Simulating Team Formation in Social Networks

Nathaniel Dykhuis
Computer Science
University of Arizona
Tucson, Arizona
Email: ndykhuis@cs.arizona.edu

Paul Cohen
School of Information: Science, Technology, and Arts
University of Arizona
Tucson, Arizona
Email: cohen@sista.arizona.edu

Yu-Han Chang
Information Sciences Institute
University of Southern California
Los Angeles, California
Email: ychang@isi.edu

***Abstract*—This research examines the problem of team formation in social networks. Agents, each possessing certain skills, are given tasks that require particular combinations of skills, and they must form teams to complete the tasks and receive payoffs. However, agents can only join teams to which they have direct connections in the social network. We find that a simple, locally-rational team formation strategy can form team configurations with near-optimal earnings, though this greedy hill-climbing search does converge to suboptimal local maxima. Under this strategy, a variety of random graph topologies not only achieve earnings competitive with complete graphs, but also are much more efficient, achieving these results in less time and with far fewer connections between agents. Several variations were tested; the best results for average earnings and equality occurred when groups were allowed to merge and expel agents, and when groups were fully connected during formation.**

I. Introduction

This research explores how people form teams within social networks. Some jobs require multiple people with varying skills – a team – to complete them. This research looks at how people form teams to complete a job, given the constraint that people can only join a team if they are connected to someone on that team.

In the generalized scenario, jobs are posted that require particular combinations of skills. Agents form teams with their neighbors to perform the jobs. A team of agents receives the payoff for a job if the team has the minimum number of required skills. This scenario relates to team-forming processes in numerous domains of business and academia. For example, research funding follows this pattern: a Request for Proposals requires particular research skills. Researchers form collaborations based on their pre-existing relationships to build teams that can successfully address the Request for Proposals.

We built a simulation to explore how social network structure affects the quality of teams, the speed at which teams are formed, the stability of the teams, and the distribution of wealth among agents. We were particularly interested in the effects of the constraint that agents only join teams with whom they are connected in the social network. Since agents can only interact with a subset of the network, they will miss opportunities to form teams that can perform higher-paying tasks. However, because agents have far fewer potential teammates, the system will converge to a stable configuration of teams much more quickly. The more connections an agent has, the better the possible team configurations will be. Even with a relatively small number of connections, however, the system is predicted to achieve disproportionately high average earnings, due to the small-world structure of the social network.

II. Related Work

Team formation has a sizeable research literature. The problem is commonly addressed using agent-based models. Research in economics deals with small problems of this nature, but for larger numbers of agents or more complex scenarios, the problem quickly becomes intractable for a purely analytical approach.

Aldrich and Kim [1] compare how entrepreneurial teams form in random, small world, and scale-free networks. They propose two strategies for team formation: rational (looking for a good mix of skills) and social (looking for a good mix of personalities). They find that in strongly-clustered topologies, teams often form from partnerships within clusters, rather than between clusters; they do not take advantage of inter-cluster links to find new talent and skills. They also conclude that search for teammates is easier in scale-free or fat-tail networks than in random or small-world graphs. Their work lends insight into the theory of graph topology's effects on team formation, but it lacks quantitative results obtained through simulation.

The Netlogo Team Assembly model [2] [3] [4] describes how teams form in a context of academic collaboration. On each iteration, agents try to form teams of a user-defined size. They must decide how many newcomers to add to the team, or how many incumbents to retain. The model shows that a large connected component of agents often forms. This work deals with a topic similar to ours, but it focuses more on network formation, rather than constraints of an existing network. In addition, our research is also concerned with economic factors.

Taramasco, Cointet, and Roth [5] study similar cases of collaborations in an academic setting. They propose a graph structure based on n-adic interactions, rather than simple

978-0-7695-5137-1/13 $26.00 © 2013 IEEE
DOI 10.1109/SocialCom.2013.42

diadic (two-agent) links. They also study team formation based on previous collaborations and subject matter, as well as the mix of newcomers and veterans on teams. The authors also build a mathematical model, and run simulations based on a dataset of real academic collaborations.

Li, Chang, and Maheswaran [6] describe a system in which agents in a social network are endowed with a set of cards (skills). Agents then "play" these cards on their links to their neighbors. Particular pairings of cards get different rewards. The authors study how various graph topologies and formation strategies influence social welfare and inequality. They find that social welfare increases with number of connections per node, and that the most effective graph topologies are those that take advantage of information about the card endowments of the individual agents.

Gaston, Simmons, and desJardins [7] propose a model very similar to ours, in which a task requires certain skills, agents each have a single skill, and agents can only join adjacent teams. Agents use local information to join teams, such as which teams are near them, and how close to achieving a task each team is. Specifically, an agent will join a team with probability proportional to the percentage of positions on that team that have been filled already (a team that is close to being successful). In this model, teams commit to specific tasks. New tasks appear every time interval, and teams must be formed within a time limit. The authors look extensively at recovery from node failure in the graph structure. Our model is similar, but it focuses on agent strategies based on estimated payoffs, and on economic features of the system, such as average payoff and equality.

III. Team Formation Model

In this simulation, agents attempt to form teams so they can complete tasks to obtain payoffs. A list of tasks is posted, each with its own payoff and required set of skills. Agents, each with their own skills, attempt to form teams that have the skills required for tasks. However, an agent can join a team only if it has a direct social connection to an agent that is already on the team.

This simulation assumes a simple task model. The skills required for a task are completely known, and the payment for the task is all-or-nothing, based on whether a team has the required number of each skill (or more). Teams receive the payoff for the highest-paying task they qualify for, and the payoff is divided evenly among the team members, regardless of each member's skills.[1] The number of tasks of each type is unlimited; any number of teams may receive the same payoff for the same task.

