A Multi-objective Formulation of the Team Formation Problem in Social Networks

Preliminary Results

Julio Juárez
Department of Computer Science
CICESE Research Center
Ensenada, Baja California, México
jjuarez@cicese.edu.mx

Carlos A. Brizuela
Department of Computer Science
CICESE Research Center
Ensenada, Baja California, México
cbrizuel@cicese.mx

ABSTRACT

The Team Formation Problem in Social Networks (TFP-SN) consists of finding a team of experts, from a social network, that better undertake a given task. It is mandatory for the team to meet the skill set required by the task and it is desired that the team members communicate effectively to achieve their goal. This problem was proven to be NP-hard for the optimization of different variants of a communication cost function. Even though, real-life instances of this problem involve the simultaneous optimization of two or more conflicting objectives, the studies of the TFP-SN under the multi-objective model has been rather scarce. In this work, we introduce the TFP-SN as a multi-objective optimization problem for the maximization of two conflicting objectives, the collaborative density and the team's ratio of expertise. We tackle this problem employing the NSGA-II framework, for which a proper representation and variation operators are proposed. Experimental results show that the proposed approach generates competitive solutions when compared with well-known heuristics for this problem. Additionally, as a response to the lack of benchmarks and to setup a baseline for future comparisons, we provide a detailed description of the generated instances.

CCS CONCEPTS

• **Mathematics of computing** → **Combinatoric problems**; • **Computing methodologies** → *Genetic algorithms*;

KEYWORDS

Team Formation Problem, Social Networks, Multi-objective optmization, NSGA-II

ACM Reference Format:
Julio Juárez and Carlos A. Brizuela. 2018. A Multi-objective Formulation of the Team Formation Problem in Social Networks: Preliminary Results. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205634

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205634

1 INTRODUCTION

Within organizations, team formation is a most important process when the success of a task or project relies on the selection of adequate team members. An appropriate selection of members should provide with the required skill set, critical to complete a particular task, and allow an effective communication between members.

The team formation problem in social networks (TFP-SN) is typically modeled as a social graph. With the candidates being the nodes and the relationships among pairs of candidates being the edges. Additionally, each node is associated with a set of skills, which the candidate possesses. Also, each edge has a weight representing the affinity of its endpoints to work cohesively. Thus, given a social graph and a required set of skills, the TFP-SN concerns with finding a subset of nodes (candidates) from the social graph, such that it meets the required skills and whose communication cost is minimum. The TFP-SN is derived from the Team Formation Problem (TFP) originally introduced in the field of Operations Research [25]. In the TFP, the objective is to assemble a team of minimum cost whose members are associated to undertake a particular task; there is an inherent cost to each possible person-task association. Both problems seem, in principle, very similar. However, the consideration of the relations among candidates modeled in a social network graph entails a different challenge. For example, based on the assumptions made, the TFP may be seen as the classic task assignment problem [7], which is known to be solvable in polynomial time [16]. In contrast, the TFP-SN considering the minimization of the communication cost was proven to be NP-hard by reduction from the Multiple Choice Cover problem [17].

This problem attracted the attention of researchers [6, 10–12, 14, 15, 18–24] since its proposal by Lappas *et al.* in 2009 [17]. Partly because of the challenge that implies to solve it at optimality. Besides, intuitively, teams of skilled people whose communication cost is minimum may deliver their assignments more efficiently. Furthermore, it is of general interest to develop a model that can predict, to some extent, the performance of a team. Additionally, the proliferation of experts' social network websites such as DBLP, IMDB, Bibsonomy, or StackOverflow, for which datasets could be obtained, have brought appreciation to the problem.

The criteria to evaluate the performance of a team, in this context, has widened over time [21]. The first works considered the minimization of the cost of the minimum spanning tree and the diameter of the team [17, 18]. Next, the total sum of the weight of the edges between team members was considered [14]. Later,

the density of the team graph was studied [11, 20]. Limitations of some of these criteria have been reported. Although, there is not yet a consensus on which criteria a team should be evaluated [21]. Moreover, the study of the multi-criteria version for this problem has been somewhat scarce.

In this work, we present a multi-objective formulation for the TFP-SN which we solve with the well-known NSGA-II framework, for the simultaneous optimization of the collaborative graph density and the team's ratio of expertise. An appropriate individual representation and two variation operators are also introduced. A set of benchmark instances are generated to set a baseline for future comparison with other approaches.

The rest of the paper is organized as follows. Section 2 states the problem and describes the related work. Section 3 describes the algorithm we propose to solve the problem. Section 4 presents the experimental setup and the corresponding results. Finally, Section 5 exposes the conclusions and some ideas for future research.

2 PROBLEM DEFINITION

Let $\mathcal{X} = \{1, 2, \cdots, n\}$ be a pool (set) of indices representing n candidates and $\mathcal{A} = \{a_1, a_2, \cdots, a_m\}$ denote the universe of m skills. Also, let $X_i \subseteq \mathcal{A}$ denote the set of skills candidate i possesses. In other words, if say candidate i can provide the skill a_j, then $a_j \in X_i$. Additionally, let us define the support set $S(a_j) \subseteq \mathcal{X}$ of an arbitrary skill $a_j \in \mathcal{A}$ as $S(a_j) = \{i \in \mathcal{X} | a_j \in X_i\}$, i.e., the subset of candidates that satisfy a_j.

Under the *basic task* team formation context, a task $T \subseteq \mathcal{A}$ is simply defined as the subset of skills required for its completion. However, we address the generalized version of this problem by defining a task $T = \{(a_j, k_{a_j})\}$ as a set of pairs (a_j, k_{a_j}), where $a_j \in \mathcal{A}$ and $k_{a_j} \in \mathbb{N}$. That is, a task T demands at least k_{a_j} people skilled in a_j, for each $(a_j, k_{a_j}) \in T$. Furthermore, let $A(T) \subseteq \mathcal{A}$ denote the set of skills in a task T.

For any given task $T = \{(a_1, k_{a_1}), (a_2, k_{a_2}), \cdots, (a_m, k_{a_m})\}$, a team is defined as $\mathcal{X}' \subseteq \mathcal{X}$ such that $|\mathcal{X}' \cap S(a_j)| \geq k_{a_j}$ for each $j \in \{1, 2, \cdots, m\}$. Note that the size of a team $\mathcal{X}'$ is $|\mathcal{X}'| \leq \sum_{j \in \{1, \cdots, m\}} k_{a_j}$, bounded by the total sum of required experts. It may be the case that a candidate is skilled in more than one of the required skills by T.

