Stable Team Formation Among Self-Interested Agents

Priyang Rathod **Marie desJardins**
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
1000 Hilltop Circle, Baltimore MD 21250
{prathod1,mariedj}@cs.umbc.edu

Abstract

Autonomous agents in a multi-agent system must sometimes collaborate with other agents in order to perform complex tasks, regardless of whether they are inherently self-interested or cooperative. However, the nature of such a collaboration may vary in duration, from a single-task short-term coalition to a stable contractual alliance. The real world is replete with instances of both types of organizations.

We present a multi-agent environment with self-interested agents whose aim is to perform tasks by forming simple teams with other agents. Team leaders recruit agents to join teams by proposing contracts that allocate various payoff shares to the team members. Contracts can be long-term (indefinite-length commitment with a buy-out penalty for leaving the team) or dynamic (commits the joining agent to the team only for a single task). We compare several alternative strategies for forming stable teams empirically and show that in the proposed environment, stable teams have greater organizational efficiency than dynamic teams. We also identify specific conditions and strategies for which stable teams have an advantage over dynamic ones.

Keywords Multi-agent Systems, Long-term Team Formation, Stable Teams

Introduction

Autonomous agents, like humans, are social entities. Even in relatively simple environments, they will often need to collaborate with other agents. This may be because of the distributed nature of the problem to be solved or the limitations of agents. The need for collaboration holds true for self-interested as well as cooperative agents. Human society is a perfect example of such a system. Humans can not survive on their own and need the help of other humans to achieve many of their goals. Completion of these goals may benefi t the entire population, as well as directly compensates those individuals who contributed. Regardless of a human agent's personal inclination to altruism, there is an incentive to collaborate on joint goals with others.

The analogy can be carried over to a multi-agent system. A number of researchers have looked at the problem of dynamically forming teams to perform tasks, where the skills of multiple agents are required for task completion. However, this research has largely been focused on cooperative agents. The research on coalition formation for self-interested agents generally focuses on the problem of forming short-term "one-shot" teams for individual tasks. In human society, by contrast, agents frequently form long-term alliances rather than short-term coalitions. This may be partly because humans tend to prefer working with people they know, but collaborating for longer period also infuses necessary elements like stability, trust and security into the alliance.

Intuition tells us that if it works for the human society then it should work for multi-agent systems as well. We analyze a simple multi-agent environment with a set of agents having different capabilities, in which tasks are introduced at regular intervals. Most of the tasks require more than one agent's capabilities for completion. Agents can either form short-term ("dynamic") teams for the duration of a single task or long-term ("stable") teams to work together for longer periods of time. We investigate different strategies for each and compare their performances empirically, concluding that even without an explicit element of trust, when competing against each other, stable teams are more successful than dynamic teams.

Related Work

The problem of forming teams in a multi-agent system has been studied by many researchers emphasizing different aspects of the problem. Task allocation, resource allocation, team formation, coalition formation, and dynamic organization all provide different perspectives on the same basic problem, and provide a variety of techniques for assigning roles to agents for individual tasks. Very little of this work has explored the need for, or advantages of, stable teams. Our work is in a sense complementary to this body of literature: Any of these techniques for allocating roles to agents could in principle be integrated with our contractual model for forming stable teams. Detailed below are some of the different approaches used for solving the team-formation problem. Unless otherwise specifi ed, work described here is all designed to form short-term dynamic teams for individual tasks that are dissolved upon task completion.

Dynamic Teams. Gerkey and Matarić (2003) define the problem of task allocation as assigning tasks to agents while taking environmental constraints into consideration. By contrast, Chavez et al. (1997) model the problem of assigning processes to machines as a resource allocation problem, where machine agents are treated as resources, and the aim is to find an assignment of resources to processes. Tambe and his team have looked at the problem of team formation in complex, dynamic multiagent domains that include uncertainty, incomplete information and the possibility of agent failure (Tambe 1998), (Tambe 1997a), (Tambe 1997b).