In addition, agent decision-making is simplified to a locally-rational heuristic: On each turn, agents switch to the group that gives them the highest payoff or, if no team receives a non-zero payoff, the team that is closest to receiving a payoff. (Agents do not think ahead about which tasks a team might be able to complete given additional members.) Every agent has this same decision-making strategy; there are no differences between agents aside from their endowed skills and their locations in the social network. Agents also have no team loyalty, aside from staying with the current team rather than switching if payoffs are equal.

Finally, social network structure is assumed to be static, and social networks are randomly generated from well-known small-world models. These small-world graphs reflect some features of real-world networks, but none is entirely realistic.

To set up the model, a set of agents is generated and randomly assigned skills. The random seed is assigned such that the set of agents is the same for every simulation with the same number of agents; this means that any two simulations with the same n should be able to achieve the same average payoff, if topology is ignored. Therefore, any variance in the results on a fully-connected graph is due to randomness in the ordering of application, consideration, and acceptance. Agents are randomly implanted into a social network topology, and each agent begins as its own degenerate group of one.

When the simulation starts, agents and groups begin iterating over an application-admission process whereby agents are allowed to join other groups. This process has three steps:

1) Each agent looks at all of the groups that its neighbors belong to. For each group, the agent calculates what its payoff would be if it joined that group. The agent compares that potential payoff with its current payoff, and "applies" to every neighboring group that will increase its payoff. Similarly, groups look at all of the other adjacent groups and calculate which merge with another group (if any) would benefit its members most.
2) Each group considers all the applications received from agents and groups, and calculates how accepting the application would affect the payoff of its members. The group then accepts the application that increases the individual member payoffs the most (if any such application exists).
3) Each agent (and group) considers the acceptances received (if any), and joins/merges with the group that increases its payoff the most.

Ties are broken by choosing the agent or group that will bring the group closest to achieving a task, and then by the order the applications were processed. The order that applications are processed is shuffled in every iteration; this alone leads to significant variance in the results (as shown in the results for the complete graph, Figure 4).

The simulation ends when one iteration has passed in which no agents change teams (or when an iteration limit

[1] Future work could model negotiations between team members to divide payoffs unequally based on skills.

is exceeded; though the limit is set high enough to never occur). The system is now in a Nash equilibrium: no agent will be accepted to a different group in which its earnings would be higher. Statistics are then calculated and output to a file.

IV. Parameters

The simulation supports numerous parameters to control social network construction, agent generation, and task structure. The major parameters are listed here; numbers in parentheses indicate either the range of values tested, or the valid possible values.

`n` Number of agents (50-150)

`graph_type`
: Topology of graph structure. Most NetworkX topologies are supported [8].

`connections`
: Connections per agent; parameter for small-world networks (2-12)

`prob_rewire`
: Probability of rewiring; parameter for small-world networks (0.05-0.35)

`tasks`
: Task structure - a list of 2-tuples where the first item is a list of the required skills to complete the task, and the second is the payoff for the task. See below for task structures used.

`maxskills`
: Maximum number of skills per agent (1)

The simulation accepts a list of values for each parameter, and runs over all combinations of the parameter values to explore how results change over the parameter space. Statistics are output for each simulation run; these include:

- Number of iterations to convergence
- Number of groups formed
- Number of successful groups
- Number of groups doing each task
- Number of successful agents
- Mean and standard devation of earnings per agent
- Gini coefficient among all agents and among only successful agents
- Average number of group affiliation changes per agent

A. Graph topologies

The simulation has been tested on six primary graph models:

Erdős-Rényi random GNM graphs – these graphs assign a subset of m edges randomly out of the set of all possible edges. They tend to have a large connected component, but low clustering. [9]

Connected Watts-Strogatz small-world graphs – a ring of agents in which every agent is connected to its c nearest neighbors, and then every edge may be rewired to a new, random destination with probability p. These graphs have short average path lengths and high clustering. [10]

Barabási-Albert preferential attachment graphs – as each node is added, it is connected to c other agents, with higher preference to agents of high degree. This creates a power law degree distibution. [11]

Complete graphs – every possible edge is present; every agent is connected to every other agent.

2D grid graphs – a two-dimensional lattice of agents where all agents have 4, 3, or 2 edges, based on whether they are on the interior, border, or corner (respectively)

2D grid graphs with random rewiring – like a grid graph, but on creation, every edge may be rewired with some probability p.

In the simulations, the number of edges in an Erdős-Rényi graph is assigned to be equal to the number of edges a Watts-Strogatz graph will have with the given connections parameter. This allows comparisons between a small-world graph and a random graph with the same number of edges.

Table I
Basic task

Number of each skill				Payoff
3	3	0	0	90
0	3	3	0	150
0	0	3	3	210
4	3	2	1	200
0	0	0	4	160
3	0	3	0	120
2	2	2	2	200

B. Task structures

The simulation results are highly dependent on the task structure. Several task structures have been tested; two of the most informative are presented here. For these tasks, skills were given different values – skill 1 was worth 10, skill 2 worth 20, skill 3 worth 30, and skill 4 worth 40. Payoffs for the tasks were the sum of the values of the required skills. This allowed explorations of efficiency and equality; we can compare the earnings of each agent to the value of its skill.

The first set of tasks was designed so that at least 6 agents are required to receive a payoff (except for task 5, which requires only 4) (Table I). Simulations were also run on this task set with the number of agents per task doubled, to generate even larger groups.