Let $G = (\mathcal{X}, E)$ be a undirected weighted social graph, where $\mathcal{X}$ is the pool of indices of candidates and E represents the set of weighted edges among pairs of candidates. For a given graph G and a team $\mathcal{X}' \subseteq \mathcal{X}$, the induced subgraph $G[\mathcal{X}']$ is the graph whose vertex set is strictly $\mathcal{X}'$ and whose edge set consists of all of those edges $e = (i, i')$ in E whose nodes i, i' are in $\mathcal{X}'$.

In the next subsections we introduce the optimization criteria we are going to use in our formulation of the problem.

2.1 Objective 1: Density

Depending on the context, the weight $w(e)$ of an edge e in E may represent communication *cost* or *gain*. In the particular context of this work, an edge $e = (i, i')$ exists to denote collaboration between any pair of candidates i and i', hence the weight $w(e) \in \mathbb{N}$ is a gain. The higher (lower) the weight of an edge, the higher (lower) the cohesion of its endpoints.

In this study, we are interested in those teams that translate into high *collaborative density*. Given a graph G and a team $\mathcal{X}' \subseteq \mathcal{X}$, the collaborative density is simply the weighted density of $G[\mathcal{X}']$, defined by the following equation

$$D(\mathcal{X}') = \sum_{e \in E_{G[\mathcal{X}']}} \frac{2 \cdot w(e)}{|\mathcal{X}'| \cdot (|\mathcal{X}'| - 1)} \quad , \tag{1}$$

where $E_{G[\mathcal{X}']}$ is the set of edges of the induced subgraph $G[\mathcal{X}']$. This objective for the TSP-SN is different from the one previously introduced in [11]. The difference is explained in Subsection 4.5.

2.2 Objective 2: Expertise

Besides the density objective, it is of a particular interest in this study to assembly a team whose members are most experienced for the required skills of a task T. It is arguably more convenient to pick candidates who are more experienced than the rest, on the skill set of interest.

Let the *level of expertise* of a candidate $i \in \mathcal{X}$ over a particular skill $a_j \in X_i$ be denoted as $z_i(a_j)$. Given a task T, a graph G and a team $\mathcal{X}' \subseteq \mathcal{X}$, the total level of expertise of the team is defined as:

$$Z(\mathcal{X}') = \sum_{i \in \mathcal{X}'} \sum_{a_j \in (X_i \cap A(T))} \frac{z_i(a_j)}{|\mathcal{X}'|} \quad . \tag{2}$$

Note that the total level of expertise $Z(\mathcal{X}')$ is scaled by $\frac{1}{|\mathcal{X}'|}$, that is to avoid any team to grow indiscriminately. To the best of our knowledge, this is the first time this objective is considered in the TFP-SN context.

2.3 Multi-objective TFP-SN

In the multi-objective optimization paradigm, every conflicting objective is equally relevant to the problem. There is no single "best" solution, in a strict sense, rather than a set of multiple trade-off solutions. Trade-off solutions may favor one or more objectives with retribution to the other objective(s), but none of these solutions excel in all objectives at once [3, 4, 8].

In this spirit, we have defined two conflicting objectives to consider when forming a team: the density objective and the expertise objective. It is intuitive to strive for the assembly of a team consisting of the most collaborative and the most experienced team possible. However, we make no distinction between the most collaborative team that happens to be the most inexperienced, and the most experienced team that cannot work together as a unit. Both are, in a sense, equally as good (or perhaps equally as bad).

The multi-objective team formation problem in social networks is formally defined in the following. Given a social graph $G = (\mathcal{X}, E)$, with $\mathcal{X} = \{1, 2, \cdots, n\}$ a set of n candidates, and a task $T = \{(a_1, k_{a_1}), (a_2, k_{a_2}), \cdots, (a_m, k_{a_m})\}$ requiring m skills,

$$\begin{aligned} \underset{\mathcal{X}' \in \mathcal{X}}{\text{maximize}} \quad & (D(\mathcal{X}'), Z(\mathcal{X}')), \\ \text{subject to} \quad & |\mathcal{X}' \cap S(a_j)| \geq k_{a_j}, \\ & \forall a_j \in A(T). \end{aligned} \tag{3}$$

2.4 Related work

2.4.1 Single criterion. Lappas et al. [17] were the first to address the Team Formation Problem in the presence of social network

graphs (TFP-SN). The team to be formed is required to meet the skills of the task at hand. They measured the effectiveness of a team using two different communication-cost functions, the cost of the minimum spanning tree and the diameter of the subgraph induced by the team. This problem was proven to be NP-hard. In an attempt to solve this problem, they proposed two algorithms, the RarestFirst algorithm for the diameter-based objective, and the EnhancedSteiner algorithm for the MST-based objective.

Li and Shan [18] took the TFP-SN one step further by generalizing the problem, in which a minimum number of persons are required for each skill in a task. They presented an adaptation of the EnhancedSteiner algorithm to tackle the generalization of the problem. This adaptation, however complete, was mentioned to be computationally costly regarding the number of skills required or in the size of the graph. Consequently, they proposed a grouping method that narrows the search for candidates within groups, for the generalized EnhancedSteiner algorithm.

Kargar and An [14] pointed out the problem of using the MST and diameter as metrics of cost for the teams. The criticism is that a minor change in the input graph may result in a radically different solution.

Gajewar and Sarma [11] addressed this issue by measuring a team's compatibility in terms of the density subgraph metric. Then, the goal is to find a team that maximizes the subgraph density such that the requirements of the task are covered. This formulation is also proven to be NP-hard. In this sense, they proposed a density-based algorithm named m-DensityAlk. However, the downside of this algorithm is that there is no guarantee to find a connected solution if any, as opposed to the distance-based methods. Rangapuram et al. [20] also proposed an utterly different density-based algorithm, although the same inconvenience is present.

2.4.2 Multiple criteria. Regarding multiple criteria for the TFP-SN, the literature is not abundant. Additionally, a considerable amount of these works [6, 10, 13, 15] are based on the linear weighted sum method for optimization, which will miss nonsupported solutions for nonconvex problems [8] like the one we study here.

Niveditha et al. [19] proposed a genetic algorithm for the TFP-SN with three objectives. This work involves the simultaneous minimization of a communication cost objective, a personnel cost objective, and a team size objective. The NSGA-II framework is also used for this study. Unfortunately, their results are not detailed or available for comparison.

2.4.3 Others. Some works have introduced variations, or extensions, for the TFP-SN such as time [22, 23], leader [14], social influence [24], or geographical location [12], to name a few. These variants are out of the scope of this study.

3 MULTI-OBJECTIVE GENETIC ALGORITHM

To solve the multi-objective TFP-SN problem defined in the section above, we employ the well-known NSGA-II framework. It is a multi-objective genetic algorithm that, through the years, has empirically proven itself useful for solving a wide variety of multi-objective problems. Notice that the MO method could also be any of other more recent MOEA approaches, however, for the sake of having a first glance at the trade-off solutions we selected this successful and well-known technique.

