What to Communicate? Execution-time Decision in Multi-agent POMDPs

Maayan Roth[1], Reid Simmons[2], and Manuela Veloso[3]

[1] Robotics Institute, Carnegie Mellon University mroth@andrew.cmu.edu
[2] Robotics Institute, Carnegie Mellon University reids@cs.cmu.edu
[3] Computer Science Department, Carnegie Mellon University veloso@cs.cmu.edu

Summary. In recent years, multi-agent Partially Observable Markov Decision Processes (POMDP) have emerged as a popular decision-theoretic framework for modeling and generating policies for the control of multi-agent teams. Teams controlled by multi-agent POMDPs can use communication to share observations and coordinate. Therefore, policies are needed to enable these teams to reason about communication. Previous work on generating communication policies for multi-agent POMDPs has focused on the question of **when** to communicate. In this paper, we address the question of **what** to communicate. We describe two paradigms for representing limitations on communication and present an algorithm that enables multi-agent teams to make execution-time decisions on how to effectively utilize available communication resources.

1 Introduction

The problem of generating optimal policies for multi-agent POMDPs is known to be NEXP-complete [1], making optimal policy-generation intractable. Therefore, the bulk of recent work in this area has focused on finding heuristic algorithms that can generate high-quality policies for multi-agent teams in a reasonable amount of time. Team members can improve their ability and the abilities of their teammates to reason about their environment by communicating their local observations. We are interested in studying heuristics for making execution-time communication decisions [5]. By assuming free communication at policy-generation time, we can generate centralized policies for multi-agent teams using a single-agent POMDP solver. We then reason about communication at execution time to enable decentralized execution of these policies.

In situations where teams have the capability to perform free and unlimited communication, the best strategy is for each agent to broadcast all of its observations to its teammates [4]. However, in general, communication is not free. To use communication effectively, multi-agent teams must trade off the benefit that can be achieved through communication with the cost of communicating. We consider two communication paradigms:

Fixed cost per communication instance - Every time the agents decide to communicate, they incur a known fixed cost [4, 6, 7]. The question that must be answered by a communication heuristic is, in essence, **when** to communicate. In some previous approaches, the use of this paradigm to focus on the question of when to communicate has justified algorithms in which agents are required to communicate their entire observation histories if they determine that communication is beneficial [8, 5]. In this paper, we extend this paradigm to include the case in which cost scales with the amount of information to be communicated. We model this by assuming a fixed cost per observation transmitted.

Limited communication bandwidth - Often, communication among agents has a strict limit on available bandwidth. For example, in robot soccer, an attempt to communicate the complete and frequent sensory data would easily overload the available communication resources [11]. In planetary exploration, communication bandwidth is limited, and in a proposed communication architecture, agents need to share a limited number of communication relays in order to stay in contact with their teammates [12]. Other domains include distributed surveillance, in which the observations themselves are very large [13]. In general, it is possible to quantify the amount of bandwidth available for communication between teammate agents. The challenge, then, is to determine **what** to communicate so as to best use the available bandwidth, an issue that has received little attention in the current multi-agent POMDP literature.

Our previous work presented an algorithm for making execution-time decisions about **when** to communicate that allows agents to successfully execute centralized policies in a decentralized fashion [5]. In this paper, we introduce an algorithm that builds on our previous work to address the question of **what** to communicate. We show the applicability of this algorithm to both communication paradigms presented above, and verify the success of our algorithm through experimental results. When there is a fixed communication cost per observation, our algorithm allows agents to identify only those observations that are relevant to team performance. In the case of bandwidth limitation, where each agent is allowed to communicate a fixed number of observations per unit time, our algorithm enables agents to choose those observations that will most improve expected team reward.

Given the complexity of generating policies for multi-agent POMDPs, it is not surprising that approaches have so far been validated on very small test problems. The *multi-agent tiger domain* , introduced by Nair *et al.* [2], has emerged as a commonly-used benchmark case [9, 5, 10]. It has the significant advantage of being small enough for easy use in explanatory examples, while still containing a challenging coordination problem. In this paper, we introduce a new domain, called the *Colorado/Wyoming problem*. This new domain contributes several key attributes that make it useful for evaluating communication heuristics, such as the presence of multiple different observations that

provide varying qualities of information. Together with the multi-agent tiger domain, it is a step toward the compilation of a comprehensive suite of benchmark domains for multi-agent POMDPs.