The second task set was designed to study inequality in the society. In this set of tasks, agents can either receive payment in a group that is segregated into a single skill or in a group with a mix of skills (Table II). Because payoff is split evenly among group members, an agent with high-valued skills will receive a higher payoff performing the

Table II
EQUALITY TEST TASK

Number of each skill				Payoff
1	1	1	1	100
2	2	2	2	200
3	3	3	3	300
4	3	2	1	200
4	0	0	0	40
0	4	0	0	80
0	0	4	0	120
0	0	0	4	160

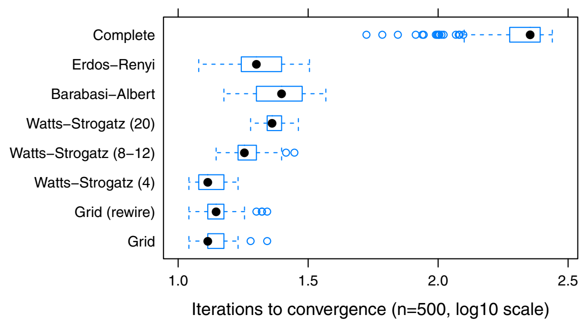

Figure 1. The complete graph takes much longer to converge than the small-world graphs.

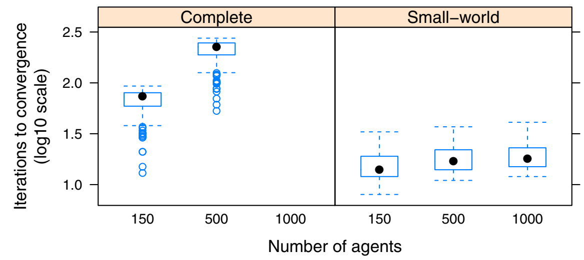

Figure 2. Convergence time grows quickly with number of agents in the complete graph.

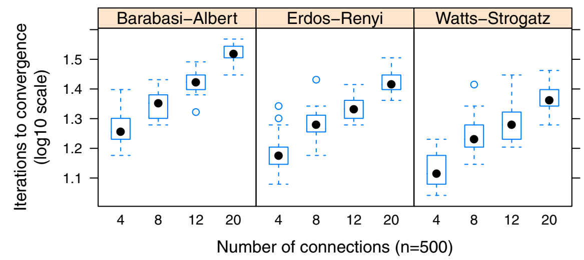

Figure 3. Convergence time depends on the number of local connections in small-world graphs.

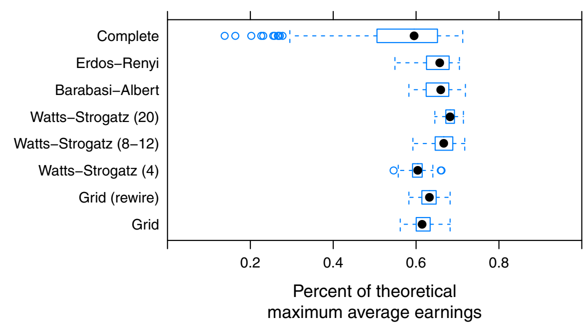

Figure 4. Average earnings as percent of theoretical maximum for various topologies.

segregated tasks, while agents with low-valued skills will prefer the mixed group tasks, where they benefit from the higher-value skills of other team members. The results will examine whether agents with high-value skills segregate into groups that are best for them, and receive higher payoffs, or whether wealth is distributed more equally in mixed groups. The figures in this paper are generated from this task structure.

V. RESULTS

A. Base results

1) Time to convergence: Convergence speed is defined as the number of iterations required before the system enters a state in which no agent wants to switch groups and is able to. As described in section III, on each iteration, every agent could apply to every group it can access, but each group can accept at most one agent or merge proposal.

Convergence time in this simulation is primarily determined by the number of edges in the graph. Therefore, complete graphs take much longer to converge than the other topologies, and convergence time grows with number of agents (Figure 1 and 2). Conversely, for small-world networks, convergence time is determined by the number of local connections per agent (exact, in Watts-Strogatz graphs; average in Erdős-Rényi and Barabási-Albert) (Figure 3).

2) Average Earnings: In the simplified case where agents can form teams with any other agent, by any algorithm, the team formation problem can be represented as an 0-1 knapsack problem: given a fixed number of agents with each skill (volume), how many of each task should be selected (items) in order to maximize average payoff (value)? Therefore, a relatively simple dynamic programming algorithm can be used to find the optimal solution for a given task structure and set of agents. Since the set of agents and task structure is the same for every simulation run of the same n, results from this algorithm will give the theoretical maximum average payoff possible, if the social network and team formation algorithm were not constraints. Simulation results can then be evaluated relative to this maximum.

As shown in Figure 4, all configurations lose about 30% relative to the theoretical maximum as they converge to local

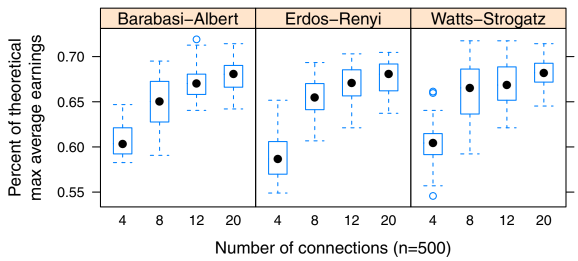

Figure 5. Average earnings increase with number of local connections in small-world graphs.

