ECAI 2016
G.A. Kaminka et al. (Eds.)

doi:10.3233/978-1-61499-672-9-444

Observation-Based Multi-Agent Planning with Communication

Luca Gasparini and **Timothy J. Norman** and **Martin J. Kollingbaum**[1]

Abstract. Models of decentralized online planning vary in the information that individual agents use to make local action decisions. Some models consider only local observations, eschewing coordination through communication. Others use communication to ensure that all agents are aware of the action decisions of others, but assume costless and delay-free communication. In this paper, we propose a model of online planning (OB-MAP) that uses estimates of the value of communicating to manage coordination through communication as costs vary. We compare this approach to existing models in widely employed benchmark problems, demonstrating that OB-MAP performs significantly better in many scenarios regardless of varying (including infinite) cost of communication.

1 Introduction

Decentralized planning problems are often modelled as Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs), where multiple agents, each with a local view of the environment, must coordinate their actions in a decentralized fashion in order to optimize some reward [1]. Goldman and Zilberstein [6] have demonstrated, however, that even approximately solving a Dec-POMDP is intractable. One of the reasons for this complexity is that the number of possible joint-histories grows doubly exponentially with the horizon. In order to address this problem, a number of online planning algorithms [4, 5, 13, 15] have been proposed, which interleave planning and enactment. These algorithms heuristically estimate the long-term value of an action and use some of the information available at runtime in order to make planning more tractable. The majority of existing algorithms (such as [5, 13, 15]) plan in such a way that each agent always has full knowledge of what actions are being performed by its team-mates. This is referred to as *strict coordination*, and is often argued to be a necessary condition for effective planning in decentralized settings [15]. In order to guarantee strict coordination, however, agents must be limited in the extent to which they exploit local observations. The argument is as follows. If agents start with a common belief (a probability distribution) about the state of the environment, and use the same planning algorithm, they will agree on a common joint action to be performed. Since each agent potentially receives a different local observation at each time step, if they take into account these observations, their beliefs may diverge. Each agent will, therefore, plan for a different joint-action, and will have incorrect beliefs about the actions of its team-mates. As a result, strict coordination is not guaranteed.

In contrast to strict coordination models, Chechetka and Sycara [4] propose BaGa-S, which extends BaGa (Bayesian Games approximation algorithm) [5] in order to take advantage of local observations. BaGa-S has been shown to provide significant advantages over strict coordination models in some scenarios. These scenarios are, however, those in which local observations provide the best evidence for good local action decisions to maximise the reward. In contrast, we show that this approach performs significantly worse in domains that require a tighter coordination, supporting the strict coordination argument, albeit in an important class of problem domains.

Another important issue to consider when planning at runtime is whether and when agents should communicate their *local observations* as opposed to action decisions. By sharing observation histories, a coalition of agents can synchronize on a common belief, and take advantage of this information while maintaining strict coordination. Then, in algorithms that trade off strict coordination for a more opportunistic exploitation of local observations, communication can be used to re-synchronize agents' beliefs once coordination is lost.

In this paper, we argue that agents are faced with an important trade-off between maintaining (almost) strict coordination, and exploiting local observations to maximize their expected reward. We analyse this trade-off by comparing the performance of different algorithms in widely employed benchmark scenarios. We propose an algorithm, OB-MAP, that attempts to capture the best of both worlds. While OB-MAP does not guarantee strict coordination, we show experimentally that it performs at least as good as strict coordination algorithms, and is able to take advantage of local observations in scenarios that favour a more opportunistic planning approach. We also propose a heuristic that takes into account the value of communication in order to decide whether or not agents should communicate.

Before formalising OB-MAP, analysing its complexity, and evaluating its performance, in the following section we provide necessary technical background. We present a précis of the Dec-POMDP approach to multi-agent planning, and then give details of the two on-line planning models that we use to represent the state-of-the-art in on-line planning algorithms.

2 Background

A Dec-POMDP [1] is a tuple, $\langle I, S, b^0, \{A_i\}, P, \{\Omega_i\}, O, R\rangle$ where:

- I is a set of agents, and S is the set of states;
- b^0 is an initial belief state, *i.e.* a probability distribution over possible initial states;
- A_i is a finite set of actions available to agent i and $\vec{a} = \langle a_1, \ldots, a_n\rangle$ is a joint-action consisting of one action for each agent;
- $P(s_j|s_i, \vec{a})$ represents the probability that taking joint-action $\vec{a}$ in state s_i will result in a transition to state s_j;

[1] Department of Computing Science, University of Aberdeen, Aberdeen, UK, l.gasparini@abdn.ac.uk, tnorman@acm.org, m.j.kollingbaum@abdn.ac.uk

- Ω_i is a finite set of local observations o_i available to agent i and $\vec{\Omega}$ is the set of joint observations $\vec{o}$ consisting of one local observation for each agent;
- $O(\vec{o} \mid s_j, \vec{a})$ specifies the probability of observing $\vec{o}$ when performing a joint-action $\vec{a}$ that leads to a state s_j;
- $R : S \times \vec{A} \to \mathbb{R}$ is a reward function, and $R(s_i, \vec{a})$ specifies the reward obtained by performing $\vec{a}$ in s_i.

