Myopic and Non-Myopic Communication Under Partial Observability

Alan Carlin and Shlomo Zilberstein
Department of Computer Science
University of Massachusetts
Amherst, MA 01003, USA
acarlin@cs.umass.edu, shlomo@cs.umass.edu

***Abstract*—In decentralized settings with partial observability, agents can often benefit from communicating, but communication resources may be limited and costly. Current approaches tend to dismiss or underestimate this cost, resulting in overcommunication. This paper presents a general framework to compute the value of communicating from each agent's local perspective, by comparing the expected reward with and without communication. In order to obtain these expectations, each agent must reason about the state and belief states of the other agents, both before and after communication. We show how this can be done in the context of decentralized POMDPs and discuss ways to mitigate a common myopic assumption, where agents tend to overcommunicate because they overlook the possibility that communication can be deferred or initiated by the other agents. The paper presents a theoretical framework to precisely quantify the value of communication and an effective algorithm to manage communication. Experimental results show that our approach performs well compared to other techniques suggested in the literature.**

I. Introduction

In multiagent settings, each agent is faced with three types of uncertainty. The first is uncertainty about the effects of its actions. This uncertainty is often addressed using the Markov Decision Process (MDP). The agent's world consists of *states*, and the agent's *actions* have probabilistic outcomes that change the state. The agent can receive a *reward* for entering a desirable state. The second type of uncertainty is about the state that the agent is in. This uncertainty can be addressed by adding *observations* to the model. The agent can reason about its state by combining its knowledge about state transitions with knowledge of its past actions and observations. The third type of uncertainty is about the state that the other agents are in and the future actions that they will take, while accounting for the fact that the other agents perform similar reasoning. In this paper, we consider the Dec-POMDP (Decentralized Partially Observable MDP) model [2], and how this third type of uncertainty manifests itself within a Dec-POMDP.

One way to alleviate the latter type of uncertainty is to communicate with the other agents. In fact, it would usually be unrealistic to assume that agents do not communicate in a cooperative setting. But assuming ubiquitous communication is unrealistic for two reasons. First, trivially, if communication were ubiquitous, then in fact the problem could be solved and executed by one *centralized* agent, removing a key feature of multiagent systems. Second, in the real world communication is often not ubiquitous. Agents may be separated by distance, or the bandwidth between them may be limited, or they may operate in a low power environment where energy must be conserved. A common approach to factor this into the model is to assign communication a negative reward or *cost*.

In this paper, we will use the *sync* model of communication [14]. That is, when one agent decides to communicate, the result will be that all agents mutually exchange all available information. Because we assume that agents can synchronize in this manner, the paper studies the question of *when* to communicate. There is a rich, separate branch of the literature that studies *what* to communicate as well [10].

The paper proceeds as follows. First we discuss previous work on communication. Then we discuss the specific model that we use to produce communication decisions. An algorithm is developed that converts the complicated multiagent domain into a Hidden Markov Model in order to estimate the state of the other agents. The algorithm is expanded so that each agent can account for the communication policies of the other agents as well as their states. Finally, we show that the resulting planner performs well empirically.

II. Related Work

The literature on communication algorithms can be divided into works that start with a centralized plan and those that do not. In the former group, agents generate a centralized policy at planning time, and then at execution time they communicate to enforce execution of the centralized plan. Xuan et al. consider the view of a "monitoring agent" whose knowledge consists only of jointly observable information since the last synchronization time [14]. Agents communicate whenever the monitoring agent notices ambiguity in what an agent should plan next. Roth et al. use the *tell* model of communication instead of the *sync* model [9]. Each agent uses its local history and the Q_{POMDP} heuristic to reason about the joint action that should be taken. The history is also used to reason about communication.

Other works do not start with a centralized policy. Nair et al. introduce the Communicative DP-JESP (Dynamic Programing Joint Equilibrium-Based Search for Policies) technique, which integrates a communication strategy into K-step pieces of the JESP algorithm and finds a Nash equilibrium of policies for multiple agents [6]. In order to keep the algorithm tractable, the authors enforce a rule that communication must occur at least every K steps.

Some recent work explores the concept of delayed communication. Spaan et al. find the best domain-level policy given that communication delays are stochastically possible [8].

The above approaches do not explicitly represent any cost to communicating. Overcommunicating is thought to be undesirable, either out of general principle, or because it can add to planning time. Williamson et al. compute an explicit reward for communicating [13]. They introduce the *dec_POMDP_Valued_Com* model, which includes a communication reward function. Reward for communicating is based on the KL Divergence in the agents' belief states.