Recently, researchers have modeled the team formation problem as a Distributed Constraint Satisfaction Problem (DCSP) (Yokoo *et al.* 1998; Yokoo & Hirayama 2000). However, DCSP fails to capture the rapidly changing environment in a dynamic multi-agent system; to address this situation, the notion of dynamic DCSP (Niemelä 1999) was introduced. Modi et al. (2001) have proposed a dynamic DCSP approach to resource allocation using the Asynchronous Weak Commitment (AWC) algorithm. In this approach, the set of constraints (roles to be filled for each task) to be satisfied are allowed to be dynamic. Their algorithm addresses a subset of dynamic DCSPs in which only the local constraints are allowed to be dynamic.

Researchers have also used market-based approaches based on methods from economics. In voting, the solution is determined from inputs taken from all the agents. Different auction mechanisms, such as sequential auctions (Boutilier, Goldszmidt, & Sabata 1999) and simultaneous auctions (Greenwald & Boyan 2001), have also been applied to this problem. Sandholm (2002) discusses methods for determining optimal winners in combinatorial auctions where agents can bid for more than one items and their valuations are different for different combinations of items. Contract nets (Huhns & Stephens 2001) have also been used to allocate tasks to contractor agents who bid for these tasks. The contractor can recruit other agents to complete the task and pay them for their services.

Wellman (1995; 1996) and Gerkey and Matarić (2002) survey and analyze methods for creating market-oriented multi-agent systems. The market approach defines costs of performing tasks and revenue earned by agents. This gives self-interested agents an incentive to complete tasks, increasing their revenue and at the same time benefiting the system. Wellman (1998) also emphasizes the necessity of market-aware agents in a multi-agent world and explains how price systems facilitate decentralized decision making. Huberman and Hogg (1995) present a model of interactions among agents and their dynamical effects on the structure and performance of the community.

Stable Teams. There has been some work on long-term teams in the computational organization theory literature. Axtell (Axtell 1999) presents a microeconomic model in which heterogeneous agents—with different preferences for effort and leisure—form firms. Agents can leave these firms or start one of their own when they think it is beneficial for them to do so. The incentive for agents to be in a firm and to contribute to the firm is the production of output, which is divided equally among agents, no matter how much effort the individual group members apply. However, as firm sizes grow, agents have little incentive to apply effort as their share in the output becomes relatively insensitive to their input. This gives rise to free-riding agents who do not contribute any effort, which in turn induces hard-working agents to leave the firm. The dynamics and distribution of firm sizes, productivity, and income are studied empirically.

However, in Axtell's framework, there is no model of tasks to be performed and it uses a very simple method for distribution of pay-off to firm members. By contrast, our contract-based model allocates profit share to agents based on both membership and contribution to tasks, balancing the need for stability with a recognition that some agents are more valuable to the team than others. Additionally, our model provides a richer environment in which the aim is to complete tasks introduced into the system at regular intervals and where team leaders must compete for member agents. We present different strategies for team formation with different inclinations for taking risks while bidding for tasks. We also study systems that include agents following different strategies and compare the performance of the different types of agents in the competitive scenario.

System Architecture

Our multi-agent system is set up with the aim of understanding and analyzing the performance of the system and its constituent agents as a result of simple team formation activities. In particular, our goal is to study the effect of forming stable or dynamic teams, and not the process of team formation itself. Therefore, our environment employs very elementary agent and task models, that are simple enough so that they do not become a digression and at the same time are generic enough to be practical and applicable to a wide range of domains.

The environment consists of a set of N agents, $A_i (i = 1, \ldots, N)$. The number of agents in the system is fixed: agents cannot leave the system, and no new agents enter the system over the course of an experiment. The skill set is defined by a set of capabilities, $C_j (j = 1, \ldots, M)$. Every agent has exactly one of the capabilities, and can participate in any task that requires that capability. Capability assignment is random, with uniform distribution, and is also permanent: i.e., agents can not discard their capability or learn new capabilities in their lifetime. At any given time, an agent is in one of three modes: Unemployed, Team-leader or Team-member.

Tasks are introduced into the system at a regular interval, t_{arr}. Each task has a required skill set associated with it. Only a set of agents with the required skills can perform the task. Each task has an associated size (number of agents required), t_s, which is selected from a uniform distribution over the range $1, \ldots, t_{s_{max}}$. Tasks have expiration times (t_{exp}, which is fixed for each experiment), and have different lengths t_l (in terms of time), again uniformly selected from $1, \ldots, t_{l_{max}}$. Team members must work on (be solely committed to) the task for its entire duration in order for it to be successfully completed. Each task has a fixed pay-off

t_{pay}, which is awarded to the agent or team that completes the task. If a team fails to complete a task that has been assigned to it before the task expires, it must pay a non-completion penalty to the controller.

