LIMITED SHARING COOPERATIVE AGENTS IN MULTI-AGENT REINFORCEMENT LEARNING

Meghan Day, Jake Roemer, Jeniya T Binte Jafar, Jhansi Lakshmi Kolla

The Ohio State University
Department of Computer Science and Engineering

ABSTRACT

Demand for intelligent agents has been a large motivation in the design and implementation of enemy AI in games. Players want opponents that can act intelligently and pose a challenge. We propose MARLs, a cooperative Multi-Agent Reinforcement Q-learning approach to AI agents with limited message sharing. We believe MARLs will give the enemy agents a human-like feel which occasionally allows players to capitalize on a wrong choice that the AI makes. To evaluate the performance of this cooperative agent, we compare it against an oracle agent with unlimited message sharing and an independent agent with no message passing capacity. We found only minor improvements in performance between the different AI when limiting policy sharing between agents.

***Index Terms*—** Q-Learning, Reinforcement Learning, Machine Learning, Exploration-Exploitation, Multi-Agent Reinforcement Learning (MARL)

1. INTRODUCTION

Computer games are among the most popular applications for artificial intelligence. When developing an AI for a game, it is important to consider creating a realistic opponent, which requires that AI agents correctly capture human intelligence. Current AI techniques use oracles to simulate cooperative decisions. In a realistic environment, units might not have the same information as other friendly units. To better simulate this scenario, each unit will act as an individual and cooperate with its allies via communication.

In this paper, we propose MARLs, Multi-Agent Reinforcement Learning with Limited Sharing, to give a competitive yet realistic challenge to the player [1]. The agents will learn independently but share information amongst their allies. In a realistic setting, two allies cannot communicate all accumulated information instantaneously, so we limit the amount of information which can be passed and between whom it can be passed per turn. This will require that the agents prioritize which information to share and with whom. We compare MARLs to an independent multi-agent that cannot share information and an oracle multi-agent with unlimited information sharing.

2. BACKGROUND AND RELATED WORK

Our game is a form of Multi-Agent System (MAS), a class which emphasizes the joint action of several agents in an environment. To cooperate in MAS, AI agents can employ three types of message sharing: **1. Percepts**, facts about the environment, such as the agent's location or intended action; **2. Events**, a tuple representing an agent's state, action, and reward; **3. Learned policies**, the learned utility of taking a particular action in a particular state from past experiences [6].

The work presented in this paper focuses on how the performance of a MAS can alter significantly with the introduction of limited message sharing. MAS can be seen in various scenarios, for example multi-agent foraging where several robots are supposed to discover particular rocks and bring them to a specific place [4].

3. GAME DESCRIPTION

Our aim is to develop a board game with two teams of intelligent units to depict the importance of message sharing in an MAS. The game takes place on a two-dimensional grid of arbitrary size. The players consist of two teams with an equal number of agents, which we will refer to as units throughout the rest of the paper. The number of units scale with the size of the grid. The teams are initially placed on opposite sides of the grid using a normal distribution around a centralized point.

The game is turn based, with each team taking its turn after the other. Each unit can move and attack within a turn, in either order. All the units of a team execute their actions simultaneously. Movement is limited in direction to forwards, backwards, left, and right and in distance to a maximum movement speed scaling with the grid. Attacking is limited to units, including allies, diagonal to the current position. After attacking, the unit moves into the position formerly occupied by the defending unit, similar to the capturing movement by pawns in chess. Movement speed is determined by equation 1, where #Rows denotes the rows of the grid and #Columns denotes the columns of the grid. Line of sight is limited to visible squares within a radius of movement speed + 1.

$$Move\_Speed = \frac{\#Rows + \#Columns}{10} \quad (1)$$

The objective of each team is to kill all units of the opposing team. For testing, the three different types of MARL AI algorithms will play against each other. In order to handle a deadlock, where one or both teams begin to act evasively, we impose a maximum number of turns per game (default is 50) and determine the winner by score. The score of each team is determined based on equations: 2 - 3.

$$Score = PPU * (NU - Alive\_EU) - TurnPenalty \quad (2)$$

$$PPU = \frac{Total\_Starting\_Points}{NU} \quad (3)$$

NU is the starting number of units per team. $Alive\_EU$ is the number of enemy units which are still alive and $TurnPenalty$ is the number of turns taken before the game ended.

