Team Formation with Negative Links

Maria Zerva
mzerva@cs.uoi.gr
Computer Science and Engineering
University of Ioannina

Ioannis Kouvatis
ikouvatis@cs.uoi.gr
Computer Science and Engineering
University of Ioannina

Abstract

Team formation is the problem of finding a team of experts that can effectively perform a task that requires a number of skills. Good communication among team members is of great importance and thus the social network of the experts must be taken into account. In this paper, we study the problem of finding a team of experts, in a social network, that also contains negative edges between experts. To the best of our knowledge, existing algorithms for team formation were designed for networks that contain only positive edges. Our goal is to minimize the communication cost function of the team's subgraph. As this problem is NP-Hard, we extend an existing approximation algorithm for team formation that is designed for this problem, to also take into account negative links. We experimented on a real dataset that contains negative edges and showed that effective teams were produced using our extension.

1 Introduction

To find a team of experts, that can effectively complete a project is not only a matter of them having the skills it takes, but also a matter of communicating well and without any problems with each other. A group of people is not only described by the good relationship and sympathy that two people can have for each other, but also by the dislikes and distrust among them. This problem applies in real life. Everyday in many different jobs, projects are assigned to teams of people. If in a certain team, two of the members, that have to work together to complete the project, do not like each other, the project is in danger and it can take ages to be accomplished.

While team formation as a problem has been studied a lot through the years, the social network that connects the experts has not been taken into consideration, in the process of finding the team, until a few years back. The studies of team formation that consider the social network between team members, to the best of our knowledge, have taken place for networks that only contain positive links between people.

In this project, we extend an existing algorithm for team formation, in order to produce results that have as many positive links as possible. The social network is modeled as a graph, with experts as nodes and edges between nodes that have a known way of communication, positive or negative. An edge that connects two nodes, that like and trust each other is a positive link and has a low weight as its communication cost, while an edge that connects two nodes, that dislike and distrust each other is a negative link with a high weight. Our goal is to produce teams, that consist of members that like each other and have no hatred between them. Our implementation accepts a team with an unfriendly link, only in the case that there is no other possible solution with only positive connections for a certain project.

We performed numerous experiments on a real dataset that contains both likes and dislikes between experts and compare the results of our implementation with another existing algorithm. Our algorithm produces teams with less negative links, while it still keeps the communication cost to the lowest possible.

Team formation in the presence of social network, that also contains negative relationships, that has not been studied before, along with the better results that have been found by the experiments conducted for this project, are our main contributions.

The remaining of this paper is organized as follows. Related work that has been conducted in this

area is presented in section 2. Definitions and the graph model are given in section 3. Section 4 describes the implementation of our work, while section 5 analyses the dataset that we used during our experiments. The experiments that took place and the evaluation of our implementation are described in section 6. The paper is concluded in section 7.

2 Related Work

The problem of finding a team of experts for a task, that takes into account the social network between the experts was first introduced by Lappas, Liu and Terzi in [1]. They used two different communication cost functions and stated that their minimization is an NP-Hard problem. Thus, they proposed three approximation algorithms that aim in minimizing these cost functions. The first communication cost function uses the diameter of the team's subgraph and one algorithm (RarestFirst) was designed as a solution. For the second communication cost function, that uses the cost of the minimum spanning tree (MST) in the team's subgraph, two algorithms (CoverSteiner and EnhancedSteiner) were designed.

Kargar and An in [2] also studied the problem of team formation. They proposed two different communication structures of a team (with and without a leader). For each one of those structures they created a communication cost function. Minimizing the first communication cost function (sum of distances) was proved to be an NP-Hard problem, so an approximation algorithm with ratio 2 was designed. For the second cost function (leader distance) an exact algorithm that minimizes it, was proposed. They also presented two procedures for producing top-K teams with minimum communication cost.