The pay-off is distributed to the agents in the form of shares of the team which can be "cashed in" when the team is dissolved or the agent leaves the team. The agents are bound by an agreement when they join a team. The agreement delineates the pay-off the agent receives while it is a member of the team and the penalty to be paid to the team if it ever decides to leave. Specifically, it contains the following information:

1. Joining Shares (S_{join}) – This is the "sign-on bonus" for joining the team.
2. Commission (S_{comm}) – This is the payout for each task completed by the team, when the agent directly participates in the task.
3. Dividend (S_{div}) – This is what sets the stable teams apart from dynamic teams. The dividend is the number of shares that the agent receives for each task completed by the team, when the agent does not actually participate in that task. Its value is set lower than S_{comm} in order to differentially reward the agents who actually work on a given task.
4. Current Share Price (P_s) – The share price represents the current total value of a team; it is calculated by dividing the total revenue earned by the number of currently outstanding shares of the team.
5. Penalty (p) – This is the amount that the agent has to pay to the team if it decides to break the contract and leave the team.

The result of this simple scheme is that agents have an incentive to be on a stable team, since they can earn a pay-off even if they do not actually perform any task. However, since $S_{comm} > S_{div}$, hard-working and rare-skilled agents are appropriately rewarded compared to others. As the team grows, however, this gap starts to diminish, and the agents may start leaving for greener pastures.

The system has a central controller agent whose job is to introduce tasks into the system. The controller models the interface of the multi-agent system with the external world. When a task is created, the controller broadcasts the availability of the task to the agents in the system, puts the task in a queue of active tasks, and waits for bids. Only team leaders can bid on tasks. If more than one team leader bids on a task, the controller assigns the task randomly to one of the teams. When a task expires, it is discarded from the queue if it has not been assigned.

There are many extensions that could be made to the model, including allowing agents to leave and join the environment, allocating multiple capabilities to agents, allowing agents to acquire new capabilities over time, having a non-uniform distribution of capabilities among the agents and in the task generator. However, this basic model captures the fundamental dynamics of the agent community that we are interested in, and allows us to perform experiments that are focused only on the stability of agent teams.

Communication

Agents need to communicate in order to be able to form teams and to keep the system updated about changes. However, they must abide by the following constraints:

1. A team leader can communicate with everyone: its own team members, any other team leader, any unemployed agent or agents that are members of other teams. This is analogous to a real-world organization, in which the leader of the organization can typically communicate with its own employees as well as other leaders, and can also learn about other organizations' employees, so that a skilled employee can be enticed into joining his own organization.
2. An unemployed agent can communicate with any team leader, allowing leaders to proactively recruit for their teams and unemployed agents to apply for job positions.
3. A team member who is not a leader can communicate with any team leader. This interaction can result in self-interested, non-stable behaviors where agents put their own interests ahead of their teams' and are willing to change teams if another team offers them a better deal.
4. All agents can communicate with the controller agent.

Strategies

In this section, we describe several strategies for forming teams; the strategies differ in how teams are formed and in how tasks are bid for.

Dynamic Team Strategy

The first strategy is the base case, in which agents form short-term teams for the duration of the task. Initially, all agents are unemployed. When a task is introduced to the system, it is broadcast to all agents; the agents who possess one of the necessary capabilities bid for the task. The task is then awarded to one of the bidders randomly.

Once the task is awarded to an agent, it has to form a team by recruiting all the required agents and start working on the task before it expires. On successful completion of the task, the task pay-off is awarded to the team, which is then equally distributed among the members. The team is dissolved and all the agents become unemployed again, ready to bid for tasks and join other teams. In the event that the task expires before it is started, the team is not awarded any pay-off, the leader is made to pay a non-completion penalty, and the team is dissolved. The agents can be either risk-taking (bid for every task) or risk-averse (bid only for tasks that can be completed with the agents available in the system).