4. ALGORITHMS

4.1. Reinforcement Learning (RL)

We use a reinforcement learning approach to implement three types of agents. Reinforcement learning is a type of machine learning where behavior is learned through trial-and-error interactions with the dynamic environment. Agents, or learners, determine which action to take in order to maximize its reward. Most of the time the action with the highest expected reward is taken, which is called exploitation. Exploration is taking a new action that has not previously been taken in the same state to learn that action's utility in the current state. The exploration-exploitation trade-off is among the biggest challenge in RL. Figure-1 illustrates this learning process.

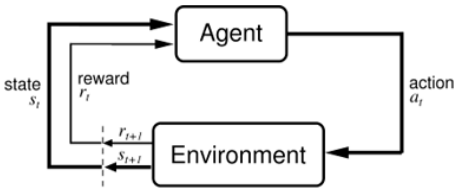

Fig. 1. Agent-Environment interaction in Reinforcement Learning [5]

In our case, agent is the unit of a team and state is the map of all visible units and empty positions. During each turn, the agent calculates all the possible actions in its current state and assigns an initial value to each state. This mapping is called the agent's policy. Units pick actions based on the exploration-exploitation techniques of the underlying MARL algorithms controlling each team.

A unit receives a positive reward when it kills an enemy unit and a negative reward when it is killed or kills an ally. Reward values range from [-2,1], with the highest reward for killing an enemy, 1, and the lowest rewards for killing an ally or getting killed, -1 and -2 respectively. Rewards are also given for proximity to enemy units, 0.5, and ally units, 0.25.

4.2. Multi-Agent Reinforcement Learning (MARL)

We use three variants of MARL to compare the effects of message-sharing between units. In MARL, multiple agents apply RL in a shared environment and collaboratively learn a single objective. Here the policy function depends not only on the environment but also on the policies of other agents. Figure-2 explains the model of MARL.

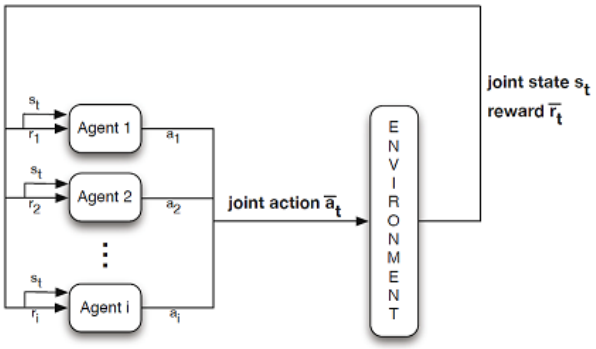

Fig. 2. Multiple agents acting in the same environment [3]

At each step, a joint action from all agents acts on the environment. A joint action is a vector of actions, one action per agent. The joint action acting on the environment changes the shared state and produces a reward for each agent based on the action the agent took.

In our game, each team has a joint action separate from the opposing team, so at each step, one of the two teams composed of multiple agents will perform a joint action on the environment. The state will change for both teams, but the rewards for each action will only go to the agents performing the action. Then the opposing team takes a step, and steps continue until the game ends.

4.3. Q-Learning

Q-Learning is a Reinforcement Learning technique in which each agent keeps an individual vector of values, Q-Values, corresponding to the possible action vector for each state, so that there is exactly one Q-Value for each state-action pair. Initially, each Q-Value is set to -2, but once the corresponding action is taken, the Q-Value of that state-action pair is updated based on equation 4.

$$Q(t,a) = Q(t,a) + \alpha * [r(t,a) + \gamma V(t) - Q(t,a)] \quad (4)$$

Here, $Q(t, a)$ is the Q-Value for action a; α is the learning rate; $r(t, a)$ is the reward function which returns the reward for taking the action a in the given state t; $V(t')$ is the value function which returns the maximum Q-Value for the next state multiplied by γ, a scaling factor, in order to back up the highest Q-Value from the next state.