2 Multi-agent POMDPs

Several representations (e.g. DEC-POMDP [1], MTDP [4], POIPSG [3], IPOMDP [10]) can be used to model cooperative teams of agents operating under partial observability. In this paper, we use the notation introduced in [1], which defines a DEC-POMDP as a tuple $\langle \alpha, \mathcal{S}, \mathcal{A}, \mathcal{T}, \Omega, \mathcal{O}, \mathcal{R} \rangle$ where α is the number of agents in the team, $\mathcal{S}$ is the set of n world states, and $\mathcal{A}$ is the set of m possible joint actions of the team, where each joint action, a^i, is composed of α individual actions. $\mathcal{T}$, the transition function, depends on joint actions and gives the probability associated with starting in a particular state s^i and ending in a state s^j after the team has executed the joint action a^k. Ω is the set of possible joint observations, where each joint observation, ω^i, is composed of α individual observations. The observation function, $\mathcal{O}$, gives the probability of observing the joint observation ω^i after taking action a^k and ending in state s^j. $\mathcal{R}$ indicates the reward that is received when the team starts in a state s^i and takes the joint action a^k. In this paper, we present example domains in which the agents are identical. However, this is not a necessary property of multi-agent POMDPs and our algorithms are equally applicable to heterogeneous teams.

It is important to note that, although the observation function is given in terms of joint observations, each agent observes only its own individual observations. Additionally, when executing a policy, the individual agents receive no explicit notification of the actions that were taken by their teammates. Multi-agent POMDPs are challenging to solve because, to accurately model the state and choose a policy, each agent must reason not only over uncertainty in the environment, but also about the possible behaviors of its teammates.

There are several classes of observability possible in cooperative multi-agent teams. In this paper, we examine domains with *collective partial observability*, meaning that even if every agent on the team had access to the local observations of all of its teammates, this union of team information may still be insufficient to uniquely identify the world state. The algorithms that we discuss in this work are equally applicable to domains with *collective observability*, sometimes called DEC-MDP, in which the union of individual observations is sufficient to uniquely determine the team's state.

3 DEC-COMM: Deciding When to Communicate

The problem of generating optimal policies for multi-agent POMDPs is known to be NEXP-complete [1], making exact solutions unfeasible and necessitating the use of heuristics. Our previous work [5] developed an approach that exploits a known property of multi-agent POMDPs, namely that the presence of

free and unrestricted communication can be used to transform a multi-agent POMDP into a centralized, or single-agent, POMDP [4], a problem that has a smaller complexity of PSPACE [14]. The approach of our previous work is to assume, at policy-generation time, that communication is free, allowing agents to know the local observations of their teammates at every timestep. This enables us to write the multi-agent POMDP as a single-agent POMDP. We are then able to use any single-agent POMDP solver (e.g. [15]) to generate a centralized policy for the team.

The challenge, then, is to enable agents to execute the centralized policy in a decentralized manner despite the fact that, in general, communication is not free. Because the transition and observation functions of a multi-agent POMDP depend on the joint action, an individual member of the team cannot compute belief independently. To correctly execute a centralized policy, the agents must form the same approximation of joint belief, and each agent must ensure that it is selecting the same joint action as its teammates. This requires agents to model joint belief based only on information that is globally available to all of the teammates.

Our approach is to have each agent calculate $\mathcal{L}^t$, the distribution of possible joint beliefs of the team. Each element, $\mathcal{L}_i^t$, is a possible joint observation history. $\mathcal{L}_i^t$ is defined as the tuple $\langle b^t, p^t, \boldsymbol{\omega}^t \rangle$, where $\boldsymbol{\omega}^t$ is the joint observation history leading to $\mathcal{L}_i^t$, b^t is the joint belief given that history, and p^t is the probability of the team observing that history. [5] provides a detailed algorithm, GROWTREE, that describes how this tree is calculated. The important detail is that the contents of this tree do not depend on any agent's local observations. Therefore, all the agents can compute identical trees independently.

Agents can calculate a joint action over the distribution of possible joint beliefs and be assured that the joint action selected is identical across teammates. This action selection can by done by means of any function that operates over the leaves in $\mathcal{L}^t$. We introduced one possible EVALUATE function, Q-POMDP, in [5]. However, since agents do not use their local observations to refine their beliefs about the state of the world, the selected action is unaffected by the agents' true experiences. Communication provides a means for agents to share their local observations with their teammates, enabling the team to use those observations when making decisions.

Since communication is not free, we want to reason about when to communicate. The DEC-COMM algorithm, presented in detail in [5], enables agents to make execution-time decisions about when to communicate their observations to teammates. An agent can hypothesize about the joint action that would be selected by the team if it chose to communicate by pruning $\mathcal{L}^t$ of all of the observation histories that are inconsistent with its own local observations. It compares the expected reward of this new joint action, a_C, with the expected reward of the joint action that would be chosen if it does not communicate, a_{NC}. If the change in expected reward is above some threshold ϵ (the cost of communication), the agent broadcasts its observation history to its team-

mates, who then prune their own $\mathcal{L}^t$ to be consistent with the communicated observations.

