# Adapting Network Structure for Efficient Team Formation

Matthew E. Gaston
mgasto1@cs.umbc.edu

John Simmons
js5@umbc.edu

Marie desJardins
mariedj@cs.umbc.edu

Multi-Agent Planning and LEarning Laboratory (MAPLE)
Department of Computer Science
University of Maryland Baltimore County

## Abstract

*The concept of team formation is central to a wide variety of disciplines, including planning and learning in multi-agent systems, artificial social systems, and distributed artificial intelligence. Typically, models of agent-based systems do not focus on the nature and structure of the interconnection network that dictates agent interaction, although recent findings suggest that real-world networks of many different types have rich, purposeful, and meaningful structures. Using a simple agent-based computational model of team formation, our previous work suggests that organizational network structure can have a significant effect on the dynamics of team formation. We present several strategies for locally adapting network structure in a simple team formation scenario, and give empirical results that show that such adaptation methods are capable of significantly improving organizational efficiency. These methods prove to be especially useful for adapting agent networks in the presence of attack faults that target the most highly connected nodes in the network.*

**Keywords:** team formation, organizational learning, adaptation, complex networks.

## 1. Introduction

In both real and artificial societies, effective organizations are highly dependent on structures that foster the efficient formation of teams to accomplish complex tasks. In many applications of multi-agent systems, agents must coordinate to solve problems, dynamically distribute resources, and collaborate to achieve collective goals. Team formation is a core concept for all of these applications.

With the continued growth of the Internet and the technologies of the Semantic Web [11], agents are required to operate in expanding and increasingly complex environments. Agents operating in these domains will be unable to maintain a working knowledge of all other agents in the domain; as a result, the network structure that governs the interactions between agents will play a fundamental role in the effectiveness of these agent societies.

Recent research on real-world networks has revealed that social systems have a rich structure [13, 14]. In understanding agent-based team formation, much effort has been devoted to the dynamics of individual agents and various algorithms for forming teams, with little focus on the role of interaction topologies. Recent work in computational learning theory [4, 3], however, highlights the importance of social structure on organizational performance for various types of organizational behaviors. Such studies have shown that clustered organizations, where agents have few connections but are tightly connected, are more stable than organizations with high connectivity [12]. Similarly, organizations with fluid (changeable) groupings are more likely to foster collaboration [10]. Motivated by these findings, this paper provides empirical evidence of the importance of network structure on multi-agent team formation. We also show that organizational learning through local network adaptation can drive agent societies toward more efficient structures.

## 2. The Team Formation Model

In the multi-agent systems community, there has been a significant amount of research on team formation and self-organization. Much of the work on team formation focuses on the mental states of the agents and their willingness to form teams and collaborate [5, 18]. These studies have driven the development and implementation of frameworks in which teams coordinate closely to develop and execute distributed plans [15, 7].

We present a simple model that emphasizes the dynamics of multi-agent team formation, without the overhead of detailed negotiations and belief modeling by the individual agents. Tasks are generated and globally advertised to the agents in the organization, and agents form teams to accomplish the tasks. Each team must consist of a connected subgraph of agents who possess the necessary skills for that task, so the network structure restricts agent interaction and participation on teams.

**Teams and Tasks.** The organization in the model consists of $N$ agents, $a_1, a_2, \ldots, a_N$, each situated at a vertex in a graph. Each agent $a_i$ is assigned a single skill, $\sigma_i \in [1, \Sigma]$, where $\Sigma$ is the organizational *skill diversity*. In the model, agents can be in one of three states: uncommitted, committed, or active. An *uncommitted* agent is available and not assigned to any task. A *committed* agent has chosen a task, but the full team to work on the task has not yet formed. Finally, an *active* agent is a member of a team that has fulfilled all of the skill requirements for a task and is actively working on that task.

