A Practical Negotiation-Based Team Formation Model for Non-Cooperative Social Networks

Wanyuan Wang, Yichuan Jiang*, *IEEE Senior Member*
Distributed Intelligence and Social Computing Laboratory,
School of Computer Science and Engineering, Southeast University, Nanjing 211189, China.
**Corresponding author, email: yjiang@seu.edu.cn*

***Abstract*—Team formation is an effective collaboration manner in social networks (SNs). Within teams, social individuals can work together to accomplish complex jobs that they are unable to perform individually. Due to its wide range of applications, team formation in SNs has been studied extensively and a number of approaches have been proposed. However, all of these proposals either build teams of individuals to accomplish jobs through a centralized manner or ignore the selfish nature of social individuals, or both. In this paper, we introduce a decentralized negotiation-based team formation model for non-cooperative SNs, where social individuals are self-interested. The proposed team formation model works by allowing the employer (i.e., *job initiator*) to recruit a team of professional employees that demand small working remuneration and incur little communication overhead and allowing the employees to join the beneficial teams from which they can achieve a high financial remuneration. The simulation results show that our model achieves about 80% social welfare of the ideal centralized models on average. Moreover, compared to other conventional distributed models, our model can reduce team formation time significantly, making our model a better choice for the real-world time-sensitive applications.**

***Index Terms*—team formation; social networks; multiagent systems; negotiation**

I. Introduction

As the increasing development of Internet technology, many online social networks (SNs) such as Facebook (www.facebook.com) and LinkedIn (www.linkedin.com) have become a flexible platform for supporting some business activities such as crowdsourcing [1]. In this paper, we study a special crowdsourcing paradigm, where enterprises (organizations or individuals) post their jobs (e.g., software product development, website maintenance and advertisement design) through these online social network platforms and wish to recruit a set of social individuals to work as a team for performing their jobs. This kind of business activity can also be called team formation in social networks [2-7].

As an example of this social team formation problem, consider an IT manager that wants to develop a software product. Due to the limited energy and skills to accomplish this job individually, he needs to recruit a team of software engineers to perform this job. To complete this job successfully, these recruited employees should be professional enough such that they satisfy all of the skill requirements of this job. Moreover, because of the limited budget on hiring workers, the manager wishes to hire the engineers *i*) that demand little working cost for their services (the recruited engineers need to be paid for their work) and *ii*) that incur little communication cost (engineers might be located in different cities, which will incur certain transportation fees when face to face meetings are necessary to improve this job's quality).

Online social networks provide good opportunities for this manager to build a team of experts, because he is connected with a large number of social individuals and can learn about their information (e.g., working cost and location) through online interaction easily. However, social networks are highly dynamic and heterogeneous (e.g., individuals own diverse skills with varying working costs and live in different cities with varying communication costs), it also present challenges for the job managers to decide which individuals to recruit and which skills of which individuals to use such that the minimum budget spent and for the individuals to decide which job to perform such that the maximum financial remuneration achieved.

Due to its wide range of applications, this social team formation problem recently has attracted great interest and a number of proposals have been presented [2-7] (see Section 2 for more details). For all of these researches, however, there are a couple of issues that we feel impractical. First, they all assume that there exists an omnipotent controller that maintains information on all of the social individuals in SNs and can directly communicate with them to form teams in a centralized manner. However, most existing online social networks do not involve such a controller and in reality, both the managers recruit employees for their jobs and the employees join teams to perform jobs are in a self-organized manner [8-10]. Second, they all hold the assumption that social individuals are cooperative, i.e., each individual is willing to join the specific team enforced by the controller to satisfy the global optimization objective. While in practice, individuals are always self-interested and their only incentive to join a team is to maximize their own benefits [11][12].

To address the above two issues, in this paper, we first model each individual in a SN as a rational and autonomous agent and then propose a distributed multiagent-based negotiation model for the agents to build or join teams. The proposed negotiation model extends the contract net protocol [8] by allowing the manager and contractor agents to negotiate with each other on skill contribution autonomously. This negotiation mechanism mainly consists of the following three phases: *i) offer*: the

manager interacts with a contractor and sends an offer to the contractor on skill contribution; *ii) Response*: the contractor responds to the manager (i.e., accept or reject this offer) with the aim of optimizing its own financial remuneration and *iii) Confirm*: the manager confirm a final agreement on skill contribution with the contractor. It should be noted that the main contribution of this paper is the decision-making strategy devised for the agents to build or join teams, rather than this negotiation mechanism used only for realizing team formation.

Although the complex bilateral bargaining-based negotiation mechanism [9][10] might also be available, they are inefficient for this social team formation problem. First, a key assumption in [9][10] is that agents have incomplete information on other agents' working costs and their mechanism can address the incompleteness effectively by allowing agents to negotiate with one another round and round until they reach an agreement on working cost. This mechanism, however, turns out to be redundant and time consuming for social team formation problem, where individual's working cost is public and the manager can learn this information easily by browsing the individual's online profile [1]. Second, social team formation should also consider the synergy satisfaction among team members, that is, the team members should form a connected subgraph of the constraining networks [2][4] and due to the arbitrary negotiation among agents in [9][10], they do not satisfy this objective. Instead, in this paper, based on the reasonable assumption that agents have complete information of others and by being aware of the synergy effect, we propose a practical social team formation model, which is proved to be more efficient than the traditional models [9][10] in reducing team formation time.

We also conduct a set of experiments to evaluate the effectiveness of our model. The experimental results show that compared to the related studies with centralized controller [2-7], our model achieves 80% social welfare of them on average and compared to the related studies with the complex bilateral bargaining protocols [9][10], our model reduces the team formation time significantly while maintaining a desirable performance on team utility, making our model a better option for real-world time-sensitive applications.

II. RELATED WORK

Team Formation in Social Networks. The problem of team formation in social networks was first studied by Lappas et al. in [2]. They model each social network as a weighted and undirected graph G=$<V, E, W>$, where vertices V represent individuals, edges E represent social connections among individuals and the weights W on the edges represent the communication cost among connected individuals. Given a social network graph G and a job J, Lappas et al. [2] target to find a team of individuals V^*∈V such that V^* not only meet all of the skill requirements of J but also incur the least team communication cost. Following [2] are several social team formation variants with different objectives, such as synergy maximization [3][4], budget minimization [5] and load balancing [6][7]. Noting that all of these variants are NP-hard, the previous researches [2-7] focus on developing centralized approximation with rigorous quality guarantees. This centralization, however, is impractical for real-world applications because many online social networks do not involve such an omnipotent central authority and social individuals are self-organized to build or join teams.