5. AGENT DESIGN

5.1. Agents

We have adopted three different MARL AI algorithms based on the amount of information shared within the Q-learning implementation [2]. We use learned policy sharing in the form of Q-Value sharing for explored state-action pairs. When one unit shares Q-Values with an allied unit, the receiving unit copies the sharing unit's Q-Values for previously unencountered action-state pairs. For previously encountered action-state pairs, the receiving unit averages its current Q-Value with that of the sharing unit. Percepts and events are not shared. The first implementation is an oracle agent with unlimited sharing of Q-Values between allied units. The second is a cooperative agent with limited sharing of Q-Values. The third implementation is an independent agent with no sharing of Q-Values.

5.2. Oracle Agent

The oracle agent makes decisions by using the collective experience of all of its units. During each turn, all the units calculate their individual Q-Values based on the outcome of their last state and action. Then, each unit shares this new value with all of its allied units, who copy this Q-Value if they have previously not encountered the corresponding state-action pair. Because it combines all the experience from all its units, we expect the oracle agent to perform better than the cooperative and independent agents.

5.3. Cooperative Agent

The cooperative agent is implemented with limited message sharing. In our implementation, during every turn each unit shares its updated Q-Values with a subset of its allies, no greater than 3, who are selected at random. The chosen allies will share their state experience, by sharing the actions associated with the shared state. This includes sharing both unencountered action-state pairs and perviously encountered action-state pair.

Q-Values are also shared when determining the result of the Value function, which helps emphasize a good action even if a unit has not taken it yet, but an ally has. Because the cooperative agent has the ability to share some limited experience between units, we expect it to perform better than the independent agent.

5.4. Independent Agent

The independent agent is implemented with no message sharing. In this implementation, each unit is controlled by itself only, so the decisions are made based only on the knowledge available to that particular unit.

6. EXPERIMENT

Due to the nature of MARL and our game rules, we don't need any prior data collection. Reinforcement learning will collect data as each game is played. We have implemented a simulation of the game described in section 3 along with the three different MARL AI implementations. The different MARL AI are pitted against each other over a series of games to measure how each AI learns based on the number of wins, and average score, calculated by equation 2.

To meausre how each AI learns, we take a set window of games and evaluate the number of wins, average score, and average turns to win seperate from all other windows. For 10000 games, each window is of size 1000 games and will show the progress in learning every 1000 games. Scores are listed above wins in each graph as a difference between the two team's scores.

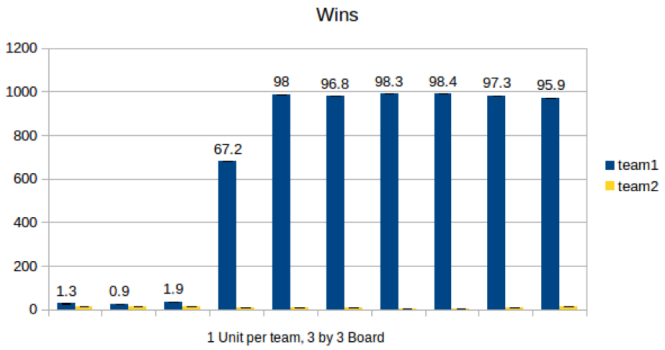

Fig. 3. One unit in each team in a 3 by 3 board

The first experiment sets both team sizes to 1 unit each with a board size of 3 by 3 and runs 10000 games. With only 1 unit per team, no information can be shared between units so each algorithm should perform the same in terms of learning. WIth a board size of 3 by 3, both units are able to reach each other and kill the enemy within a single turn. As a unit learns from game to game it should explore most, if not all, of the state-action space and eventually the action of killing an enemy will be discovered and reinforced positively. After enough runs, the team who takes the first turn should win every time since it will have positively reinforced the action of killing the enemy within a single turn. Looking at Figure 3, it is evident that one team was the clear overall winner. As you can see, during the first couple thousand games neither team

was beating the other. Eventually, team 1 learned how to kill team 2. At this point, due to the bias nature of the game setup, team 1 fully learned how to win almost every game.