The approach most similar to ours has been developed by Becker et al. [1]. Communication incurs a negative reward, determined by the domain. Each agent determines the *Value of Communication* (VoC), which is the difference between the expected value of future policies with and without communication. However, the technique assumes that the world has joint full observability, that each agent fully observes its own local state, and furthermore that the other agents cannot affect its transitions or observations. The only interaction between the agents is via the joint reward function. The resulting problem is "only" NP-Complete [4], as the elimination of observations means that each agent only needs to reason about the global state, and not the belief states or observation histories of the other agents. In this paper, we will use a similar methodology to solve instances of the more complicated Dec-POMDP model, where each agent receives partial observations, and the agents are not transition or observation independent. Computing the value of communication in this more general context is substantially more complicated and is one of the key contributions of this paper.

We retain the *sync* model of communication, as we will see in the next section. Other models of communication include the Dec-POMDP-Comm [3] and Comm-MTDP [7]. In both of these models, which are equivalent to each other under an assumption of perfect recall, agents are able to select from an alphabet of messages, and messages have unique costs.

III. Dec-POMDP Model

A Dec-POMDP is a Decentralized Partially Observable Markov Decision Process with Communication [3]. It is defined by the following components:

- A set of agents numbered $1..n$
- S, the set of domain states.
- $b_0 \in \Delta S$, the initial belief state distribution.
- $A = \times_i A_i$,the set of joint actions, where A_i is the set of actions available to agent i. At each time step, agents take one joint action $\mathbf{a} = \langle a_1, .., a_n \rangle$.
- T, the transition model: $T(s'|s, \mathbf{a})$ is the probability of transitioning to state s' given the previous state was s and joint action $\mathbf{a}$ was taken by the agents.
- R, the reward function: $R(s, \mathbf{a})$ is the immediate reward for taking joint action $\mathbf{a}$ in state s.
- $\Omega_1..\Omega_n$, the sets of observations possible for each agent. Each agent i receives only its own observation $o_i \in \Omega_i$ at each timestep. The vector of received observations is $\mathbf{o} = \langle o_1, .., o_n \rangle$.
- O, the observation function. It specifies joint observation probability $O(\mathbf{o}|s', a_1..a_n)$, the joint probability that every agent i sees corresponding observation o_i after the agents took joint action $\mathbf{a}$ causing a state transition to s'.
- H, the horizon, or number of steps, in the problem.

We add communication to the model. Each agent has the option to initiate communication before taking an action. We restrict this paper to the *sync* communication model, so the communication language simply allows transmission of the agents' action/observation histories before each action. Communication is instantaneous, a message is received without delay as soon as it is sent. We also include C, a fixed cost on each step of communicating these synchronization messages. The fixed cost of C is incurred if *any* number of agents choose to communicate. Otherwise, if no agent communicates, they incur no penalty. This problem is NEXP-hard. Indeed, when communication is prohibitively expensive, the model becomes a Dec-POMDP with no communication.

Since the problem has a finite horizon H, we can use a policy tree to represent a non-communicative policy of an agent. In the policy tree representation, nodes represent actions and branches represent observations. Each agent i follows its own policy tree generated at the last synchronization step, referred to as π_i^0 with its first action corresponding to the root at time $t = 0$, and its last action corresponding to the leaves. π_i^0 contains a number of subpolicies, each corresponding to an observation sequence as the tree is traversed. We refer to an observation sequence as $\bar{o}$ and the resulting subpolicy as $\pi_i(\bar{o})$. Note that if we know an agent's initial policy and its sequence history of observations, we can derive its sequence of actions. Furthermore, the next sections will show that the local history of an agent can be combined with Bayesian reasoning on the Dec-POMDP model and the initial policies of the other agents to form a belief about the histories of other agents. To summarize, each node of an agent's policy tree maps to:

- A unique action/observation sequence $\bar{o_i}$
- A future local subpolicy rooted at the node $\pi_i(\bar{o_i})$
- A belief about the global S as well as the action/observation histories of the other agents.

We will use these mappings throughout the paper. Unless stated otherwise we will also assume some housekeeping on the part of the algorithms that we describe, that knowledge of $\pi_i(\bar{o_i})$ implies knowledge of $\bar{o_i}$.

Let $b(s)$ be a belief state, and let q be a variable representing a successor state. Let a_i and a_{-i} be the root actions of policies $\langle \pi_i, \pi_{-i} \rangle$. Standard theory on Dec-POMDPs says that the value of a joint policy tree, $\langle \pi_i, \pi_{-i} \rangle$ at a given belief state is recursively defined as the expected sum of the rewards of the subpolicy trees. That is:

$$V(\langle \pi_i, \pi_{-i} \rangle, b) = \sum_{s,q,o_i,o_{-i}} \Big[b(s) T(q|s, a_i, a_{-i}) \\ O(o_i, o_{-i}|q, a_i, a_{-i}) V(\langle \pi_i(o_i), \pi_{-i}(o_{-i}) \rangle, q) \Big]$$

The above equation says that the value of the joint policy at b_0 can be decomposed into cases where the root actions result in a transition to state q, resulting in observations o_i and o_{-i}. The base case of the recursion occurs at the last step of a finite horizon problem, where value simply corresponds to the reward $R(s, a_i, a_{-i})$ of the last actions taken.