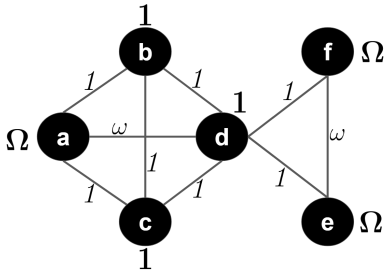


Figure 1: Example of social network graph.

The NSGA-II features three key aspects. First, a fast non-dominated sorting routine, which partitions efficiently the pool into non-dominated fronts under the Pareto's concept of non-dominance. Second, elitism, as the best individuals are carried out from one generation to the other; this is a necessary condition for convergence to the Pareto front. Third, a diversity preservation mechanism, which favors solutions that are more isolated than other solutions in the same non-dominated front. For further details of the algorithm, the reader is referred to the original work [5].

In general, genetic algorithms are highly customizable, the outline of the algorithms are merely a backbone with placeholders for the operators. The components of a typical genetic algorithm are mostly inherent to the problem. The representation of the individuals, initialization procedure, recombination and mutation operators that we propose for the TFP-SN, are described throughout this section. The selection method is crowded-based binary tournament, and the replacement is also native to the NSGA-II.

3.1 Representation of the solution

For the multi-objective TSP-SN, we represent any feasible solution or individual I straightforward as the definition of a team presented in Section 2. Given a graph $G = (\mathcal{X}, E)$ and a task T, a *feasible solution* $\mathcal{I}$ is a subset of $\mathcal{X}$ such that $|\mathcal{I} \cap S(a_j)| \geq k_{a_j}$, for each $a_j \in A(T)$.

3.1.1 Example. Suppose there is a necessity to form a team with the skill set defined by $T = \{(s_1, 1), (s_2, 3), (s_3, 1)\}$ in an organization. Also, suppose a set of available candidates $\mathcal{X} = \{a, b, c, d, e, f\}$ whose social network graph G is depicted in Figure 1. Finally, suppose that the skill set of each candidate is as follows $X_a = X_d = \{s_2\}$, $X_b = X_c = \{s_1\}$, $X_e = \{s_1, s_2\}$, $X_f = \{s_2, s_3\}$. Then, each of the following subsets $\mathcal{I}_1 = \{a, e, f\}$, $\mathcal{I}_2 = \{a, d, e, f\}$, $\mathcal{I}_3 = \{a, b, d, f\}$ are feasible individuals. Note that, the induced subgraphs of the solutions are not necessarily connected, as it is the case of $G[\mathcal{I}_1]$.

3.1.2 Fitness of solutions. The fitness of any solution $\mathcal{I}$ is given by the pair $(D(\mathcal{I}), Z(\mathcal{I}))$ whose elements are defined by equations (1) and (2), respectively; and the objective function is given by equation (3).

To illustrate this, let us recycle the example from above. The respective level of expertise $Z(\{i\}) = \sum_{a_j \in (X_i \cap A(T))} z_i(a_j)$, of each

candidate i, is highlighted with a boldface number next to its corresponding node. Similarly, for each edge e, the weight $w(e)$ is highlighted with an italics number next to its corresponding edge. Both Ω and ω are arbitrary large constants. Then, the fitness of individuals $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{I}_3$ are $(\frac{\omega}{3}, \Omega)$, $(\frac{\omega+1}{3}, \frac{3\Omega+1}{4})$, $(\frac{\omega+3}{6}, \frac{\Omega+1}{2})$, respectively. Solutions $\mathcal{I}_1$ and $\mathcal{I}_2$ are non-dominated, and $\mathcal{I}_3$ is dominated by the formers.

3.2 Initialization of the pool

In the context of genetic algorithms, it is desired an initialization routine to generate individuals randomly [9]. Even better, if those individuals also belong to the feasible search space.

Algorithm 1 shows the heuristics used to generate N random and feasible individuals. The `ReservoirSampling`(S, k) function, in Line 5, returns a randomly sampled subset of size k from a given set S. Thus, in lines 4 to 6, we ensure that every generated individual meets the requirements of having at least k_{a_j} team members for each $a_j \in A(T)$.

Algorithm 1: Initialization

input : a set of n candidates $\mathcal{X}$, a task T of m skills, and an integer N
output : a set P of $|P| = N$ (feasible) individuals

```
1  P ← ∅;
2  while |P| < N do
3  |   I ← ∅;
4  |   for a_j ∈ A(T) do
5  |   |   I ← I ∪ ReservoirSampling(S(a_j), k_{a_j});
6  |   end
7  |   P ← P ∪ {I};
8  end
9  return P;
```

3.3 Operators

We introduce two variation operators for the TFP-SN, one corresponding to the recombination and one to the mutation. Both operators ensure that the resulting individuals are within the feasible search space.

3.3.1 Recombination. Algorithm 2 ensures that the resulting offsprings are always valid. Iteratively, for each skill $a_j \in A(T)$, with a probability $p = 0.5$, the support set $S_{P_1}(a_j)$ for the skill a_j of parent P_1 is combined with the child O_1 and $S_{P_2}(a_j)$ of P_2 is combined with O_2. In the same iteration, with a probability of $1 - p$ the opposite combination is made, that is, $S_{P_1}(a_j)$ is combined with O_2 and $S_{P_2}(a_j)$ is combined with O_1. Where $S_{P_i}(a_j)$ is the support set of skill a_j from parent P_i, $i \in \{1, 2\}$.

For instance, consider the parents $P_1 = I_1$ and $P_2 = I_3$ and the task T, from Subsection 3.1.1, as input for Algorithm 2. Suppose that for each skill $\{s_1, s_2, s_3\}$ in $A(T)$, we got Head, Tail, and Head, respectively, from the `CoinFlip()`. Then, the resulting offspring O_1 is $S_{P_1}(s_1) \cup S_{P_2}(s_2) \cup S_{P_1}(s_3) = \{e\} \cup \{a, d, f\} \cup \{f\} = \{a, d, e, f\}$. Consequently, the mirroring offspring is given by $O_2 = S_{P_2}(s_1) \cup S_{P_1}(s_2) \cup S_{P_2}(s_3) = \{a, b, e, f\}$. Note that both individuals are feasible, as all the skill positions required by T are covered.

However, there is an inconvenient with this method. If $m = |A(T)|$ is relatively small, say less than 4, the probabilities of getting an offspring identical to the parents is high, but decreases exponentially with respect to m. Suppose an input of a task T of m skills, and two parents P_1 and P_2 such that $P_1 \neq P_2$ and $S_{P_1}(a_j) \neq S_{P_2}(a_j)$, for each $a_j \in A(T)$. The chances of the `CoinFlip()` of getting either consecutive Heads or Tails m times is $\frac{2}{2^m}$ (if fair). That means that there is at least a probability of $\frac{1}{2^{m-1}}$ that the resulting offspring from Algorithm 2 is identical to the parents.

