Learning To Identify Beneficial Partners

Sandip Sen
sandip@utulsa.edu
University of Tulsa, USA

Anil Gursel
anil-gursel@utulsa.edu
University of Tulsa, USA

Stéphane Airiau
stephane@utulsa.edu
University of Tulsa, USA

ABSTRACT

Human and artificial agents routinely make critical choices about interaction partners. The decision about which of several possible candidates to interact with, either for a limited or extended time period, has significant importance on the competitiveness, survivability, and overall utility of an agent. We assume that an agent has time and resource constraints that limit its participation to only a fixed number, k, of relationships or interactions with other agents in a particular time period. Therefore, in a given time period, an agent is free to choose to interact with any k other agents from a society of N agents. A bilateral relationship is established in a time period, however, if both agents choose to do so. The goal of this research is to investigate the extent to which known learning schemes can identify and sustain mutually beneficial relationships in these conditions. While exploration is necessary to locate possible fruitful relationships, resource constraints limit the extent of exploration. The desired emergent phenomena of mutual cooperation is uncertain and fragile as it is predicated on the convergence of the learning of multiple, concurrent learners. We investigate the success of individual learners in identifying and sustaining mutually beneficial relationships in a multiagent society under varying envrionmental conditions.

1. INTRODUCTION

"Whom should I interact with?" Whether "I" is a human being, a business, or an agent, the answer to this question will play an important role on its performance, e.g., happiness, capital, utility, etc. There are two key aspects of this problem. First is the location problem: finding a set of "right" agents to interact with. Though the set of candidates is potentially large, an agent must quickly find other agents with whom interactions are particularly rewarding. It may also take several interactions to build an approximately correct estimate of the true rewards or interacting with another agent. Effective exploration schemes are needed to rapidly identify potentially good candidates. Secondly, a desirable interaction requires both agents to benefit from it. This mutual benefit requirement adds another constraint on the partner selection problem: when an agent encounters a beneficial agent, it needs to determine whether the other agent is also interested in developing a lasting relationship. When the amount of interactions over a period of time is limited, recognizing mutual benefit may not be easy. If another agent does not interact with our agent in a particular time period, it may mean that the agent is not interested in continuing the interactions with our agent, or that the agent was busy exploring relationships with others in that time period.

In this paper, we investigate the effectiveness of learning whom to interact with in a society of agents. We consider that time and resource constraints limit the number of possible interactions, k, that an agent can have per time period, i.e., an agent can choose to interact with any k agents in a population of N agents in each time period. For an interaction to occur successfully, both agents must choose to interact with each other. Each agent receives a utility for every such interaction and the utility of selecting an agent unsuccessfully (when the other agent does not want to interact) is zero. Each agent can update its rating for the other agent based on the received utilities. This in turn determines their willingness to select the other agent for future interactions. Agents are interested in identifying partners such that corresponding interactions generate high utility. Only mutually profitable interactions, however, are desirable and self-sustaining.

The goal of this research is to investigate the extent to which known learning schemes can identify and sustain mutually beneficial relationships in these conditions. The primary difficulty of efficiently finding mutually beneficial relationships, without prior knowledge of others' preferences and needs, is the number of potential candidates to evaluate. Large-scale problems can involve a massive number of agents. The heterogeneity and the number and types of knowledge, services, and resources that are of interest to agents further compound the problem: a fairly small portion of the agent population might be of interest to an agent. So the search for effective partnerships may amount to the proverbial searching for a needle in a haystack!

In addition, agents can enter and leave open environments at any time, e.g., agents may be unable to quickly find desirable interaction partners and are compelled to leave the environment. Also, new services may be needed, and some agents may enter the environment to meet this demand. The environment can, therefore, be highly dynamic. As learned knowledge may be quickly outdated, agents have to carefully manage the exploration-exploitation tradeoff.