4 Choosing What to Communicate

The Dec-Comm algorithm described above answers the question of **when** to communicate, making run-time communication decisions in the context of decentralized execution of a centralized policy. However, it does not address the question of **what** to communicate. Each time an agent communicates, the algorithm requires it to broadcast all of the observations received since the last time that it communicated. There are several shortcomings to this approach. First, it is unnecessarily wasteful, forcing agents to broadcast observations that do not serve to improve team performance. Second, it can deal only with communication limitations that are represented through a fixed cost of communication. The algorithm in its original form does not enable agents to make efficient use of limited communication bandwidth.

Given a limited bandwidth availability of k observations, the goal is to find those k observations that would most increase the expected team reward if communicated. While this can be done exhaustively by calculating the value of information of each subset of size k of an agent's observation history, it is intractable for run-time decision making.

Instead, we introduce the BuildMessage heuristic. The intuition is as follows: The agent can calculate a_C, the joint action that the team would perform if the agent could broadcast its entire observation history. From this agent's perspective, a_C is the best possible action that the team could take, given all of the available information. If a_C is the same as the action that would be performed without communication, it is clear that the agent cannot expect that communicating only a subset of observations will improve expected reward. If, however, communication could potentially improve the team's selection of a joint action, then it seems logical to select those observations that most increase the desirability of choosing a_C. In essence, BuildMessage is a hill-climbing heuristic that greedily selects those observations that, when integrated into the joint belief, result in the highest expected reward for the action a_C.

While BuildMessage is not optimal, its run time is only polynomial in the length of the observation history. The parameters of the heuristic make it applicable to both paradigms of communication that were discussed earlier. If communication has a fixed cost and the goal is simply to minimize the number of observations communicated, k can be set to t, the number of observations in the agent's observation history. This enables BuildMessage to select as many observations as needed to change the joint action to a_C, but no others. If there is a bandwidth limitation of k observations, ϵ should be close to 0, indicating that communication of up to k messages is allowed as long as there is even a marginal improvement in expected reward. Table 2 shows the new Dec-Comm-Selective algorithm, which utilizes the BuildMessage heuristic to

```
BUILDMESSAGE(L, ω_j, ε, k)
  a_NC ← arg max_a EVALUATE(a, L)
  L' ← PRUNE(ω_j, L)
  a_C ← arg max_a EVALUATE(a, L')
  if EVALUATE(a_C, L') - EVALUATE(a_NC, L') ≤ ε
    return ∅
  else
    ω_C ← ∅
    while (|ω_C| ≤ k) ∧ (a_NC ≠ a_C)
      v_MAX ← −∞
      for each ω ∈ ω_j
        L' ← PRUNE(ω, L)
        v ← EVALUATE(a_C, L')
        if v > v_MAX
          v_MAX ← v
          ω_MAX ← ω
      ω_C ← ω_C ∘ ⟨ω_MAX⟩
      L ← PRUNE(ω_MAX, L)
      ω_j ← ω_j − ω_MAX
      a_NC ← arg max_a EVALUATE(a, L)
    return ω_C
```

Table 1. The BUILDMESSAGE heuristic greedily selects the observations that lead to the greatest increase in expected reward for a_C, the action that would be executed if the agent communicated its entire observation history.

choose when and what to communicate. It is invoked in any timestep when a particular agent is allowed to communicate (i.e. there is bandwidth available for it to use in this timestep).

```
DEC-COMM-SELECTIVE(L^t, ω_j^t, ε, k)
  ω_C ← BUILDMESSAGE(L^t, ω_j^t, ε, k)
  if |ω_C| > 0
    communicate ω_C to teammates
    L^t ← PRUNE(ω_C, L^t)
    ω_j^t ← ω_j^t − ω_C
  if message ω_i^t was received from teammate i
    L^t ← PRUNE(ω_i^t, L^t)
  a ← arg max_a EVALUATE(a, L^t)
  take action a
  receive observation ω_j^{t+1}
  ω_j^{t+1} ← ω_j^t ∘ ⟨ω_j^{t+1}⟩
  L^{t+1} ← ∅
  for each L_i^t ∈ L^t
    L^{t+1} ← L^{t+1} ∪ GROWTREE(L_i^t, a)
  return [L^{t+1}, ω_j^{t+1}]
```

Table 2. One time step of the DEC-COMM-SELECTIVE algorithm for an agent j

5 Experimental Results

5.1 Multi-agent Tiger Domain

The multi-agent tiger problem [2] is a two-agent extension to the classical tiger problem [17]. This domain is comprised of a room with two doors. Behind one door is a tiger, and behind the other is a treasure. Each agent may either choose to open a door or to perform LISTEN, an information-gathering action that provides a noisy observation about the position of the tiger. The goal of the problem is to avoid the tiger and instead to open the door hiding the treasure.