Stable Team Strategies

We have developed a number of alternative strategies for stable team leaders to recruit teams and bid on tasks. The strategies are summarized in this section. In our preliminary results, we found that the strategy used did not have a significant impact on team performance in the environments we were studying. Therefore, for the experiments described

later in the paper, we selected the moderate risk-taking strategy. In future work, we plan to investigate alternative strategies in more depth.

In all of the stable team strategies, the agents try to form teams to complete tasks, with the goal of maximizing team profit. They can recruit agents from the pool of unemployed agents or from other teams (except in Naive, Cautious Strategy).

Each joining member is given shares of the team for becoming a part of the team, and is awarded a commission and dividend for every task completed by the team. When the team is dissolved, every member is paid its share of the team revenue.

Unemployed agents wait for some period of time to be recruited by teams, simultaneously monitoring the system's performance. When an agent has not been recruited for a predefined length of time and the system performance falls below a threshold, it starts a team and becomes a team leader.

Unemployed agents always act to maximize their pay-off, following a simple heuristic for doing so. If they are offered memberships from multiple teams, they choose the one that offers the maximum average utility. This utility is calculated based on the commission (S_{comm}) and dividend (S_{div}) being offered and the current share price (P_s) of that team.

$$average\ utility = (\frac{S_{comm} + S_{div}}{2})P_s$$

A more sophisticated method would take into consideration the growing size of the team, predicting the future value of the shares of the team.

Naive, Cautious Strategy In this strategy, team leaders are very cautious when it comes to bidding for tasks, bidding only on those tasks for which there are agents free in the team at that particular moment. This obviates the possibility of the team having to pay any non-completion penalties. The team is dissolved when the performance of the team relative to other teams falls below threshold.

When agents start a team, they start recruiting proactively for as many different capabilities as possible. Then they stop recruiting, and start bidding for and performing tasks, monitoring their performance regularly. If their performance falls below a threshold and there are unemployed agents available to be recruited, they start recruiting again. In this case, they try to recruit agents with capabilities that their team is lacking. If the team already has all the capabilities, it randomly selects a capability and tries to recruit an agent with that capability.

Team leaders are not allowed to recruit agents that already belong to another team. Also, team members are loyal to their teams and will stay with the team through "good times and bad." Once they join a team, the only way they will become unemployed again is if the team is dissolved by the leader.

Intelligent Recruiting Strategy Team leaders, when recruiting reactively, select the agents to be recruited more intelligently. They first try to recruit an agent with the capability the team does not possess. If all capabilities are present in the team, a capability is selected randomly and agents try to recruit an agent with that capability, using the following method:

1. If there are unemployed agents, try to recruit a skill that the team does not possess.
2. Otherwise, if there are employed agents, try to recruit a new skill from employed agents.
3. Otherwise, try to recruit a random skill from the pool of unemployed agents.
4. Otherwise, try to recruit a random skill from the employed agents.

To avoid trying to recruit forever, teams try to recruit agents a fixed number of times (five in this case) and on failure stop recruiting. On success, the count is reset to 0.

Moderate Risk-Taking Bidding Team Strategy The only difference between this and the previous strategy is that the team leaders are more risk-taking as far as bidding is concerned. They bid for a task if the team has the agents to perform the task, regardless of whether the agents are free or performing some other task. This leads to the possibility of the team being unable to complete the task in time, risking a non-completion penalty.

Risk-Taking Bidding Team Strategy The difference between this and the Moderate Risk-Taking Strategy is that agents following this strategy bid for tasks even if they don't have all the necessary team members to complete them. They recruit the required agents once the task has been awarded to them, further increasing the possibility of the team not being able to complete the task and paying a penalty. In some ways this strategy is a mix of stable long-term strategy and dynamic short-term strategy.

Specialized Team Strategy In our experiments, the performance of the previous strategies seems to indicate that good teams usually end up having a diverse set of capabilities. As a result, some of the tasks that require many agents with the same skill are never completed. Although the number of such tasks was small, we developed a "specialized-team" strategy in order to test whether there is any advantage in having some team leaders form teams of agents who all have the same skill, only hiring other agents as and when needed.