Tasks are introduced at fixed *task introduction intervals*, where the length of the interval between tasks is given by a parameter, $\mu$. Tasks are globally advertised (i.e., announced to all agents). Each task $T$ has an associated size requirement, $|T|$, and a $|T|$-dimensional vector of required skills, $R_T$. The skills required for a given task $T$ are chosen uniformly from $[1, \Sigma]$. Each task is advertised for a finite number of time steps proportional to its size (namely $\delta|T|$, where $\delta$ is a model parameter) to ensure that the resources (i.e., agents) assigned to the tasks are freed if the full requirements of the task cannot be met. Similarly, teams that form to fill the requirements of a given task are only active for a finite number of time steps (namely $\alpha|T|$, where $\alpha$ is a model parameter).

**Agent Dynamics.** The organizational network structure, represented by the graph, dictates the collections of agents that can be on teams together.

> **Definition**: A **valid team** is a set of agents whose corresponding set of vertices induce a *connected subgraph* and whose collective skill set fulfills the skill requirements for a given task $T$.

Given the restriction of teams to connected subgraphs of the organizational structure, agents must use heuristics based on localized knowledge to decide which teams to join. The local knowledge available to an agent includes the number of positions currently filled on each team, the number of the agent's uncommitted immediate neighbors, and the number of immediate neighbors on each team. Different heuristics for using this local information to decide which team to join yield varying model dynamics [9].

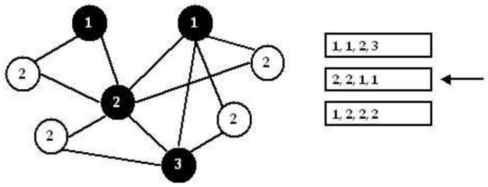


**Figure 1. An example of blocking in the team formation model for a simple graph. The list on the right represents tasks and their respective skill requirements.**

One interesting phenomenon that can occur in this team formation model is *blocking*. This phenomenon is illustrated in Figure 1 for a simple graph. In the figure, the list on the right corresponds to tasks and their skill requirements. Each vertex is labeled with its corresponding skill. Notice that if the shaded vertices form a team to meet the requirements of the first task, teams attempting to complete the second and third tasks are blocked by the agents working on the first team.

## 3. Network Effects

In our previous work [9], we showed that the network structure of an organization can have significant effects on the overall team formation performance. In this work, we used four different network structures to model different interaction topologies among the agents. The graphs are designed to have a fixed density (i.e., a constant number of edges for a given graph size).[1]

- 2-dimensional regular lattice: In this graph, each vertex is connected to exactly four others, as in a typical spatial agent-based model.
- 2-dimensional small-world network: This network is derived from the 2-dimensional lattice described above. The small-world network is formed by randomly "rewiring" each edge (i.e., changing one of its vertices to a random vertex) with a probability $p$ [16, 17]. For all of the experiments below, $p = 0.05$.
- Random graph: This is the Erdos-Renyi random graph model [8], where there is an undirected edge between

[1] Holding the graph density constant allows us to focus only on the topology of the network. In practice, graphs with higher density may be more able to form unblocked teams, but may also have higher communication and computational overhead. Determining the ideal density for an agent society is an interesting problem, but is beyond the scope of this paper.

each pair of vertices with probability $\rho$. To approximate the density of the lattice and small-world networks, $\rho = 2/(N-1)$.

- Scale-free graph: In this graph model, the graph is iteratively grown. As new vertices are added the graph, they "preferentially" attach to existing vertices that are highly connected [1]. The resulting graph is a random structure that has "hub" vertices. To match the density of the other networks, $2N$ undirected edges are introduced.

These network structures were chosen because they are well studied, and because they are reflective of typical agent-based and real-world networks.

In order to measure collective organizational performance, we use the following definition of organizational efficiency:

$$\text{efficiency} = \frac{\text{\# of teams successfully formed}}{\text{total \# of tasks introduced}}. \qquad (1)$$

This measure provides a global view of the task completion rate of the agent society. In our previous work, we studied how the organizations perform over a range of parameters, where the agents are embedded in each of the four networks described above. Each of the graphs has 100 vertices and 200 undirected edges. The experimental results presented are the result of 10 simulations run for 5000 time steps of the model with $\delta = 2$ and $\alpha = 4$.