Team Formation in Networked Multiagent Systems. In networked multiagent systems, agents need interacting with their neighbors to form teams for accomplishing tasks. Noticing that the underlying network topology has a dramatic effect on the quality of the formed teams, Gaston and desJardins [13] and Kota et al. [14] develop a dynamic structural adaption method to improve system performance, where agents can adjust network structure by deleting their costly connections and rewiring to these agents with better connections. Following [13], Glinton et al. [15] investigate the network topologies that result from the structural adaption method and they find that scale-free networks are consistently formed. All of these proposals [13-15] struggle to optimize system performance by adapting network structure, while we are interested in finding a team of individuals to perform a specific job from a static network. An et al. [9] and Ye et al. [10] have made great effort on distributed team formation in static networks, where agents are assumed to have incomplete working cost information of others. Their proposed bilateral bargaining-based negotiation mechanism is efficient in addressing the incompleteness by allowing the agents to negotiate with each other round and round until they reach an agreement on working cost. However, this mechanism turns out to be redundant and inefficient for the social team formation problem, where individuals are willing to let others know their working cost and from an individual's perspective, letting others know his true information may help him get a desirable occupation [1]. Moreover, their mechanism might form a unconnected team, dissatisfying the synergy objective in social team formation issue [4].

III. PROBLEM DESCRIPTION

In this section, we provide notations necessary for understanding the problem of team formation in social networks.

Social Networks. Each social network *SN*=$<A, E, D>$ is an undirected graph, where A=$\{a_1, a_2, ..., a_m\}$ is the set of agents (hereafter, we use the terms "agent" and "individual" interchangeably). $\forall(a_i, a_j)$∈E indicates the existence of a direct social interaction between a_i and a_j. Each pair of agents a_i and a_j is associated with a value $d(a_i, a_j)$∈D indicates the cost (e.g., transportation fee) incurred by the necessary communication between a_i and a_j and $d(a_i, a_j)$=$d(a_j, a_i)$. Moreover, we assume that there are l type skills S= $\{s_1, s_2, ..., s_l\}$ available in a *SN*.

Agents. Each agent a_i∈A is defined by a 3-tuple $<$ $F(a_i), Wc(a_i), Nei(a_i)>$, where $F(a_i)$∈S indicates the set of skills that agent a_i owns and if s_j∈$F(a_i)$, a_i has the skill s_j. $Wc(a_i)$=$\{Wc(a_i, s_1), ..., Wc(a_i, s_{|F(a_i)|})\}$ indicates a_i's working cost of providing each owned skill s_j∈$F(a_i)$. And $Nei(a_i)$ indicates the social neighbors of a_i, i.e., $Nei(a_i)$=$\{a_j|(a_i, a_j)$∈$E\}$. Furthermore, we postulate that

each agent cannot join more than one team at a time, while as a member of a team it can contribute more than one skill to this team job. This assumption is reasonable since each individual has limited energy and participating in multiple teams will degrade the performance of each team that it joins [13].

Jobs. We consider a set of jobs K=$\{\kappa_1, \kappa_2, ..., \kappa_n\}$ initiated by these agents A independently, in this paper, at each time step, jobs are initiated by agents independently with a specific probability. Then, each job $\kappa \in K$ can be defined by a tuple $< Ia_\kappa, R_\kappa, It_\kappa, Dl_\kappa, Wt_\kappa, v_\kappa(t) >$, where Ia_κ: $\kappa \rightarrow A$ indicates the initiator agent at which job κ is initiated. R_κ is the set of skills required by job κ and if $s_j \in R_\kappa$, job κ requires the skill s_j. It_κ is the initialization time of κ, i.e., at time It_κ, agent Ia_κ posts κ. Dl_κ is the deadline (the latest execution start time) of κ. Wt_κ indicates the working time it will take to finish κ after execution starts (note that job κ must finish before Dl_κ+Wt_κ). Finally, $v_\kappa(t)$ represents the value associated with the job κ, which is a function of finish time *t*. Here, referring to the related definition in [9], we define the value function as:

$$v_\kappa(t) = \begin{cases} v_\kappa(\frac{Dl_\kappa + Wt_\kappa - t}{Dl_\kappa - It_\kappa})^\delta & t \leq Dl_\kappa + Wt_\kappa; \\ 0 & t > Dl_\kappa + Wt_\kappa. \end{cases} \quad (1)$$

v_κ is the predetermined value of κ, set by its initiator Ia_κ at job κ's initialization time It_κ; δ ($0 \leq \delta \leq 1$) is the parameter modeling how job value decreases with the increase of finish time *t*. If job κ starts execution before the deadline Dl_κ, $v_\kappa(\cdot)$ has the maximum value at time It_κ+Wt_κ and the minimum value at time Dl_κ+Wt_κ.

Teams. Each team T_κ is responsible for a job κ, which is denoted by a 3-tuple $<\Omega_\kappa, Cont(\Omega_\kappa, \kappa), us_\kappa>$, where Ω_κ is the set of agents that have joined the team T_κ. $Cont(\Omega_\kappa, \kappa)$=$\{(a_i, s_j), ..., (a_p, s_q)\}$ is the skill contribution function that associates with each teammate $a_i \in \Omega_\kappa$ a required skill $s_j \in R_\kappa$ and $\forall (a_i, s_j) \in Cont(\Omega_\kappa, \kappa)$ indicates teammate a_i contributes skill s_j to job κ. us_κ represents the skills of job κ that have not been satisfied by the current teammates Ω_κ, i.e., $us_\kappa = R_\kappa \backslash \{s_j | (a_i, s_j) \in Cont(\Omega_\kappa, \kappa)\}$. A team T_κ is called a ***complete*** fulfilled team if and only if each skill of job κ is satisfied by at least one teammate $a_i \in \Omega_\kappa$. As soon as a complete team T_κ is formed for job κ, the execution for κ starts. Otherwise, i.e., $us_\kappa \neq \emptyset$, team T_κ is a ***partial*** fulfilled team.

IV. TEAM FORMATION IN SOCIAL NETWORKS

Prior to describing the social team formation model, we first provide the definition of three roles of agents used throughout the team formation process, i.e. *Manager*, *Contractor* and *Freelancer*.