IV. Solution Method

In our method, plans and communication strategies are determined offline and stored for use at runtime. The planner starts by precomputing optimal joint policies without communication. Any non-communicative planner which generates policy trees can be used for this step. It also precomputes non-communicative joint policies for various reachable belief states of horizons $1...T$ (more details on this are in the next section), and stores these policies and their value in a cache. It uses these to construct a cache function for reachable belief distributions on the global state, and at runtime the cache will be accessed by each agent through a function call:

$$\text{CACHE}_i(b(S), h) \rightarrow \langle \pi_i^*(b(S)), \pi_{-i}^*(b(S)) \rangle$$

where i is the identity of the local agent accessing the cache, $b(S)$ is the belief state it wants to query, h is the depth of the policy and π^* represents that the policy is specific to that belief state. It also optionally stores a mapping of some or all observation sequences to communications decisions (if these are not stored, they could be recomputed by the agent at execution time).

$$\langle b(S), \bar{o_i} \rangle \rightarrow \{\textit{true}, \textit{false}\}$$

where $\bar{o_i}$ is a vector composed of the observations agent i has made on prior steps. At execution time, each agent follows its policy and its communications policy. Upon communication, it retrieves the appropriate policy from the cache for the discovered belief state. We note trivially that if agents' policies are known to each other, then a joint observation sequence $\langle \bar{o_i}, \bar{o}_{-i} \rangle$ also determines a unique action history, and a unique $b(S)$ can be constructed by starting at the initial belief state and performing a forward computation as in a POMDP.

Before each action, each agent must decide whether to communicate. To do this, it uses the *Value of Communication*. Let $P(q, \bar{o}_{-i}|\bar{o_i}, \langle \pi_i, \pi_{-i} \rangle, b_0)$ represent the probability of reaching state q while the other agents receive observations $\bar{o}_{-i}$ after $|\bar{o_i}|$ steps, given a starting belief state b_0 with policies $\langle \pi_i, \pi_{-i} \rangle$, and local observations $\bar{o_i}$. (The computation of this probability will be deferred to the next section). Let $\langle \pi_i, \pi_{-i} \rangle$ be the joint policy before communication and $\langle \pi_i^*(b_h), \pi_{-i}^*(b_h) \rangle$ be the joint policy that results from communication and discovery of joint belief state b_h.

Definition 1: The Value of Communication (VoC) is the difference between the expected value when communicating and the expected value for remaining silent.

$$VoC(\bar{o_i}, \langle \pi_i, \pi_{-i} \rangle, b_0) = \sum_q \sum_{\bar{o}_{-i}} P_{q,\bar{o}_{-i}} (V^* - C - V)$$

where

$$\begin{aligned} P_{q,\bar{o}_{-i}} &= P(q, \bar{o}_{-i}|\bar{o_i}, \langle \pi_i, \pi_{-i} \rangle, b_0) \\ V^* &= V^*(\langle \pi_i^*(b_h), \pi_{-i}^*(b_h) \rangle, q, t) \\ V &= V(\langle \pi_i(\bar{o_i}), \pi_{-i}(\bar{o}_{-i}) \rangle, q, t) \end{aligned}$$

b_h is the belief distribution at time h given $\langle o_i, o_{-i} \rangle$ and b_0.

To understand the above definition, consider the perspective of agent i. It has synchronized with the other agents and determined that they synchronized in belief state b_0, it knows that the other agents have been following policies π_{-i} since then, and that it has observed $\bar{o_i}$ since synchronization. In order to contemplate the value of remaining silent, it must consider the joint probability that the other agents' have observed $\bar{o}_{-i}$, and that the real current state is q. If this is the case, it knows that the agents will continue along subpolicies $\langle \pi_i(\bar{o_i}), \pi_{-i}(\bar{o}_{-i}) \rangle$, and the value of staying silent is simply the value of the joint subpolicy from state q. If the agents do communicate, they will combine observations to form a new joint belief state b_h, and they will follow a new joint policy for the belief state, $\langle \pi_i^*(b_h), \pi_j^*(b_h) \rangle$. The new joint belief state does not affect the fact that the true state is q, and so it computes the value of the new joint policy for q.

For example, consider the well-known multiagent Tiger problem [5]. In this problem, agents take a joint action of opening either the right or the left door, or they listen for the tiger. If both agents choose to listen, each agent will then receive its own observation with some defined probability of error, and because of this probability of error it is possible that both agents will not receive the same observation. Consider the perspective of an agent after it has listened and observed Tiger-Left. In order to evaluate the value of communicating, the agent must consider each scenario that occurs after communication, one of which is the chance that the other agent has also observed Tiger-Left, that they use the combined observations to open the door on the right, but that the true state was Tiger-Right, resulting in a large penalty. Although the possibility of this happening is small, it will motivate communication if the penalty is large enough.

