentrates on finding a group of individuals  $X_0 \subseteq X$  such that  $\cup_{i \in X_0} X_i \supseteq T$ . The method then determines the lowest cost tree that spans all of the nodes in  $X_0$  and maybe other nodes in  $X \setminus X_0$  in the second stage. As a result, a collection of nodes  $X'$  such that  $X_0 \subseteq X' \subseteq X$  is reported. This two-step algorithm is known as the CoverSteiner algorithm.

---

**Algorithm 2** The CoverSteiner algorithm for the MST-TF problem.

---

**Input:** Graph  $G(\mathcal{X}, E)$ ; individuals' skill vectors  $\{X_1, \dots, X_n\}$  and task  $T$ .

**Output:** Team  $\mathcal{X}' \subseteq \mathcal{X}$  and subgraph  $G[\mathcal{X}']$ .

1:  $\mathcal{X}_0 \leftarrow \text{GreedyCover}(\mathcal{X}, T)$

2:  $\mathcal{X}' \leftarrow \text{SteinerTree}(G, \mathcal{X}_0)$

---

Figure 5: CoverSteiner Algorithm [2]

According to Figure 5, the first step's goal is to solve a fundamental Set Cover problem: the elements to be covered is the task  $T$ 's criteria, and everyone in  $X$  is a subset of the set. For the Set Cover problem, we use the standard GreedyCover [3] technique. The GreedyCover algorithm is a repetitive procedure that at every stage adds the individual  $X_i$  with the most still unknown required abilities in  $T$ .

The CoverSteiner algorithm solves a Steiner Tree problem on graph  $G$  in its second iteration. The following is an example:

An undirected graph with positive edge costs is displayed to us. This graph's vertices are divided into two categories: necessary and Steiner vertices. The Steiner Tree [4] challenge then requests the lowest-cost tree in the graph that includes necessary vertices as well as any part of the Steiner vertices.

In our scenario, the GreedyCover algorithm's nodes  $X_0$  corresponds to the needed vertices, while the vertices in  $X \setminus X_0$  describe the Steiner vertices. Given a graph  $G(X, E)$ , the purpose of Algorithm 2 line 2 is to find the solution  $X'$  that reduces Cc-MST ( $X'$ ), with the condition that  $X_0 \subseteq X'$ .

This below figure gives us the Steiner Tree algorithm that the CoverSteiner algorithm called in its algorithm.

---

**Algorithm 3** The SteinerTree algorithm.

---

**Input:** Graph  $G(\mathcal{X}, E)$ ; required nodes  $\mathcal{X}_0$  and Steiner nodes  $\mathcal{X} \setminus \mathcal{X}_0$ .

**Output:** Team  $\mathcal{X}_0 \subseteq \mathcal{X}' \subseteq \mathcal{X}$  and subgraph  $G[\mathcal{X}']$ .

- 1:  $\mathcal{X}' \leftarrow v$ , where  $v$  is a random node from  $\mathcal{X}_0$ .
  - 2: **while**  $(\mathcal{X}_0 \setminus \mathcal{X}') \neq \emptyset$  **do**
  - 3:      $v^* \leftarrow \arg \min_{u \in \mathcal{X}_0 \setminus \mathcal{X}'} d(u, \mathcal{X}')$
  - 4:     **if**  $Path(v^*, \mathcal{X}') \neq \emptyset$  **then**
  - 5:          $\mathcal{X}' \leftarrow \mathcal{X}' \cup \{Path(v^*, \mathcal{X}')\}$
  - 6:     **else**
  - 7:         Return Failure
- 

Figure 6: Steiner Tree Algorithm [2]

The procedure above incrementally adds  $\mathcal{X}'$  nodes from the needed set  $\mathcal{X}_0$  to the present solution. Each step adds an additional node from  $\mathcal{X}_0$  which is the node  $v^*$  with the nearest separation to the collection of nodes  $\mathcal{X}'$  already in the solution (line 3). If such a node exists,  $v^*$  and all nodes on the closest route from it to  $\mathcal{X}'$  are added to the solution set. Otherwise, failure is recorded.

The CoverSteiner algorithm's execution time is the sum of the execution times of GreedyCover and SteinerTree. The time required for the GreedyCover algorithm to execute is  $O(|T| * |\mathcal{X}|)$  or  $O(mn)$ . The time required to execute the SteinerTree in Algorithm 3 is  $O(|\mathcal{X}_0| * |E|)$ . Thus, in the worst-case scenario, CoverSteiner's execution time is  $O(n^3)$ .

One major drawback of the CoverSteiner algorithm is that it entirely ignores the underlying graph structure in the first stage. This might result in teams with a high communication cost, or even failure, even when a solution to the MST-TF problem exists. So, to overcome this issue, we use EnhancedSteiner algorithm.