Definition 1. Manager, Contractor and Freelancer. *Given a team T_κ of job κ, the agent who initiates κ is called the* Manager, *the agent who has joined the team T_κ is called the* Contractor *and other agents are* Freelancers *that the team* Manager *can negotiate with.*

The team formation model, employed by the manager Ia_κ to build a set of freelancers to work as a team for job κ is presented in Algorithm 1.

Algorithm 1. Team Formation Model (Ia_κ,κ)
1. Initialize Ω_κ=Ia_κ, $Cont(\Omega_\kappa, \kappa)$=$F(Ia_\kappa)$, R_κ=$R_\kappa \backslash F(Ia_\kappa)$.
2. Initialize T_κ=$< \Omega_\kappa, Cont(\Omega_\kappa, \kappa), R_\kappa >$.
3. **While** ($It_\kappa \leq t \leq Dl_\kappa$) *% t is the real time %*
4. **For** each contractor $a_j \in \Omega_\kappa$
5. **For** each freelancer $a_i \in Nei(a_j)$
6. Negotiate(Ia_κ,a_i).
7. **If** $\forall s_y \in R_\kappa$, $\exists a_x \in \Omega_\kappa$: $(a_x, s_y) \in Cont(\Omega_\kappa, \kappa)$
8. Finish team formation and start job execution.
9. **End For**
10. **End For**
11. **End While**

In Step 1, as being the initiator of job κ, agent Ia_κ is postulated to contribute all of its owned skills to κ. Meanwhile, it is also necessary for Ia_κ to create a team T_κ to manage the team contractors and recruit new freelancers (Step 2). Before the deadline of job κ (Step 3), each team contractor $a_j \in \Omega_\kappa$ is responsible for introducing its neighbor freelancers $a_i \in Nei(a_j)$ to the team manager Ia_κ through the social interaction (a_j,a_i) (Steps 4∼5). By this introduction, the team manager Ia_κ is responsible for negotiating with the freelancer a_i for skill contribution (Step 6). The negotiation mechanism adopted to make a contract between Ia_κ and a_i about which skills to contribute will be discussed in Section IV-A. Finally, after adequate negotiations, if Ia_κ has recruited sufficient contractors such that they can satisfy all of the skill requirements of κ, Ia_κ will terminate the team formation process and start to execute job κ (Steps 7∼8).

It is worth noting that this social team formation model captures a couple of desirable properties that are coincide with the nature of real-world team formation.

Property 1. *Team manager prefers to recruit the freelancers that have many neighbor agents.*

In reality, popular well-connected individuals have large social influence and thus they can access the relevant resources easily. Certainly, recruiting these influential social individuals will expedite team formation [16].

Property 2. *Given an freelancer a_i, the more neighbors of a_i join the team T_κ, the larger probability a_i will join T_κ.*

In reality, because of the social affinity among close friends [17], an individual prefers to join a community if many of his friends have been members of that community.

Property 3. *Each team induces a connected subgraph of the social networks.*

This property satisfies the synergy objective of social team formation problem [4], requiring that for any contractor $a_x \in \Omega_\kappa$ in team T_κ, there must exist at least one social neighbor $a_y \in Nei(a_x)$ being a member of T_κ.

A. The Negotiation Mechanism

The negotiation mechanism, employed in the team formation model, extends the traditional contract net protocol [8] by allowing a team manager, say Ia_κ, and a freelancer, say a_i, to make a contract on skill contribution. This mechanism mainly consists of three basic phases: *i) offer*: Ia_κ sends an offer to a_i on skill contribution; *ii) Response*: a_i responds to Ia_κ (i.e.,

accept or reject this offer) with the aim of optimizing its own remuneration and *iii) Confirm*: Ia_κ confirms a final contract on skill contribution with a_i. In the following, we will describe the three phases in detail.

[Phase 1] ***Offer*[*O*]**. When Ia_κ begins to negotiate with a_i, Ia_κ first initiatively sends a_i an offer on which skills should a_i contribute. The manager Ia_κ's only incentive to recruit the freelancer a_i for using a_i's skills is to maximize its own expected profit. Thus, it is necessary to discuss the definition of the team manager's expected profit before presenting the offer strategy. Due to the dynamics and uncertainty in social networks, there are many factors correlated with the manager's expected profit, such as,

- The expected value (*Ev*) attached to job κ at time τ, which is given by the average value of finishing the job κ within the earliest completion time (τ+Wt_κ) and the latest completion time (Dl_κ+Wt_κ), i.e.,
$$Ev(\kappa,\tau)=\frac{\int_{\tau+Wt_\kappa}^{Dl_\kappa+Wt_\kappa} v_\kappa(t)dt}{Dl_\kappa-\tau} \quad (2)$$
As time τ approaches closer to the deadline Dl_κ, job κ produces the less expected value to Ia_κ.
- The success rate (*Sr*) of forming a complete team, which is given by the fraction of job κ's skills that have been satisfied by team contractors Ω_κ, i.e., $Sr(T_\kappa)=1-|us_\kappa|/|R_\kappa|$ ($|X|$ denotes the number of elements in the set *X*). The larger the value of $Sr(T_\kappa)$, the higher possibility a complete team will be formed.
- Team contractors' total communication cost (*TCC*) incurred by the necessary communication between team manager and contractors, which is given by $TCC(T_\kappa)=\sum_{a_j\in\Omega_\kappa} d(Ia_\kappa,a_j)$.
- Team contractors' total working cost (*TWC*) for performing job κ, which is given by $TWC(T_\kappa)=\sum_{(a_x,s_y)\in Cont(\Omega_\kappa,\kappa)} Wc(a_x,s_y)$.

It is beneficial for the manager of job κ, Ia_κ, to recruit the freelancer a_i if a_i demands little remuneration, incurs little communication cost and its joining increases the success rate for Ia_κ to form a complete team.

