Grouped Team Formation in Social Networks

Ze Lv[1,2], Jianbin Huang[1(✉)], Yu Zhou[1,2], Heli Sun[3], and Xiaolin Jia[3]

[1] School of Software, Xidian University, Xi'an, China
jbhuang@xidian.edu.cn
[2] School of Computer Science and Technology, Xidian University, Xi'an, China
[3] Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China

Abstract. Given an expert collaboration social network and a task, the team formation problem in social networks aims at forming a team which satisfies the skill requirements of the task with efficient collaboration. Different communication cost functions have been proposed in the existing work, but the grouped organization structure inside the team is not considered. With a novel communication cost function as objective function, we define the Grouped Team Formation problem. We propose an exact algorithm for solving the problem, and evaluate it by experiments.

1 Introduction

Lappas et al. [4] first bring the social network to the team formation problem. In the existing work, all the communication cost functions consider the cost among all the experts in the team, ignoring the internal grouped organization structure. In this paper, we propose the Grouped Team Formation problem with a novel communication cost function which considers the intra-costs among the group members in the same group and the inter-costs among the leaders. A group member only needs to communicate with his/her group leader and the other group members in the same group. A group leader only needs to communicate with his/her group members and the other leaders. To solve the problem, an exact algorithm is proposed. This algorithm is based on a method of solving a subproblem which can be constructed into an assignment problem, and a pruning method is proposed to filter the subproblems.

2 Problem Definition

Let $A = \{a_1 ... a_m\}$ be a set of m skills. Let $G = (V, E)$ to be an expert collaboration social network. Each expert node $v \in V$ owns a set of skills $\chi_v \subseteq A$; each edge $(u, v) \in E$ represents the previous collaboration, and the weight on the edge represents the communication cost between the two experts. We assume $P = \{g_i = (s_i, k_i) \,|1 \leq i \leq q\}$ to be a project. g_i represents a group; $s_i \subseteq A$ represents the skillset required by the group; k_i is an integer, representing the number of experts that g_i requires; q is the number of groups. We assume l_P

F. Li et al. (Eds.): APWeb 2016, Part II, LNCS 9932, pp. 398–401, 2016.
DOI: 10.1007/978-3-319-45817-5_33

to be a project leader of project P, also the team leader. A grouped team is a set of q expert subsets, denoted by $T = \{V(g_1) ... V(g_q)\}$. Each expert subset $V(g_i) \subseteq V$ represents a set of experts including a group leader $l_{g_i} \in V(g_i)$ in group g_i. For two expert nodes u and v, we define the communication cost between them as the shortest path distance of them, denoted by $d(u, v)$. We compute the shortest path distances between each pair of experts on the offline. The Grouped Leader Distance function is formulated as

$$C(T) = \sum\nolimits_{g_i \in P} \sum\nolimits_{v \in V(g_i)} d(v, l_{g_i}) + \sum\nolimits_{g_i \in P} d(l_{g_i}, l_P) \ . \tag{1}$$

The total communication cost consists of two parts. To measure the communication costs among the group members in same group and among the leaders, we make reference to the Leader Distance proposed in [3].

The Grouped Team Formation problem can be formulated as

$$\begin{aligned} & \min_{T} C(T) \\ \text{s.t.}:\ & \forall v \in V, |\{g_i | g_i \in P \ and \ v \in V(g_i)\}| \leq 1 \\ & \forall 1 \leq i \leq q, |\{v | v \in V(g_i) \ and \ s_i \subseteq \chi_v\}| \geq k_i \ . \end{aligned} \tag{2}$$

The first constraint is the expert assignment constraint. It ensures that each expert can be assigned to at most one group. The second constraint is the group cardinality constraint with the skill cover constraint which requires that each group has a specific number of experts with corresponding skills.

3 Assignment and Pruning Algorithm (AP)

In this section, we propose an exact algorithm called AP with its framework shown in Fig. 1.

Consider the subproblem: when given the group leaders for each group, how to find its optimal team with the minimum communication cost. In fact, this subproblem can be constructed into an assignment problem from the candidate group members to the groups. We form a matrix where rows represent candidate group members and columns represent groups as shown in Fig. 1(c). For each group, we add at most $|T| - 1$ candidate group members closest to the group leader, while the others will never be chosen in the optimal assignment. For each group g_i, we accordingly add $k_i - 1$ columns representing the groups. With the matrix as input, we can find the optimal assignment for the given group leaders by calling the auction algorithm [1].