The second experiment sets both team sizes to 3 on a board of size 5 by 8, and runs for 10000 games. This is a larger board and both teams are bigger, giving each algorithm room to learn differently. With more than 1 unit per team, sharing becomes a factor as well so here we expect to see Oracle outperform both Independent and Cooperative. We also expect to see Cooperative outperform Independent. Over 10000 games, we see in Figure- 5 and Figure- 6 that indeed oracle does win more often and with a higher score than the other two algorithms. Cooperative also outperforms Independent, but by an even slighter margin [Figure- 4]. As the win rate, and score are within small bounds when comparing one algorithm to another, we believe this leaves room for further improvement, discussed in section 7. The red curve in figure-4, figure-5, figure-6 is reflecting the learning rate of the agents.

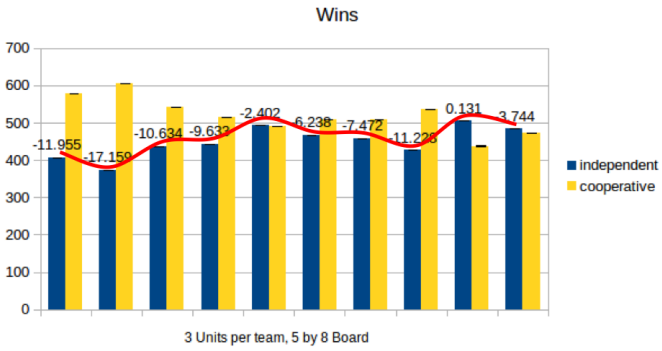

Fig. 4. Independent vs Cooperative teams with 3 units in each team in a 5 by 8 board

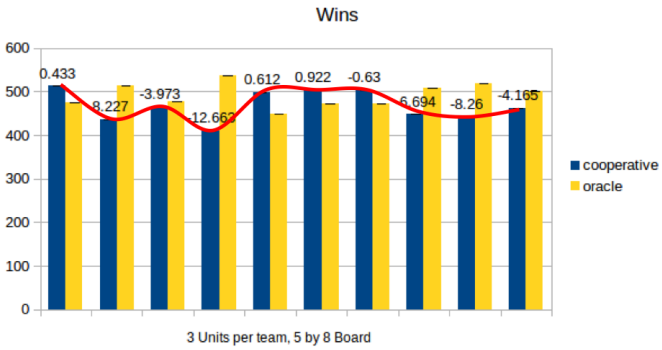

Fig. 5. Cooperative vs Oracle teams with 3 units in each team in a 5 by 8 board

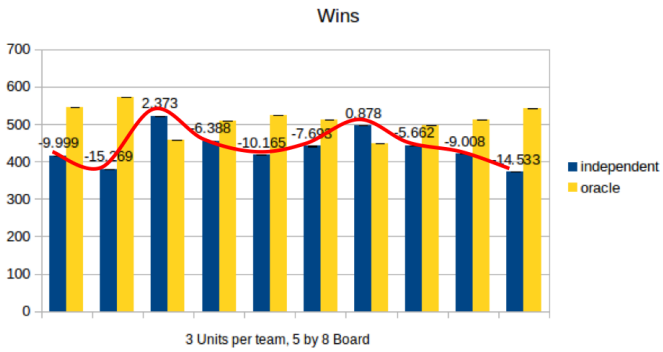

Fig. 6. Independent vs Oracle teams with 3 units in each team in a 5 by 8 board

7. FUTURE WORK

The performance increase obtained by information sharing in our results was modest. We have identified a number of ways in which the algorithms and game could be modified to potentially increase performance by information sharing. The message sharing could be modified to include sharing percepts and events. In this context, that could mean allowing units to know the locations of their allied units beyond their line of sight and the locations of enemy units seen by their allies. The units could also share not just new Q-Values, but every state-action-reward tuple so that allied units can update their Q-Values with this information even after a given state-action pair has been explored once. Sharing can also be extended from only sharing with allies to sharing with yourself as well. It is possible to find the degree of similarity between a unit's current state and different states a unit has already visited and predict the best next action based on the actions taken in similar previous states. The agents could also be modified to have a commander unit, which would make decisions for all of its allied units to help the team as a whole.

8. CONCLUSION

From the experiments, it is observed that Oracle, where there is unlimited message sharing, has the best performance. However this can never be applied in a real time scenario where there can only be limited message sharing among agents acting in any field (e.g. a battlefield). Also, we have seen that the performance is poor in the case of independent agents, which implies that message sharing is important in a multi-agent environment. From our results, we can also see that the difference in performance between the MARL AI algorithms is negligable leaving room for further development and future improvement.