**EnhancedSteiner Algorithm:**

To construct the enhanced graph  $H$ , the EnhancedSteiner method first augments graph  $G$  with new nodes and edges. SteinerTree is then used to solve the Steiner Tree problem on the augmented graph  $H$ .

The pseudocode for these two phases of the EnhancedSteiner algorithm is depicted in Algorithm 4.

---

**Algorithm 4** The EnhancedSteiner algorithm for the MST-TF problem.

---

**Input:** Graph  $G(\mathcal{X}, E)$ ; individuals' skill vectors  $\{X_1, \dots, X_n\}$  and task  $T$ .

**Output:** Team  $\mathcal{X}' \subseteq \mathcal{X}$  and subgraph  $G[\mathcal{X}']$ .

- 1:  $H \leftarrow \text{EnhanceGraph}(G, T)$
  - 2:  $\mathcal{X}_H \leftarrow \text{SteinerTree}(H, \{Y_1, \dots, Y_k\})$
  - 3:  $\mathcal{X}' \leftarrow \mathcal{X}_H \setminus \{Y_1, \dots, Y_k\}$
- 

Figure 7: EnhancedSteiner Algorithm [2]

Let the task at hand necessitate  $k$  skills, i.e.,  $T = \{a_1, a_2, \dots, a_k\}$ . The Enhance procedure performs a linear pass through the graph  $G$  and enhances it.

an extra node  $Y_j$  is generate for each skill  $a_j \in T$ . Every new vertex  $Y_j$  is attached to a node  $i \in X$  if and only if  $a_j \in X_i$ . The separation between node  $Y_j$  and nodes  $i \in S(a_j)$  is  $d(Y_j, i) = D$  where  $D$  is a huge real value that is greater than the sum of all the pairwise distances between nodes in the graph  $G$ . Finally, for each node  $i \in X$  that has capabilities  $X_i$  is replaced by a clique  $C_i$  of size  $|X_i|$ . Every node in the clique  $C_i$  should be seen as a duplicate of individual  $i$  who possesses just one distinct skill from the set. Each node in the clique  $C_i$  initialized to zero. Each node in the clique  $C_i$  keeps all of node  $i$ 's current connections to the remaining part of the graph, with connections to nodes  $\{Y_1, Y_2, \dots, Y_k\}$ .

Calling the SteinerTree algorithm with required nodes  $\{Y_1, Y_2, \dots, Y_k\}$  returns nodes  $X_H$  that involve in the Steiner tree of the enhanced graph  $H$ .

Finally, the algorithm removes the falsely created nodes  $\{Y_1, Y_2, \dots, Y_k\}$  from set  $X_H$  to produce the final answer  $X'$ . Note that the substitution of everyone  $i$  with a clique  $C_i$  of size  $|X_i|$  is just conceptual. In practice, the algorithm's implementation does not necessitate this. As a result, the enhanced graph  $H$  has just  $k$  additional nodes than the original graph  $G$ , namely nodes  $Y_1, Y_2, \dots, Y_k$ . As a result of the SteinerTree analysis in the preceding section, the execution time of the EnhancedSteiner algorithm is  $O(k * |E|)$ .

## 4. Findings of the paper:

### 4.1. Findings of paper 1

#### 4.1.1. Experimental Evaluations

The paper describes the experimental evaluations conducted to answer two questions: effectiveness and efficiency of their proposed algorithms for team member replacement. Three datasets were used: DBLP, Movie, and NBA. The DBLP dataset [9] provided information on computer science journals and proceedings, which was used to build a co-authorship network. The Movie dataset [10], which is an add-on of MovieLens dataset, contained information on movies, actors/actresses, and genres, and was used to set up the social network of actors/actresses. The NBA dataset [11] contained NBA and ABA statistics and was used to label players' positions as their skills.

Data	n	m	# of teams
DBLP	916,978	3,063,244	1,572,278
Movie	95,321	3,661,679	10,197
NBA	3,924	126,994	1,398

Table 2: Summary of Datasets. [1]

#### 4.1.2. Effectiveness Results

The paper presents an algorithmic solution for the Team Member Replacement problem, which arises when a team member becomes unavailable and needs to be replaced. The proposed algorithm aims to find a replacement that not only matches the skills of the unavailable member but also preserves the team's structural properties [12]. The

algorithm uses graph kernels to capture the structural similarity between team members and a Bayesian framework to model the skill similarity.

To evaluate the effectiveness of the algorithm, the paper presents both qualitative and quantitative evaluations. The qualitative evaluations include case studies on three of the datasets. In each case study, the algorithm recommends replacements for an unavailable team member, and the authors provide an explanation for why the recommended replacements are suitable.