3 Preliminaries

3.1 Set Definitions

We define a set $S = \{s_1, s_2, ..., s_n\}$ that contains n skills, a set $E = \{e_1, e_2, ..., e_m\}$, that contains m experts, a set $S_i = \{s_{i1}, s_{i2}, ..., s_{ik}\}$ that contains the k skills that expert i has, where $S_i \subseteq S$, a set $C_j = \{c_{j1}, c_{j2}, ..., c_{jx}\}$, that contains x experts that have skill j, where $C_j \subseteq E$ and a set $P = \{p_1, p_2, ..., p_r\}$ that contains r skills, that are required for a certain project to be completed, where also $P \subseteq S$. Given the aforementioned sets, a team is defined as a set $T = \{t_1, t_2, ..., t_z\}$, that contains z experts that together cover the r skills that the certain project requires, where $T \subseteq E$.

3.2 Graph Model

Our model is a weighted undirected graph $G = (E, L)$, where E is the set of nodes that represent the m experts defined above and $L = \{l_1, l_2, ..., l_e\}$ is the set of e edges that connect the experts in E. Each edge in L has a weight w that represents the communication cost between two nodes. An edge between two nodes that can communicate well with each other can be labeled as positive and its weight is 1, while an edge between two nodes that can not communicate well with each other can be labeled as negative and its weight is a higher number. This higher number should be noted that is larger than the diameter of G. Since we want our team to communicate with ease, we apply our algorithm in the biggest connected component of G (contains over 90% of the nodes). We define the distance function $d(i, j)$ as the sum of the weights of the edges in the shortest path between node i and node j. We also define $minPath(i, j)$ as the function that returns the shortest path between node i and node j.

3.3 Problem Definition

Problem: Team formation is defined as the problem of finding a team of experts T, given a project $P = \{p_1, p_2, ..., p_r\}$, a set of experts $E = \{e_1, e_2, ..., e_m\}$, a set of skills $S_i = \{s_{i1}, s_{i2}, ..., s_{ik}\}$ for each expert $i \in E$ and the graph $G = (E, L)$, where $T \subseteq E$, so that the skills of the experts in T cover the skills in P and the communication cost function between the experts in the team is minimized.

In addition to the classical definition of team formation, taking into consideration that G also contains negative labeled edges, we allow negative links between team members only in the case when there is no possible team with no negative links. More specifically, we assume set $T_c = \{T_{c1}, T_{c2}, ..., T_{cb}\}$, to be the set of candidate teams that can complete P. Also let $Neg = \{neg_1, neg_2, ..., neg_q\}$ be the set of negative labeled edges in G, where $Neg \subseteq L$, $Pos = \{pos_1, pos_2, ..., pos_h\}$, be the set of positive labeled edges in G, where $Pos \subseteq L$ and $Pos \cup Neg = L$. Furthermore, we assume $Neg_{T_{ci}} = \{n_1, n_2, ...n_c\}$, to be the set of negative labeled edges in subgraph $G[T_{ci}]$, where $i = 1, ..., b$ and $n_j \in Neg$ for each $j = 1, ..., c$.

T_{ck} is the solution team of the problem when $|Neg_{T_{ck}}| \leq |Neg_{T_{ci}}| \; \forall T_{ci} \in T_c$.

Communication cost function definition: Assume team's T subgraph of $G = (E, L)$, $G[T]$, where $W_T = \{w_1, w_2, ...w_f\}$ is the set of the weights of the edges $\in G[T]$. Sum of weights (SW) is defined as follows:

$$SW = \sum_{i=1}^{f} w_i, \quad where \quad w_i \in W_T$$

4 Implementation

In order to solve the predefined problem we modified the *RarestFirst* algorithm designed by Lappas, Liu, Terzi and introduced in [1]. *RarestFirst* algorithm aims in finding the best team, for a project, that has the minimum diameter, that is the longest shortest path between any two nodes in the team. In [1] they prove that the minimization of the diameter is an NP-Hard problem and so *RarestFirst* is an approximation algorithm with ratio 2.