Definition 2. Expected profit of team manager. *For a partial fulfilled team* T_κ=$<\Omega_\kappa, Cont(\Omega_\kappa,\kappa), us_\kappa>$ *at time step* τ*, the team manager* Ia_κ*'s expected profit is:*

$$Ep(T_\kappa,\tau)=Sr(T_\kappa)\cdot Ev(\kappa,\tau)–\lambda TCC(T_\kappa)–TWC(T_\kappa) \quad (3)$$

where Sr, Ev, TCC and TWC are the terms defined above and the parameter λ *is the tradeoff factor that determines the importance of TCC and TWC.*

Given the freelancer a_i that the team manager Ia_κ is negotiating with, Ia_κ should take all of a_i's available skills $as(a_i,R_\kappa)$=$F(a_i)\cap R_\kappa$ into consideration to identify a_i's optimal skill contribution that produces the maximal expected profit for itself. Note that even if there are certain skill $s_y\in as(a_i,R_\kappa)$ that has been satisfied by one of the team contractors, i.e., $\exists a_x\in\Omega_\kappa$: $(a_x,s_y)\in Cont(\Omega_\kappa,\kappa)$, skill s_y still needs to be considered because the new freelancer a_i might incur less communication cost and demand less working cost for providing s_y. However, to compute a_i's optimal skill contribution, there are $O(2^{|as(a_i,R_\kappa)|})$ possible skill combinations that the manager Ia_κ needs to evaluate. Conceptually, Ia_κ could exclude some skill combinations. However, excluding even a single skill combination may cause Ia_κ's expected profit to be arbitrarily far from the optimum because the excluded skill combination may produce the largest expected profit value. To illustrate this situation, consider a simple social team formation example.

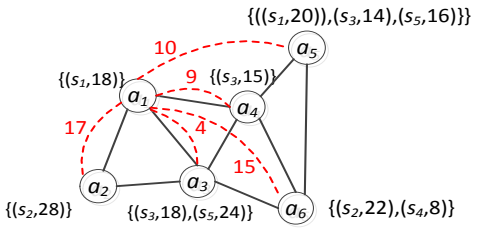


Fig. 1: A simple social network. Each agent is associated with the skills it can provide and the costs of providing these skills. Solid lines indicate the social connections. Each red dotted line is attached a value indicating the communication cost between a_1 and any other agent.

Example 1. Fig. 1 is a social network constituting six agents $\{a_i|1\leq i\leq 6\}$. Now suppose that a job κ is initiated by agent a_1 and its skill requirements are R_κ=$\{s_i|1\leq i\leq 5\}$. To complete κ successfully, agent a_1 first needs to negotiate with his direct neighbors $\{a_i|2\leq i\leq 5\}$ for the unsatisfied skills $\{s_i|2\leq i\leq 5\}$. Assume that after the negotiation with agents a_2, a_3 and a_4, the team for job κ becomes T_κ=<$\{a_1,a_2,a_3\}$,$\{(a_1,s_1),(a_2,s_2),(a_3,s_3),(a_3,s_5)\}$,$\{s_4\}$>, i.e., a_2 contributes skill s_2 to κ and a_3 contributes skills s_3 and s_5. It is easy to achieve that team T_κ's working cost $TWC(T_\kappa)$=$Wc(a_1,s_1)$+$Wc(a_2,s_2)$+$Wc(a_3,s_3)$+$Wc(a_3,s_5)$ =88 and communication cost $TCC(T_\kappa)$=$d(a_1,a_2)$ +$d(a_1,a_3)$=21. Obviously, T_κ is a partial fulfilled team and then a_1 proceeds to negotiate with a_5. In the case that a_5 only contributes s_3 to κ, the team becomes $T_\kappa^{'}$ =<$\{a_1,a_2,a_3,a_5\}$,$\{(a_1,s_1),(a_2,s_2),(a_3,s_5),(a_5,s_3)\}$,$\{s_4\}$> with $TWC(T_\kappa^{'})$=84 and $TCC(T_\kappa^{'})$=31. Team $T_\kappa^{'}$ is less beneficial than T_κ because $TWC(T_\kappa^{'})$+$TCC(T_\kappa^{'})$=115>109=$TWC(T_\kappa)$+$TWC(T_\kappa)$. It has the similar result if a_5 only contributes skill s_5. However, if a_5 contributes skills s_3 and s_5 to job κ simultaneously, the team will become T_κ^*=<$\{a_1,a_3,a_5\}$,$\{(a_1,s_1),(a_2,s_2),(a_5,s_3),(a_5,s_5)\}$,$\{s_4\}$> with $TWC(T_\kappa^*)$=76 and $TCC(T_\kappa^*)$=27. Team T_κ^* is more beneficial than T_κ because $TWC(T_\kappa^*)$+$TCC(T_\kappa^*)$=103>109=$TWC(T_\kappa)$+$TWC(T_\kappa)$. ■

Therefore, given any freelancer that a team manager is negotiating with, the manager faces the exponential computations to find the most beneficial skill contribution. To tackle this computationally complex problem, we propose an efficient greedy polynomial algorithm by first sorting a_i's available skills $as(a_i,R_\kappa)$ in increasing order of working cost and then evaluating these skills in the order of their ranking. A formal description of the offer strategy implemented by the manager Ia_κ to compute the appropriate skill contribution of the freelancer a_i is shown in Algorithm 2.

Algorithm 2. Offer($Ia_\kappa, T_\kappa, a_i, ns, \tau$)
/* ns: the set of skills that freelancer a_i should provide;
τ: the current time. */
1. Initialize cur_ep=$Ep(T_\kappa, \tau)$, max=-∞, ns=$\emptyset$.
2. Rank $s_j \in as(a_i, R_\kappa)$ such that $Wc(a_i, s_1) \leq \ldots \leq Wc(a_i, s_l)$.
3. **For** $1 \leq j \leq l$
4. **If** $Ep(T_\kappa \oplus \{ns \cup s_j\}, \tau) \geq max$
5. ns=$ns \cup s_j$, max=$Ep(T_\kappa \oplus ns, \tau)$.
6. **End If**
7. **End for**
8. **If** $ns \neq \emptyset$ && $(max - d(Ia_\kappa, a_i)) > cur\_ep$
9. Send the offer O=<Ia_κ, T_κ, ns> to a_i.