1. Recruit as many same-skill agents as possible.
2. Bid for all tasks. The bid is awarded to the team that fulfills the maximum number of requirements (i.e., that already has the most agents matching the necessary capabilities).
3. Temporarily hire the remaining agents from other teams, paying each agent's team a "consulting fee" f, equal to the task payoff, divided by the total number of agents needed. The consulting fee is allocated as follows:
   - $f/2$ to the team's "bank account."
   - $f/4$ directly to the agent that was hired.
   - $f/4$ directly to the team leader.
4. When the task is completed, free the agent.

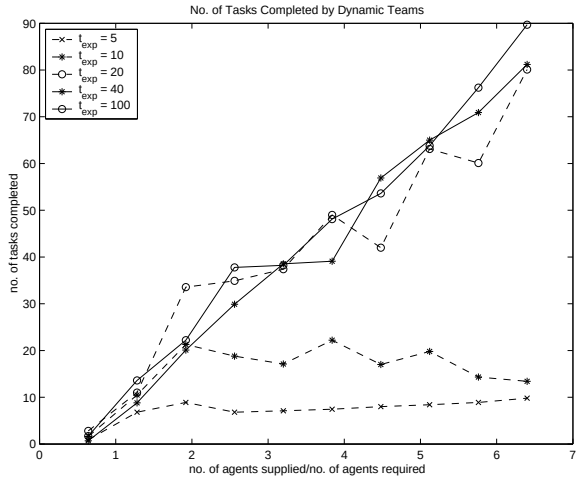

Figure 1: Tasks Completed by Dynamic Teams for $t_{exp} = 5 \ldots 100$

5. If the team fails to complete the task, the bidding team has to pay a non-completion penalty, but the hired agent team still receives the consulting fee.

Experiments and Results

We ran a series of experiments with a simulated multi-agent system consisting of a mixture of dynamic and stable agents. Only when acting as team leaders do the agents behave differently: that is, dynamic agents will only form dynamic teams, and stable agents will only form stable teams, but each agent is free to join either type of team.

Tests were run varying the size of the system in ten increments, starting with four agents and going up to 40 agents, each time increasing the size by four agents. The set of capabilities (C_j=5), maximum task size ($t_{s_{max}}$=4), maximum task length ($t_{l_{max}}$=4) and task frequency (t_{arr}=1) were fixed for all sets of experiments. All the tasks carry the same payoff (t_{pay}=10) and non-completion penalty(same as t_{pay}) irrespective of their size or length.

In the first set of experiments, we studied the effect of varying task expiration times (t_{exp}) on the performance of agents. The intuition behind this was that as the time needed for agents to recruit members decreases, more tasks assigned to dynamic agents would fail to be completed, and they would end up losing revenue in the form of non-completion penalties. The setup consisted of equal numbers of agents of each type in the same environment, competing for tasks.

Figures 1 and 2 show the number of tasks completed by dynamic and stable agents respectively for different task expiration times in a typical setup. One can see that stable teams are able to complete far more tasks than dynamic teams (almost ten times as many). The two dominant factors responsible for this are the scarcity of recruitable agents and the failure of dynamic team leaders to form teams before the tasks expire. We can also see that as t_{exp} decreases, the performance of dynamic teams starts to degrade but at the same time, increasing the task expiration time beyond