The first step of *RarestFirst* is to compute for each skill j in P, the set C_j, that contains the experts that have this skill. After that, the algorithm finds the skill r in P, that has the lowest $|C_r|$, which means that r is the rarest skill. Then, for every expert in C_r, computes a team with the closest experts in all other C_a sets, where $a \in P$ and $a \neq r$. *RarestFirst* chooses the expert in C_r, that has the team with the minimum diameter.

While *RarestFirst* aims in minimizing the diameter, our implementation aims in minimizing the sum of weights (SW) of the produced team. The main idea is that the minimization of the diameter does not consider the weights of all the edges, only the shortest path's edges, and thus the result team may contain one or more negative links, while there are available options with only positive links. Since our graph contains experts that dislike each other, our goal is to find a team with no conflicts if possible. Therefore, we consider sum of weights (SW) to be a more suitable cost function for solving this problem, in the sense that it considers the weights of all the edges in the team and since negative links have a higher weight than the positive ones, SW's minimization means that teams with no, or as less as possible negative edges, are preferred. Our implementation is shown in Algorithm 1.

Algorithm 1 The RarestFirst Algorithm using SW as cost function.

Input: Graph $G(E, L)$: experts' skill vectors $\{S_1, S_2, ..., S_m\}$ and project P.
Output: Team $T \subseteq E$ and subgraph $G[T]$.

1: **for** every $a \in P$ **do**
2: C_a={i | $a \in S_i$ }
3: $r \leftarrow arg\ min_{a \in P} |C_a|$
4: **for** every $i \in C_r$ **do**
5: **for** $a \in P$ and $a \neq r$ **do**
6: $Path_{ia} \leftarrow minPath(i, C_a)$
7: $Cand_i = \{x | x \in Path_{ia}\}$
8: $Sub_{Gi} \leftarrow G[Cand_i]$
9: $i* \leftarrow arg\ min_{sw} Sub_{Gi}$
10: $T \leftarrow Cand_{i*}$
11: $G[T] \leftarrow Sub_{Gi*}$

Notes: In line 6 of Algorithm 1 $minPath(i, C_a)$ gives the shortest path that expert i with skill r has to cross to get to an expert with another skill $a \neq r$, where $a \in P$. In line 7 $Cand_i$ set holds all the nodes(experts) that appear in all the shortest paths that expert r has to cross to get to all the other skills in P. In line 8 Sub_{Gi} is a subgraph of $G = (E, L)$ with the nodes that $\in Cand_i$. In line 9, $min_{sw} Sub_{Gi}$ gives the subgraph G_i that has the minimum value of cost function sum of weights(SW). In the same line, we denote $i*$ as the expert with skill r that produces the team with the minimum communication cost SW.

5 Dataset

We use *Epinions'* dataset to test our algorithm. *Epinions* contains information about users and the reviews that they make about products. Those users form a network, as they declare their trust or distrust with each other. When a user trusts another user, then there is an edge with label 1 between them. If there is distrust among them, then the label of the edge is -1.

Epinions' users are the experts in our model. Each product that has been reviewed, belongs to a certain category. These categories are used as the skills that the experts have.

Since we did not find a complete *Epinions* dataset with both information about reviews and negative links between users, we combined two

different datasets found online. The first one [1] is a mysql relational database containing information about the reviews, the users and the network between those users. The problem though, was that this network contained only positive edges. Then came the need for a second one [2], which is a directed signed network for *Epinions'* users, with both positive and negative edges. Using *Epinions'* unique user identifier, we created and use a dataset, that contains the network from the second dataset and the reviews from the first one only for users that participate in the aforementioned network.

Since the produced network was directed and we wanted to work with an undirected one, we replaced all duplicate edges with different labels (-1 and 1) with an edge of negative (-1) label and replaced duplicate edges of the same label with one edge. We followed this procedure, because we believe that negative edges are more important in the communication between two nodes. Edges that are not duplicate, are assumed to be not directed.