A number of multiagent learning algorithms have been developed recently that converge to equlibria in repeated play [3, 7]. Most of these algorithms are evaluated in two-player situations with few actions per agent. We believe, however, that in real open-world environments, where agents interact with different set of agents in each interaction, it is much more likely that simple, single-agent re-

AAMAS07 '07 Honolulu, HI USA

inforcement learning techniques will be used by a large majority of the agents. Hence, it is useful and highly instructive to evaluate the resultant dynamics of a relatively large population of simple reinforcement learners searching for mutually beneficial partnerships. We, therefore, use Q-learning [13] as the learning algorithm used by our population of agents.

We do not know of any research that has attempted such massively concurrent learning by a large number of utility maximizing agents using single-agent reinforcement learning techniques where the agent utilities are closely coupled. Not only is the likelihood of convergence of such interlinked learning to effective selections unclear *a priori*, no weak guarantees about performance can also be provided. That is what, however, makes this empirical evaluation interesting as we can develop important insights about the effects of different environmental parameters on the learned outcomes. It will also be illuminating to observe and analyze the dynamics of agent selection strategies as agents develop preferences for different partners over time and adapt their selections.

We conduct a series of experiments varying population size, agent preferences, exploration schemes, survival rates, etc. We show that the agents can learn to effectively select k partners from a population of N agents, provided sufficient exploration, in dynamic domains, and both with deterministic and stochastic payoffs from interactions. We also study the effect of the diversity of payoffs from interactions with different agents as well as the effect of agent "deaths" if unsuccessful in generating a minimum threshold utility from interactions in successive generations.

2. RELATED WORK

The agent location problem has been an active area of research in multiagent systems. A well-studied approach is to use referrals from other agents [11, 12, 14]. With referral systems, the agents are not only providing services, but also act as referrers. The issue is not simply to locate agents that offer a particular service, but also to locate services offering a high quality of service, which are likely to be recommended by many peers. When an agent receives a recommendation, the value of this information depends on how trustworthy the recommender agent is and also on its expertise. Agents use learning to determine the value of a recommendation. In the particular case where two agents know each other and interact frequently, a trust relationship is established and can be used to guide the search for other useful agents. One of the agents can provide its opinion about another agent, propose agents to meet, and even organize an interaction between two of its trusted friends. In referral systems, however, the mutual interest constraint is not taken into account. In our work, the agents do not provide referral, they locate agents with complementary interest only through repetitive personal interaction.

Another solution to finding useful agents is to use a matchmaker: agents can reveal some of their interest and competence to a trusted third party. From this information, the matchmaker can identify agents with complementary expertise. This solution is effective for finding optimal matches, as the matchmaker can evaluate all possible matchings, but is computationally costly. Information about all agents in the system must be collected at a such centralized repository. Distributed matchmaking [5, 6, 8] alleviate the scalability and fault tolerance issues inherent with centralized system. Ben-Ami and Shehory compare centralized and distributed agent location mechanisms [2].

To find agents with complementary competence, it is possible to create a model of the competence of other agents. Plaza and Ontañón propose an approach for learning the competence of other agents where agents face a classification problem and can convene a committee to improve the classification accuracy [10]. They show that agents successfully learn when to ask for collaboration and which agents to ask for help. In their work, the agents are cooperative and there is no mutual interest constraint.

Bringing together agents with complementary interest can be done by building a social network. The goal of the agents in such a network can be to modify the connections so that neighbors of an agent have complementary service: one agent consumes the service that the neighbor produces [4]. In [1] a peer-to-peer approach is used to bring together agents with complementary interest and competence: agents are grouped together so as to form self-sustaining groups of agents who can share their knowledge. An agent will provide knowledge to some agents and receive knowledge from other agents in such peer clusters.

As discussed briefly in the last section, this work is also related to work on multiagent learning: in our domain a potentially large number of self interested agents are learning concurrently to optimize their private utility. See [9] for a survey on multiagent learning.

3. PROBLEM DESCRIPTION

We consider a population of N agents. Each agent can try to interact with any other agent in the society. Because of time and resource constraints, an agent can interact with only k other agents in any time period. In each time period, each agent selects a list of k agents it wants to interact with.