In Step 1, before negotiating with a_i, the manager Ia_κ first initializes its state: the value cur_ep records the current expected profit of the current team T_κ and ns stores the skills that freelancer a_i should contribute. In Step 2, Ia_κ sorts a_i's available skills $as(a_i, R_\kappa)$ in increasing order of working cost and then evaluates these skills in the order of their ranking (Steps 3∼7). In step 4, we use $Ep(T_\kappa \oplus \{ns \cup s_j\}, \tau)$ to represent Ia_κ's updated expected profit after a_i contributes skills $\{ns \cup s_j\}$. $Ep(T_\kappa \oplus \{ns \cup s_j\}, \tau)$ is computed as follows: first Ia_κ add a_i's each contributed skill $s_y \in \{ns \cup s_j\}$ to the current skill contribution function $Cont(\Omega_\kappa, \kappa)$ and if there exists a team contractor $a_x \in \Omega_\kappa$, that has agreed to contribute s_y, remove this skill contribution (a_x, s_y) from $Cont(\Omega_\kappa, \kappa)$ and remove this team contractor a_x from Ω_κ if removing a_x's skill contribution (a_x, s_y) leads to a_x does not make any skill contribution to this team; and then Ia_κ computes the updated expected profit $Ep(T_\kappa \oplus \{ns \cup s_j\}, \tau)$ of the updated team configuration $T_\kappa \oplus \{ns \cup s_j\}$. If the updated team $T_\kappa \oplus \{ns \cup s_j\}$ produces a larger expected profit than the previous team $T_\kappa \oplus ns$ with the expected profit max, add skill s_j to the required skill set ns (Step 5). Finally, if Ia_κ finds it is beneficial to recruit a_i by using its skills ns ($ns \neq \emptyset$) as well as considering a_i's communication cost (i.e., $max - d(Ia_\kappa, a_i) > cur\_ep$), Ia_κ will send an offer O=<Ia_κ, T_κ, ns> to a_i for skill requirement ns (Steps 8∼9).

Computation complexity of Algorithm 2. In Step 2, sorting the available skills in increasing order of their working cost needs $O(l^2)$ computations, where l is the number of skills available in the network. Next in Step 4, evaluating each skill $s_j \in as(a_i, R_\kappa)$ by calculating the expected profit of the updated team $T_\kappa \oplus \{ns \cup s_j\}$ takes $O(3l)$ computations (one $O(l)$ is used for checking whether s_j has been contributed and the other $O(2l)$ is used for computing $TCC(\cdot)$ and $TWC(\cdot)$). Finally, because there are at most l skills to be evaluated, the total computation complexity of Algorithm 2 then is $O(l^2 + 3l^2)$=$O(l^2)$.

Besides its low computation load, Algorithm 2 can always achieve the optimal solution under some conditions.

Theorem 1. *Given a freelancer a_i that the team manager Ia_κ is negotiating with, if a_i's available skills $as(a_i, R_\kappa)$ are exactly what the current team T_κ lacks, i.e., $as(a_i, R_\kappa) \subseteq us_\kappa$. Algorithm 2 returns the optimal skill contribution of a_i that produces the maximal expected profit for Ia_κ.*

Proof. Before negotiation, assume that Ia_κ's current team T_κ=< $\Omega_\kappa, Cont(\Omega_\kappa, \kappa), us_\kappa$ > and its expected profit is:

$$Ep(T_\kappa, \tau) = (1 - \frac{|us_\kappa|}{|R_\kappa|}) Ev(\kappa, \tau) - \lambda TCC(T_\kappa) - TWC(T_\kappa) \quad (4)$$

Suppose that the available skills $as(a_i, R_\kappa)$ have been sorted in the increasing order of their working cost such that $Wc(a_i, s_1) \leq \ldots \leq Wc(a_i, s_{|as(a_i, R_\kappa)|})$. Without loss of generality, we assume that the first skill, derived from Algorithm 2, that decreases Ia_κ's expected profit is the pth ($p \geq 1$) skill. This means that the skill contribution returned by Algorithm 2 is $ns = \bigcup_{1 \leq j \leq p-1} s_j$ and the following inequality holds.[1]

$$\begin{aligned} &Ep(T_\kappa, \tau) + \frac{p-1}{|R_\kappa|} Ev(\kappa, \tau) - \sum_{1 \leq j \leq p-1} Wc(a_i, s_j) \\ &> Ep(T_\kappa, \tau) + \frac{p}{|R_\kappa|} Ev(\kappa, \tau) - \sum_{1 \leq j \leq p} Wc(a_i, s_j) \end{aligned} \quad (5)$$

Derived from inequality (5), we have

$$Wc(a_i, s_p) - Ev(\kappa, \tau)/|R_\kappa| > 0 \quad (6)$$

Now for any alternative skill contribution ns^* with q (=$|ns^*|$) skills, we need to prove $Ep(T_\kappa \oplus ns, \tau) > Ep(T_\kappa \oplus ns^*, \tau)$. Here, we are mainly concerned with the case where $q \geq p$ (for the case $q \leq p-1$, the proof is similar to this case). It is easy to achieve that in the case of $q \geq p$, the first (p-1) skills with the least working costs in $as(a_i, R_\kappa)$ are selected by ns^* and the other (q-p+1) skills in ns^* are arbitrarily selected from the remaining available skills $as(a_i, R_\kappa) \backslash ns$. Now we have:

$$\begin{aligned} &Ep(T_\kappa \oplus ns, \tau) - Ep(T_\kappa \oplus ns^*, \tau) \\ &= Ep(T_\kappa, \tau) + \frac{p-1}{|R_\kappa|} Ev(\kappa, \tau) - \sum_{1 \leq j \leq p-1} Wc(a_i, s_j) \\ &\quad - \Big[Ep(T_\kappa, \tau) + (\frac{p-1}{|R_\kappa|} + \frac{q-p+1}{|R_\kappa|}) Ev(\kappa, \tau) \\ &\quad - \sum_{1 \leq j \leq p-1} Wc(a_i, s_j) - \sum_{s_j \in ns^* \backslash ns} Wc(a_i, s_j) \Big] \\ &\geq (q-p+1) Wc(a_i, s_p) - \frac{q-p+1}{|R_\kappa|} Ev(\kappa, \tau) > 0 \end{aligned} \quad (7)$$

The equality (7) follows directly from equality (6). Up to this point, we can conclude that for a given freelancer a_i, if its available skills $as(a_i, R_\kappa)$ do not overlap with the team T_κ's satisfied skills $R_\kappa \backslash us_\kappa$, Algorithm 2 can always return the optimal skill contribution. ■

As soon as the team manager Ia_κ sends the offer O=< Ia_κ, T_κ, ns> to the freelancer a_i, a_i needs to evaluate this offer and responds to Ia_κ.

[Phase 2] ***Respond[R].*** Prior to describing the response strategy of the freelancer, we first provide the definition of three states (i.e., *Finally-contracted*, *Tentatively-contracted* and *Free*) of agents during social team formation.