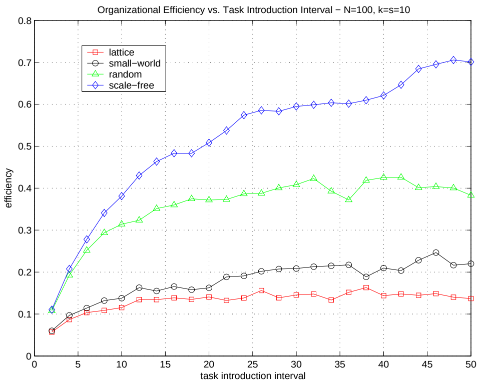


**Figure 2. Organizational efficiency vs. task introduction interval for the team formation model with $N = 100$, $|T| = \Sigma = 10$**

Figure 2 shows the effect of varying the task introduction interval on system performance for each of the four network structures. In this experiment, agents joined teams using only information about how many positions on the team are previously filled (i.e., how many agents have previously committed to a specific team). Each agent sequentially considers joining each of the teams to which one of its neighbors belongs, in random order. The agent joins the team with probability

# positions filled on team / team size

If it decides not to join a team, it moves on to consider the next candidate team, until no options remain. This decision process is performed at each time step.

Although the scale-free network dominates, there are several important performance differences among the other structures. First, in general, the more stochastic network structures (i.e., the scale-free and the random networks) appear to consistently outperform the more regular network structures (i.e., the lattice and the small-world). Second, within the "regular" graphs, the small-world network (–*o*–) outperforms the lattice network structure, despite their high degree of similarity. In these experiments, the small-world network was constructed by simply rewiring edges of the lattice network with a fairly low probability, $p = 0.05$. Therefore, 95% of the edges in these two network structures are identical, yet the small-world network shows a statistically significant improvement in organizational efficiency over the lattice structure.

In the other experiments we performed—varying network size, team size, skill diversity, and agent decision strategies—the results were similar: scale-free networks dominated, followed by random, small-world, and then lattice network structures.[2] Other researchers have found that similar results hold for the adoption of social conventions in agent societies [6].

## 4. Organizational Learning

The results presented above demonstrate that the network structure can have a substantial effect on organizational performance, and that some network structures consistently outperform others. In considering how to apply this finding to the implementation of multi-agent systems, the obvious choice is to use the network structure that performs the best, but this implies that the designer has knowledge of all possible network structures. It also ignores the possibility of agent failures, which can lead to a substantial drop in performance. Scale-free networks, which have nodes of very high degree, are particularly susceptible to targeted *attack faults,* as we discuss later. To address these issues, we propose a set of strategies for network adaptation. The results we present here suggest that organizational learning

[2] These results will be presented in more detail in a paper in preparation, to be submitted to the Journal of Artificial Intelligence Research (JAIR).

(i.e., improvement in organizational efficiency) can be accomplished through localized network adaptation.

Adaptation decisions occur on a local level: when adaptation occurs, each agent is responsible for deciding which, if any, of its incident edges should have its other end moved, and to which other agent it should be attached. For the sake of simplicity, in the work presented here, adaptations occur in a batch scenario, but the results should be applicable to an online scenario. Since agents decide locally whether or not to join a forming team, it makes sense to adapt locally as well.

The *performance* of nodes and edges is assessed as a tradeoff between their rate of successfully forming teams and their rate of joining failed teams. Specifically, each time a team succeeds, each of its vertices and edges has its performance measure incremented (+1); each time a team fails, the performance of each of its vertices and edges is decremented (-1).

## 4.1. Network Faults

To study the effectiveness of the adaptation strategies, we model failures in the simulation environment by constructing faulty graphs. These are created by first constructing a graph of some network structure., then selecting a subset of the vertices to be deleted from the graph (along with their incident edges). The *fault rate* of a graph is the fraction of vertices that are removed.