The quantitative evaluations focus on two aspects: (1) whether simultaneously considering both design objectives (skill and structural match) outperform considering only one of them, and (2) how the proposed graph kernel formulation compares to alternative choices. The evaluations are performed on the same three datasets, and the results show that the proposed algorithm outperforms alternative approaches in both aspects.

The algorithm can be used in various scenarios where team member replacements are needed, such as academic collaborations, project teams, and sports teams. The proposed algorithm not only considers the skill match but also preserves the team's structural properties, which can lead to better performance and higher team harmony.

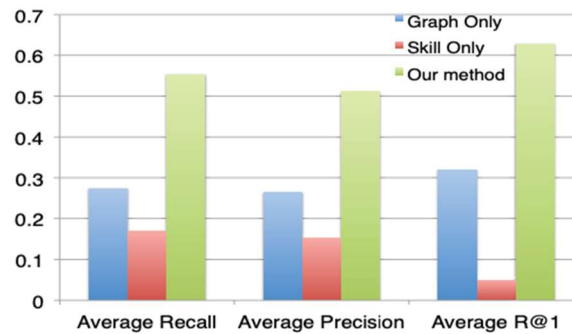


Figure 8: performance of three comparison methods [1]

A higher score indicates better performance.

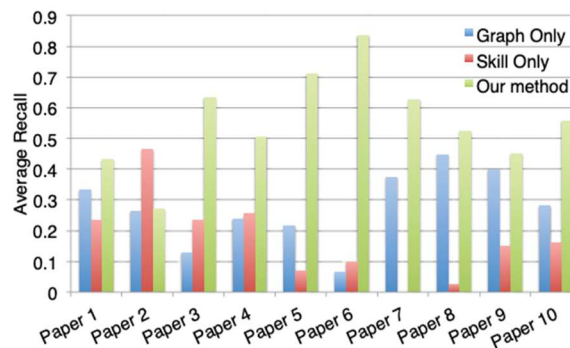


Figure 9: recall score comparison [1]



The higher the recall score, the better the performance of the paper.

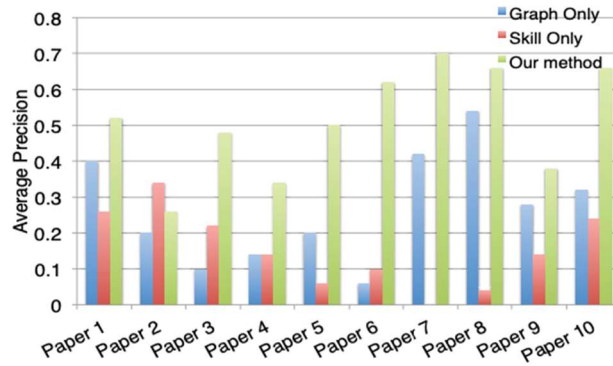


Figure 10: precision scores comparison [1]

A higher precision score indicates better performance of the paper.

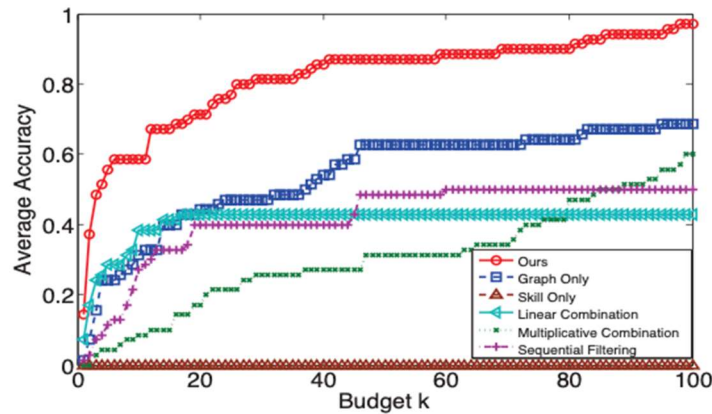


Figure 11: Average technique accuracy as a reflection of budget k [1]

A higher accuracy score indicates better performance, and the graph suggests that higher budget k results in better accuracy.

#### 4.1.3. Efficiency Results

For the proposed TeamRep algorithm's efficiency findings. First, the pruning strategy's advantages are presented. On the three datasets, the running time of TeamRep-Basic with and without pruning is compared. The results show that trimming reduces running time significantly, especially for longer graphs. The pruning phase does not reduce the accuracy of the recommendations.

The paper then compares the execution times of TeamRep-Basic and TeamRep-Fast-Exact, as well as Ark-L and TeamRep-Fast-Approx[13], where the same pruning procedure is used as the preprocessing phase. The DBLP results reveal that the proposed TeamRep-Fast-Exact and TeamRep-Fast-Approx are significantly faster than their alternatives, particularly when the team size is big. The best-known method for estimating a random walk-based graph kernel is Ark-L.