Algorithm 2: Recombination

input : two individuals, P_1 and P_2, and a task T
output : two individuals, O_1 and O_2

```
1  O_1 ← ∅;
2  O_2 ← ∅;
3  for a_j ∈ A(T) do
4  |   if CoinFlip() = Heads then
5  |   |   O_1 ← O_1 ∪ S_{P_1}(a_j);
6  |   |   O_2 ← O_2 ∪ S_{P_2}(a_j);
7  |   else
8  |   |   O_1 ← O_1 ∪ S_{P_2}(a_j);
9  |   |   O_2 ← O_2 ∪ S_{P_1}(a_j);
10 |   end
11 end
12 return O_1, O_2;
```

3.3.2 Mutation. Algorithm 3 also ensures that the resulting individual is feasible. First, from a copy of the input individual, a random team member is removed. Then, if there is no deficit on the number of required people in each skill, the new individual is returned. Else, one random person is added to the new team (individual) out of the support set of any of the skills (also selected at random) where there is a deficit. This last operation is repeated until no deficit to cover remains.

For instance, let us consider once again the example from Section 3.1.1. Suppose the individual $\mathcal{I} = \{a, d, e, f\}$ along with T are the input to Algorithm 3. Suppose also that the team member d was selected and deleted from a copy $\mathcal{I}'$ of $\mathcal{I}$, from the Line 1 of the Algorithm 3. The resulting individual is $\mathcal{I}' = \{a, e, f\}$. Since, $T = \{(s_1, 1), (s_2, 3), (s_3, 1)\}$, and $X_a = \{s_2\}$, $X_e = \{s_1, s_2\}$, $X_f = \{s_2, s_3\}$, then the deficit is $D = \emptyset$ and $\mathcal{I}'$ is returned.

4 EXPERIMENTS

In this section, we evaluate the solutions produced by the MOGA-TFP-SN using the DBLP dataset to model the social network graph over some randomly generated tasks as the benchmark. We present non-dominated fronts and examples of the diversity of solutions. We also show that the solutions returned by our method disputes just fine over criteria other than the evolved, when compared to other algorithms. Finally, we discuss the differences between the graph density and the subgraph density objectives, and the advantages of the first over the second when evolving solutions.

Algorithm 3: Mutation

input : one individual I and a task T
output: one individual I'

$\mathcal{I}' \leftarrow \mathcal{I} \setminus$ `ReservoirSampling`$(\mathcal{I}, 1)$;
$D \leftarrow$`SkillDeficit` $(\mathcal{I}', T)$;
while $|D| > 0$ **do**
  $a_j \leftarrow$ random skill from D;
  $\mathcal{I}' \leftarrow \mathcal{I}' \cup$ `ReservoirSampling`$(S(a_j), 1)$;
  $D \leftarrow$`SkillDeficit` $(\mathcal{I}', T)$;
end
return I';

4.1 The DBLP dataset

The DBLP is an online bibliographic information system on computer science publications and authors. The data is openly available through webpage[1] or XML file[2]. We used the DBLP XML data from a snapshot taken on March 02, 2015, to produce the input social network graph. To derive experts, skills, and relationships from the dataset to the graph, we adopted most of the methodology described in [11, 17]. The papers published in 16 selected conferences are categorized in the domains of Artificial Intelligence (AI), Database (DB), Data Mining (DM), and Theory (T) as follows: AI = {ICML, ECML, COLT, UAI}, DB = {SIGMOD, VLDB, ICDE, ICDT}, DM = {WWW, KDD, SDM, PKDD}, and T = {SODA, FOCS, STOC, STACS}. The set of candidates $\mathcal{X}$ contains the authors with at least three publications within at least one of the defined domains, independently. In other words, if the candidate i has three or more publications in, say AI, then $i \in S(\text{AI})$ and $\text{AI} \in X_i$. Moreover, there is an edge $e = (i, j)$ in E, $i, j \in \mathcal{X}$, if candidates i and j are coauthors in at least two publications. Then, the weight of any edge is $w(e) \geq 2$, determined by the number of publications whose endpoints coauthored. The level of expertise $z_i(a_j)$ of a candidate $i \in \mathcal{X}$ over a particular skill $a_j \in \mathcal{A}$ is given by the total number of publications within the skill domain a_j. For example, $z_i(\text{AI})$ is determined by the number of papers in which the candidate i appears as author, within the conferences ICML, ECML, COLT, and UAI.

4.2 Test instances

Due to the lack of bechmark instances, we have randomly generated five tasks for each size of $k = \{4, 8, 12\}$, 15 tasks in total, as benchmark for experimentation. Let us recall that a task T is defined as a set of m pairs $\{(a_j, k_{a_j})\}$, $j \in \{1, 2, \cdots, m\}$, where k_{a_j} specifies the minimum required number of experts in a team to cover the skill a_j, and its size is given by $k = \sum_j k_{a_j}$. Each task T is generated by choosing k skills with replacement from the universe of skills $\mathcal{A} = \{AI, DM, DB, T\}$. The test instances (tasks) are presented in Table 1.

Unlike other works [11, 20], we limit k to smaller values. Since any size efficient team $\mathcal{X}'$ on T is bounded by $\max_{a_j \in A(T)} k_{a_j} \leq |\mathcal{X}'| \leq k$, other than to measure scalability of the method, there is no practical reason to solve for larger teams. Furthermore, in practice, team performance will drop to a significant degree concerning the size of the team [1, 2].

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

Tasks	k	AI	DM	DB	T
Test 1	4	1	1	0	2
Test 2	4	1	1	2	0
Test 3	4	2	0	0	2
Test 4	4	0	3	1	0
Test 5	4	2	1	1	0
Test 6	8	3	2	3	0
Test 7	8	1	2	4	1
Test 8	8	2	2	2	2
Test 9	8	2	3	2	1
Test 10	8	2	3	2	1
Test 11	12	4	2	3	3
Test 12	12	3	3	3	3
Test 13	12	2	3	5	2
Test 14	12	4	0	6	2
Test 15	12	1	5	4	2

Table 1: Randomly generated benchmark set of test instances where each row represents a task.