The notion of *attack* faults was coined by Albert *et al.* [2], with the intent of simulating a malicious attack on a network, where the attackers specifically target the most important agents – i.e., those with highest degree. This scenario could also be seen as a more benign "wear-and-tear" situation, where the most connected vertices are more prone to failure simply because of the inherent added stress and traffic volume. Vertices are selected for deletion by selecting a random edge in the graph and then deleting one of its incident vertices; as a result, the probability of a vertex failure is proportional to that vertex's degree.

Although we have studied both *random* faults (in which all agents have an equal probability of failure) and *attack* faults (in which each agent's probability of failure is proportional to the degree of its corresponding vertex), in this paper we only report results for attack faults. We are interested in this scenario because it is realistic for many types of environments. More importantly, scale-free networks—which yield the best performance in non-fault scenarios—are the most susceptible to attack faults of any of the network structures studied. Therefore, one of our major objectives is to develop adaptation strategies that allow scale-free networks to maintain their high organizational efficiency under an attack fault scenario.

We focus on graphs with relatively low fault rates (less than 10%) for two reasons. First, if we can identify a scheme that works well for low fault rates, that scheme could easily be generalized to a continuous environment, where faults occur over time. That is, if small failures happen once in a while, the adaptation scheme could be used repeatedly, each time a fault is detected. Second, a single massive failure is less realistic and is very difficult to recover from under any scheme since so many edges have been lost.

## 4.2. The Adaptation Process

There are two major aspects of each adaptation strategy. First, a strategy must specify which agents will be allowed to adapt. The options that we explored for this aspect are:

- Random: Allow a random set of agents to adapt (move an edge). The number of adaptations $N_A$ is given as a parameter to the system. (In the experiments described here, $N_A = N$.)
- Team Failure: When a team fails to finish, allow one of its agents to adapt (move an edge).
- Node Failure: When an agent fails, allow each of its neighbors to adapt (replace the "lost" edge). This has the effect of replacing most of the edges lost due to failure. The only edges which will not be replaced are those where both incident vertices have failed.

Second, a strategy must define an initial pool of candidate neighbors for each adapting agent. The options we explored are:

- All: All agents may be new neighbors
- Teammate: Only ex-teammates may be new neighbors
- Referral: Neighbors of neighbors may be new neighbors

These two aspects yield nine different adaptation strategies, which cover a wide range of possible scenarios. The realism and requirements of each will be discussed later, but examining all of these possibilities should give a sense of which methods are most promising.

Some of the strategies require prior knowledge regarding teammate and performance histories. This data is collected in a preliminary simulation; adaptation is then performed in a batch step. The performance results in Section 5 are based on the network's performance in a second simulation following the batch adaptation.

The Random and Team Failure strategies require an incident edge to be deleted before creating a new edge. In this case, the edge with the worst performance is chosen for deletion, with ties broken randomly. (In the Node Failure strategy, edges are replaced when one of their vertices fails.)

New neighbors are selected through a four-step process that filters an initial pool of candidates down to a final new neighbor.

First, the candidate pool is created using the current strategy (All, Teammate, or Referral, as described above). Any candidates that are already neighbors are excluded. In the teammate strategy, each agent keeps a queue of its 20 most recent teammates. This data is collected during the preliminary simulation (before adaptation occurs).

Second, with the All strategy, an optional structure filtering may be performed. Each candidate is subjected to the same criteria used in the graph's creation, according to its structure type. For example, in a lattice graph, the only candidates kept are those directly above, below, to the left, or to the right, and having no other vertices in between. The intuitive purpose of this filter is to model a scenario in which the underlying graph structure arises from some environmental influence (e.g., line-of-sight constraints for a lattice graph).

Third, a skill filtering is performed. The only candidates kept are those that have a skill that is currently underrepresented among the agent and its neighbors. This works to diversify the local skill distribution. Different strategies can optionally specify a specific skill requirement, which will be treated with highest priority. Specifically, the Team Failure strategy uses this approach to find a new neighbor having whichever skill was missing on the failed team.

Fourth, a degree filtering is performed. The only candidates kept are those with the single highest degree; if some candidate has a degree one less than another, it is eliminated. The idea behind this filter is that a well connected agent is most likely to have neighbors which could complete a forming team. This strategy is in keeping with the general "hub" architecture of scale-free networks. This filter primarily serves as a tiebreaker among promising candidates. If more than one candidate still remains after this filter, ties are broken randomly.