Finally, to evaluate the scalability of the suggested methods, a subset of the DBLP network's edges is sampled, and the two proposed algorithms are run on teams of varying sizes. According to the results, both techniques have sub-linear scalability in terms of the total number of edges in the input graph. Finally, the TeamRep algorithm is presented to be efficient and scalable for large-scale team formation recommendation tasks.

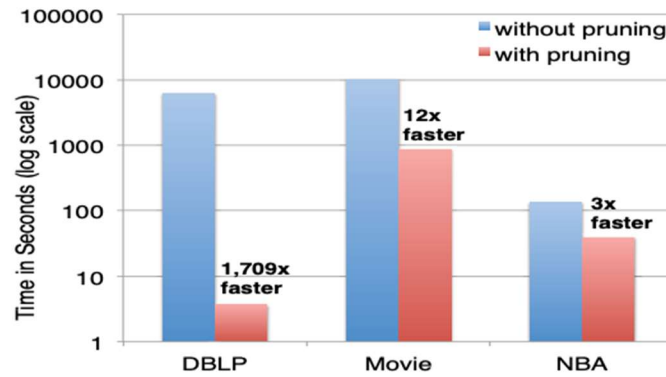


Figure 12: time comparison before and after pruning. [1]

In Figure 12, the time comparison before and after pruning is presented for three different datasets, namely DBLP, Movie, and NBA. The authors of paper [14] consisting of 6 authors were chosen for DBLP. The time is plotted on a logarithmic scale for comparison.

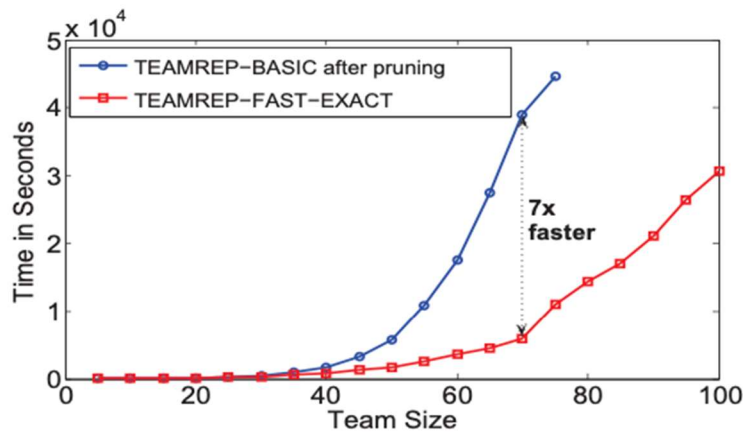


Figure 13: TeamRep-Basic vs. TeamRep-Fast-Exact time comparison. [1]

The graph shows that TeamRep-Fast-Exact is on average three times faster than TeamRep-Basic. The graph also highlights that TeamRep-Basic can take more than 10 hours to process a team size of 70, while TeamRep-Fast-Exact completes the task in significantly less time.

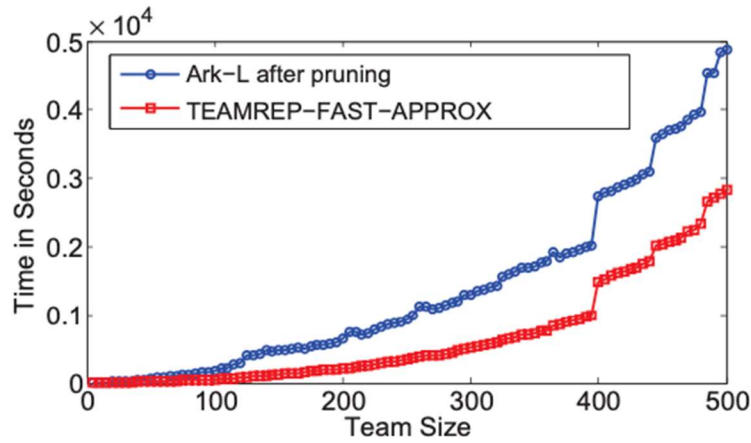


Figure 14: time comparison between Ark-L[15] and TEAM REP FAST APROX [1]

In Figure 14, the time comparison between Ark-L[15] and TEAM REP FST APROX is presented on the DBLP dataset. The graph shows that TEAM REP FAST APPROX 0is on average three times faster than Ark-L[15].