[1]During the negotiation with a_i, the values $Ev(\kappa, \tau)$ and $Ep(\kappa, \tau)$ are assumed to be invariable because the time used for computing the appropriate skill contribution by Algorithm 2 is so short that its effect on $Ev(\kappa, \tau)$ and $Ep(\kappa, \tau)$ and can be neglected.

Definition 2. States of agents. *During social team formation, the agents who initiates a job or has been a member of a complete fulfilled team is in state* Finally-contracted; *the agent who has been a member of a partial fulfilled team is in state* Tentatively-contracted; *and the agent who neither initiates a job nor has joined a team is in state* Free .

Agents in different states might have different response strategies. Thus, it is necessary to devise different response strategies for the freelancer a_i within different states.

Case 1) a_i **is Free.** In this case, a_i will accept T_κ's offer. This is because joining a team to perform a job can obtain some financial remuneration, which is a rather economical choice compared to the free state that has no remuneration.

Case 2) a_i **is Tentatively-contracted.** Suppose that a_i has contracted with a partial team T_{κ^*}. Now a_i receives a new offer O=$<Ia_\kappa, T_\kappa, ns>$ from T_κ. A dilemma now is faced by a_i: breaking the contract with the original team T_{κ^*} to participate in this new team T_κ or rejecting the new offer $O(\cdot)$. Here, we use the measure of expected remuneration (per unit time) to quantify how much a_i gains by rejecting or accepting. On one hand, staying in the original team T_{κ^*}, a_i will achieve the expected remuneration (per unit time):

$$Er(a_i, T_{\kappa^*}, \tau) = \frac{Sr(T_{\kappa^*}, \tau) \sum_{(a_i,s_j)\in Cont(\Omega_{\kappa^*},\kappa^*)} Wc(a_i, s_j)}{Wt_{\kappa^*}} \tag{8}$$

In (8), $Sr(T_{\kappa^*}, \tau)$=(1-$|us_{\kappa^*}|/|R_{\kappa^*}|$)$(Dl_{\kappa^*} - \tau)$ indicates the success rate of team T_{κ^*} that will be formed completely; $\sum_{(a_i,s_j)\in Cont(\Omega_{\kappa^*},\kappa^*)} Wc(a_i, s_j)$ indicates the remuneration obtained by a_i if job κ^* is finished successfully. Wt_{κ^*} indicates the period it will take to finish κ^*. On the other hand, joining the new team T_κ that sends offer O=$<Ia_\kappa, T_\kappa, ns>$, a_i will obtain the expected remuneration (per unit time):

$$Er(a_i, T_\kappa, ns, \tau) = \frac{Sr(T_\kappa, \tau) \sum_{s_j\in ns} Wc(a_i, s_j)}{Wt_\kappa} \tag{9}$$

where $Sr(T_\kappa)$=(1-$|us_\kappa \backslash ns|/|R_\kappa|$)$(Dl_\kappa - \tau)$ indicates the success rate of team T_κ formed completely if a_i contributes the required skill ns. The meanings of other terms are similar to those discussed in (8). Finally, as a self-interested individual, a_i prefers to join the team that produces a larger expected remuneration (per unit time) for itself. In other words, if $Er(a_i, T_{\kappa^*}, \tau) \geq Er(a_i, T_\kappa, ns, \tau)$, a_i will stay in the original team T_{κ^*} and reject the offer of T_κ; otherwise, a_i will break the contract with the original team T_{κ^*} and accept T_κ's offer.

Case 3) a_i **is Finally-contracted:** In the case that the team T_κ of which a_i is a member has recruited sufficient contractors that can satisfy all of job κ's skill requirements and has started job execution, a_i will reject any new offer until job κ finished. This is because each contractor can only be one team member and breach the contract with the complete team will suffer tremendous monetary penalty or reputation loss [11].

As soon as the freelancer a_i makes a decision on the new offer $O(\cdot)$, a_i will send a response R=$< acceptance/rejection>$ to the team manager Ia_κ. And if Ia_κ receives the acceptance response from a_i, Ia_κ needs to make a final confirmation for this acceptance.

[Phase 3] ***Confirm[C].*** For the freelancer a_i that accepts its offer O=$<Ia_\kappa, T_\kappa, ns>$, the team manager Ia_κ first needs to make a *tentative* contract with a_i on skill contribution ns (a *tentative* contract means that before T_κ is formed completely, Ia_κ can adjust a_i's skill contribution to T_κ). And if a_i's skill contribution leads to a complete team for job κ, this contract will become a *final* contract such that a_i cannot breach the contract unilaterally until κ is finished. Then, Ia_κ updates its team configuration by adding the skills $\bigcup_{s_j\in ns}(a_i, s_j)$ to $Cont(\Omega_\kappa, \kappa)$, removing the duplicated skill contributions provided by other team contractors and removing the team contractors that do not contribute skills any more.

Up to this point, we have illustrated the entire negotiation process between a team manager and a freelancer. A formally description of this negotiation mechanism adopted in Algorithm 1 (Step 6) can be seen in Algorithm 3.

Algorithm 3. Negotiate(Ia_κ,a_i)

1. Manager Ia_κ calls Algorithm 2 to generate the offer O=$< Ia_\kappa, T_\kappa, ns>$ and sends the offer $O(\cdot)$ to a_i.
2. **If** a_i is *Free*
3. Freelancer a_i sends response R=$< acceptance >$ to Ia_κ
4. **Else If** a_i is *Tentatively-Contracted* && $Er(a_i, T_{\kappa^*}, \tau) < Er(a_i, T_\kappa, ns, \tau)$
   % T_{κ^*} is the current team of which a_i is a member %
5. Freelancer a_i sends R=$< acceptance >$ to Ia_κ
6. **Else** Freelancer a_i sends R=$< rejection >$ to Ia_κ.
7. If R==$< acceptance >$
8. Ia_κ contracts with a_i and updates team configuration.