9. REFERENCES

[1] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, March 2008.

[2] J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm, 1998.

[3] A. Now, P. Vrancx, and Y.-M. De Hauwere. Game theory and multi-agent reinforcement learning. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning*, volume 12 of *Adaptation, Learning, and Optimization*, pages 441–470. Springer Berlin Heidelberg, 2012.

[4] L. Panait and S. Luke. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*, 11:2005, 2005.

[5] R. S. Sutton and A. G. Barto. Reinforcement learning i: Introduction, 1998.

[6] M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *In Proceedings of the Tenth International Conference on Machine Learning*, pages 330–337. Morgan Kaufmann, 1993.

10. REVIEWS

The reviews and responses are listed below. Since the section number in the final version of the paper is different from the initial version, we added the name of the section which the reviewer mentioned in brackets.

10.1. Review 1

1. Please include more related work to explain the message sharing (e.g. facts and events)
   Response : Message sharing is explained in more detail in the later section-5.2, 5.3, 5.4.

2. Although introduced in details later, could you explain action, state and rewards in more details in section 3.1 and 3.2 (Reinforcement learning and multiple agent learning) maybe with some examples.
   Response : We added some additional detail to section-4.1.

3. I suppose some parameters for the variables in Equation 1 are not shown explicitly, e.g. 'r' has parameters 'state' and 'action'.
   Response : We added explanations for each variable in the equation.

4. Maybe you could introduce each kind of agent first and compare them in a new section
   Response : We briefly added a section-5

5. For the result, could you run examples with larger board size and team size?
   Response : We include results for 5x8. Results for larger sizes perform similarly but take longer, so we opt to not include these in the paper due to space limitations.

10.2. Review 2

1. In Equation (1),neither the description nor the intuitive reasoning behind the parameter '' is not given. From the context, I am to understand that it is a learning rate similar to the stochastic gradient descent method. It would be nice to clarify the same in this proposal.
   Response : See review 1, point 3.

2. The section about the usage of Q-Values in the overall algorithm is a little handwavy - the precise way how it determines the overall running of the algorithm is not mentioned in sections 3.3 (Q-Learning) which manifests in an unclear understanding of section 4.1 (Oracle Agent) - The sentence "On the other hand if the black circled units are controlled by oracle agent then the agent will choose either unit-1 or unit-2 based on their Q-values to attack unit-A" is hard to comprehend when the exact role of Q-Values if not known.
   Response : We added more detail to section-4.3, and the section quoted above was completely removed from the paper as we took a different direction with our message sharing implementation.

3. I believe that there is a discrepancy in the results for table 1. In the event that there are 2 teams with 1 member each where the game ends in a single turn - both independent and oracle play should produce the same result (since they are the same in this particular case). However it seems from the table that there is an imbalance in the two. That is the ratio is 0.82:0.17 whereas it should have been 0.5 to 0.5 since the two are essentially equivalent. (This result is surprisingly reflected in table 2). It may be a problem in my understanding of the table semantics however it would behoove the authors to provide clarity in the said aspect in the final draft.
   Response : New figures are added instead of the tables in the revised version to clarify this confusion.

4. The authors don't provide any analysis for the convergence of the gameplay algorithm. More specifically, I am curious if there can be any situations where there is a deadlock in the game play (I can envision a situation where there are two agents left in a board and one agent plays the game offensively while the other agent plays

defensively essentially producing a "cat-mouse" situation which can extend the game for a large turns - such a situation while uncommon may occur). I believe that the parameters in the Q-Value computation can help in determining if such an analysis exists or not.
Response : If by convergence, the reviewer meant when we should stop training our algorithm to start testing with. We agreed that there is no true convergence since agents learn indefinitely. As an evaluation, we show that there is a point in which learning for an agent in a particular game setup will stop making significant process.