We can find the optimal solution if we traverse every possible combination of the group leaders and get the optimal assignment. Next we construct the pruning lists $L = \{L_1 ... L_q\}$ to filter the combinations of group leaders as shown in Fig. 1(a). A list L_i corresponds to a group g_i. Given a group g_i and a candidate expert $v \in S(g_i)$, the group leader expectation cost $EC(v, g_i)$ is the minimum communication cost caused by v when v is the group leader l_{g_i}, i.e., $EC(v, g_i) = d(v, l_P) + \min_{V(g_i)} \sum_{u \in V(g_i)} d(v, u)$. Each tuple $t = (v, EC)$ consists of an expert

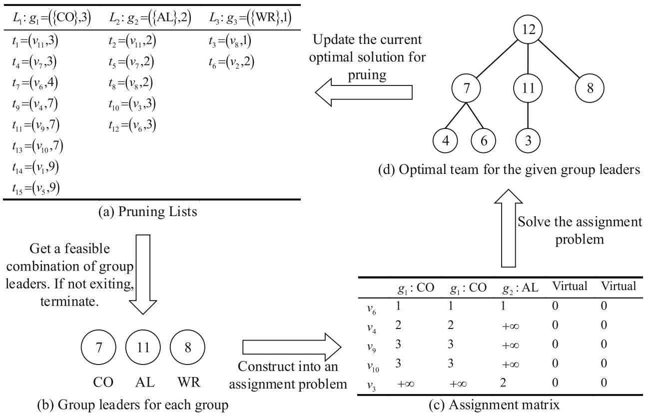


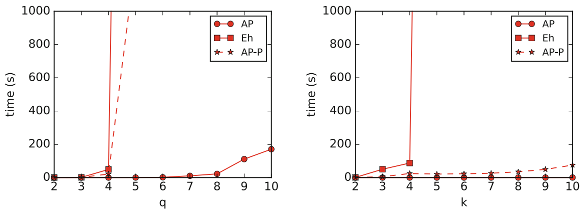
Fig. 1. Illustration of the procedure of the AP algorithm.

node v and the value of the group leader expectation cost $EC(v, g_i)$. The tuples in the same list are sorted in ascending order of E. We can guess that the optimal combination of group leaders should be in the upper layers of L. We adopt a pruning strategy which is similar to the upper bound algorithm in [2] to traverse the pruning lists layer by layer and filter the combinations of the group leaders. A set of p tuples $\{t_1...t_p\}$ is called a feasible combination, denoted by c, if each tuple is from a distinct list in L and has a distinct expert. A feasible combination c is called a full combination if $p = q$. The d-th layer is called depth d of L. Let $x_1^d...x_q^d$ be the values in the tuples each from $L_1...L_q$ at depth d of L. If there is no tuple in L_i at depth d, then $x_i^d = +\infty$. Suppose that here is a feasible combination c, and $L_{i_1}...L_{i_j}$ are the lists from which we have not selected any tuple for c. Then the lower bound communication cost of c at depth d is $lower(c)^d = \sum_{t=(v,EC)\in c} EC + {x_{i_1}}^d + ... + {x_{i_j}}^d$. At depth d, we can prune c if $lower(c)^d \geq C_{min}$, where $C_{\min}$ is the communication cost of the current optimal solution.

Initially, there is an empty combination. We traverse the tuples in L layer by layer and check if the tuple can be combined with any existing combination to become a new feasible combination. On each layer, for a new formed feasible combination, we can compute a lower bound for it, and check if it can be pruned. And for a full combination, we can compute its optimal assignment and update the current optimal solution. When no combinations are left, the algorithm terminates.

4 Experimrntal Evaluations

There are 100 nodes and 1,000 random linked edges in the dataset. Each node owns 5 skills that randomly selected from a total of 50 skills. We evaluate the efficiency of the AP algorithm by comparing with the exhaustive method (Eh)



(a) $k = 2$ (b) $q = 3$

Fig. 2. The running time of the AP algorithm (AP), the exhaustive method (Eh) and the AP algorithm without pruning (AP-P).

and the AP algorithm without pruning (AP-P). Though the AP algorithm is far more efficient than the exhaustive method, it becomes time consuming when the input scale rises. So the comparison with the exhaustive method is on the small scale dataset. We can see that the exhaustive method and the AP algorithm without pruning work badly even on a small scale dataset. And the effect of the pruning method in the AP algorithm is very clear (Fig. 2).

5 Conclusions

Considering the grouped organization inside team, we define the Grouped Team Formation problem. The basic idea of our exact algorithm is as follows. When given the group leaders, we can transform it into a classical assignment problem. A pruning strategy is proposed to improve the efficiency of selecting the group leaders. Experiments verify the efficiency of our algorithm.

Acknowledgements. This work was supported by Natural Science Foundation of China (61474299, 61540008).

References

1. Bertsekas, D.P.: The auction algorithm: a distributed relaxation method for the assignment problem. Ann. Oper. Res. **14**(1), 105–123 (1988)
2. Feng, K., Cong, G., Bhowmick, S.S., Ma, S.: In search of influential event organizers in online social networks. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp. 63–74. ACM (2014)
3. Kargar, M., An, A.: 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, pp. 985–994. ACM (2011)
4. Lappas, T., Liu, K., Terzi, E.: Finding a team of experts in social networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 467–476. ACM (2009)