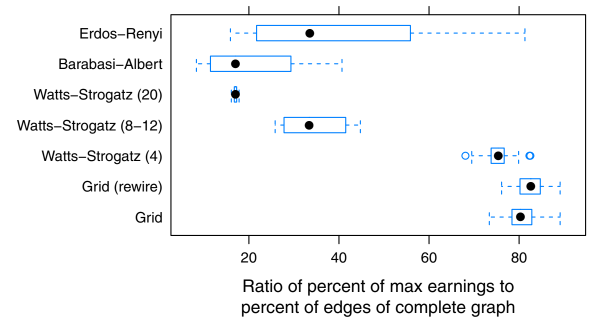

Figure 6. Small world graphs achieve a high percentage of theoretical earnings with a low percentage of connectivity (a high earnings-per-edge ratio).

maxima. Even though the optimal teams should be able to form on the complete graph due to lack of social network constraints, the team formation algorithm is a local, greedy process, and is likely to find sub-optimal solutions. Short convergence time correlates strongly with reduced earnings on the complete graph (see top-left box in Figure 9).

Comparison of the complete graph with other topologies shows that small-world networks are competitive in average earnings, even though the restricted set of edges could prevent optimal solutions from being possible. From the results, it appears that the Watts-Strogatz and grid topologies suffer most from the reduced number of links.

Breaking down results by model parameters reveals that all of the random graphs achieve higher average payoffs with additional local connections (Figure 5). For the Watts-Strogatz graph, the connections parameter refers to the number of connections to adjacent neighbors in the initial ring structure. For the Barabási-Albert graph, this refers to the number of connections made when adding a new node to the graph. The Erdős-Rényi graph does not control local connections, so the results for each connection value are from an Erdős-Rényi graph with number of edges equivalent to a Watts-Strogatz graph of that connection value.

3) Efficiency: The next results deal with the efficiency of the random graphs. Every graph can be thought of as a complete graph with some edges removed. Efficiency is a measure of how much payoff the new graph can achieve with its reduced number of edges; essentially, if some percentage of the edges are removed, by what percentage is the payoff reduced? The next results will deal with this ratio of percent of maximum edges (if the graph were fully connected) to percent of maximum pay (theoretically possible on a complete graph) (Equation 1).

The earlier results have already made clear that random graphs achieved earnings comparable to a complete graph. Even more striking is that random graphs achieve these earnings with less than 15% of the edges of a complete graph, even down to 5% as many edges. Figure 6 shows the ratio of percent of full connectivity to percent of maximum earnings (essentially, the earnings-per-edge ratio), illustrating that random graphs are up to 90 times more efficient with their edges than complete graphs.

$$\text{ratio} = \frac{\%\text{ of earnings}}{\%\text{ of edges}} = \frac{\frac{(\text{earnings})}{(\text{theoretical max earnings})}}{\frac{(\#\text{ of edges})}{(\#\text{ of edges on complete graph})}} \qquad (1)$$

This result is also due to the local structure of the problem. The number of agents required per task does not scale with number of agents total, and since skills are randomly distributed through the population, agents need only the g closest neighbors to them to form a successful group, where g is proportional to the number of agents required to complete a task. Agents do not need additional connections across the network; rather, these connections only slow down the team formation process, and can even lead to reduced earnings (compare with complete graph results in Figure 4).

In addition, the Watts-Strogatz graph and rewired grid graph use a parameter that controls the probability of randomly rewiring each edge during graph formation, but this parameter showed no significant effect on average earnings. This is because skills are uniformly distributed throughout the network, so rewiring an edge from one part of the network to another merely connects it to a region with the same distribution of skills, with no real effect.

4) Equality: Equality in a population can be described by the Gini coefficient [12], as defined in Equation 2, where X_i are the per-agent payoffs sorted in descending order. This function varies from 0 to 1, where 0 indicates total equality, and 1 indicates that a single agent has the entire payoff. This measure will indicate in these simulations whether all agents are receiving a payoff, or whether some are not successful in building a team, as well as whether some teams are paying their agents better than others.

$$G = \frac{N+1}{N-1} - \left(\sum_{i=1}^{n} i \cdot X_i \right) \frac{2}{N(N-1)\bar{X}} \qquad (2)$$

In this simulation, there are two possible factors that influence equality. The first factor is the percent of agents who successfully form teams that receive some payoff. This is the most important factor in both Gini inequality and

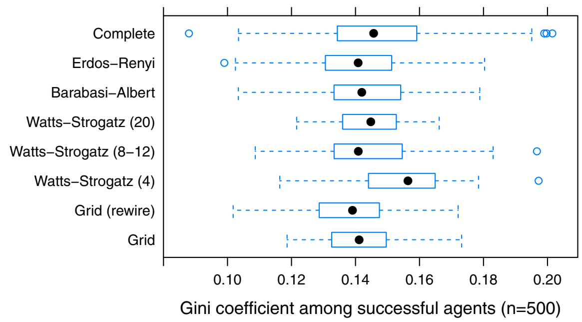

Figure 7. Equality is similar among topologies.

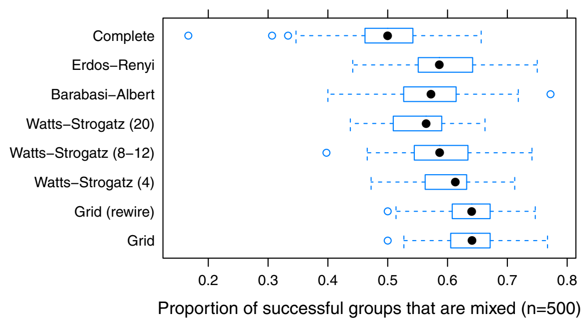

Figure 8. With fewer local connections, agents are more likely to be in a mixed group.

average earnings, as agents receiving zero payoff will skew average earnings and contribute to inequality. The second factor is the Gini inequality among agents that did receive payoffs. This is an important metric for the "equality" task (Table II). If all agents formed mixed groups, Gini inequality should be zero, but if agents segregated into groups by skill, inequality should be much higher, since some skills are worth more than others. Since overall inequality is so tightly correlated with average earnings, and earnings have been discussed already, this section will focus on equality among successful agents.