## 5. Results and Discussion

Figure 5 shows empirical results for the nine adaptation strategies. As in Section 3, each of the graphs has 100 vertices and 200 undirected edges. Each point in the graphs is the average of ten 5000-step simulations with $\delta = 2$ and $\alpha = 4$.

The next two sections discuss the results with respect to different candidate pool and adaptation source strategies, respectively. In Section 5.3, we analyze the best performing strategy in more detail. Section 5.4 presents some preliminary results on analyzing the adapted network structures.

### 5.1. Candidate Pool Strategies

The "All" candidate generation strategy performed well, and the "Referral" strategy managed to outperform the unadapted graph by at least a small margin.

Only the "All" strategy enables an agent to reconnect to a different (and larger) component of the network, if it becomes disconnected as the result of agent failures. Two of the "Referral" strategies managed to succeed even without this capability, suggesting that connecting between different network components is not a critical part of a good strategy.

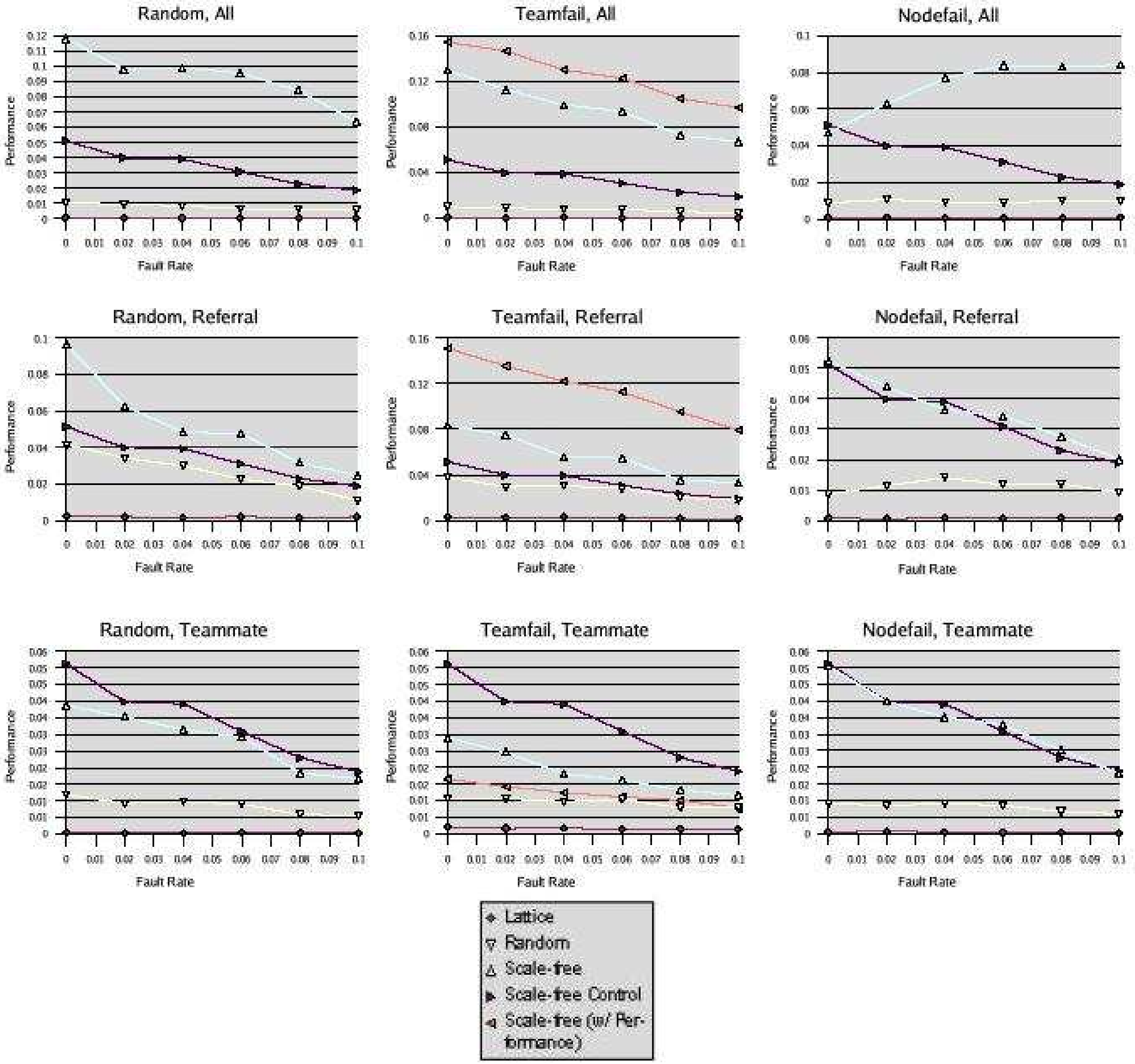
The "Teammate" strategy did not succeed at all. This is a little surprising, since the "Teammate" approach seems very intuitive, and since it allows agents to find more distant neighbors than the Referral strategy. In the current implementation, teammates are identified by running a simulation after the faults are introduced, but before adaptation. If the teammate strategy was modified to remember teammates from before the failure incident, it might show better results, since teammates that ended up in other components of the faulty graph would still be available.