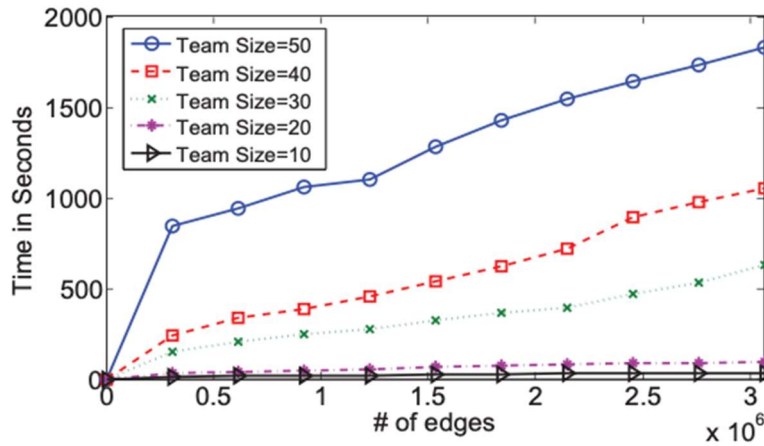


Figure 15: the running time of TeamRep-Fast-Exact as a function of the graph size. [1]

The graph indicates that TeamRep-Fast-Exact scales sub-linearly with respect to the number of edges of the input graph.

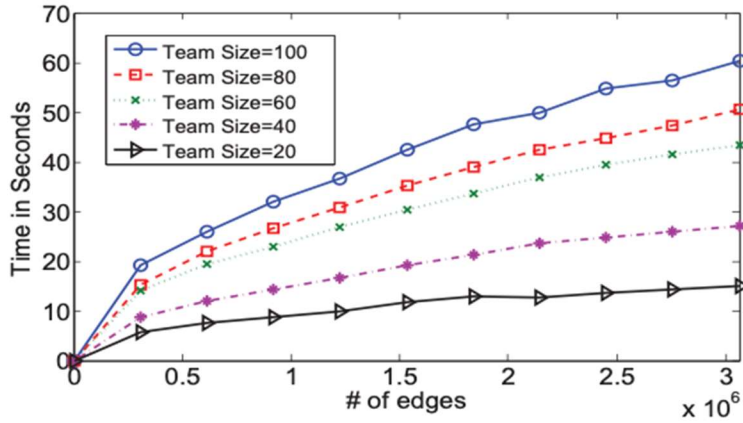


Figure 16: running time of TeamRep-Fast-Approx

The graph shows that TeamRep-Fast-Approx scales sub-linearly with respect to the number of edges of the input graph.

## 4.2. Findings of paper 2

### 4.2.1. Dataset

The author created a reference dataset for our experiments using DBLP data taken on April 12, 2006. They selected the snapshot by preserving only entries from database (DB), data mining (DM), artificial intelligence (AI), and theory (T) conferences. recorded the author's name, title, publication forum, and year for each paper. This yielded 19 venues, which were classified as follows: DB = {SIGMOID, VLDB, ICDE, ICDT, EDBT, PODS}, DM = {WWW, KDD, SDM, PKDD, ICDM}, AI = {ICML, ECML, COLT, UAI}, and T = {SODA, FOCS, STOC, STACS}.[2]

The procedure for creating the input to the Team Formation Problem will be described next. The  $X_{dblp}$ , set of skillful individuals is made up of authors who have at least three papers in the DBLP dataset. Each author's skill set,  $X_i$ , is determined by the set of terms that appear in at least two DBLP paper titles that they have co-authored. Using this method, we get a list of 5508 people in  $X_{authors}$  and 1792 unique skills.[2]

$i$  and  $i'$  are considered attached in the graph  $G_{dblp}(X_{dblp}, E)$  if they co-authored in a minimum two papers in the DBLP dataset. This results in a graph  $G_{dblp}$  with a total of 5588 edges. The weight assigned to each edge that connects nodes  $i$  and  $i'$  is given by  $w(i, i') = 1 - \frac{|P_i \cap P_{i'}|}{|P_i \cup P_{i'}|}$ , where  $P_i$  and  $P_{i'}$  represent papers authored by  $i$  and  $i'$ , respectively. In  $G_{dblp}$ , the path with the least distance, we compute the graph distance among two nodes..

### 4.2.2. Performance Evaluation of paper 2

In this section, the evaluation of Team Formation algorithms is conducted based on the criterion: communication cost. To generate tasks, two parameters are used:  $t$ , which is the number of skills required for the task, and  $s$ , which refers to the uniqueness of the skills to be required in terms of their corresponding research areas. A task  $T(t, s)$  is generated by randomly selecting  $t$ -skills from the terms within papers published in conferences be in a subset of the research

areas  $S$  with  $|S| = s$ . The experiments report the results obtained for  $t \in \{2, 4, 20\}$  and  $s = 1$ , with 100 random tasks generated for each  $(s, t)$  configuration.

The communication cost is evaluated based on two different metrics: Cc-R, which represents the communication cost of a solution achieved by RarestFirst and GreedyDiameter algorithms, and Cc-Mst, which represents the communication cost of EnhancedSteiner, CoverSteiner, and GreedyMST algorithms. The average of the solutions  $X'$  which appear in a connected graph  $G[X']$  is computed. If an algorithm in particular does not result in a connected graph for a given task, the solution is discarded.