Figure 7 shows that equality is similar among the topologies, with the complete graph showing the greatest variance. For comparison, the maximum Gini coefficient if all agents are successful and completely segregate into groups by skill is approximately 0.20-0.25. Figure 8 shows the proportion of mixed groups (not segregated by skill). One notable feature of this graph is that, among Watts-Strogatz topologies, the proportion of mixed groups increases as number of local connections decreases. With fewer neighbors to choose from, an agent is more likely to be happy with any group that receives a payoff, even if the agent has a more valuable skill.

5) Summary: Overall, these results show that the greedy team-building search algorithm performs at a large fraction of optimal efficiency, but still converges to suboptimal local maxima. Results for small-world graphs depend mostly

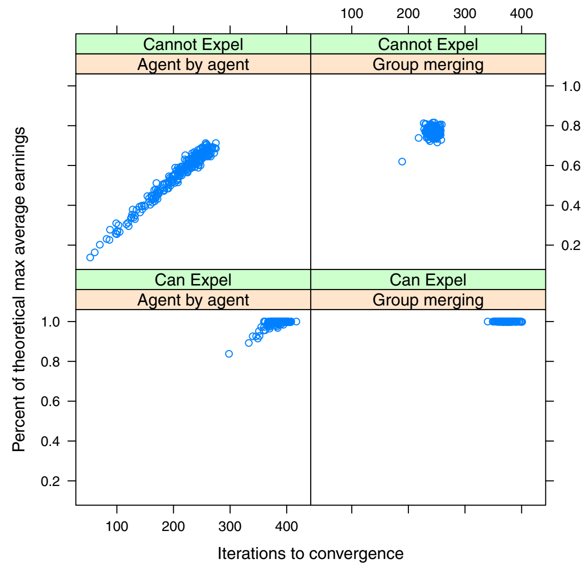

Figure 9. Earnings by iterations on the complete graph. Simulations that converge early reach suboptimal solutions. Group merging reduces variance in convergence time; expelling agents forms better groups.

on local structure, rather than total number of agents. As the number of local connections increases, earnings also increase. Small-world graphs are efficient in that they can achieve a large fraction of the optimal earnings with only a small fraction of the possible edges. There appears to be little significant difference in equality between the topologies, though the percent of mixed groups increases with fewer local connections.

B. Group merging

After exploring the baseline behavior of the simulation, several variations were evaluated for their impact on convergence time, efficiency, and equality. In the first, entire groups are allowed to merge together. At the application phase of the simulation, not only agents, but also entire groups can submit applications to other groups. As before, the other group accepts the application that increased average payoff for all of its members the most.

On a complete graph, when agents join groups individually, convergence time varies greatly. When groups are allowed to merge, convergence takes slightly longer, but the variance is greatly reduced (Figure 9). Convergence time is highly left-skewed for individual joining, with many outliers finishing unusually early. This affects the resulting earnings. Average earnings are highly correlated with number of iterations to convergence; a longer convergence time often results in a better configuration of teams. Joining individually often finds slightly better solutions than merging groups, but it also often converges early to an inferior solution, while merging groups is more consistent. This indicates that the search process of joining individually tends to find worse local

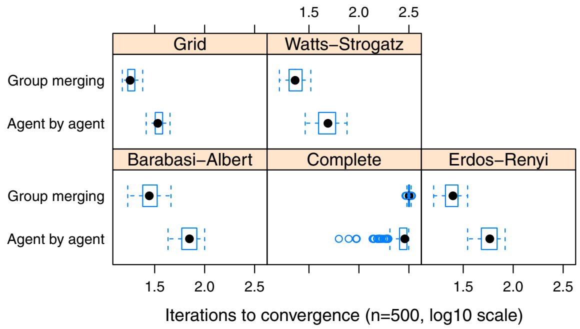

Figure 10. Group merging speeds up convergence and reduces variance (for the complete graph)

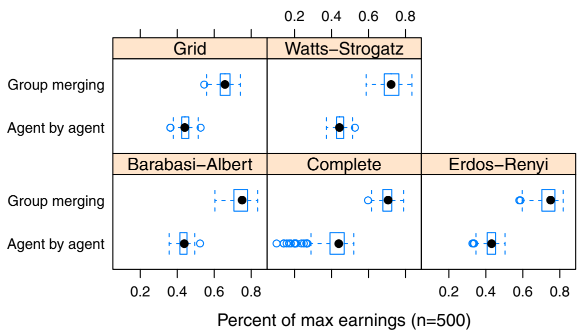

Figure 11. Average earnings are slightly higher with group merging, due to more agents being included in a successful group.

maxima more commonly than merging groups together.

One likely reason for this is that, as soon as a group can successfully complete a task, it has little reason to add more members. In the individual-joining case on the complete graph, groups are able to select specifically the agents that will complete their task, and no more, potentially stranding the remaining agents in groups that are unable to complete any task. In contrast, with group merging, these leftover groups would be able to merge together, allowing them to complete some task at least, even though the individual payoff is lower.

On random graphs, group merging converges slightly faster (Figure 10). This could be simply due to the fact that groups can grow in size more quickly by merging than by adding one agent at a time.