### 5.2. Adaptation Source Strategies

Of the three strategies for choosing agents to adapt, the "Team Failure" strategy performed best. The "Random" strategy was fairly successful, which is interesting because it was performing far fewer adaptations than the "Team Failure" strategy: the former performed about 100 adaptations, while the latter performed about 500. These values were given as parameters (the number of random adaptations to perform and the teamfail queue limit). Increasing the amount of random adaptation could lead to better performance for the random strategies.

The "Node Failure" strategy performed poorly, especially considering that its resulting graphs can potentially have many more edges than those from the other strategies. Perhaps its poor performance could be blamed on the candidate pools, since it performed almost as well as the others under the "All" strategy. The "Nodefail, All" strategy has some very unusual behavior, where its performance actually increases with the fault rate, to a point. This is because the "Node Failure" strategy only allows neighbors of a failed vertex to adapt; the extremely low fault rates simply have too little adaptation to achieve the full benefit. When a single agent fails, only its neighbors may adapt.

The "Team Failure" strategy has a distinct advantage over the others, in that it is much more sensitive to the real performance of the graph, and is able to specifically target problematic areas. It even performs well on a fault-free graph, achieving a significant performance gain. In the "Team Failure" charts, the "Scale-free (w/ Performance)" data shows a modification of the team failure scheme where



the best performing team member is chosen for adaptation, rather than a random team member. In general, this variation performs very well, though it fails when used in conjunction with the "Teammate" candidate pool strategy. (While the adaptation mechanisms were being developed, it was observed informally that if the *worst* performing team members are chosen, performance falls near to or below the control data.)

### 5.3. The Rich Get Richer

It is interesting to notice that the best performing strategy ("Teamfail with performance, All") uses a rich-get-richer sort of approach: the best performing agents have the chance to further increase their performance; agents with higher degree are more likely to get new neighbors through adaptation; and the worst performing edges are dropped. This would seem to create a situation where a few agents increase their performance at the expense of others, but the results indicate that a majority of the agents benefit from this approach. This likely results from the nature of the team formation environment: the performance of an agent is directly related to the performance of its neighbors, since only complete teams can increase their performance ratings. Thus, when an agent adapts for better performance, all of its neighbors (except the one that was disconnected) will see an indirect increase in performance. This suggests that even in a situation where agents behave in a greedy, self-serving manner, the network as a whole can benefit.