In real-life, agents are unlikely to have preferences for individual agents and are more likely to have preferences for other classes of agents. For example, if an agent needs complementary resources or capabilities, any other agent who can provide those, i.e., belong to a type or class with corresponding properties, will be preferred. Therefore, we introduce a type for each agent. The set of all types T is finite. We consider that each agent knows its own type, that it is private (an agent does not know the type of other agents), and that the utility received by an agent i in an interaction with another agent j is a function of the type of j. Note that the preference over the type is individual: two agents of the same type can have different preferences.

For an interaction between agent i and agent j to be successful, i's selection list should contain j and vice versa. When an interaction is successful, each agent will receive a utility (note that for an unsuccessful interaction, e.g., when agent i wanted to interact with j but j did not, i receives zero utility). The reward is stored in a matrix $\mathcal{R}$, that is not known to the agents. The value $\mathcal{R}(i, t_j)$, $(i, t_j) \in N \times T$, is the payoff received by agent i in a successful interaction with an agent j of type t_j. The entries of $\mathcal{R}$ are either the maximum utility(H) one can receive from an interaction or a relatively low value(L). Agent i's goal, then, is to select k agents $\{i_1, ..., i_k\}$ such that $\forall m \in [1..k]$, $\mathcal{R}(i, t_{i_m}) = \mathcal{R}(i_m, t_i) = H$, assuming such a perfectly matching scenario is feasible given the matrix $\mathcal{R}$. To illustrate, let's consider an environment which consists of 6 agents that can be of 3 types, and each agent seeks 2 partners. The types of each agent and the matrix $\mathcal{R}$ are represented in Table 1. The optimum solution to this environment is when the list $l(i)$ of each agent i are as follows: l(0)={2,3}, l(1)={4,5}, l(2)={0,3}, l(3)={0,2}, l(4)={1,5}, l(5)={1,4}.

Note that the agents may not have complementary interest: it can be the case that agent i gets a high value when it interacts with agent j, but agent j may have only a low value for this interaction. Thus, if they interact once, i may be eager to interact again with j, but j may not be enthusiastic about such an interaction in the future. If i keeps on choosing j without j selecting i, i's utility for j will decrease. For example, considering the example in Table 1, when

Agent	Type	Preferences		
		Type 1	Type 2	Type 3
1	Type 1	L	H	L
2	Type 1	L	L	H
3	Type 2	H	H	L
4	Type 2	H	H	L
5	Type 3	H	L	H
6	Type 3	H	L	H

Table 1: Preference example with 6 agents and 3 types.

agent 3 interacts with agent 2, agent 3 receives a high reward, but agent 2 receives a low reward. Agent 3 may seek to interact again with agent 2, but the converse is not true. More generally, utility to an agent for another agent is determined both by the payoff matrix (obtaining low or high reward) and the selection strategies used by the agents (agents needs to select each other to receive a positive reward).

We generated matrices $\mathcal{R}$ for different number of agents and types with unique known solutions. These matrices allow us effectively determine whether the agents are learning to interact with the optimal set of agents. Also, this setting provides a challenging learning problem for the agents as there exists only a unique solution to the problem.

3.1 Learning and Exploration scheme

At each time step, each agent builds its list of k agents, and receives a payoff for each successful interaction. This process is repeated for many time steps. The goal of each agent is to learn to select the k agents that will maximize its utility from interactions.

In our formulation, an agent j maintains an estimate $Q_j(i)$ of the interaction reward with each agent i in the society. The Q-learning [13] rule is used by agent j to update the reward estimate for another agent i as follows

$$Q_j(i) \leftarrow \alpha(e_j(i)) \cdot r + (1 - \alpha(e_j(i))) \cdot Q_j(i),$$

where r is the utility received for interacting with i and $\alpha(e_j(i))$ is the learning rate that is function of the number of times agent j has tried to interact with agent i, $e_j(i)$.

We use the ϵ-greedy exploration scheme for forming the list of k agents. First, each agent builds a sorted list l of all the other agents in decreasing order of Q-values. To draw the k agents to interact with, the agent chooses the next agent in l with probability $1 - \epsilon$, or chooses a random agent in the rest of l with probability ϵ. The exploration rate ϵ determines the tradeoff between exploitation and exploration. At the beginning of the learning process, agents must explore to develop utility estimates for the other agents, and the k agents are chosen at random. Later in the learning process, agents must exploit their learned knowledge: by decreasing the value of ϵ with time, more agents that rank high are chosen with certainty. At the end of the simulation, the k agents chosen are the one with the highest Q-values. In this setting, exploiting corresponds to commiting in a relationship with another agent, i.e., continuing to select that agent to benefit from that partnership. With the decrease of ϵ, the agents commit to more relationships.