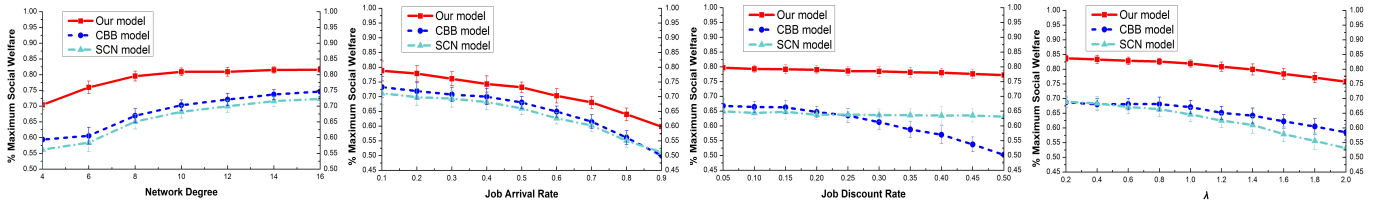
V. Experiments

A. Experiment setting

In this section, we will validate the effectiveness of the proposed social team formation model in simulated agent networks. Each network consists of 200 agents, which are interconnected by a random network model used in [18]. There are 16 type skills available in each network. Each agent owns U(1,4) type skills ($U(a,b)$ returns the value in the range [a,b] uniformly) and the working cost of each skill is set U(1,10). The communication cost between each pairwise agents is set U(1,10). At each time step, a job κ arrives at the network with a probability p=0.1∼0.9. The number of skills required by a job is set U(1,16). The period that is needed to accomplish a job (i.e.,Wt_κ) is drawn from U(20,40) and the deadline of a job (i.e., Dl_κ) is set in the range [It_κ, It_κ+20], which must be greater than the job κ's initialization time It_κ. The parameter λ used to trade-off the communication cost and working cost is set 0.2∼2.

We compare our social team formation model (**Our model**) with three other models, i.e., the benchmark centralized optimal model (**OPT**), the distributed simple contract net (**SCN**) and complex bilateral bargaining model (**CBB**) models.

- **Optimal Model (OPT) [5].** In this model, there is a central controller maintaining information on all of the agents' working costs and locations. When a job submitted to the system, the controller can build a team



(a) (b) (c) (d)

Fig. 2: Social welfares of different models relative to OPT on (a) network degree, (b) job arrival rate, (c) job discount rate and (d) tradeoff factor λ.

of agents that have the least working and communication costs in the network. This is an ideal model, which is impractical, but it can be used as an upper bound of system performance.

- **Simple Contract Net Model (SCN) [8].** In this model, each agent submits the bid to the manager whose job has the largest profit value and the team manager only select the bidders that have the skills the current team lacks. Compared to this model, the significance of accounting for all of the bidder's available skills could be revealed.
- **Complex Bilateral Bargaining Model (CBB) [10].** In this model, job managers are assumed to have no prior knowledge of workers' working price. And the manager negotiates with the workers round and round until they reach an agreement on skill price. Compared to CBB, the advantage of our model with simple negotiation mechanism could be revealed.

We evaluate the performance of these models through social welfare (*SW*) and the team formation time. *SW* is computed as the sum of all manager agents' profits, i.e.,

$$SW=\sum_{\kappa_i\in K}\left(v_{\kappa_i}(ct)-\lambda TCC(T_{\kappa_i})-TWC(T_{\kappa_i})\right) \quad (10)$$

where ct is job's completion time and for the job that is not completed successfully, its completion time is set infinite.

B. Simulation results

Fig. 2(a)∼Fig. 2(d) show the percentage social welfares achieved by our model, CBB and SCN, respectively, compared to the social welfare of the OPT. These results are recorded by averaging over 40 instances.

Fig. 2(a) shows the percentage social welfares of these models on network degree, where job arrival rate is 0.4, job discount rate is 0.1 and the trade-off parameter λ is 1. In Fig. 2(a), as the network degree increases, the social welfares of our model, CBB and SCN increase gradually and our model can evenly reach to 80% of OPT. This can be understood from two perspectives: 1) when the managers have more neighbors, they can access more skills, leading to an increment probability in forming complete team; and 2) the more neighbors, the more easily these managers can build teams of professional agents, and consequently, the less time is required for team formation. Thus, these distributed models perform better in the system with larger network degree.

Fig. 2(b) shows the percentage social welfares of these models on job arrival rate, where network degree is 8, job discount rate is 0.1 and λ is 1. In Fig. 2(b), the social welfares of the three distributed models decrease with job arrival rate, i.e., the more frequency jobs arrive at the system, the less social welfare they will achieve compared to OPT. This finding can be explained as follows: as job arrival rate becomes larger, the manager will have to manage more jobs; and because the manager has limited neighbors, these overloaded jobs either have to wait the manager's neighbors' skills that are being used by other jobs or have to take much time to negotiate with the remote agents. However, these remote negotiations might be unaccessible because there is no social interaction available for introducing these remote negotiations, thereby making certain team formation unsuccessful. While OPT is capable of communicating all of system agents, therefore it can take full advantage of the skills distributed on the system.

Fig. 2(c) shows the percentage social welfares of these models on job discount rate, where network degree is 8, job arrival rate is 0.4 and λ is 1. In Fig. 2(c), as the job discount rate increases, the social welfare of the CBB decreases as well, but our model and SCN perform almost invariably. The potential reason is that CBB always takes much time for agents to reach an agreement on working price, leading to a large team formation time (the team formation time results can be seen in Fig. 3). Because the value function attached to each job is a conciliatory function x^{δ} $(0<\delta<1)$ and when the variable $x\to 0$ (i.e., the team formation time in CBB approaches to the deadline), $x^{\delta_1}\gg x^{\delta_2}$ if $0<\delta_1\ll\delta_2<1$. Therefore, the larger the discount rate, the more social welfare of CBB will decrease. On the other hand, when $x\to 1$ (i.e., it does not take much time for team formation in our model and SCN), $x^{\delta_1}\backsimeq x^{\delta_2}$ if $0<\delta_1\ll\delta_2<1$. Therefore, the job discount rate has no significant effect on our model and SCN's performance.

Fig. 2(d) shows the percentage social welfares of these models on the trade-off parameter λ, where network degree is 8, job arrival rate is 0.4 and the job discount rate is 0.1. In Fig. 2(d), as the communication factor becomes more crucial (i.e., λ becomes larger), these distributed models perform worse. This is because that the OPT model can always find a team of agents that incur the least communication cost, while these distributed local models cannot perform that well. It is also should be noted that λ has a larger effect on SCN: its performance decreases from 70% to 50% when λ varies from 0.2 to 2. This can be explained by the fact that the agents in SCN always submit bids to the job that has the highest profit and the manager only selects these bidders that have the skills it lacks, ignoring their communication costs.