A. Estimating the Joint History

We now explain how $P(q, \bar{o}_{-i}|\bar{o_i}, \langle \pi_i, \pi_{-i} \rangle, b_0)$ is computed. There are three sources of difficulty in this computation: (1) the local agent's history of actions has affected

Algorithm 1: Find SSTs for other agents at current step

input : Synchronized Belief State b, Nonlocal Policies Q_{-i}, Local Observation History $\bar{o}_i$, Local Action History $\bar{a}_i$, steps

output : An array of SSTs, each containing the true state, the remaining policies of the other agents, and a probability

begin
- $D, E \leftarrow$ arrays of StateSubTrees, initialized to empty
- **for** $i = 1$ *to* $|S|$ **do**
  - $D[i] \leftarrow \langle i, Q_{-i}, b(i),$ *false*$\rangle$
- **for** $step = 1$ *to* $steps$ **do**
  - $E \leftarrow$ empty
  - **for** $i = 1$ *to* $|D|$ **do**
    - $\bar{a}_{-i} \leftarrow$ the root actions of $D[i].tree$
    - $a_i \leftarrow \bar{a}_i[step]$
    - $o_i \leftarrow \bar{o}_i[step]$
    - **for** $s' = 1$ *to* $|S|$ **do**
      - **for** $o_{-i} = 1$ *to* $|\Omega_{-i}|$ **do**
        - $SST \leftarrow$ new SST
        - $\alpha \leftarrow (D[i].p)T(s, a_i, a_{-i}, s')$
        - $O(s, a_i, a_{-i}, o_i, o_{-i}, s')$
        - **if** *nonmyopic* **then**
          - Lookup $SST.comm$
          - **if** $SST.comm == true$ **then**
            - prune SST
        - $SST.s = s'$
        - $SST.p = \alpha$
        - SST.Q = $D[i].Q.subTrees[o_{-i}]$
        - Add SST to E
  - Merge SSTs with equivalent subpolicies
  - Prune SSTs with $p <$ threshold from E
  - Normalize each $SST.p$ in E
  - $D \leftarrow E$
- return D

end

the transition matrix of the global state; (2) the other agents have adjusted their actions based on their observation history, not the true state; and (3) each local agent only holds its own observations, not necessarily the observations of the other agents.

Definition 2: Let a State SubTree (SST), be a tuple $\langle s, Q, p, comm\rangle$, where s is a state, Q is a finite-horizon policy, p is a probability, and comm is a boolean.

Algorithm 1 shows how $P(\bar{o}_{-i}, q|\bar{o}_i, \langle \pi_i, \pi_{-i}\rangle, b_0)$ is estimated. The algorithm takes as input initial belief state b_0, the action and observation histories of the current agent i, and the known policies of the other agents at b_0. It outputs a set of SSTs at the current time step, each SST assigns a probability to one world state, composed of the actual state and the current policy of the other agents. SSTs are computed in a forward fashion. The set of SSTs is initialized to contain one element for each nonzero entry in b_0, with its p being its probability in b_0, and its Q being the initial policies of the other agents. At each time step, the current set of SSTs are used to generate a new set. Each SST in the new set represents a joint action taken by the other agents, a joint observation received, and a global state transition from an old SST, resulting in the new SST's state and subpolicy. The forward probability α is the probability of the old SST times the probability that the other agents made this transition, given the local agent's knowledge of its own action and observation on that step.

We also take the opportunity to merge SSTs with the same subpolicy. That is, if two observation histories of the other agent lead to the same subpolicy, there is no need to distinguish the two cases. Formally, if there are two SSTs:

$$\langle s, Q, p1, comm\rangle \quad \text{and} \quad \langle s, Q, p2, comm\rangle,$$

they can be merged into a single SST:

$$\langle s, Q, p1 + p2, comm\rangle.$$

This can be particularly useful in practice, if the non-communicative plans were built by an algorithm such as IMBDP [12], which builds plans where only a limited set of subpolicies are generated, and different observations lead to the same subpolicy.

There are other augmentations that can be made to Algorithm 1 which are not explored in this work. (1) The cache can be smaller and only contain likely decision points. At runtime, when a non-cached state is encountered, the agent can either initiate an online computation, or it can use the joint-policy from the least (Manhattan) distant cached belief-state. (2) SSTs can be generated by sampling from agent histories, rather than direct computation.

Theorem 1: The problem of estimating $M.p$ has an equivalent Hidden Markov Model (HMM) representation. Furthermore, the algorithm is correct. That is, suppose agent i calls Algorithm 1 with threshold 0 at time t after observing $\bar{o}_i$, and the algorithm returns a set Z of SSTs. Then $\forall M \in Z$, if $M = \langle s, Q, p, comm\rangle$, then p is the probability that the global state is s and the other agents' policies on this step are Q at time t.