4. RESULTS

In the following experiments, the population $N = 48$ and $k = 8$. High reward, H, is 1 whereas low reward, L, is a uniform random number in the range [0,0.8] which is unique for each agent. This choice for the low value should make the problem harder to learn. While it is true that after one successful interaction agents will know whether the other agent is a preferred partner or not, it might take more time to select optimal partnerships in a given environment. This is because a relatively high Q-value, e.g., 0.75 can be obtained by mostly successful interactions with a preferred partner and few unsuccessful interactions or a more consistent successful interactions with a somewhat less preferred partner. Differentiating between these two partners becomes more difficult as the upper bound of the range of L increases.

The initial value of ϵ is set to 0.8 and we use exponential decay of exploration with different rates of decay d: at each iteration, $\epsilon \leftarrow \epsilon \cdot d$. All results are averaged over 10 runs.

4.1 Effect of exploration in static environments

The first question we address is the optimality of the policy found by the learners. We designed an environment where each of the agents can interact with exactly k agents to receive high payoffs. We want to find out if the agents can learn a policy that is close from the optimal in reasonable time for this highly constrained environment. In the following figures, the y-axis represents the average reward obtained by an agent for an interaction, averaged over all the agents. The optimum payoff is H, which is 1 for these experiments. In the first experiment, we vary the value of d, the rate of decay of the exploration. When d is small, ϵ decreases quickly, and the exploration period is short. When d is close to 1, ϵ decreases less rapidly, and the agents have more time to explore. The agents may not have time to exploit (if d is too low). Figure 1 shows that a slow decay ($d > .995$) is required for agents to learn a close to optimal policy. When the decay is fast, (e.g., for $d = 0.98$ and 0.95) agents converge prematurely to a suboptimal solution. The fact that concurrent learning by a relatively large number of individual Q-learners is able to find close to optimal solution to such a tightly coupled problem is pleasantly surprising and bodes well for open agent societies that require agents to locate mutually beneficial partners through trial and error.

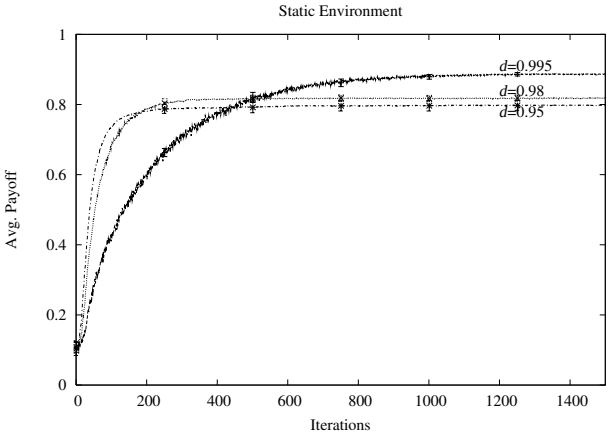

Figure 1: Effect of *d* in static environment

4.2 Dynamic environments

An agent may be forced to leave the environment if it does not quickly find interesting agents to interact with. This may be because it does not have enough resource reserves to sustain itself. We model this situation in the following experiments: an agent survives when the total payoff accumulated over the past n iterations is larger than a threshold value $v = \delta \cdot k \cdot n$. For low value of

δ, an agent needs to find only few good partners. For larger values of δ, the agents need to quickly identify many good partners: this requires an aggressive exploration. When an agent is forced to leave the environment, it is replaced by a copy. The type and utility function remains unchanged, which ensures that the same solution exists (Q-values, history of past interactions, and ϵ are reset). The other agents are notified of the departure of the agent, but they do not know the utility function of the new agent is the same. Hence, they reset the corresponding Q-values to a high value. This will force the other agents to try to interact with the newcomer. As a result, this helps the new agent to quickly find beneficial partners. If few agents have found good partners, the arrival of a new agent may allow them to find a new preferred partner. This might help the system to converge to a close to optimal solution. If many agents frequently die, however, agents will keep on exploring, and hence they may loose fruitful interactions with agents, which in turn can lead to instability in the environment.