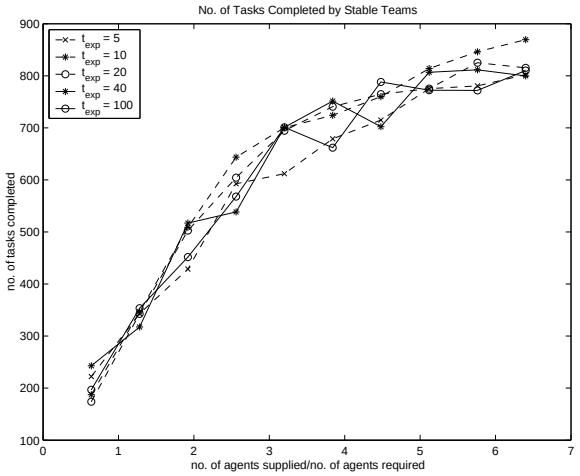

Figure 2: Tasks Completed by Stable Teams for $t_{exp} = 5 \ldots 100$

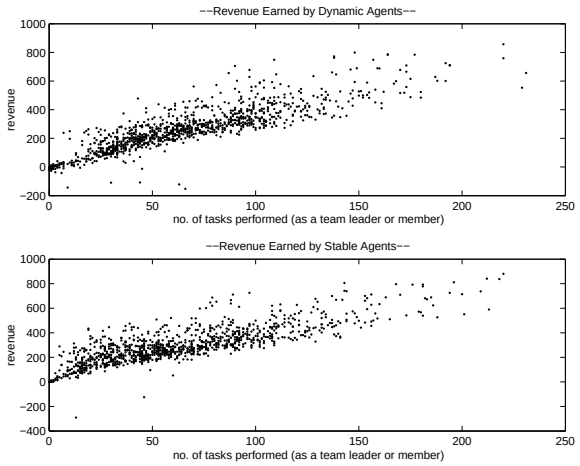

Figure 3: Revenue Earned vs. Tasks Completed

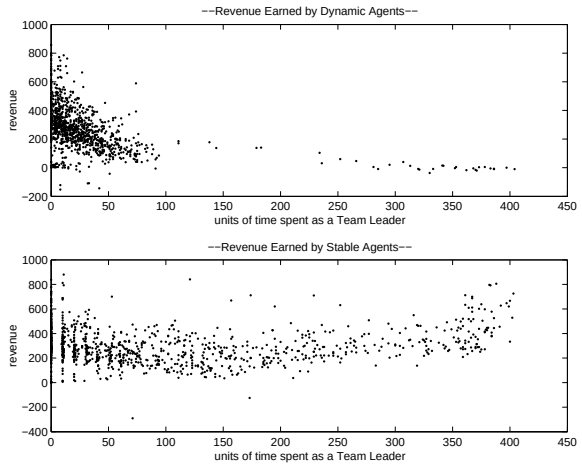

Figure 4: Revenue Earned vs. Team Leading Initiative

a limit (t_{exp}=20) does not have much effect on the performance. On the other hand, stable teams are not affected much by the change, since they do not generally need to recruit new agents to perform a task.

Figure 3 compares the revenue earned by the two different types of agents against the number of tasks completed (as a team leader or a team member). There is no difference in agent earnings because agents are not restricted to joining one of the two types of teams —i.e., a short-term team forming agent can join a stable team and earn revenue.

This result prompted us to investigate further to find out whether agents who take the initiative to lead a team are rewarded accordingly. As shown in Figure 4, the dynamic agents that were highly entrepreneurial (i.e., initiated many teams) had the lowest earnings. In fact, there were only a handful of such agents. Most of the agents were content with leading teams for a short period of time and mostly joining other teams to earn revenue. By contrast, stable team leaders are uniformly rewarded irrespective of the length of time spent as team leaders.

We also compared the performance of the system for different proportions of agent types. In this case, the task expiration time was fixed and the system was run for different proportions of stable team forming (N_s) and dynamic team forming agents (N_d). The dynamic teams are able to complete more tasks as their proportion in the population increases but the difference is not really significant among cases where there are some stable agents present in the system (Figure 5). But when all of the stable team forming agents are removed, there is a noticeable improvement in the performance of dynamic agents. From this we can conclude that dynamic teams are not able to compete with stable teams, even if there are a very few stable team forming agents. Therefore, in environments that contain any stable agents, other team leaders will be predisposed to form stable, rather than dynamic teams.

On the other hand, there is no significant difference in the performance of stable team forming agents when their proportion in the population is changed (Figure 6). The