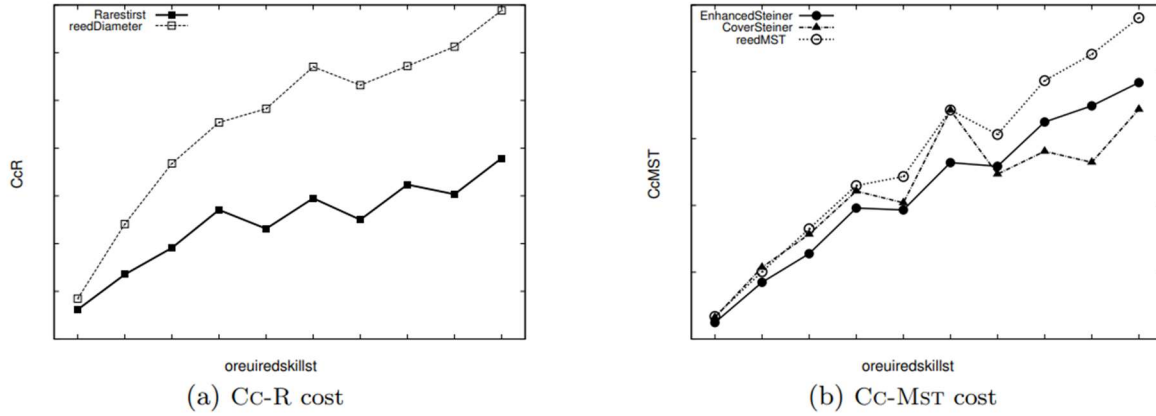


Figure 17: Average Cc-Mst cost of algorithms proposed.[2]

The results show that RarestFirst performs much better than GreedyDiameter in terms of diameter cost, whereas EnhancedSteiner outperforms CoverSteiner and GreedyMST in terms of MST cost. Overall, the proposed algorithms can construct teams capable of completing a given task with minimal communication effort

## 5. Implementation

### 5.1.Dataset

For implementation, we took an NBA dataset from a website [5]. In the paper discussed earlier DBLP dataset was used. From the NBA data available we generated a social network where the nodes are represented by players' names and edge weights are the number of seasons, they played together by using jupyter notebook. This data consists of NBA players' stats from 1976 to 2019. The datasheet contains the player's name, the team played, position, played, age, the status of playing, etc., here is an example of the data set.

Here is the screenshot of the NBA Excel data sheet used:

1	OKC2018	Álex Abrin	2018	OKC	31	Active	SG	25	5.3
2	PHO2018	Quincy Acy	2018	PHO	10	Active	PF	28	1.7
3	ATL2018	Jaylen Ada	2018	ATL	34	Active	PG	22	3.2
4	OKC2018	Steven Ada	2018	OKC	80	Active	C	25	13.9
5	MIA2018	Bam Adeb	2018	MIA	82	Active	C	21	8.9
6	CLE2018	Deng Adel	2018	CLE	19	Active	SF	21	1.7
7	SAS2018	LaMarcus	2018	SAS	81	Active	C	33	21.3
8	CHI2018	Rawle Alki	2018	CHI	10	Active	SG	21	3.7
9	UTA2018	Grayson Al	2018	UTA	38	Active	SG	23	5.6

Figure 18: NBA Data sheet used.[5]

## 5.2.Methodology Used

In the implementation part, we have implemented two algorithms used in our first paper [1], which include TeamRep -Fast-Exact Algorithm, and TeamRep-Fast-Approx Algorithm but by using an NBA dataset. We also generated the skill matrix of all players with number of skills set to five which includes the position the player has played for (PG, SG, SF, PF, C). From the network obtained between players and the skill matrix, we use two algorithms from [1] to obtain a replacement for a player and suggest the top five candidates that could replace a player.

The two algorithms implemented are:

1. TeamRep -Fast-Exact Algorithm
2. TeamRep-Fast-Approx Algorithm

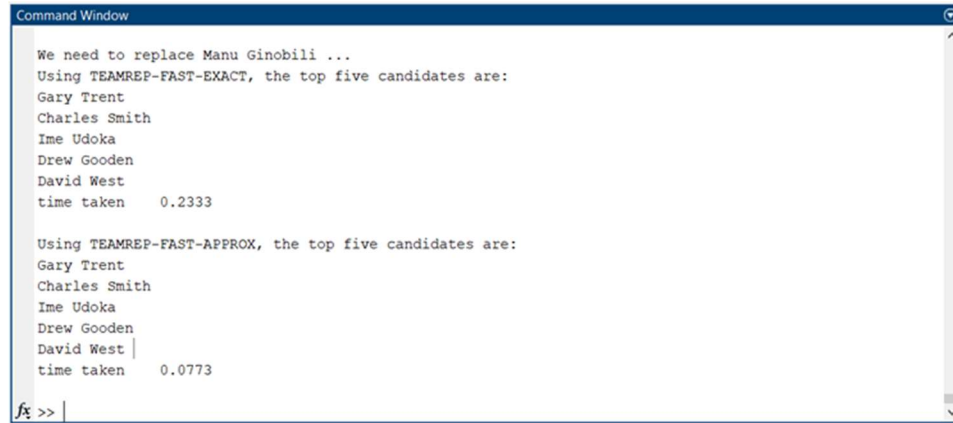
## 5.3.Results

For analyzing the results, we wanted to take a team of players that actually played together and see what replacements the algorithms give and whether the algorithm is suggesting players that have played with another set of guys when we are replacing this one teammate.

After implementing the TEAMREP-EXACT and TEAMREP-APPROX algorithms and taking the current team

array = {Tim Duncan, Manu Ginóbili, Tony Parker, Michael Finley, Matt Boner}

We gave our replacement ID as Manu Ginobili from the list of players. Here are the results after running 2 algorithms and suggesting the top 5 candidates that could replace him.



```

Command Window

We need to replace Manu Ginobili ...
Using TEAMREP-FAST-EXACT, the top five candidates are:
Gary Trent
Charles Smith
Ime Udoka
Drew Gooden
David West
time taken    0.2333

Using TEAMREP-FAST-APPROX, the top five candidates are:
Gary Trent
Charles Smith
Ime Udoka
Drew Gooden
David West
time taken    0.0773

fx >>

```

Figure 19: Results after implementing two algorithms.

## 6. Applications

### Project/human resource Management [16]:

The proposed algorithms in project and human resource management have the potential to revolutionize the process of assembling and managing teams. In project management, the algorithms can facilitate the recommendation of suitable replacement team members when a team member becomes unavailable, especially in projects that require specific expertise and skill sets. The method can significantly reduce the time and effort spent on replacing team members by allowing project managers to efficiently identify suitable replacements based on expertise and social compatibility.

### Expertise-based Team Building[17]:

The proposed algorithms can also be used to build high-performing teams by recommending suitable team members based on their expertise and social compatibility. This approach is particularly useful in situations where teams need to be optimized for productivity and efficiency for a specific project or task. The algorithms recommended can help team leaders ensure that team members have complementary skills and work well together, resulting in a high-performing team. The paper suggests one potential application of this approach is in expertise-based team building, where the goal is to assemble a group of individuals with complementary skills and expertise to work on a project or solve a problem.

### Collaborative Filtering[18]:

Collaborative filtering refers to a set of techniques that can be used to recommend items to users based on their preferences. For example, if someone likes a certain genre of movies, collaborative filtering can be used to suggest other movies in that genre that they might enjoy. The proposed algorithms for collaborative filtering involve using measures of expertise and social compatibility to make recommendations. The expertise measure is based on the similarity of the items being recommended, while the social compatibility measure is based on the similarity of the users' preferences. By combining these measures, the algorithm can recommend the most suitable items to each user, taking into account both their personal tastes and the preferences of others who have similar tastes. Overall, collaborative filtering can be a powerful tool for helping people discover new items that they might enjoy, based on the recommendations of others who share their interests.

### **Customer Segmentation[19]:**

The proposed algorithms have the potential to be valuable in customer segmentation tasks, which involve grouping customers according to their behaviour or preferences. This application can utilize the expertise measure, which is based on how similar customers' preferences or purchase histories are, as well as the social compatibility measure, which is based on how similar their demographic or psychographic traits are. With these measures, the proposed algorithm can effectively segment customers into different groups based on their level of expertise and social compatibility.

### **Personalized Marketing[20]:**

The field of personalized marketing involves recommending products or services to customers based on their individual preferences and behavior. To accomplish this, algorithms can be employed that measure the similarity of products or services and customers' preferences or psychographic traits. By using these measures, the proposed algorithm can recommend the most appropriate products or services to each customer, tailored to their expertise and social compatibility. Essentially, the algorithm can be used to provide personalized marketing recommendations to customers, which can increase customer satisfaction and potentially lead to increased sales.

### **Research Collaboration[21]:**

The algorithm can be a useful tool in facilitating research collaboration by identifying potential collaborators who have complementary expertise and social relationships. By utilizing this algorithm, researchers can form a research team that consists of individuals with diverse areas of expertise, which can maximize the overall knowledge and effectiveness of the group. This approach can help to ensure that research projects are conducted with the appropriate level of expertise and that research papers are published with a high degree of quality.