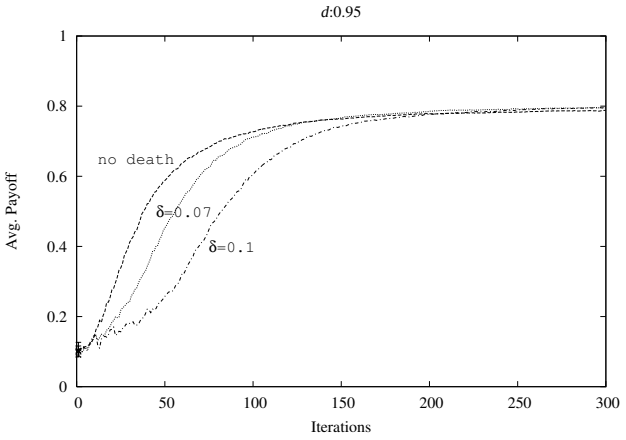

Figure 2: Effect of δ

In Figure 2, we compare learning in a static environment (no death) with learning in dynamic environments with two different values for δ. In each case, the agents are able to select the close to optimal partners. Compared to the curve of the static environment which present a concave gradual increase, the curves for the dynamic environments show a rise in utility after an initial exploration period that increases with δ. At the start of the simulation, as the exploration is large, many agents have to leave: the agents do not exploit enough to be able to survive. However, some agents manage to rapidly find good partners and are able to survive. As there are some stable agents in the system, new agents can establish a relationship with these agents. With the increase of the number of survivors, the number of deaths should decrease faster and faster over a run. Earlier in the simulation, if an agent found a good partner, the partner may not live long enough to provide sufficient payoff. When an agent leaves, a (surviving) agent resets the corresponding Q-value. Hence, it is likely to try to interact with newcomers. This exploration may be harmful if many agents leave at the same time.

If we further increase the value of δ beyond 0.1, it becomes really hard to learn effectively: agents need to achieve a high utility in a small amount of time after birth. They have to explore quickly as well as rapidly exploit potentially good relationships.

4.2.1 Eager exploitation to avoid extinction

In Figure 3, we present the effect of different *d* values on the

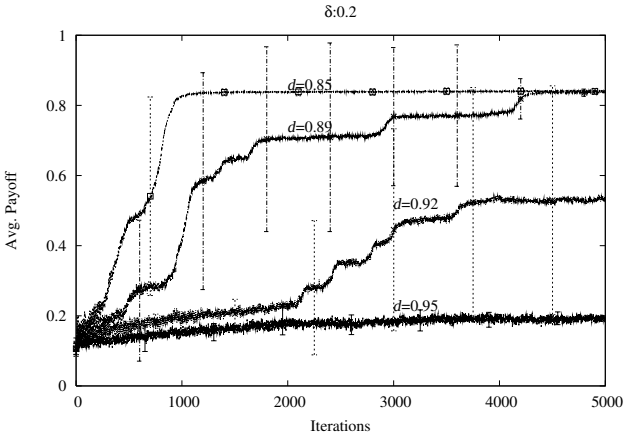

Figure 3: Effect of *d* on dynamic environment

learning abilities of agents where $\delta = 0.2$. Agents are not able to establish initial mutual relationships when *d* is 0.95. Faster decay, e.g., 0.85, allows them to create initial relationships. For slower decay (higher d), payoffs increase more slowly over a run. We note the presence of plateaus where agents are not able to find partners, and sharp increase where a small set of agents that were repeatedly dying finally manage to survive.