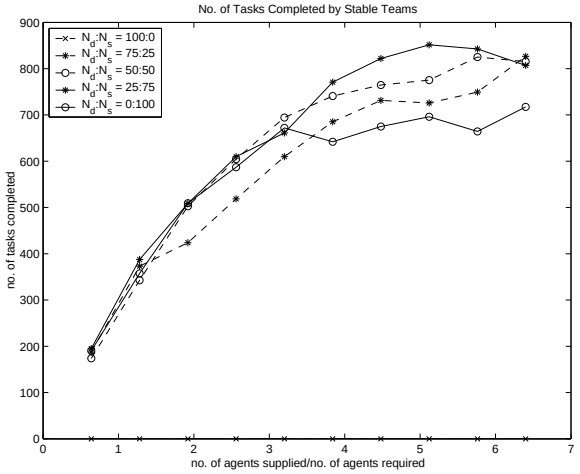

Figure 5: Tasks Completed by Dynamic Teams for Different Agent Proportions

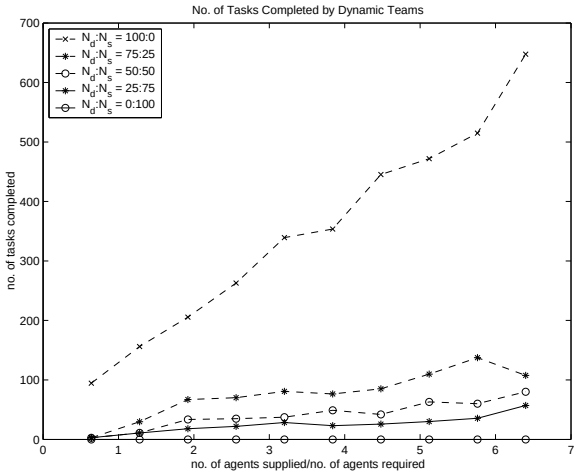

Figure 6: Tasks Completed by Stable Teams for Different Agent Proportions

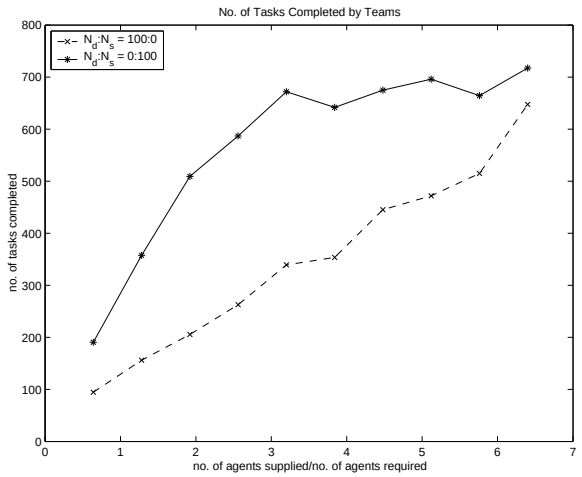

Figure 7: Tasks Completed in Environments with All Dynamic Team Agents vs. All Stable Team Agents

small decrease in their performance when the proportion is increased is due to the decrease in the size of the pool of free dynamic agents that can be hired by stable teams. Despite this effect, Figure 7 clearly shows that if the population contains only one of these two types of agents, stable teams are able to complete many more tasks than dynamic teams.

Future Work

We have used a very simple model of team formation in our framework, with relatively straightforward strategies for joining teams and bidding for tasks. We would like to simulate the behavior of the system with a more sophisticated set of strategies. In particular, we are interested in studying how models of trust and reputation would affect the stability and performance of the system. Agents can be allowed to enter or leave the system, adding uncertainty to the environment.

In the current set of strategies, team recruitment offers are evaluated by potential team members based on the current value of the pay-off. Agents could potentially use learning strategies to predict the future value of a team share when selecting from among multiple offers, or in deciding whether to start or leave a team.

One could study a system with a richer capability model with non-uniform distribution of skills. Agents could have more than one capability, which would make the task bidding strategy for team leaders more interesting. In the current model, agents are assigned to a task by performing a simple capability match. We could make this process more realistic by incorporating factors like experience and proficiency. We could take this one step further and create ontologies of tasks and capabilities. When coupled with a measure of the quality of task performance and a proportionate reward, such models would result in a complex but realistic environment.

We would also like to investigate the effect on system performance of varying task features such as:

- *Task Structure.* In the current environment, a small task requiring only one skill is as likely to occur as a large task requiring several different capabilities. If this restriction is removed, agents may adapt to the actual task distribution, or find a niche and specialize in it.
- *Pay-off.* What happens if rather than all tasks carrying the same pay-off, the pay-off depended on the size of the task or the rarity of the skills required by the task?
- *Task dependencies.* How would the system be affected if tasks were dependent on each other, i.e., if completion of one task was a prerequisite for beginning another?