One concern is the possible creation of an "uber-team," where a majority of the agents become connected in a star pattern with a single hub agent. This sort of formation would perform well on individual non-overlapping tasks, since there is a very good chance of finding all of the skills required for any given task. The absolute performance of the adapted graphs never rose above around 15%; it may be the case that only such non-overlapping tasks were being completed. Such a network has two significant drawbacks. First, it does not scale well, as its performance is bounded above due to the blocking effect [9]. While the hub agent is engaged, only a few of the peripheral agents would also be engaged, leaving the others unoccupied yet unable to form any teams. Therefore this formation would almost always be able to complete any single task, but would only be able to complete one task at a time regardless of the size of the formation. A second concern is fault tolerance. The high connectedness of the hub agent makes this formation extremely susceptible to further attack faults: if the hub agent failed, recovery would be very difficult. If the goal of adaptation was to create fault tolerance, then this would be counterproductive. If, however, the goal was simply performance gain, then formations of this sort could be promising.

### 5.4. Effects of Adaptation on Network Structure

A few of the schemes demonstrate a remarkable performance gain, even exceeding the unadapted scale-free network without any failures. One interesting question is how the network structure is being affected by the adaptations. Figure 5.4 shows the (cumulative) distributions of degree and vertex performance for a single instance of a scale-free graph under a few selected adaptation schemes. The adapted graphs were not derived from the same original graph.

It appears that the adaptation schemes have increased the disparity of the degree distribution: there are even fewer vertices of high degree, and those few vertices have even higher degree than before. The performance distribution is affected in an inverse fashion: many vertices are now grouped in the higher performance range, where before performance was fairly evenly distributed.

## 6. Future Work

Future directions on this work fall into four main categories: further investigation and data gathering for the strategies described in this paper; studying adapted network structures and characteristics; developing new adaptation methods; and exploring methods for online adaptation.

Gathering additional data and running further experiments would enable us to better explain our results. In particular, to understand the performance differences between the adaptation strategies, it would be useful to gather more statistics about the adaptation process. Data such as the size of the candidate pool after each filtering step, the number of adaptations performed, and the distribution of adaptations among the agents could help indicate why some strategies outperformed others. We also plan to vary the parameter settings (number of adaptations, size of teammate history list, etc.) in order to study their influence on the adaptation process.

Further analysis of the adapted network structures is warranted. Given that some strategies achieved a performance gain in a fault-free environment, characterizing those resulting structures could lead to better team formation networks in general. Perhaps an optimal network structure for the team formation environment could be discovered, which would be an important result with many interdisciplinary applications.

Many other local adaptation strategies are possible. In particular, given that many of the current strategies and mechanisms operate deterministically, it would be interesting to investigate more weighted probabilistic approaches. For example, the degree filter will eliminate a candidate if its degree is just one less than the maximum, even if it is a very good candidate in other aspects. Relaxing this behavior could allow the adaptation mechanism to trade off different factors.

The current implementation takes a "single-shot" approach to adaptation. We plan to study the application of similar strategies in an online adaptation scenario, where the network structure is gradually changed over time.