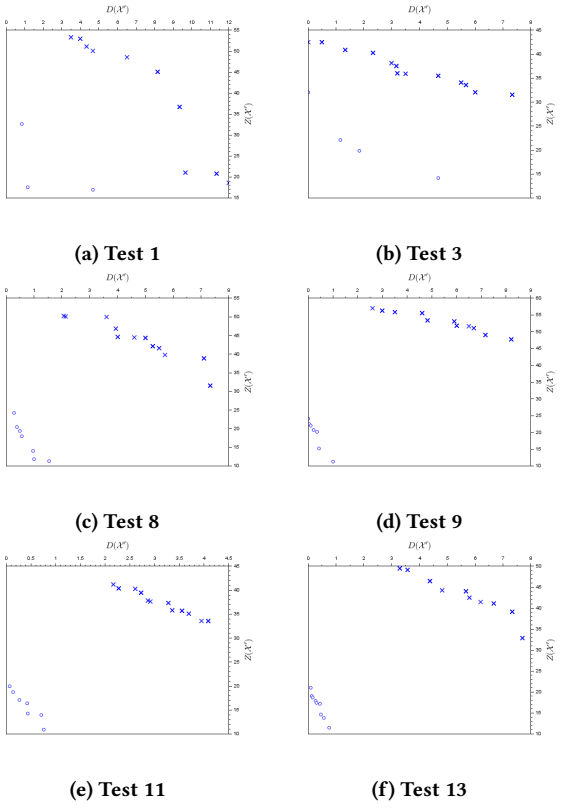
4.3 Experimental setup and results

The MOGA-TFP-SN algorithm and the variation operators were implemented in Java (1.8) and the corresponding tests were performed on a Dell Precision T3610, Intel Xeon CPU (64 bits), Windows 10 machine with 16 GB of memory.

The running parameters for the NSGA-II are set to the following values, population size = 100, tournament size = 2, generations = 200, recombination probability = 0.95, mutation probability = 0.08. No special effort is spent in optimizing the input parameter values. The input network social graph is generated as explained earlier in Subsection 4.1 and the input tasks are given by Table 1. We run the MOGA-TFP-SN 10 times, for each test (task).

In each run, the MOGA-TFP-SN outputs the non-dominated set of solutions of the last population. The non-dominated sets resulting from each independent run are stored by test case. Then, the consolidated non-dominated front is computed for each case. Figure 2 shows the consolidated non-dominated fronts for the test instances 1, 3, 8, 9, 11, and 13. From this figure, it is clear that there is an improvement in the solutions generated initially, wherein a significant gap separates both consolidated, initial and resulting, non-dominated fronts. The same pattern is observed in the non-dominated fronts of the other test cases (not shown here). The average number of non-dominated solutions in the consolidated fronts is $8.\bar{6}$. The smallest populated front corresponds to Test 2 with two solutions, while the most populated corresponds to Test 6 with 14 solutions.

Regarding solutions, it is worth noting that the resulting solutions exhibit a broader diversity than that inherent in their values of density ($D(\mathcal{X}')$) and expertise ($Z(\mathcal{X}')$). Particularly in the sense of the topology of the induced subgraph of the solution. These subgraphs may be connected or disconnected (in two or more components). In other words, within non-dominated fronts, solutions' induced subgraphs may be found in the form of cliques, diamond graphs, bipartite graphs, isolated vertices, or a combination of them. Figure 3 shows the consolidated non-dominated solutions found for



(a) Test 1 **(b) Test 3**

(c) Test 8 **(d) Test 9**

(e) Test 11 **(f) Test 13**

Figure 2: Consolidated initial (○) and resulting (×) non-dominated fronts from multiple test instances.

Test 12, wherein the induced subgraph of four different solutions is displayed. This is an example of how solutions that differ in team size and topology also reside in the same front. Apart from the objective function values, the decision maker may benefit from the populational nature of this method which generates a diverse set of teams.

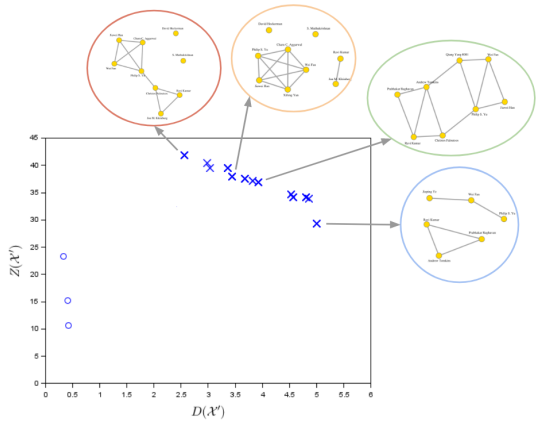


Figure 3: Consolidated non-dominated front of Test 12 and the topology of some of its solutions.

Despite the advantages of the density-based objective over the distance-based objective discussed in [11, 20]. The algorithms for the former [11, 20] do not guarantee connectivity of the solution if any, as opposed to the algorithms for the later [14, 15, 17, 18]. Our method, however density-based, was able to find at least one connected solution in 14 out the 15 test cases. It should be noted that there is no particular mechanism implemented, not a restriction or objective, for driving that to happen. The aforementioned suggests a natural affinity for this method to find connected solutions. However, more experiments are required to have substantial evidence of this result.

Finally, not a single solution was found to have more than k experts, i.e., all solutions within the consolidated non-dominated fronts are size efficient teams. Moreover, the average size of a team per size of $k = \{4, 8, 12\}$ are 3.57, 5.06, and 7.7, respectively. This is mainly attributed to two things, the initialization procedure generates solutions of size at most k, and the mutation operator decreases the size of the team if a team member with redundant skill to the task is selected randomly.

4.4 Other algorithms and objectives

Any given solution $\mathcal{X}'$ can be evaluated through multiple criteria, such as diameter, density, cardinality, to name a few. In this subsection, we will compare MOGA-TFP-SN solutions with those obtained by other algorithms, over multiple criteria.

The criteria to be compared are presented below:

D($\mathcal{X}'$) Density of the graph $G[\mathcal{X}']$ (Eq. 1).
Z($\mathcal{X}'$) Level of expertise (ratio) of team members in $\mathcal{X}'$ (Eq. 2).
sD($\mathcal{X}'$) Subgraph density of the graph $G[\mathcal{X}']$.
Cc-mst($\mathcal{X}'$) Cost of the minimum spanning tree of $G[\mathcal{X}']$.
Cc-R($\mathcal{X}'$) The diameter cost of $G[\mathcal{X}']$.
$\bar{\kappa}(\mathcal{X}')$ The number of disconnected components in $G[\mathcal{X}']$.
$|\mathcal{X}'|$ The size of the team $\mathcal{X}'$.

As defined earlier, the weight $w(e)$ of an edge e represents collaboration between candidates. Thus, we defined the distance cost $c(e)$ of an edge e to be $c(e) = |w(e) - \max_{e' \in E} w(e')|$, which is used for Cc-mst($\mathcal{X}'$) and Cc-R($\mathcal{X}'$) criteria. Also, please be advised that the graph density, although closely related, is different from the subgraph density. This difference is addressed and discussed in the next subsection.