Proof: We can convert the problem of estimating $M.p$ into an HMM, and then solve using the forward-backward algorithm [11]. Each state of the HMM corresponds to a global state and an observation history of the other agents (we use the fact that each joint observation history maps to a specific joint subpolicy such as Q). State transition probabilities of the HMM correspond to state transition probabilities of the Dec-POMDP, given the local agent's action histories, times the probability of making the last observation. The transition probability is zero if the new observation history can not follow from the old. That is, a state with an observation history w_1w_2 can not transition to a state with an observation history $w_2w_2w_3$, but it can transition to a state with observation history $w_1w_2w_3$.

Given this transition model, it is clear through induction (with the base case consisting merely of S when Algorithm 1 is initialized) that the forward computation used to generate the leaves in the last step of Algorithm 1 are the same as the

steps used to generate the corresponding states in the Hidden Markov Model. ■

We note that this proof of Algorithm 1 is similar in spirit to proofs of algorithms in the literature such as JESP-belief, with the addition of the SST data structure that allows it to reason more precisely and formally over this belief space, as well as some nuances such as the merging operation described above. However, we are unaware of an equivalency proof to an HMM in the literature, and we are hopeful that such an equivalency can be used in future work to leverage the rich HMM literature in Decentralized POMDPs.

Proposition 1: Assume the agents synchronize at belief state b_0 and form policies $\langle \pi_i, \pi_{-i} \rangle$, and the cost of communication is C. Assume a myopic perspective (the local agent may communicate only once, and the other agent can not communicate at all). The error introduced when agent i makes a single communication decision after observation sequence $\bar{o}_i$ is at most:

$$P(\bar{o}_i | \langle \pi_i, \pi_{-i} \rangle, b_0) \cdot C$$

Proof: Assuming Theorem 1, if an agent computes its expectation of communication and decides not to communicate, it can never be wrong in the expected case. However, if it decides to communicate, it may be making an error. Since $V^* \geq V$, that is, the policy after communication is always at least as good as the one before, the error on this case bounded by C, and weighing by the probability of encountering the case in the first place, we have $P(\bar{o}_i | \langle \pi_i, \pi_{-i} \rangle, b_0) \cdot C$. Note that we are only considering the single communication decision; not possibilities that involve multiple future communication actions. This analysis is considerably more complex and will be handled in future work. ■

The communication may not be necessary, as there are cases where (1) the other agent may initiate communication in all of the necessary states, or (2) communication can be deferred to a future step when more information is known.

Note that the number of SSTs can grow exponentially in each step, in the worst case. In practice, however, the number only grows with reachable belief states, and often on real-world problems only a small number of observations will be possible on each step. In order to keep the algorithm tractable, the algorithm can optionally prune SSTs with low probabilities at each step.

V. Non-myopic Reasoning

Algorithm 1 does not take into account the communication policy of the other agents, nor does it take into account the fact that communication need not be immediate, it may be deferred to future steps. In this section, we discuss how we improve the algorithm past this *myopic assumption*. The algorithm can be improved in three ways, first by using the fact that other agents did not communicate since the last *sync*, second by using the fact that other agents can communicate in the present, and finally by using the fact that communication can be deferred to the future.

A. Other agents in the past

We can use the knowledge that the other agents have not communicated since the last synchronized state. To do this, we use the *comm* field in the SST structure. At planning time, each agent computes VoC given its possible observation sequences and synchronized belief states. If VoC is positive, it sets *comm* to true. The *comm* value is stored for this history.

As Algorithm 1 is executed, each SST represents one possible observation history of this agent, and its children represent a continuation of that history. If the *comm* field is set to true for a corresponding observation history, this means that the agents would have communicated at this point. But any agent executing Algorithm 1 knows that didn't happen, since the algorithm initialized at the last communication point. Therefore it is known that the observation histories represented by such an SST never occurred, and the SST can be pruned.

(As an aside, the algorithm only generates accurate probabilities for SSTs in the current step. SST probabilities from previous steps do not necessarily reflect the probabilities of those histories. This is due to the nature of the forward-backward algorithm. In the last step, only a forward computation is necessary, such as that provided in the algorithm. For past steps, however, backwards information from subsequent steps would be necessary. Let this backwards probability be β, and define it to be the probability that an SST's policy Q was executed from its state s given future observations. Finding β can be computationally complex, as each leaf of Q must be evaluated. Therefore in implementation we limit the analysis of Q to the next h steps, and only computing β_h. For instance define β_1 for an SST with state s' and whose policy Q corresponds to an observation history $\bar{o}_{-i}$ to be:

$$\beta_1 \leftarrow \sum_{s''} \sum_{\bar{o}'' \in O''} T(s'' | \mathbf{a}, s') O(\bar{o}'' | \mathbf{a}, s'')$$

where **a** is a joint action composed of the known local action for that step as well as the root action for each other agents' policies in the SST, and O'' is the set of joint observations which must include the known observation from the local agent's history.)