Group merging also leads to higher average earnings on the random graphs (Figure 11), though it does not reduce the variance in earnings and convergence time as significantly as on the complete graph. In addition, Figure 12 shows that group merging reduces inequality; this is due to a larger percentage of the agents being in mixed groups (Figure 13). When groups are allowed to merge, it is more likely that two adjacent groups of two agents will be mixed than segregated. When these groups merge and the resulting group of four receives a payoff, the group will be more likely to stay this way rather than to try to segregate. However, in the agent-

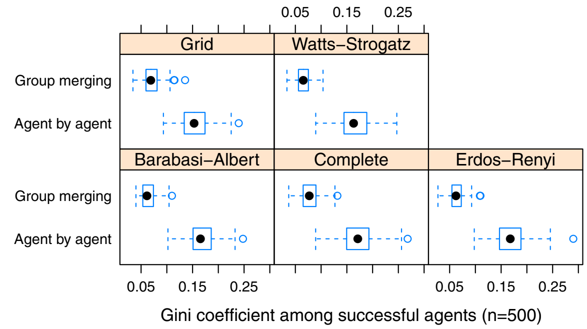

Figure 12. Inequality is reduced among successful agents when teams form by group merging.

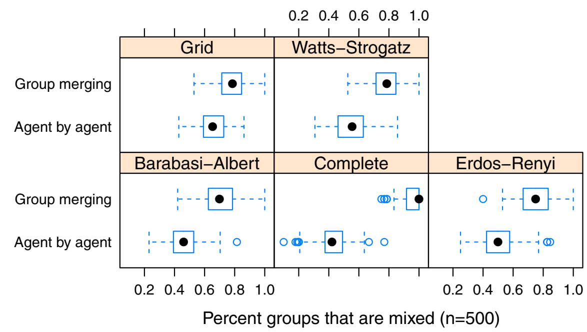

Figure 13. More groups are mixed when teams form by group merging.

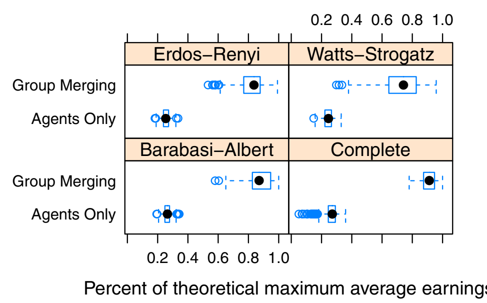

Figure 14. If the task structure is set so a group of twice the size receives four times the pay, group merging achieves much higher earnings.

by-agent case, groups may be more selective about accepting only agents of the same type to achieve a higher payoff.

The behavior of the group-merging team formation strategy was validated on another task designed to show a situation in which group merging is advantageous. The task payoffs were set so groups of size n receive some payoff, but groups of size $2n$ receive four times the payoff (Table III). When agents join individually, they quickly form a small group that receives some payoff. At this point, the group has no incentive to add another member, since it would then need to split the same payoff among more members. However, when groups are allowed to merge, two adjacent groups can join together and receive a superior payoff for all of their members. As predicted, average payoffs were significantly

Table III
GROUP MERGING TEST TASK

Number of each skill				Payoff
1	1	1	1	100
2	2	2	2	400
4	4	4	4	1600
2	2	0	0	60
4	4	0	0	240
0	0	2	2	140
0	0	4	4	560

higher when groups were allowed to merge (Figure 14).

This result further illustrates the dynamics of the search process. Essentially, this team formation strategy is a greedy, hill-climbing search – at each iteration, the agent or group considers all of the possible moves, and chooses the one that increases payoff the most, stopping when no action increases payoff. Group merging is equivalent to taking a larger step size in that it can allow an agent or group to escape one local maximum and jump to a better one. However, the more fine-grained approach of forming teams agent-by-agent avoids adding more agents than necessary, and can therefore sometimes find slightly better solutions.

C. Expelling agents

Group merging tends to distribute the wealth more evenly across agents in the simulation: most agents are able to be part of a successful group. However, the groups are inefficient in that they have more members than necessary to accomplish their respective tasks. The next variation of the simulation, therefore, allows groups to eject members. Each iteration, each group iterates through its members, and analyzes what its payoff would be without each member. If the group can achieve the same payoff without one of its agents, that agent is ejected from the group.

Results on the complete graph are immediately striking (Figure 9). Expelling agents leads to the much higher average earnings, approaching optimal solutions on some topologies. Variance in convergence time and earnings on the agent-by-agent case, though still higher than with group merging, is also greatly reduced. These benefits come with only a modest increase in convergence time.

Expelling agents leads to more efficient teams, since each team will have only exactly as many agents as necessary to receive the payoff (Figure 15). This mechanism helps correct the “hasty mistakes” of the greedy team formation algorithm, as unnecessary agents that cause suboptimal solutions are pruned out. In addition, the expelled agents are then allowed to form new teams, and with more groups receiving payoffs, average earnings increase. However, in random and small-world graphs, many expelled agents are left stranded, unable to connect with other free agents to form a successful team

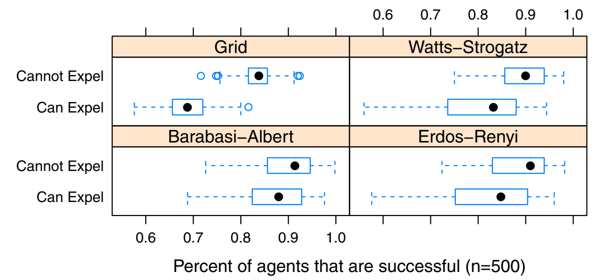

Figure 16. Expelling agents decreases equality in the system, as expelled agents are unable to form successful teams.

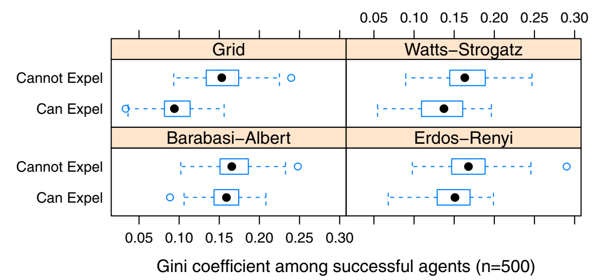

Figure 17. Expelling agents increases equality among successful agents in the agent-only formation model.