Two well-known heuristics were implemented for the TFP-SN, a generalization of Lappas et al.'s RarestFirst algorithm [17], and Gajewar and Sarma's m-DensestAlk algorithm [11]. The RarestFirst algorithm objective is to find a team $\mathcal{X}'$ that minimizes the diameter (Cc-R($\mathcal{X}'$)) of the induced subgraph $G[\mathcal{X}']$, in addition to the skill set constraint imposed by the task T. A generalization of this algorithm is presented in [11]. The m-DensestAlk algorithm objective is to find a team $\mathcal{X}'$ that maximizes the subgraph density (sD($\mathcal{X}'$)) of $G[\mathcal{X}']$.

The generalized Rarestfirst and the m-DensestAlk algorithms were run for each test instance. For comparison, up to three of the consolidated resulting non-dominated solutions were chosen from the previous experiment of the MOGA-TFP-SN, from each test instance. These three solutions include, an arbitrary solution, the one with the lowest diameter cost, and the one with the highest subgraph density.

Tables 2, 3, and 4, show a comparison of the values of the solutions from each algorithm, evaluated on multiple criteria. The column names identify the criterion function along with a mark ↑ if the criterion is to be maximized, or ↓ if minimized. The MOGA-TFP-SN was able to find at least one solution whose value is equal to or better than that of both RarestFirst and m-DensestAlk algorithms in the 100% of the tests instances for the criteria $D(\mathcal{X}')$, 100% for $Z(\mathcal{X}')$, 20% for $sD(\mathcal{X}')$, 93.3% for Cc-mst($\mathcal{X}'$), 66.6% for Cc-R($\mathcal{X}'$), 93.3% for $\bar{\kappa}(\mathcal{X}')$, 100% for $|\mathcal{X}'|$. It was expected for our method to excel in both $D(\mathcal{X}')$ and $Z(\mathcal{X}')$ objectives. However, our method was able to outperform Rarestfirst algorithm in 10 out of 15 tests, and m-DensestAlk algorithm in 3 out of 15 tests, when compared to their corresponding optimization criteria Cc-R($\mathcal{X}'$) and $sD(\mathcal{X}')$, respectively. Additionally, our method shows either superior or competitive results in Cc-mst($\mathcal{X}'$), $\bar{\kappa}(\mathcal{X}')$, and $|\mathcal{X}'|$ criteria.

Algorithm	Criteria						
	$D(\mathcal{X}')\uparrow$	$Z(\mathcal{X}')\uparrow$	$sD(\mathcal{X}')\uparrow$	Cc-mst($\mathcal{X}'$)↓	Cc-R($\mathcal{X}'$)↓	$\bar{\kappa}(\mathcal{X}')\downarrow$	$\lvert\mathcal{X}'\rvert\downarrow$
Test1: k = 4, T = (AI:1, DM:1, DB:0, T:2)							
RarestFirst	5.0	24.428	5.0	57.0	57.0	**0.0**	**3.0**
DensestAlk	5.75	37.368	**20.125**	n/a	n/a	2.0	8.0
	3.5	**53.333**	5.25	n/a	n/a	2.0	4.0
MOGA-TFP-SN	6.5	48.599	9.75	n/a	n/a	1.0	4.0
	11.333	20.75	11.333	**41.0**	**33.0**	**0.0**	**3.0**
Test2: k = 4, T = (AI:1, DM:1, DB:2, T:0)							
RarestFirst	5.666	33.0	8.5	76.0	58.0	**0.0**	4.0
DensestAlk	14.333	46.857	14.333	30.0	30.0	**0.0**	3.0
MOGA-TFP-SN	21.0	**73.8**	10.5	15.0	15.0	**0.0**	**2.0**
	32.0	56.799	**16.0**	**4.0**	**4.0**	**0.0**	**2.0**
Test3: k = 4, T = (AI:2, DM:0, DB:0, T:2)							
RarestFirst	3.0	20.777	4.5	**90.0**	**90.0**	**0.0**	**4.0**
DensestAlk	2.819	33.382	**19.7333**	n/a	n/a	2.0	15.0
	0.0	**42.555**	0.0	n/a	n/a	3.0	**4.0**
MOGA-TFP-SN	3.0	38.099	4.5	n/a	n/a	2.0	**4.0**
	7.333	31.583	11.0	n/a	n/a	1.0	**4.0**
Test4: k = 4, T = (AI:0, DM:3, DB:1, T:0)							
RarestFirst	4.333	21.428	6.5	84.0	57.0	**0.0**	4.0
DensestAlk	14.333	46.857	14.333	30.0	30.0	**0.0**	**3.0**
	8.0	**71.285**	8.0	48.0	48.0	**0.0**	**3.0**
MOGA-TFP-SN	10.333	64.444	15.5	52.0	48.0	**0.0**	4.0
	19.666	64.285	**19.666**	**19.0**	**19.0**	**0.0**	**3.0**
Test5: k = 4, T = (AI:2, DM:1, DB:1, T:0)							
RarestFirst	4.333	32.25	6.5	85.0	61.0	**0.0**	4.0
DensestAlk	2.819	33.382	**19.733**	n/a	n/a	2.0	15.0
	4.0	**62.222**	6.0	n/a	n/a	1.0	4.0
MOGA-TFP-SN	16.0	42.0	8.0	**20.0**	**20.0**	**0.0**	**2.0**
	16.333	41.625	16.333	24.0	24.0	**0.0**	3.0

Table 2: Comparison results of solutions obtained for test instances of size k = 4 and evaluated over multiple criteria.

4.5 Graph-density objective vs. subgraph-density objective

The subgraph density objective, in the context of TFP-SN was first introduced in [11]. The definition of the subgraph density is given by the following equation

$$sD(\mathcal{X}') = \sum_{e \in E_{G[\mathcal{X}']}} \frac{w(e)}{|\mathcal{X}'|} \quad , \tag{4}$$

which is similar to the graph density definition (Eq. 1). However, note that the subgraph density function decreases linearly with each team member, as opposed to the graph density function which decreases quadratically with the size of the team. To illustrate the difference, suppose there are three teams $\mathcal{X}^1$, $\mathcal{X}^2$, and $\mathcal{X}^3$, as shown in Figure 4. Their corresponding subgraph density values are $sD(\mathcal{X}^1) = \frac{1}{2}$, $sD(\mathcal{X}^2) = \frac{2}{4}$, $sD(\mathcal{X}^3) = \frac{3}{6}$, all the same. In contrast, their corresponding graph density values are $D(\mathcal{X}^1) = \frac{2}{2}$,