To make this an interesting benchmark for multi-agent systems, an explicit coordination problem is built into the domain. The maximum reward is obtained when both agents simultaneously open the door with the treasure. A penalty is incurred when both agents open the door with the tiger. However, the worst penalty occurs when each agent opens a different door. This coordination problem requires the agents to consider the actions of their teammates when making their own decisions.

Our experimental results demonstrate that the DEC-COMM-SELECTIVE algorithm enables a team of agents to make execution-time communication decisions not only about **when** to communicate, but also about **what** to communicate, ensuring that they do not send unnecessary information. Table 3 summarizes the results of the experiment. We generated centralized a policy for the team using the Cassandra POMDP solver [15]. We then ran 1000 trials each of the DEC-COMM and DEC-COMM-SELECTIVE algorithms, allowing the team to execute for 6 timesteps in each trial. The DEC-COMM-SELECTIVE algorithm enables agents to broadcast almost 30% less observations with only a small reduction in performance.

	Average Reward	Average # Communications
FREE COMMUNICATION	11.95	10.0
DEC-COMM	9.35	5.14
DEC-COMM-SELECTIVE	8.41	3.68

Table 3. Results for the tiger problem.

5.2 Colorado/Wyoming Domain

While the tiger domain is useful for evaluating communication strategies, in that it encodes a non-transition independent coordination problem in which agents must act jointly to maximize expected reward, it is missing other characteristics that are necessary to illustrate the full range of communication decisions. In particular, the tiger domain has only two possible individual observations. In this paper, we introduce the Colorado/Wyoming domain that, in addition to sharing the useful characteristics of the tiger domain, also has

many possible observations, and those observations have different utilities with respect to team performance.

In this domain, two agents start one of two possible 5x5 grid worlds, Colorado or Wyoming, and must meet in a predetermined location. If they are in Colorado, their goal is to meet up in Denver, at grid position (2,4). If the agents are in Wyoming, they must rendezvous in Cheyenne, located at grid position (5,5) (see Figure 1). Each agent can move NORTH, SOUTH, EAST, or WEST, with each move succeeding with probability $p = 0.95$ and incurring a cost of -1. An agent can also STOP or send up a SIGNAL. Similarly to the multi-agent tiger domain, the Colorado domain contains an explicit coordination problem. If both agents are at the correct goal location when they simultaneously send up a SIGNAL, they receive a joint reward of +20. If they send up simultaneous SIGNALs from an incorrect location, they receive a reward of -50. However, if only one agent SIGNALs, or if they signal in different locations, the team incurs a penalty of -100.

Fig. 1. Figure (a) is one possible configuration of the two agents in Colorado, with the goal, Denver, at (2,4). Figure (b) is one possible configuration of the two agents in Wyoming, with Cheyenne located at (5,5).

In order to progress toward the correct goal location, the agents must observe their environment. Both Colorado and Wyoming contain flat and mountainous regions. However, the probability that an agent will observe MOUNTAIN in Colorado is slightly higher than observing it in Wyoming. Likewise, the observation PLAIN is more probable in Wyoming. Colorado and Wyoming also contain distinctive tourist attractions. It is somewhat likely that an agent will see a sign for PIKESPEAK in Colorado or a sign for OLDFAITHFUL in Wyoming, but very unlikely that these would be observed in the opposite state. Because an agent is much more likely to observe PIKESPEAK in Colorado than in Wyoming, but only slightly more likely to see a MOUNTAIN, it is clear that a PIKESPEAK observation would be more valuable to communicate to a teammate.

In our experiment, we demonstrate that the BUILDMESSAGE heuristic is able to identify and choose to communicate important observations. We compared its performance to the performance achieved by choosing random observations. A centralized policy for the team was generated using the Q-MDP heuristic [16]. We ran 1000 trials of each heuristic, with 10 timesteps per trial. The bandwidth limitation that we applied allowed agents to communicate one observation every two timesteps. Table 4 shows the results of the experiment. The BUILDMESSAGE heuristic clearly outperforms a random selection of ob-

servations, demonstrating that it successfully identifies observations that have high value of information.