Taking into account the labels of the edges mentioned above, we assigned appropriate weights to negative and positive links. An edge between two nodes that can communicate well with each other and is labeled with 1, is assigned a weight of 1, while an edge between two nodes that can not communicate well with each other and is labeled with -1 is assigned a weight of a higher number. This higher number is larger than the diameter of G.

Using the dataset that we produced, we precomputed the frequency of each skill in it, in order to find the rarest skill with ease. We also pre-computed the set of skills (categories) of every user in the dataset, using the information contained in the reviews.

The created dataset contains 31.322 experts(users), 587 skills(categories) and in our network there are 210.078 edges between the experts. From the 210.078 edges, the 35.150 are negative (circa 16,7%).

The first dataset also contains information about users that are considered specialists in certain categories. From this, we produced in the same way as before, the network that connects them. This network contains 253 users and 212 edges. The different categories (skills), that these users have are 21. We also used this mini-dataset for testing.

[1] http://liris.cnrs.fr/red/

[2] https://snap.stanford.edu/data/soc-sign-epinions.html

6 Experiments and Evaluation

In order to compare our results with those of *RarestFirst's*, we also implemented original *RarestFirst* algorithm presented in [1]. Both *RarestFirst* and our modification of *RarestFirst* are implemented in python using networkx's libraries. The experiments were executed on an Intel(R) Core(TM) i5 2.53GHz with 8GB RAM.

To evaluate the results of our implementation with the results of *RarestFirst* algorithm we use a set of three metrics. The first metric is the diameter of the team, that is the longest shortest path between any two nodes in the team. The second one is the sum of weights (SW) and the third one is the cardinality of the team.

In figures 1, 2 and 3 we show the result of our modified *RarestFirst* implementation and original *RarestFirst* in terms of the diameter, SW and team cardinality on the best team on the *Epinions'* dataset.

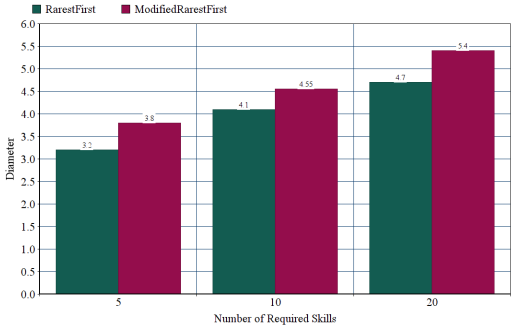

Figure 1: Comparison of modified *RarestFirst* and *RarestFirst* using diameter as performance metric.

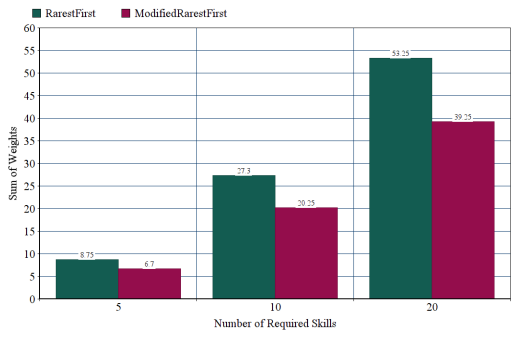

Figure 2: Comparison of modified *RarestFirst* and *RarestFirst* using SW as performance metric.

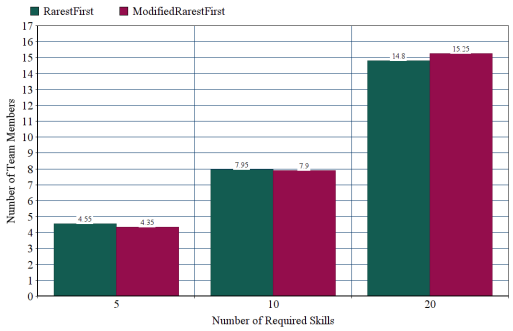

Figure 3: Comparison of modified *RarestFirst* and *RarestFirst* using team cardinality as performance metric.

In figures 1, 2 and 3 we can see the mean value of each measure after a set of experiments for the same random project for different number of skills using the two algorithms.