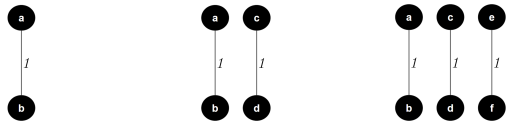
Algorithm	Criteria						
	$D(\mathcal{X}')\uparrow$	$Z(\mathcal{X}')\uparrow$	$sD(\mathcal{X}')\uparrow$	Cc-mst($\mathcal{X}'$)↓	Cc-R($\mathcal{X}'$)↓	$\bar{\kappa}(\mathcal{X}')\downarrow$	$\lvert\mathcal{X}'\rvert\downarrow$
Test6: k = 8, T = (AI:3, DM:2, DB:3, T:0)							
RarestFirst	2.952	24.357	8.857	168.0	63.0	**0.0**	7.0
DensestAlk	2.367	31.421	**18.941**	n/a	n/a	3.0	17.0
	2.799	**54.636**	5.599	n/a	n/a	1.0	**5.0**
MOGA-TFP-SN	5.9	51.454	11.8	n/a	n/a	2.0	**5.0**
	9.1	42.583	18.2	**70.0**	**55.0**	**0.0**	**5.0**
Test7: k = 8, T = (AI:1, DM:2, DB:4, T:1)							
RarestFirst	3.799	27.461	9.5	133.0	91.0	**0.0**	6.0
DensestAlk	9.699	42.0	**19.399**	n/a	n/a	1.0	5.0
	6.4	**59.363**	12.8	**86.0**	**82.0**	**0.0**	5.0
MOGA-TFP-SN	9.833	53.8	14.75	n/a	n/a	1.0	**4.0**
	10.0	37.363	15.0	n/a	n/a	1.0	**4.0**
Test8: k = 8, T = (AI:2, DM:2, DB:2, T:2)							
RarestFirst	3.0	26.076	7.5	138.0	90.0	**0.0**	6.0
DensestAlk	2.819	33.382	**19.733**	n/a	n/a	2.0	15.0
	4.0	**44.6**	10.0	124.0	**82.0**	**0.0**	6.0
MOGA-TFP-SN	5.266	42.187	13.166	**109.0**	87.0	**0.0**	6.0
	7.099	38.928	14.199	n/a	n/a	1.0	**5.0**
Test9: k = 8, T = (AI:2, DM:3, DB:2, T:1)							
RarestFirst	3.799	27.999	7.599	109.0	**61.0**	**0.0**	**5.0**
DensestAlk	2.819	33.382	**19.733**	n/a	n/a	2.0	15.0
	4.599	**55.545**	9.199	**100.0**	82.0	**0.0**	**5.0**
MOGA-TFP-SN	5.9	53.09	11.8	n/a	n/a	2.0	**5.0**
	8.199	47.666	16.399	n/a	n/a	1.0	**5.0**
Test10: k = 8, T = (AI:2, DM:1, DB:5, T:0)							
RarestFirst	2.928	25.611	10.25	189.0	89.0	**0.0**	8.0
DensestAlk	2.819	33.382	19.733	n/a	n/a	2.0	15.0
	6.4	**54.416**	12.8	87.0	68.0	**0.0**	**5.0**
MOGA-TFP-SN	10.8	45.833	21.6	57.0	**33.0**	**0.0**	**5.0**
	12.1	45.25	**24.2**	**48.0**	35.0	**0.0**	**5.0**

Table 3: Comparison results of solutions obtained for test instances of size k = 8 and evaluated over multiple criteria.

Algorithm	Criteria						
	$D(\mathcal{X}')\uparrow$	$Z(\mathcal{X}')\uparrow$	$sD(\mathcal{X}')\uparrow$	Cc-mst($\mathcal{X}'$)↓	Cc-R($\mathcal{X}'$)↓	$\bar{\kappa}(\mathcal{X}')\downarrow$	$\lvert\mathcal{X}'\rvert\downarrow$
Test11: k = 12, T = (AI:4, DM:2, DB:3, T:3)							
RarestFirst	2.311	23.181	10.399	250.0	90.0	**0.0**	10.0
DensestAlk	2.15	30.424	**18.277**	n/a	n/a	3.0	18.0
	2.722	**39.428**	10.888	208.0	**82.0**	**0.0**	**9.0**
MOGA-TFP-SN	3.277	37.318	13.111	n/a	n/a	1.0	**9.0**
	3.688	35.119	16.6	**194.0**	139.0	**0.0**	10.0
Test12: k = 12, T = (AI:3, DM:3, DB:3, T:3)							
RarestFirst	2.5	24.399	10.0	220.0	**90.0**	**0.0**	9.0
DensestAlk	2.367	31.789	**18.941**	n/a	n/a	3.0	17.0
	3.355	**39.5**	15.1	n/a	n/a	1.0	10.0
MOGA-TFP-SN	3.928	36.904	13.75	**163.0**	105.0	**0.0**	8.0
	5.0	29.352	12.5	n/a	n/a	1.0	**6.0**
Test13: k = 12, T = (AI:2, DM:3, DB:5, T:2)							
RarestFirst	2.928	25.611	10.25	189.0	89.0	**0.0**	8.0
DensestAlk	2.819	33.382	**19.733**	n/a	n/a	2.0	15.0
	4.38	**46.47**	13.142	**139.0**	**87.0**	**0.0**	7.0
MOGA-TFP-SN	7.333	39.125	18.333	n/a	n/a	1.0	6.0
	7.699	32.857	15.399	n/a	n/a	1.0	**5.0**
Test14: k = 12, T = (AI:4, DM:0, DB:6, T:2)							
RarestFirst	2.781	21.959	13.909	258.0	**90.0**	**0.0**	11.0
DensestAlk	2.117	29.975	**18.0**	n/a	n/a	2.0	18.0
	2.833	**42.714**	11.333	202.0	117.0	**0.0**	9.0
MOGA-TFP-SN	3.555	39.227	14.222	176.0	117.0	**0.0**	9.0
	4.428	36.4	15.5	**142.0**	103.0	**0.0**	**8.0**
Test15: k = 12, T = (AI:1, DM:5, DB:4, T:2)							
RarestFirst	3.904	27.357	11.714	153.0	91.0	**0.0**	7.0
DensestAlk	5.75	37.368	**20.125**	n/a	n/a	2.0	8.0
	3.099	**55.0**	6.199	**115.0**	**82.0**	**0.0**	**5.0**
MOGA-TFP-SN	4.599	52.928	11.5	119.0	**82.0**	**0.0**	6.0
	8.699	44.384	17.399	n/a	n/a	1.0	**5.0**

Table 4: Comparison results of solutions obtained for test instances of size k = 12 and evaluated over multiple criteria.