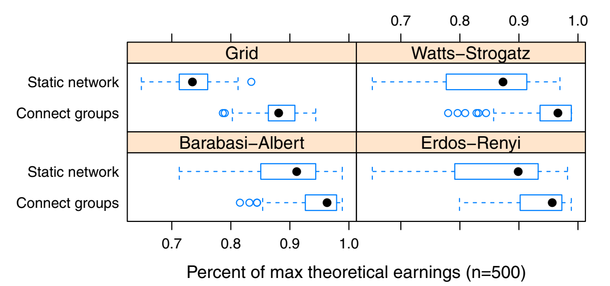

Figure 18. Fully-connecting groups often achieves earnings approaching the optimal solution.

(Figure 16). Therefore, equality globally decreases, even as equality among successful agents increases (Figure 17).

D. Fully-connected groups

The final variation attempted to reduce the number of stranded agents by allowing agents to retain their social connections from previous groups. This scenario models the effect of social networks like LinkedIn, which allow users to retain past social connections that they can later leverage during a job search.

In this variation, when an agent joins a group, it is immediately given direct connections to all of the other agents in the group. When an agent leaves or is ejected from a group, these connections remain. This allows agents that have been expelled to more easily find a new group to join, which affects the overall earnings of the system (Figures 18 and 19). In fact, the combination of group merging, expelling

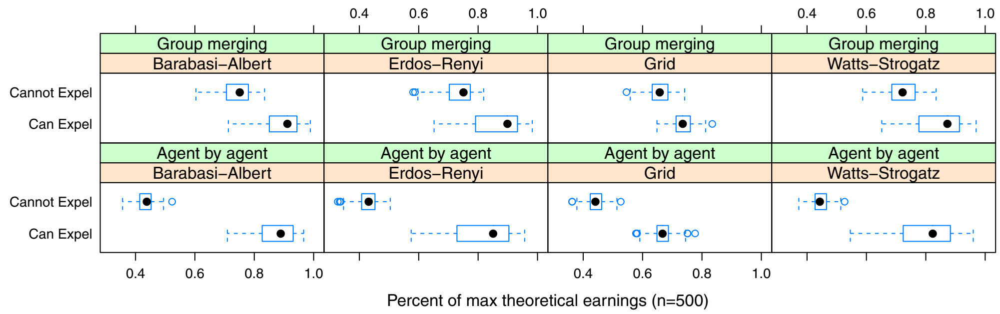

Figure 15. Expelling agents increases efficiency on all topologies.

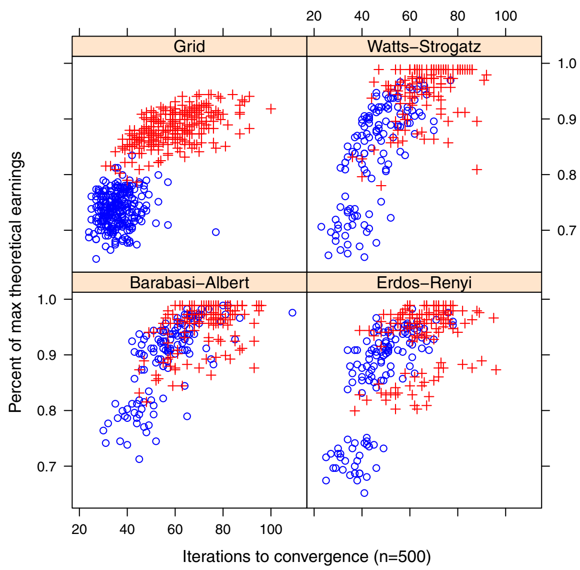

Figure 19. Fully connecting group members increases earnings at the expense of longer convergence time. (+ indicates fully-connecting, o indicates without)

agents, and fully connecting groups generates the highest average earnings of any configuration, even approaching the optimal solution. Almost all of the agents are able to join successful groups (Figure 20). However, inequality also increases among successful agents in this configuration (Figures 21 and 22); it may be that increasing the number of connections allows agents to be more selective in the groups they join and segregate by skill.

VI. Conclusion

This research explored the performance of a simple, agent-based team-formation strategy in a variety of graph topologies and task structures. Even though they do not achieve optimal performance, small-world graph topologies, which reflect characteristics of real social networks, are efficient structures for team formation. Because they have a limited number of local connections, convergence is very fast, but because of the small-world structure, agents have quick access to a majority of the other agents, leading to efficient use of these few connections in finding effective solutions. When groups of agents shortsightedly look to maximize the payoff for agents in the group, as in the individual-joining examples, solutions are often sub-optimal, as the group will stick with a lower-payoff task rather than temporarily reduce their pay by adding more agents to pursue a more valuable task. However, when groups are willing to look ahead and cooperate, they are able to merge to complete tasks they could not handle on their own. The overall outcome is better when every agent has a job, even if some of the teams have more members than necessary. To reduce the inefficency caused by group merging, groups can expel members. However, in this situation, equality drops, as expelled agents are often unable to connect with a new successful group. This problem can be alleviated by forming connections between agents and their previous team members so that agents can more easily find a new team after being expelled.

Future work will look at the effects of multiple agent personalities, where not every agent follows the same strategy in applying to groups and considering acceptances. Other modifications could include cases where payoff is not split evenly between group members, and simulations where multiple tasks are offered sequentially, and only one group is awarded the payoff for each task. Also, instead of using random graphs, the simulation could form the graph structure based on the skills of the agents, as in [6].