4.2.2 Protecting young agents

In the previous experiments, we varied the exploration-exploitation balance. Now, we keep the exploration schedule fixed but we provide additional time for the agents to explore by protecting the agents in their early ages, i.e., they are eliminated only after at least *pp* interactions. Various schemes can be used in practice to provide such protection, e.g, by providing a new agent an initial endowment upon entering the environment. In Figure 4, we display the effect of protecting the young agents. When the protection period (*pp*) is increased, the agents are able to survive more easily, and the system converges faster to near optimal value. To shed some light on this phenomenon we plot the average number of agent departures that occur every 10 iterations in Figure 5: when *pp* is higher less deaths occur early in the simulation. Furthermore, the plateaus and rises in Figures 4 and 5 are correlated. An increase in average payoffs corresponds to a decrease in average number of departures.

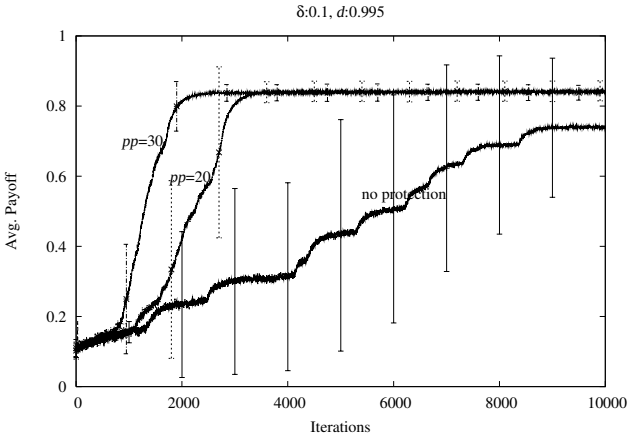

Figure 4: Effect of *pp* on average payOffs

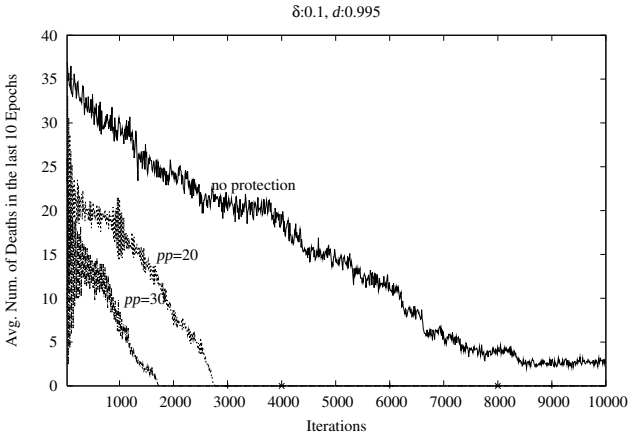

Figure 5: Effect of *pp* on number of deaths

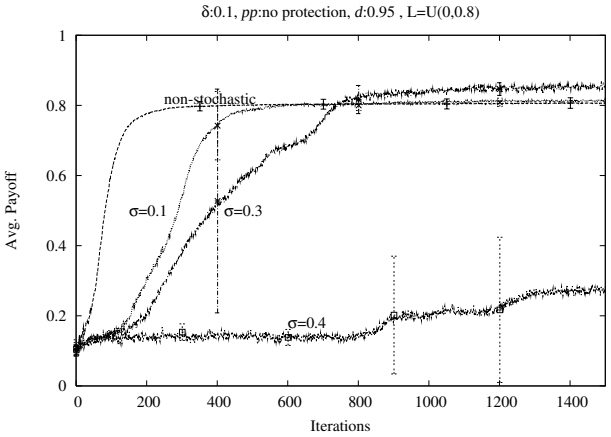

Figure 7: Effect of Gaussian noise on payoffs

4.2.3 Robustness to noise

Next, we present results to show that the learners are robust to noise. We present two scenarios. First, as a result of an interaction, an agent gets a high value H if the interaction is preferred and gets a low value L otherwise. The more the difference between H and L, the easier it is to differentiate a potentially beneficial agent from one that is not. In Figure 6, we present the average payoffs gained by agents with two different low value settings. In one of them, L is set to a constant value 0.01. In the other, L is a uniform random number in the range [0,0.8] which is different for each agent. Although it is harder for agents to differentiate between close to optimal and sub-optimal relationships in the second case, it can be seen that the performance difference is not significant. The reason is that even if two agents, which should have beneficial interactions, fail to take advantage of that opportunity, they are able to limit their loss by interacting with agents that provide lesser, but still acceptable payoffs. In the second study, we consider stochastic payoffs:

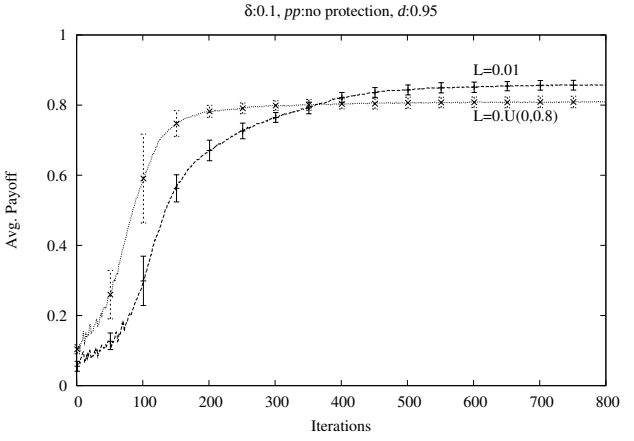

Figure 6: Effect of varying L values

for each successful interaction, a Gaussian noise $N(0, \sigma)$ is added to the payoff, e.g., if the interaction is a preferred one, instead of getting a high value of 1 an agent might get a lesser value, or vice versa. Thus, it is again harder for agents to identify most appropriate agents. In Figure 7 the performances of noisy experiments are compared with a non-stochastic one. For σ values up to 0.3, agents are still able to find partners quickly. When σ=0.4 it takes around 10000 epochs to reach close to optimal pairings. The figure shows that when noise is present, it takes longer time but the agents are still able to discover the best set of partners.

4.3 Experiments with agent preferences

We also investigated the scenario when an agent leaving the environment is replaced by another agent with a different valuation function. With the type based environment, we constructed the matrix $\mathcal{R}$ such that the optimal solution was unique. We now consider direct agent-to-agent preferences in the population. To do so, we use a reward matrix $\mathcal{R}$ of dimension $N \times N$: for each row, representing the reward for a given agent, we randomly set $\gamma \cdot N$ values to 1. The other values of the matrix are drawn from a uniform distribution $\mathcal{U}(0, 0.8)$. When $\gamma \cdot N > k$, it is possible that all agents can get a maximum payoff, though it is not guaranteed. As the difference $\gamma \cdot N - k$ increases, more optimal solutions to the problem should exist, i.e., the agents should have less problem in finding k optimal partners. On the other hand, when $\gamma \cdot N < k$, agents will not be completely satisfied with their agent selections. They can still find $\gamma \cdot N$ agents with optimal interaction value, but they must also interact with some agents with whom the interactions are not optimal. In Figure 8 we observe that agents discover best partners if they have preferences over agents instead of types. Additionally, we note that there does not exist a significant change when the agents are replaced by the ones with different utility functions.

5. CONCLUSION

We studied the problem of agents learning to select interaction partners from a large population of concurrent learners. This corresponds to real-world problems of selecting mutually beneficial relationships. Resource and time limitations constrain the learners' search for close to optimal partnerships. Current multiagent learning does not scale upto problems involving so many concurrent learners and large action spaces. Traditional single-agent learning algorithms are not guaranteed to converge to effective solutions when agents learn concurrently. So, *a priori* it was unclear if this concurrent search for selection partners would converge to desirable system states. Our experiments show that independent Q-learning by concurrent learners with sufficient exploration is surprisingly robust in identifying most of the mutually beneficial relationships in the society. This is particularly credible for the type-based agent preference situation, where agents prefer other agents

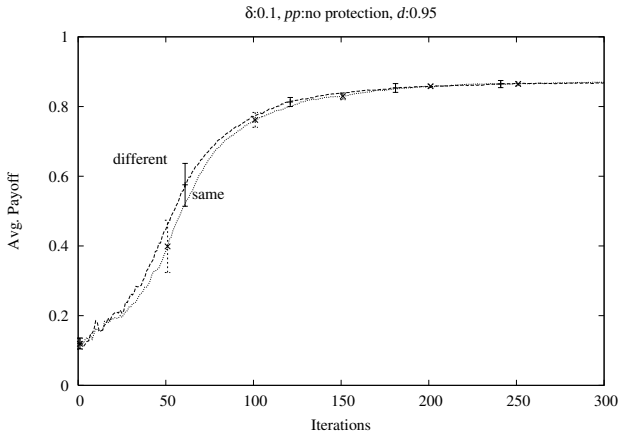

Figure 8: Agent based environment. Eliminated agents are replaced by the ones with same or different utility functions.

of certain types and the problems were constructed so that only one optimal global configuration exists.