Figure 1 shows that original *RarestFirst* algorithm gives slightly better results in terms of the diameter, which is normal, because the aim of *RarestFirst* is to minimize diameter.

Figure 2 shows that modified *RarestFirst* gives better results in terms of sum of weights, which means that a team has a smaller total communication cost.

From figure 3 we see that both algorithms give almost the same number of experts for a certain project, so our modification does not make a lot of difference in the number of skills assigned per expert.

Our main goal was to produce teams with small communication cost, that have as less negative links as possible. In order to achieve this goal we had to sacrifice the diameter cost of the team, to avoid a negative link. Smaller diameter does not mean a better result in a network with negative edges, as it measures the cost between the two experts that are furthest away from each other, and does not measure the cost of all the required communication. A team with a negative link may have a smaller diameter than one with no negative links.

Figures 4 and 5 show the subgraph of the produced team of *RarestFirst* and modified *RarestFirst* for the same random project of 5 skills.

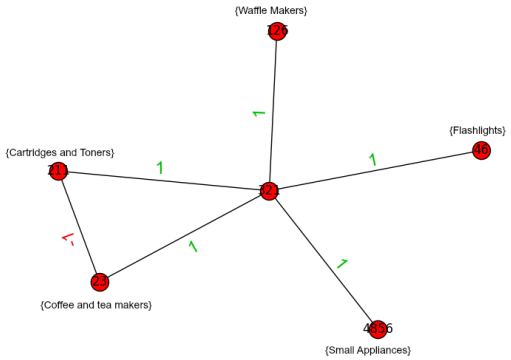

Figure 4: Subgraph of team produced with *RarestFirst* algorithm for project P={Coffee and Tea Makers, Waffle Makers, Small Appliances, Cartridges and Toners, Flashlights}.

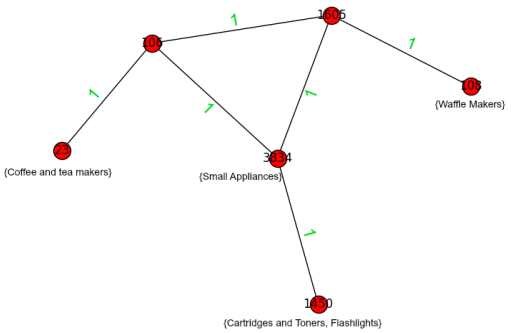

Figure 5: Subgraph of team produced with modified *RarestFirst* algorithm for project P={Coffee and Tea Makers, Waffle Makers, Small Appliances, Cartridges and Toners, Flashlights}.

In figures 4 and 5 we have a random project P={Coffee and Tea Makers, Waffle Makers, Small Appliances, Cartridges and Toners, Flashlights} as an example. The numbers on the edges represent the label -1 for negative links and 1 for positive. A negative link in our graph has a weight higher than the diameter of G (10), while a positive has weight 1. *RarestFirst* algorithm gives as best team the graph of figure 4. This team has diameter value 2 and SW value 15, since it contains a negative labeled edge. Modified *RarestFirst* gives figure 5 as the best team. This team has diameter 3 and SW value 6. This is an example of a case that smaller diameter is not preferred, as there is a better solution with no negative links and with slightly bigger diameter available.

7 Conclusions

In the work that we presented, the problem of finding a team of skilled experts that can complete a project, with a small communication cost and with no unfriendly links if possible, was studied. We modified an existing team formation algorithm, in order to make it avoid selecting teams with negative links in them. A great number of experiments were conducted on the real dataset of *Epinions'* network and we evaluated the results comparing them with those of an existing algorithm's.

8 References

[1] Theodoros Lappas, Kun Liu, and Evimaria Terzi, 2009. Finding a team of experts in social networks. In Proceedings of KDD09.
[2] Mehdi Kargar and Aijun An, 2011. Discovering Top-k Teams of Experts with/without a Leader in Social Networks. In Proceedings of CIKM11.