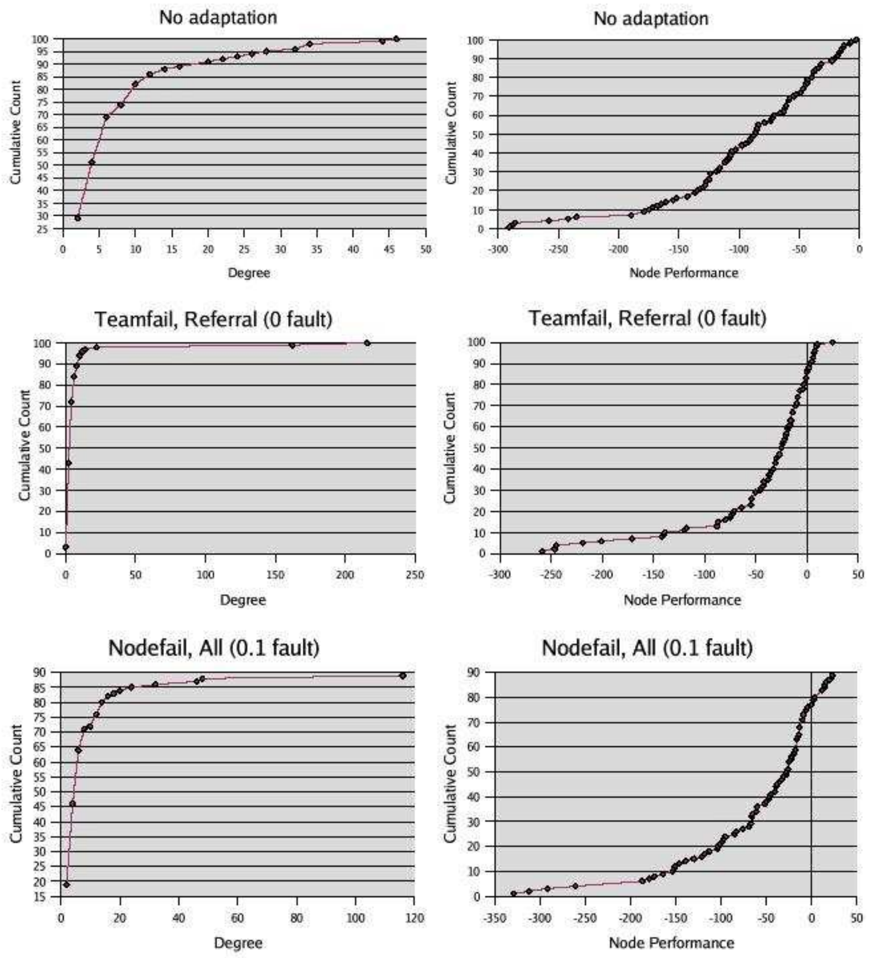
## 7. Conclusions

We have shown that network structures can have a significant effect on organizational team formation in a society of agents. These results support the conclusion that the interaction topology should be a consideration in the design and function of multi-agent systems, especially those that are embedded in large and expanding domains.

Our empirical results showed that organizational efficiency can be improved by allowing agents to adapt their local interactions. Although many avenues remain to be explored, we believe that our initial results point the way to many interesting opportunities for future work in this area.

## References

[1] R. Albert and A.L. Barabási. Statistical mechanics of complex networks. *Review of Modern Physics*, 99(3):7314–7316, May 2002.



[2] Reka Albert, Hawoong Jeong, and Albert-Laszlo Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.

[3] K. Carley. Computational organization science: A new frontier. *Proceedings of the National Academy of Science*, 99(3):7314–7316, May 2002.

[4] Kathleen Carley and Les Gasser. Computational organization theory. In Gerhard Weiss, editor, *Introduction to Multiagent Systems*. Morgan Kaufmann, 1999. Chapter 7.

[5] P. Cohen, H. Levesque, and I. Smith. On team formation. In J. Hintikka and R. Tuomela, editors, *Contemporary Action Theory*. Synthese, 1997.

[6] J. Delgado. Emergence of social conventions in complex networks. *Artificial Intelligence*, 141:171–185, 2002.

[7] Edmund H. Durfee and Victor R. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5):1167–1183, 1991.

[8] P. Erdos and A. Renyi. On random graphs. *Publicationes Mathematicae*, (6):290–297, 1959.

[9] Matthew Gaston and Marie desJardins. Team formation in complex networks. In *Proceedings of the North American Association for Computational Science and Organizational Science (NAACSOS) Conference*.

[10] Natalie S. Glance and Bernardo A. Huberman. Organizational fluidity and sustainable cooperation. In *Computational Organization Theory*. Lawrence Erlbaum Associates, 1994.

[11] James Hendler. Agents and the semantic web. *IEEE Intelligent Systems*, 16(2):30–37, March/April 2001.

[12] Bernardo A. Huberman and Tad Hogg. Communities of practice: Performance and evolution. *Computational and Mathematical Organizational Theory*, 1(1):73–92, 1995.

[13] Mark Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.

[14] Steven Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2002.

[15] Milind Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[16] D. Watts. *Small Worlds*. Princeton University Press, 1999.

[17] D. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[18] Michael Wooldridge and Nicholas Jennings. The cooperative problem solving process. *Journal of Logic and Computation*, 9(4):563–592, 1999.