	Average Reward	Average # Communications
HEURISTIC	4.70	4.29
RANDOM	0.94	4.56

Table 4. Results for the Colorado/Wyoming problem.

We alse performed an experiment to demonstrate the utility of our approach even in domains in which agents can operate independently. In this experiment, we added an absorbing state to the domain. Each agent transitions to that state when it SIGNALs. Reward is additive, with no requirement that agents SIGNAL simultaneously. This is a problem that can be solved with independent single-agent POMDPs. However, as the results in Table 5 show, the team still benefits from communication. When agents communicate their observations to each other, they are able to solve the problem more efficiently, accruing greater reward. Our algorithm enables the team to communicate those observations that will improve team performance.

	Average Reward
INDEPENDENT POMDPS	3.78
DEC-COMM-SELECTIVE	4.23
FREE COMMUNICATION	5.27

Table 5. Mean discounted reward for the modified Colorado/Wyoming problem.

6 Conclusions and Future Work

This paper discusses the need to reason about **what** to communicate when coordinating a multi-agent team. We identify two paradigms of communication, and show that it is insufficient, particularly in the case where the communication paradigm is limited bandwidth availability, to reason only about **when** to communicate. We provide a polynomial-time heuristic for selecting those observations that are, within the parameters of limited communication, most valuable for team performance and demonstrate the success of this algorithm experimentally.

In this work, we make decisions about which observations to communicate. There are domains in which a finer granularity would be beneficial, where the question to be answered is which features of the state are most relevant to team performance. Factored representations operate over these state and observation features, and we intend to investigate their applicability to our work. We also intend to apply our approach to domains in which observation probabilities vary more from state to state. We believe that these domains pose an interesting challenge to the problem of reasoning about value of information.

References

1. Bernstein D S, Zilberstein S, Immerman N (2000) The complexity of decentralized control of Markov decision processes. In: Uncertainty in Artificial Intelligence
2. Nair R, Pynadath D, Yokoo M, Tambe M, Marsella S (2003) Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In: International Joint Conference on Artificial Intelligence
3. Peshkin L, Kim K-E, Meuleau N, Kaelbling L P (2000) Learning to cooperate via policy search. In: Uncertainty in Artificial Intelligence
4. Pynadath D V and Tambe M (2002) The communicative multiagent team decision problem: Analyzing teamwork theories and models. In: Journal of AI Research
5. Roth M, Simmons R, Veloso M (2005) Reasoning about joint beliefs for execution-time communication decisions. In: International Joint Conference on Autonomous Agents and Multi Agent Systems
6. Xuan P, Lesser V, Zilberstein S (2000) Formal modeling of communication decisions in cooperative multiagent systems. In: Workshop on Game-Theoretic and Decision-Theoretic Agents
7. Goldman C V and Zilberstein S (2003) Optimizing information exchange in cooperative multi-agent systems. In: International Joint Conference on Autonomous Agents and Multi Agent Systems
8. Nair R, Roth M, Yokoo M, Tambe M (2004) Communication for improving policy computation in distributed POMDPs. In: International Joint Conference on Autonomous Agents and Multi Agent Systems
9. Emery-Montemerlo R, Gordon G, Schneider J, Thrun S (2004) Approximate solutions for partially observable stochastic games with common payoffs. In: International Joint Conference on Autonomous Agents and Multi Agent Systems
10. Doshi P and Gmytrasiewicz P J (2005) Approximating state estimation in multiagent settings using particle filters. In: International Joint Conference on Autonomous Agents and Multi Agent Systems
11. Roth M, Vail D, Veloso M (2003) A real-time world model for multi-robot teams with high-latency communication. In: International Joint Conference on Intelligent Robots and Systems
12. Bhasin K, Hayden J, Agre J R, Clare L P, Yan T Y (2001) Advanced communication and networking technologies for Mars exploration. In: International Communications Satellite Systems Conference and Exhibit
13. Rosencrantz M, Gordon G, Thrun S (2003) Decentralized sensor fusion with distributed particle filters. In: Uncertainty in Artificial Intelligence
14. Papadimitriou C H and Tsitsiklis J N (1987) The complexity of Markov decision processes. In: Mathematics of Operations Research
15. Cassandra, A R (2005) Tony's POMDP page.
At: http://www.cassandra.org/pomdp/code/index.shtml
16. Littman M L, Cassandra A R, Kaelbling L P (1995) Learning policies for partially observable environments: Scaling up. In: International Conference on Machine Learning
17. Kaelbling L P, Littman M L, Cassandra A R (1998) Planning and acting in partially observable domains In: Artificial Intelligence