In summary, our model performs consistently better than the CBB and SCN models. The potential reason is that compared

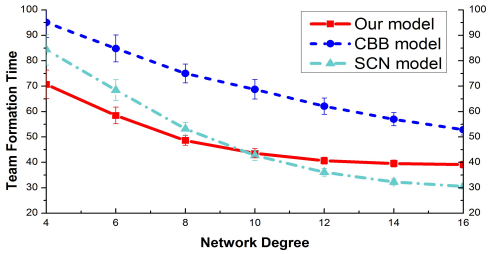


Fig. 3: The team formation time of distributed team formation models.

to the CBB, our model reduces much team formation time and compared to SCN that only negotiates on the unsatisfied skills, our model considers each available skill, because this kind of skill might demand less working cost and its owner might incur less communication overhead.

Fig. 3 shows the average team formation time of each job in the three distributed models. From Fig. 3, we can determine that: 1) The team formation time of these models decreases with network degree. The reason is that when the managers have more neighbors, they can access the lacked skills more easily. 2) The managers in CBB take much more time than our model and SCN on building teams. This is because that in CBB, the negotiation between a manager and contractor proceeds multiple rounds until they reach an agreement on working price. However, in our model, when the manager and contractor do not reach an agreement on skill contribution within one negotiation round, the manager will terminate this negotiation and proceeds to the next round negotiation with another freelancer that demands the less remuneration and reside in closer location that incurs less communication cost. 3) When network degree becomes larger ($\geq$10), SCN forms teams faster than our model. This is because that compared to our model with redundant negotiation on satisfied skills, in SCN model, if each manager has sufficient social neighbors, it is more timesaving for him to send offers for requiring these skills what exactly he lacks.

VI. Conclusion

In this paper, we propose a practical and efficient team formation model for the non-cooperative social networks where individuals are self-interested. In terms of practicability, we mean that compared to the traditional models with cooperativeness assumption, our model considers individuals' selfish behaviors and individuals build and join teams in a self-organized manner, which is in accordance with real-world applications. By the efficiency, we mean that our model achieves 80% social welfare compared to the ideal centralized model on average, and compared to other conventional distributed models, our model reduces the team formation time significantly while maintaining beneficial teams where team members demand little working and communication costs.

In this study, the manager and contractor are allowed to decommit from tentative agreement arbitrarily. In reality, however, if one contract party breaches the agreement unilaterally, he needs to pay a compensation to the other contract party [10]. Therefore, in the future work, it is essential for us to extend the current social team formation model by allowing for individual's compensatory decommitment strategy.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No.61170164, No. 61472079), the Program for Distinguished Talents of Six Domains in Jiangsu Province (No.2011-DZ023), and the Funds for Distinguished Young Scholars of Jiangsu Province (BK2012020).

References

[1] L. Tran-Thanh, S. Stein, A. Rogers and N.R. Jennings, Efficient Crowdsourcing of Unknown Experts Using Multi-Armed Bandits, *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI-12)*, pp.768-773, Montpellier, France, 2012.

[2] T. Lappas, K. Liu and E. Terzi, Finding a Team of Experts in Social Networks. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD-09)*, pp.467-475, Paris, France, 2009.

[3] A. Anagnostopoulos, L. Becchetti and C. Castillo, A. Gionis and S. Leonardi, Online Team Formation in Social Networks. *Proceedings of the 21st international Conference on World Wide Web (WWW-12)*, pp.839-847, Lyon, France, 2012.

[4] S. Liemhetcharat and M. Veloso. Modeling and Learning Synergy for Team Formation with Heterogeneous Agents. *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems(AAMAS-12)*, pp.365-374 Valencia, Spain, 2012.

[5] M. Kargar, A. An and M. Zihayat, Efficient Bi-objective Team Formation in Social Networks, *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD-12)*, pp.483-498, Bristol, UK, 2012.

[6] S. Datta, A. Majumder and K. Naidu, Capacitated Team Formation Problem on Social Networks, *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining (KDD-12)*, pp.1005-1013, Beijing, China, 2012.

[7] S. Rangapuram, T. Buhler and M. Hein, Towards Realistic Team Formation in Social Networks based on Densest Subgraphs, *Proceedings of the 22nd International Conference on World Wide Web(WWW-13)*, pp.1077-1087, Rio de Janeiro, Brazil, 2013.

[8] M.de Weerdt, Y. Zhang and T. Klos, Multiagent Task Allocation in Social Networks, *Autonomous Agents and Multi-Agent Systems*, 25(1): 46-86.

[9] B. An, V. Lesser, D. Irwin and M. Zink, Automated Negotiation with Decommitment for Dynamic Resource Allocation in Cloud Computing, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems(AAMAS-10)*, pp.981-988, Toronto, Canada, 2010.

[10] D.Y. Ye, M.J. Zhang and D. Sutanto, Self-Adaptation-Based Dynamic Coalition Formation in a Distributed Agent Network: A Mechanism and a Brief Survey, *IEEE Transactions on Parallel and Distributed Systems*, 24(5): 1042-1051, 2013.

[11] Y. Jiang, Y. Zhou and W. Wang, Task Allocation for Undependable Multiagent Systems in Social Networks, *IEEE Transactions on Parallel and Distributed Systems*, 24(8): 1671-1681, 2013.

[12] Yichuan Jiang and J.C. Jiang, Understanding Social Networks from a Multiagent Perspective, *IEEE Transactions on Parallel and Distributed Systems*, 2014.

[13] M.E. Gaston and M. desJardins, Agent-Organized Networks for Dynamic Team Formation, *Proceedings of the 4th international Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pp.230-237, Utrecht, Netherlands, 2005.

[14] R.Glinton, K. Sycara and P. Scerri, Agent Organized Networks Redux. *Proceedings of the 23rd National Conference on Artificial Intelligence(AAAI-08)*, pp.83-88, Chicago, Illinois, 2008.

[15] R. Kota, N. Gibbins and N.R. Jennings, Self-Organising Agent Organisations, *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-10)*, pp.797-804, Budapest, Hungary, 2009.

[16] Yichuan Jiang and J.C. Jiang. Diffusion in Social Networks: A Multiagent Perspective. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, DOI: 10.1109/TSMC.2014.2339198, 2014.

[17] W. Wang and Y. Jiang, Community-Aware Task Allocation for Social Networked Multiagent Systems, *IEEE Transactions on Cybernetics*, 44(9):1529-1543, 2014.

[18] A.L. Barabási and R. Albert. Emergence of Scaling in Random Networks, *Science*, 286(5439): 509-512, 1999.