References

[1] H. E. Aldrich and P. H. Kim, "Small worlds, infinite possibilities? how social networks affect entrepreneurial team formation and search," *Strategic Entrepreneurship Journal*, vol. 1, no. 1-2, pp. 147–165, 2007.

[2] E. Bakshy and U. Wilensky, "Netlogo team assembly model," *Center for Connected Learning and Computer-Based Modeling. Northwestern University, Evanston, IL*, 2007, http://ccl.northwestern.edu/netlogo/models/TeamAssembly.

[3] U. Wilensky, "Netlogo," *Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL*, 1999, http://ccl.northwestern.edu/netlogo/.

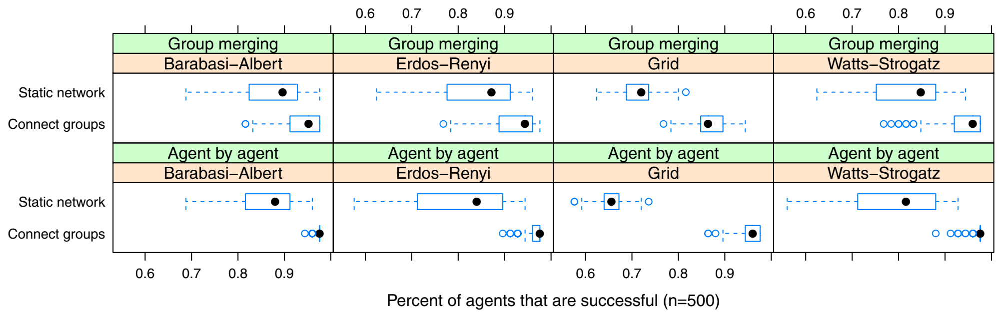

Figure 20. Fully connecting group members enables more agents to find a successful group.

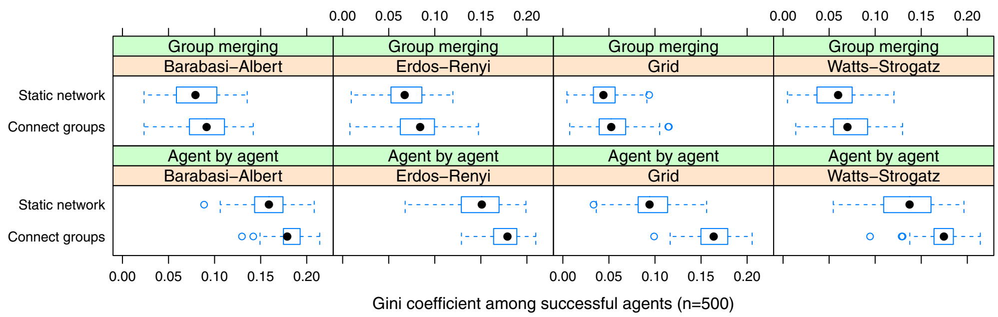

Figure 21. Fully connecting group members reduces equality among successful agents.

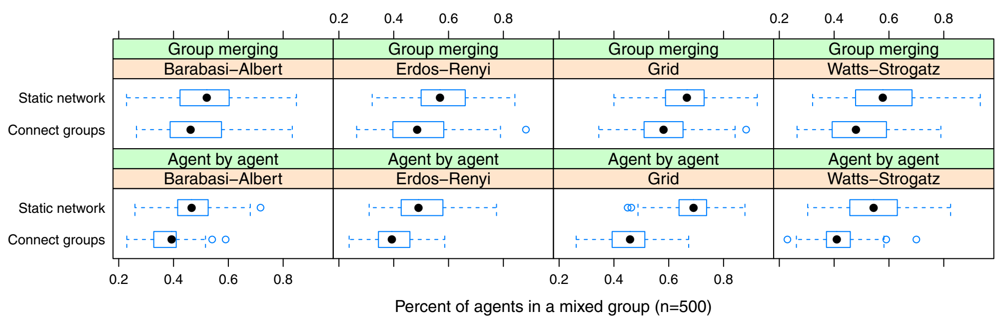

Figure 22. Fewer agents are in mixed groups when groups are fully connected.

[4] R. Guimera, B. Uzzi, J. Spiro, and L. A. N. Amaral, "Team assembly mechanisms determine collaboration network structure and team performance," *Science*, vol. 308, no. 5722, pp. 697–702, 2005.

[5] C. Taramasco, J.-P. Cointet, and C. Roth, "Academic team formation as evolving hypergraphs," *Scientometrics*, vol. 85, no. 3, pp. 721–740, 2010.

[6] Z. Li, Y.-H. Chang, and R. Maheswaran, "Graph formation effects on social welfare and inequality in a networked resource game," in *Social Computing, Behavioral-Cultural Modeling and Prediction*. Springer, 2013, pp. 221–230.

[7] M. Gaston, J. Simmons, and M. DesJardins, "Adapting network structures for efficient team formation," in *Proceedings of the AAAI 2004 Fall Symposium on Artificial Multi-agent Learning*, 2004.

[8] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference (SciPy2008)*, Pasadena, CA USA, Aug. 2008, pp. 11–15.

[9] P. Erdős and A. Rényi, "On the evolution of random graphs," *Magyar Tud. Akad. Mat. Kutató Int. Közl*, vol. 5, pp. 17–61, 1960.

[10] D. Watts and S. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[11] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[12] C. Gini, "Variabilità e mutabilità," *Reprinted in Memorie di metodologica statistica (Ed. Pizetti E, Salvemini, T). Rome: Libreria Eredi Virgilio Veschi*, vol. 1, 1912.