B. Non-myopia with respect to other agents

Having modeled the communication strategy of the other agent on past steps, we turn to modeling the present step. To do this, we construct a matrix. For the two agent case, each row of the matrix corresponds to the SSTs for one agent, and each column corresponds to the SSTs for the other agent (for the multiagent case, each dimension represents another agent). Entries in the matrix correspond to the VoC given the history represented by the corresponding joint history, multiplied by the probability of that joint history. Each agent has the ability to communicate or not to communicate given a history. Communicating after a history corresponds to turning

a row (or column, for the other agent) "on" or "off". The value of a joint communication strategy is the sum of the "on" values in the matrix. The myopic strategy discussed in above sections corresponds to turning each row or column on if its entries sum to a positive number. However, this illustrates the flaw of myopia, it does not maximize the value of the whole matrix, only its individual rows and columns. Since the row agent and column agent are not coordinating, they may double count entries. We improve on this by finding a better joint strategy. The approach is similar to the one described in [1], except (1) The rows and columns and probabilities correspond to observation histories, not states. (2) To reduce time of computation, agents can only alter K rows, where K is a parameter specified by the users. The remaining rows are toggled through myopic computation. As noted in [1], it takes an exponential amount of time with respect to the matrix size to find an optimal row/column strategy, but finding a Nash equilibrium is a reasonable alternative which can be done in polynomial time. Thus, in our implementation we find a Nash equilibrium.

An example can be found in Figure 1. Each entry in the table corresponds to VoC for a single joint history. Using an agent-myopic strategy, Agent 1 has decided that it should communicate given the history represented by s_1^2, because its VoC of 2 (the sum of its row) is positive, and Agent 2 has decided that it should communicate from state s_2^1, because its VoC of 1 is positive. VoC decisions are shown in the figure as π_{1c} and π_{2c}. In the figure, all joint-histories that result in communication under a myopic strategy are bolded. This strategy double counts certain elements in the table and can result in choosing a communication policy worse than not communicating at all. The expected value of a joint communication policy for one step is the sum of all entries in the table where communication happens. Because we are using the sync model of communication, an entry is only counted once, even if both agents initiate communication. This corresponds to all joint-states where communication happens, weighted by their probability. In the example, the myopic policy has a value of -1, computed by summing the bold entries. The reason for this negative value is because the joint history in the first column of the second row was counted twice for determining the policies (once for each policy), but only once for determining the value of the table. If agent 2 did not communicate in s_1 then the value would be 2. Never communicating ($\pi_{ic} = \{no, no, no\}$) will always have a value of 0.

Creating the table costs no more than the original approach since each entry represents a reachable joint history. Note that in problems with structure, or where communication has occurred on a recent step, the number of reachable joint histories is limited. For larger problems, though, there will be a large number of reachable joint histories, and in future work we plan on reducing the dimensionality of the matrix while minimizing loss of information. Equilibrium solutions to the reduced matrix problem will correspond to reasonable joint communication policies.

	s_2^1	s_2^2	s_2^3	π_{1c}	VoC
s_1^1	**-1**	0	-1	*no*	-2
s_1^2	**4**	**-1**	**-1**	***yes***	**2**
s_1^3	**-2**	-1	1	*no*	-2
π_{2c}	***yes***	*no*	*no*		
VoC	**1**	-2	-1		

Fig. 1. A simple table M showing the expected gain in value for communicating for the two agent case. Each row represents an agent history for agent 1. The table represents 3 possible histories for each agent, or 9 belief states overall.

C. Value of deferring communication

The value of deferring communication to the future can be computed. For a given SST, the value of delay is the reward achieved by not communicating on the current step, added to the expected reward after communicating on the next step. The immediate reward is $pR(s, \mathbf{a})$ and it is added to:

$$p \sum_{s', \mathbf{o}'} T(s'|\mathbf{a}, s) O(\mathbf{o}'|\mathbf{a}, s') \mathbf{V}(\langle \pi_i^*(b_{h+1}), \pi_{-i}^*(b_{h+1}) \rangle, s')$$

where p is the probability associated with the SST, $\mathbf{a}$ is the joint action specified by continuing the current policy of the local agent and the SST, s is the state in the SST, $\mathbf{o}'$ the next joint observation, and b_{h+1} is the belief state that would result at the next step. $\mathbf{V}$ is used to represent the fact that VoC must be retrieved for the local agent's observation in $\mathbf{o}'$, and if it is positive then $\mathbf{V} = V^*$ and b_{h+1} is the belief state that results from communication while if $\mathbf{V}$ is negative, $\mathbf{V} = V$ and the joint policy merely continues. To compute the value of delaying communication, the computation above is summed for all SSTs returned by algorithm 1. If the sum is greater than or equal to the value of communicating on the current step, the agent does not communicate. A new value of delay will be computed after the next action is executed. Because of this, it is possible that the decision to postpone communication will cascade across several steps.