In our multi-agent system, we haven't experimented with different parameters of the share model. Changing the values of the commission, dividend and penalty could induce interesting agent reactions.

Finally, our long-term goal is to let the agents determine their optimal strategy dynamically, based upon the current situation. A lack of competition may trigger risk-averse behavior, whereas an abundance of agents of a particular type may encourage an agent to form a specialized team. To adapt successfully would require the agents to have a lot of knowledge about the system and a sophisticated algorithm to analyze the effects of not only their own actions but also the actions of other agents' opportunistic actions.

Conclusions

We have presented a multi-agent environment in which self-interested agents can form stable or dynamic teams to complete tasks. From our experiments in simulated multi-agent environments, we conclude that stable teams are able to complete significantly more tasks than dynamic teams, especially in time-critical domains in which the tasks must be performed quickly. Such a competitive environment does not favor entrepreneurial dynamic agents: in fact, they end up being among the lowest-earning agents. The addition of even a small number of stable team forming agents drastically reduces the performance of dynamic teams. Finally, even in the absence of competition from other type of agents, stable teams are more beneficial to the society, since they are more productive.

References

Axtell, R. 1999. The emergence of firms in a population of agents: Local increasing returns, unstable Nash equilibria, and power law size distributions. Technical Report Working Paper No.3, Brookings Institution.

Boutilier, C.; Goldszmidt, M.; and Sabata, B. 1999. Sequential auctions for the allocation of resources with complementarities. In *IJCAI*, 527–523.

Chavez, A.; Moukas, A.; and Maes, P. 1997. Challenger: A multi-agent system for distributed resource allocation. In Johnson, W. L., and Hayes-Roth, B., eds., *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, 323–331. New York: ACM Press.

Gerkey, B. P., and Matarić, M. J. 2002. Sold!: Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation* 18(5):758–768.

Gerkey, B. P., and Matarić, M. J. 2003. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 3862–3868.

Greenwald, A., and Boyan, J. 2001. Bid determination in simultaneous auctions—a case study. In *Proceedings of the Third ACM Conference on Electronic Commerce*, 210–212.

Huberman, B. A., and Hogg, T. 1995. Communities of practice: Performance and evolution. *Computational and Mathematical Organization Theory* 1:73–92.

Huhns, M. N., and Stephens, L. M. 2001. Multiagent systems and societies of agents. In Weiss, G., ed., *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. The MIT Press. 79 – 120.

Modi, P. J.; Jung, H.; Tambe, M.; Shen, W.-M.; and Kulkarni, S. 2001. A dynamic distributed constraint satisfaction approach to resource allocation. In *Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming*, 685–700.

Niemelä, I. 1999. On the complexity of dynamic constraint satisfaction. In *Proceedings of the FLoC Workshop on Complexity-Theoretic and Recursion-Theoretic Methods in Databases, Artificial Intelligence and Finite Model Theory*, 38–47.

Sandholm, T. 2002. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence* 135(1–2):1–54.

Tambe, M. 1997a. Agent architectures for flexible, practical teamwork. In *Proceedings of the 14th National Conference on Artificial Intelligence*, 22–28.

Tambe, M. 1997b. Towards flexible teamwork. *Journal of Artificial Intelligence Research* 7:83–124.

Tambe, M. 1998. Implementing agent teams in dynamic multi-agent environments. *Applied Artificial Intelligence* 12:189–210.

Wellman, M. P., and Wurman, P. R. 1998. Market-aware agents for a multiagent world. *Robotics and Autonomous Systems* 24:115–125.

Wellman, M. P. 1995. The economic approach to artifi cial intelligence. *ACM Computing Surveys* 27(3):340–342.

Wellman, M. P. 1996. Market-oriented programming: Some early lessons. In Clearwater, S., ed., *Market-Based Control: A Paradigm for Distributed Resource Allocation*. River Edge, New Jersey: World Scientifi c. 74–95.

Yokoo, M., and Hirayama, K. 2000. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems* 3(2):185–207.

Yokoo, M.; Durfee, E. H.; Ishida, T.; and Kuwabara, K. 1998. The distributed constraint satisfaction problem: Formalization and algorithms. *Knowledge and Data Engineering* 10(5):673–685.