We demonstrated that the learning is robust against the difference between close to optimal and sub-optimal payoffs. We also experimented with the problem of agent "deaths" if their payoffs over the last few iterations dropped below a threshold. This caused some turnover in the population in the initial stages, but finally the learners converged to the close to optimal set of pairings. Results improve if agents are allowed more time to locate beneficial partnerships. These results attest to the robustness of concurrent learning for mutually beneficial relationships with existing single-agent reinforcement learning algorithms.

We would be interested in evaluating scale up to significantly larger problem sizes. Development of smarter exploration-exploitation mechanisms would enable faster convergence in larger problems. It would be interesting to evaluate a more realistic scenario involving heterogeneous group of learners using different reinforcement learning algorithms. On a micro-level, phase transitions in the learning process, e.g., rapid transitions from relatively low to high payoffs needs to be analyzed.

Acknowledgment: US National Science Foundation award IIS-0209208 partially supported this work.

6. REFERENCES

[1] S. Airiau, S. Sen, and P. Dasgupta. Effect of joining decisions on peer clusters. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 609–615, New York, NY, USA, 2006. ACM Press.

[2] D. Ben-Ami and O. Shehory. A comparative evaluation of agent location mechanisms in large scale mas. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 339–346, New York, NY, USA, 2005. ACM Press.

[3] M. H. Bowling and M. M. Veloso. Existence of multiagent equilibria with limited agents. *Journal of Artificial Intelligence Research (JAIR)*, 22:353–384, 2004.

[4] M. E. Gaston and M. desJardins. Agent-organized networks for dynamic team formation. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 230–237, New York, NY, USA, 2005. ACM Press.

[5] A. Iamnitchi and I. Foster. A peer-to-peer approach to resource location in grid environments. In J. Weglarz, J. Nabrzyski, J. Schopf, and M. Stroinski, editors, *Grid Resource Managenment*. Klower Publishing, 2003.

[6] S. Jha, P. Chalasani, O. Shehory, and K. Sycara. A formal treatment of distributed matchmaking (poster). In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, pages 457–458, New York, NY, USA, 1998. ACM Press.

[7] M. Littman and P. Stone. A Polynomial-time Nash Equilibrium algorithm for repeated games. *Decision Support Systems*, 39:55–66, 2005.

[8] E. Ogston and S. Vassiliadis. Matchmaking among minimal agents without a facilitator. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 608–615, New York, NY, USA, 2001. ACM Press.

[9] L. Panait and S. Luke. Cooperative multi-agent learning: The state of the art. *Journal of Autonomous Agents and Multi-Agent Systems*, 11(3):387–434, November 2005.

[10] E. Plaza and S. O. nón. Learning collaboration strategies for committees of learning agents. *Journal of Autonomous Agents and Multi-Agent Systems*, 13(3):429–461, 2006.

[11] S. Sen and N. Sajja. Robustness of reputation-based trust: Boolean case. In *Proceedings of the First Intenational Joint Conference on Autonomous Agents and Multiagent Systems*, pages 288–293, New York, NY, 2002. ACM Pres.

[12] M. P. Singh, B. Yu, and M. Venkatraman. Community-based service location. *Commun. ACM*, 44(4):49–54, 2001.

[13] C. J. C. H. Watkins and P. D. Dayan. Q-learning. *Machine Learning*, 3:279 – 292, 1992.

[14] P. Yolum and M. P. Singh. Engineering self-organizing referral networks for trustworthy service selection. *IEEE Transactions on Systems, Man and Cybernetics- Part A: Systems and humans*, 35(3), May 2005.