VI. Experiments

We considered our algorithm, labeled *VoC-NM* (Value of Communication - Non-Myopic), as compared to the algorithms of *No Communication* (labeled No-Comm), *Full Communication* (communicating on every step), *Periodic Communication*, as well as the algorithm of Roth et al. For the *Periodic* strategy, we ran an algorithm which communicated every K steps, and we used results from the best value of K from 1 to the horizon of the problem. Thus, *Periodic* will provably outperform *No Communication* and *Full Communication*, so

horizon	Cost	No-Comm	Periodic	VoC-NM
3	0	5.19	12.5	12.5
3	5	5.19	5.46	7.99
3	10	5.19	5.19	6.03
5	0	4.92	26.2	26.2
5	5	4.92	6.3	9.14
5	10	4.92	4.92	5.62
8	0	9.00	41.8	41.8
8	5	9.00	12.3	24.3
8	10	9.00	9.00	10.6
10	0	9.4	53.2	53.2
10	5	9.4	12.87	22.7
10	10	9.4	9.4	11.9

TABLE I
COMPARISON OF VARIOUS COMMUNICATION STRATEGIES FOR THE TIGER PROBLEM.

horizon	C=0	C=5	C=10	C=15
3	3.0	.44	0	0
5	5.0	1.4	.92	0
8	8.0	1.9	.79	.02
10	10	2.5	.72	0

TABLE II
AVERAGE NUMBER OF COMMUNICATIONS FOR EACH RUN OF THE *VoC-NM* STRATEGY ON THE TIGER PROBLEM. EACH ROW REPRESENTS RESULTS FOR A DIFFERENT HORIZON.

horizon	Cost	No-Comm	Periodic	VoC-NM
5	0	59.6	78.7 (4.0)	78.7 (4.0)
5	15	59.6	64.3 (1.0)	64.9 (.89)
5	30	59.6	60.3 (1.0)	64.1 (.80)

TABLE III
COMPARISON OF VARIOUS COMMUNICATION STRATEGIES FOR THE BOXPUSHING-5 PROBLEM. PARENTHESES SHOW THE MEAN NUMBER OF COMMUNICATIONS FOR EACH SIMULATION.

we do not separately list results for full communication. Our algorithm was implemented as follows: we precomputed values of communication for each agent for reachable histories at planning time by running a large number of simulations, and then stored this in a cache. We used a pruning threshold of 0, thus we did not prune SSTs. We used the IMBDP planner [12] as the non-communicative submodule for this step. Then we ran a new 100,000 simulations of the non-myopic algorithm, referencing this cache on each simulation. Since MBDP-based planners only store a handful of subpolicies for each horizon step (using the same subpolicies for various branches of the larger policy tree), this choice of planners kept the size of the cache smaller.

Results for the Multiagent Tiger problem [5] on various horizons are shown in TABLE I. Results show that the *VoC-NM* planner was able to successfully communicate for both lower and higher costs of communication. We also performed experiments using the planner from Roth et al. [9], which was available for use with the Tiger domain. Note that this planner was not constructed with Cost of Communication in mind; it develops the policies first and then each agent communicates when it considers the state of the other agent ambiguous. It also uses the *tell* model of communication. Thus we do not present the results side-by-side in the table. Still, it is interesting to compare [9] as an alternative to *VoC-NM*. *VoC-NM* outperformed the Roth et al. planner on all experiments. On the horizon 10 problem, *VoC-NM* outperformed the Roth et al. planner by 11% and 36% respectively on communications costs 5 and 10. Naturally, the difference in performance continues to increase as the cost of communication increases. We also sought to compare to other works in the literature by requesting Comm-MTDP [7] and Communicative DP-JESP [6]. The former is implemented within a framework which evaluates based on input non-communicative policies but does not generate the policies themselves, and the latter was unavailable for experimentation.

Results for the *VoC-NM* strategy are more closely examined in Table II. Execution of the *VoC-NM* strategy was simulated 100,000 times for each cost of communication (C). Each entry is the mean number of communications for per simulation, given that cost of communication. For example, the column C=0 represents a configuration where there was no cost of communication, and thus the agents communicated at every step. As expected, the table shows that the expected number of communications decreases as the cost of communication increases.

Running time for *VoC-NM* was 9 seconds for the precomputation, and 2 seconds for the $100,000$ simulated runs after that. We also ran a myopic variant of the *VoC* planner, it did not include the algorithm enhancements of Section 4.2. The result across all tests was an approximately 10% decrease in score at C equal to 5 or 10.