$D(\mathcal{X}^2) = \frac{4}{12}$, $sD(\mathcal{X}^3) = \frac{6}{30}$. The two main advantages of the graph density over the subgraph density are described in the following. First, the graph density favors the search over smaller teams. Second, disconnected teams are less favored.



(a) Team $\mathcal{X}^1$. **(b) Team $\mathcal{X}^2$.** **(c) Team $\mathcal{X}^3$.**

Figure 4: Three teams $\mathcal{X}^1$, $\mathcal{X}^2$, and $\mathcal{X}^3$, with unitary weights on the edges.

To support this claim, a small experiment was conducted over test instances 11 and 14, the ones with the largest teams in size ($|\mathcal{X}'|$) from the above experiments. The MOGA-TFP-SN was run ten times, for Test 11 and Test 14, but instead of using graph density ($D(\cdot)$) and expertise level ratio ($Z(\cdot)$) as objective functions, we evolved for subgraph density ($sD(\cdot)$) and expertise level ratio ($Z(\cdot)$). When comparing the output sets of solutions, the ones evolved for graph density presented an average team size of 8.481 and 8.603, while the ones evolved for subgraph density presented an average team size of 13.582 and 12.026, for Tests 11 and 14, respectively. Moreover, the number of connected solutions of the ones evolved for graph density is 446 and 332, while the number of connected solutions of the ones evolved for subgraph density is 305 and 120, for Tests 11 and 14, respectively.

5 CONCLUSION AND FUTURE WORK

In this paper, we presented a multi-objective formulation of the Team Formation Problem in Social Networks that optimize the collaboration and expertise of the team members. We introduced two objective functions, the graph density objective, that models the collaboration among team members, and the expertise level ratio, that represents the ratio of experience of a team, on the required skill set, divided by its size. We solved this problem using the well-known NSGA-II framework for which an individual representation and two variation operators were proposed. The experimental evaluation of our method over different tasks, on a DBLP-derived social network graph, showed a diverse set of competitive solutions from either density or expertise, including other implicit traits such as size and connectivity. An additional experiment suggests that the graph density objective compared to the subgraph density objective, yields to finding smaller teams with a larger number of connected experts. We also generated a set of benchmark instances intended to serve as a reference for future comparison studies.

Future work is aimed at exploring and comparing the performance of alternative cutting-edge evolutionary algorithms (based on decomposition) as well as specialized operators for the problem. A more diverse set of test instances and all criteria described in Subsection 4.4 are planned to be included.

REFERENCES

[1] Frederick P Brooks Jr. 1995. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition, 2/E.* Pearson Education India.

[2] Shi-Jie Chen and Li Lin. 2004. Modeling team member characteristics for the formation of a multifunctional team in concurrent engineering. *IEEE Transactions on Engineering Management* 51, 2 (2004), 111–124.

[3] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. 2007. *Evolutionary algorithms for solving multi-objective problems.* Vol. 5. Springer.

[4] Kalyanmoy Deb. 2001. *Multi-objective optimization using evolutionary algorithms.* Vol. 16. John Wiley & Sons.

[5] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *International Conference on Parallel Problem Solving From Nature.* Springer, 849–858.

[6] Christoph Dorn and Schahram Dustdar. 2010. Composing near-optimal expert teams: a trade-off between skills and connectivity. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems".* Springer, 472–489.

[7] Thomas E Easterfield. 1946. A combinatorial algorithm. *Journal of the London Mathematical Society* 1, 3 (1946), 219–226.

[8] Matthias Ehrgott. 2013. *Multicriteria optimization.* Vol. 491. Springer Science & Business Media.

[9] Agoston E Eiben, James E Smith, et al. 2003. *Introduction to evolutionary computing.* Vol. 53. Springer.

[10] Farnoush Farhadi, Maryam Sorkhi, Sattar Hashemi, and Ali Hamzeh. 2011. An effective expert team formation in social networks based on skill grading. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on.* IEEE, 366–372.

[11] Amita Gajewar and Atish Das Sarma. 2012. Multi-skill collaborative teams based on densest subgraphs. In *Proceedings of the 2012 SIAM International Conference on Data Mining.* SIAM, 165–176.

[12] Yuqiang Han, Yao Wan, Liang Chen, Guandong Xu, and Jian Wu. 2017. Exploiting Geographical Location for Team Formation in Social Coding Sites. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining.* Springer, 499–510.

[13] Kareem Kamel, Noor Tubaiz, Osama AlKoky, and Zaher AlAghbari. 2011. Toward forming an effective team using social network. In *Innovations in Information Technology (IIT), 2011 International Conference on.* IEEE, 308–312.

[14] Mehdi Kargar and Aijun An. 2011. Discovering top-k teams of experts with/without a leader in social networks. In *Proceedings of the 20th ACM international conference on Information and knowledge management.* ACM, 985–994.

[15] Mehdi Kargar, Aijun An, and Morteza Zihayat. 2012. Efficient bi-objective team formation in social networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Springer, 483–498.

[16] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics (NRL)* 2, 1-2 (1955), 83–97.

[17] 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.* ACM, 467–476.

[18] Cheng-Te Li and Man-Kwan Shan. 2010. Team formation for generalized tasks in expertise social networks. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on.* IEEE, 9–16.

[19] M Niveditha, G Swetha, U Poornima, and Radha Senthilkumar. 2017. A genetic approach for tri-objective optimization in team formation. In *Advanced Computing (ICoAC), 2016 Eighth International Conference on.* IEEE, 123–130.

[20] Syama Sundar Rangapuram, Thomas Bühler, and Matthias Hein. 2013. Towards realistic team formation in social networks based on densest subgraphs. In *Proceedings of the 22nd international conference on World Wide Web.* ACM, 1077–1088.

[21] Xinyu Wang, Zhou Zhao, and Wilfred Ng. 2015. A Comparative Study of Team Formation in Social Networks.. In *DASFAA (1).* 389–404.

[22] Xinyu Wang, Zhou Zhao, and Wilfred Ng. 2016. Ustf: A unified system of team formation. *IEEE Transactions on Big Data* 2, 1 (2016), 70–84.

[23] Yan Yang and Haiyue Hu. 2013. Team formation with time limit in social networks. In *Mechatronic Sciences, Electric Engineering and Computer (MEC), Proceedings 2013 International Conference on.* IEEE, 1590–1594.

[24] Hongzhi Yin, Bin Cui, and Yuxin Huang. 2011. Finding a wise group of experts in social networks. In *International Conference on Advanced Data Mining and Applications.* Springer, 381–394.

[25] Armen Zakarian and Andrew Kusiak. 1999. Forming teams: an analytical approach. *IIE transactions* 31, 1 (1999), 85–97.