5. I believe that section 4.3 can benefit from some clarity. The two parameters which are crucial to correct information sharing - " limited to only sharing with a set number of allies and only specifically requested information. " It is not clear how the first parameters will be determined. Also not clear is how the agents will learn the information to be requested - is it a part of the learning or will it be fixed. In the former case, the ML complexity of the algorithm increases while in the latter case, it will suffer from slow learning rate since it may keep on optimization over a set of values on which we already have sufficient information.
Response : The number of allies to share information with is random. So when sharing, we choose a random subset of allies to share with. The actual process of sharing information with allies happens by sharing Q-Values after an action is taken. Given a set of allies to share with, if a chosen ally has also taken the same action as some point the Q-Value for the unit and the ally are averaged together. In addition to averaging Q-Values, the value function will also average the Q-Values of the next state. The same subset of allies will be chosen to share with and if a chosen ally has also taken the same next action at some point the Q-Value for the unit and the ally used to calculate the Value Function will average together.

10.3. Review 3

1. would suggest moving the description of the game that is being played earlier in the paper.
Response : We restructured the paper to introduce the game earlier.

2. I was confused on the specific type of game being played when I started reading. I was confused about the results. Why, in the case of a single unit, would one algorithm outperform another.
Response : In case of single unit, the point is that there is no difference in the performance of different AI agents. Whichever team starts first wins.

3. Even in section 5 you mention that no sharing is being done, so each algorithm should perform the same, but Table 1 shows differently. Could you address this or clarify this?
Response : Corrected this and replaced the table with our new results Fig-3.

4. In Table 2 the results between the algorithms seem close enough to maybe be statistical error. How are you sure that one algorithm outperforms another?
Response : We changed the presentation of results to emphasize that the differences in performance are not, in fact, that strong.

5. I would like to see some questions asked at the end about generalizing. Could this technique work for a less contrived game, like a modern turn based or RTS game?
Response : The different AI make decision based on different states, the possible actions from those states, and what kind of reward a unit should receive for taking an action. If states, actions, and rewards are properly defined for each game then the algorithm simply needs a new state as an input and will provide an action to take.

6. Does cooperative passing provide a better experience for a user, or do they not care?
Response : for 1 user there is no difference

7. I noticed a small typo in section 2(BACKGROUND AND RELATED WORK): messages passing
Response : Typo corrected

8. All in all I think this is a good and interesting paper.

10.4. Review 4

1. The biggest problem is that you didn't describe Cooperative Agent clearly in section 4.3, while Cooperative Agent is supposed to be your key innovation in this paper. You may add some explanation of how the Cooperative Agent incorporates (uses) the received information and in what ways it is different from the Oracle Agent.
Response : Added the explanation for cooperative agent in section-5.3.

2. It is not appropriate to say that Reinforcement Learning is a form of Unsupervised Learning. The "prior knowledge" in a Reinforcement Learning system is the rewards, or the rewards generating system. A more rational division of learning algorithms should be Supervised Learning, Unsupervised Learning and Reinforcement Learning.
Response : We changed how we introduce and describe Reinforcement Learning.

3. In your first experiment, where there is only one unit, the result shows a clear relationship that Oracle Agent outperforms Cooperative Agent, and Cooperative Agent is better than Independent Agent. How do you explain this? If it is totally because of the order that who runs first, you may start from a randomly selected agent at each episode.
   Response : See review 3, point 2

4. The order of your bibliography is confusing. For example, the number of your bibliography jumps from "[1]" to "[6]", and then jumps back to "[4]" in your introduction and background section.
   Response : We use abbrv format, which is an acceptable bibliography format.

5. The explanation to Formula 1 in your paper is correct in general. However, if you are familiar with the mathematical derivation of the Markov Decision Process, it may not be appropriate to use "V" as the optimal Q value, since "V" is commonly used to denote the state-value function. In addition, the explanation of "V" in your paper should be more accurate. In another word, you should tell explicitly for which state "V" is the maximal Q-value. Perhaps it is helpful for you to take a deeper study of the mathematical foundation of Reinforcement Learning and Bellman Equation.
   Response : See review 1, point 3

6. The keywords you selected may be inappropriate, since the concept of one keyword may belong to another one.
   Response : Conference papers also have some overlapping keywords, so we think this is fine

7. Some prior art you mentioned may not be directly relevant to the topic of this paper, for example, the multi-agent foraging example in the background section.
   Response : It is added as a concrete example of Multi Agent Systems, especially demonstrating a situation where multiple agents may have limited bandwidth with which to share information.