We also ran the larger BoxPushing problem [12] for horizon 5, a problem in which the value of the generated centralized and decentralized plans only differ by 20. Still, results similarly show that a *VoC-NM* methodology outperformed the other strategies because it communicates less, resulting in a gradual decrease in value as communication cost gets higher. The time taken for BoxPushing-5 was 4300 seconds at the planning stage, and then $.38$ seconds to run each simulation at execution time.

Across all experiments, a simple communication policy such as *Periodic* can be adequate when communication cost is low, or when communication points can easily be picked from the domain. As the cost of communication cost gets higher, and agents are motivated to avoid communication if possible, the richer *VoC-NM* approach is required. Even assuming, as we did, that the best period can be determined, a periodic communicator is forced to either choose to not communicate at all, or else to overcommunicate. This was shown as the *VoC-NM* approach reduced the amount of communication by 10% on BoxPushing when communication cost was 15, and by 20% when cost was 30.

VII. Conclusion

We have presented a general approach for reasoning about costly communication within the Dec-POMDP framework. Computing the value of communication is challenging because each agent receives different partial observations and must reason about the observations of the other agent, as well as the possible synchronized state of the system after communication. We have shown that computing the value of communication can be used effectively to determine the utility of communicating versus the utility of staying silent. We have also shown that there are several myopic assumptions made by a simple implementation of a value of communication algorithm, and that these assumptions can be mitigated by accounting for the communication of other agents as well as the possibility of future communication. The fact that we have shown that this can be done under conditions of partial observability represents an important contribution.

The approach allows each agent to make an exact estimate of the state of the other agents. We implemented and tested this capability using several standard benchmark problems. The results show that our approach uses communication effectively and outperforms a naive algorithm based on periodic communication.

One area not explored in this paper is the tradeoff between building the cache of new policies at planning time, versus building it at runtime. Our implementation used for experiments generated the full cache at planning time, and always accounted for all possible observation histories. We do not claim that this is the best possible implementation, only a simple one that allows us to characterize an "offline" algorithm. In future work, it would be interesting to trade accuracy for speed. This can be done by pruning more improbable belief state histories for the other agents as the algorithm progresses. Furthermore, it may be possible to drop the requirement that the exact value of the post-communication policies be used. Instead, perhaps a quicker heuristic could be used, such as the value of the centralized policy. These techniques create practical, yet disciplined ways to manage communication in decentralized multiagent systems.

Acknowledgment

This work was supported in part by the National Science Foundation Grant No. IIS-0812149 and the Air Force Office of Scientific Research Grant No. FA9550-08-1-0181.

References

[1] Raphen Becker, Alan Carlin, Victor Lesser, and Shlomo Zilberstein. Analyzing myopic approaches for multi-agent communication. In *Computational Intelligence*, 25(1):31–50, 2009.

[2] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. In *Mathematics of Operations Research*, 27:819–840, 2000.

[3] Claudia V. Goldman and Shlomo Zilberstein. Optimizing information exchange in cooperative multi-agent systems. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 137–144. ACM Press, 2003.

[4] Claudia V. Goldman and Shlomo Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research*, 22:143–174, 2004.

[5] Ranjit Nair and Milind Tambe. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 705–711, 2003.

[6] Ranjit Nair, Milind Tambe, Maayan Roth, and Makoto Yokoo. Communication for improving policy computation in distributed POMDPs. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1098–1105, 2004.

[7] David V. Pynadath and Milind Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. In *Journal of Artificial Intelligence Research*, 16:389–423, 2002.

[8] Matthijs T.J. Spaan, Frans A. Oliehoek, and Nikos Vlassis. Multiagent planning under uncertainty with stochastic communication delays. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 338–345, 2008.

[9] Maayan Roth, Reid Simmons, and Manuela Veloso. Decentralized communication strategies for coordinated multi-agent policies. In *Multi-Robot Systems: From Swarms to Intelligent Automata, volume IV*. Kluwer Academic Publishers, 2005.

[10] Maayan Roth, Reid Simmons, and Manuela Veloso. What to communicate? Execution-time decision in multiagent POMDPs. In *Proceedings of the Eighth International Symposium on Distributed Autonomous Robotic Systems*, pages 177–186, 2007.

[11] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[12] Sven Seuken and Shlomo Zilberstein. Improved memory-bounded dynamic programming for decentralized POMDPs. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*, 2007.

[13] Simon A. Williamson, Enrico H. Gerding, and Nicholas R. Jennings. A principled information valuation for communications during multi-agent coordination. In *AAMAS Workshop on Multiagent Sequential Decision Making (MSDM)*, 2008.

[14] Ping Xuan, Victor Lesser, and Shlomo Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *In Proceedings of the Fifth International Conference on Autonomous Agents*, pages 616–623. ACM Press, 2001.