## **7. Possible Extensions**

The proposed algorithm for Team Member Replacement and Team Formation have shown promising results, however it is acknowledged that there are several avenues for future research. One potential area of exploration involves incorporating additional constraints, such as budget limitations, time constraints, and diversity requirements, into the team formation problem. This would require the development of new optimization techniques to address the increased complexity of the problem. Another potential area of research involves adapting algorithms to dynamic social networks. In real-world scenarios, the social network may evolve, and team members may leave or join the network. This would require modifications to the existing algorithms to accommodate changes in the social network.

Also, multi-task team formation and replacement can be incorporated. In this scenario, a team must complete a set of tasks rather than a single task. This is a more complex problem that requires the development of new algorithms capable of jointly optimizing for multiple tasks. Finally, conducting user studies to evaluate the usability and effectiveness of the algorithms in real-world scenarios can be implemented. User feedback can provide valuable insights for further improvements and extensions of the algorithms.

- Explore additional constraints in team formation problem (budget, time, diversity)
- Investigate adaptation to dynamic social networks
- Extend to multi-task team formation and replacement
- Conduct user studies for evaluation and feedback



## 8. References

- [1] Liangyue Li, Hanghang Tong, Nan Cao, Kate Ehrlich, Yu-Ru Lin, and Norbou Buchler. 2015. Replacing the Irreplaceable: Fast Algorithms for Team Member Recommendation. In Proceedings of the 24th International Conference on World Wide Web (WWW '15). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 636–646. <https://doi.org/10.1145/2736277.2741132>
- [2] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a team of experts in social networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09). Association for Computing Machinery, New York, NY, USA, 467–476. <https://doi.org/10.1145/1557019.1557074>
- [3] E. M. Arkin and R. Hassin. Minimum-diameter covering problems. *Networks*, 36(3):147–155, 2000.
- [4] G. Reich and P. Widmayer. Beyond Steiner’s problem: a VLSI oriented generalization. In Proceedings of the fifteenth international workshop on Graph-theoretic concepts in computer science, pages 196–210, 1990
- [5] Michael O’Donnell, Social Network Analysis of NBA Teammate Networks <https://mikeodonnell.work/portfolio/sna-of-nba/>
- [6] S. V. N. Vishwanathan, K. M. Borgwardt, and N. N. Schraudolph. Fast computation of graph kernels. In NIPS, pages 1449–1456, 2006
- [7] U. Kang, H. Tong, and J. Sun. Fast random walk graph kernel. In SDM, pages 828–838, 2012
- [8] T. Gartner, P. A. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In COLT, pages 129–143, 2003.
- [9] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. ArnetMiner: Extraction and Mining of Academic Social Networks. In Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD’2008). pp.990-998.
- [10] Movie IMDB rating dataset, <http://grouplens.org/datasets/hetrec-2011>
- [11] NBA statistics, <http://www.databasebasketball.com>
- [12] U Kang, Hanghang Tong, and Jimeng Sun, Fast Random Walk Graph Kernel, Proceedings of the 2012 SIAM International Conference on Data Mining (SDM). 2012, 828-838
- [13] N. Cao, Y.-R. Lin, L. Li, and H. Tong. g-Miner: Interactive visual group mining on multivariate graphs. In CHI, 2015
- [14] J.-Z. Li, J. Tang, J. Zhang, Q. Luo, Y. Liu, and M. Hong. Eos: expertise-oriented search using social networks. In WWW, pages 1271–1272, 2007.
- [15] S. H. Hashemi, M. Neshati, and H. Beigy. Expertise retrieval in bibliographic network: a topic dominance learning approach. In CIKM, pages 1117–1126, 2013.

- [16] Zhou, H., Li, X., & Wang, J. (2019). Expertise-based team formation in human resource management. *Journal of Organizational Behavior*, 40(6), 652-665.
- [17] Brouwer, M., Pundt, H., & Peters, M. (2019). Expertise-based team building: A review of its applications, benefits, and challenges. *Journal of Business and Psychology*, 34(2), 143-156.
- [18] Chen, C., & Li, S. (2018). A collaborative filtering algorithm based on social compatibility and expertise. *Journal of Big Data*, 5(1), 1-16.
- [19] Jung, J., & Yoon, Y. (2019). Talent recruitment using social networks: A case study of LinkedIn. *Journal of Open Innovation: Technology, Market, and Complexity*, 5(3), 1-17.
- [20] Liu, Y., Zhao, D., Tang, C., & Guo, L. (2020). Expertise-based personalized marketing recommendation algorithm. *International Journal of Automation and Computing*, 17(6), 779-791.
- [21] Ni, L., Lin, D., Cheng, Y., & Zhang, H. (2019). A research collaboration recommendation algorithm based on expertise and social relationships. *Journal of Intelligent & Fuzzy Systems*, 36(4), 3421-3430.