We define a local history h_i for agent i up to time t as a sequence of interleaved local actions and observations. $h_i = (a_i^0, o_i^1, a_1, \ldots, o_i^t)$ and a joint-history as a tuple $\vec{h}$ consisting of one local history for each agent $\vec{h} = \langle h_0, \ldots, h_n \rangle$. A belief state at time t, $b^t : S \to \mathbb{R}$, is a function that represents the probability that the system is in each state. Given a belief b^t at time t, the belief state at time $t+1$ after the agents have executed joint-action $\vec{a}_i$ and received joint-observation $\vec{o}_j$ can be computed as follows:

$$b^{t+1}(s) = \frac{\sum_{s'} b^t(s') \cdot P(s|s', \vec{a}_i) O(\vec{o}_j|s, \vec{a}_i)}{\sum_{s', s''} b^t(s') \cdot P(s''|s', \vec{a}_i) O(\vec{o}_j|s'', \vec{a}_i)} \quad (1)$$

We denote the updated belief state as $p(*|b^t, \vec{a}_i, \vec{o}_j)$. The denominator corresponds to the probability of observing $\vec{o}_j$ after performing $\vec{a}_i$ from belief b^t. Where we are interested in the expected joint observation, we also use the explicit notation $p(\vec{o}_j|b^t, \vec{a}_i)$. A local policy for agent i is a mapping from local histories to actions. A local policy $u_i^t \in U_i^t$ for an horizon length t can be represented as a tree where each node represents an action, and each edge of the tree an observation. We denote with $a_{u_i^t}$ the local action prescribed by a local policy u_i^t and with $u_i^t(o_i)$ the sub-policy (for horizon length $t-1$) that should be followed after receiving an observation o_i. A joint-policy $\vec{u}^t \in \vec{U}^t$ is defined as consisting of one local policy for each agent. We denote with $\vec{a}_{\vec{u}^t}$ the joint action prescribed by policy $\vec{u}^t$ and with $\vec{u}^t(\vec{o})$ the joint sub-policies that the agents follow after receiving joint observation $\vec{o}$. The value of executing a joint-policy $\vec{u}$, from a state s with t steps to go can be computed recursively as follows:

$$V(\vec{u}^t, s_i) = R(s_i, \vec{a}_{\vec{u}^t}) + \sum_{s_j, \vec{o}} P(s_j|s_i, \vec{a}_{\vec{u}^t}) O(\vec{o}|s_j, \vec{a}_{\vec{u}^t}) V(\vec{u}(\vec{o}), s_j) \quad (2)$$

Given a belief state, solving a Dec-POMDP means finding a joint-policy $\vec{q}$ that maximizes $\sum_s b(s) V(\vec{q}, s)$.

Goldman and Zilberstein [6] demonstrated that even approximately solving a Dec-POMDP is intractable (NEXP-COMPLETE). A great deal of research on offline planning for Dec-POMDPs, therefore, focuses on tractable approximate algorithms that do not provide a guarantee on solution quality. A different body of work has explored heuristic-based online planning for Dec-POMDPs. These algorithms interleave planning and execution, and use heuristics to make decisions on what actions to perform next. The long-term expected value of executing joint-action $\vec{a}_j$ from a belief state b_i is $Q(b_i, \vec{a}_j)$. This Q function can be computed, for example, by performing an l-step lookahead and assuming that the state of the system becomes fully observable after the l-th step.

Given our aim is to propose a model of on-line planning that effectively balances the trade-off between maintaining coordination through communication and exploiting local observations, we choose as our comparators MAOP-COMM [15] and BaGa-S [4].

Wu *et al.* [15] propose the MAOP-COMM algorithm for online planning in Dec-POMDPs with communication. At each step, each agent maintains a pool of possible joint-histories H (one local history per agent), each associated with a probability and a joint belief; that is, the belief that would be obtained by an hypothetical agent that has complete knowledge of all the agents' histories. Given a joint-history $\vec{h}^t$, we denote $b_{\vec{h}^t}$ to be the belief associated with $\vec{h}^t$, and $p(\vec{h})$ to be its probability.

Each agent approximates a one-step lookahead policy $\vec{\pi}$ that maps, for each agent, a local history to an action. We denote π_i to be the local component of $\vec{\pi}$ and $\pi_i(h_i)$ to be the action associated with history h_i. Given a joint-history $\vec{h} = \langle h_0, \ldots, h_n \rangle$ the joint-action executed will be $\vec{\pi}(\vec{h}) = \langle \pi_0(h_0), \ldots, \pi_n(h_n) \rangle$. The objective is to find a policy $\vec{\pi}$ that optimizes the following value function:

$$V(\vec{\pi}) = \sum_{\vec{h} \in H} p(\vec{h}) Q(b_{\vec{h}}, \vec{\pi}(\vec{h})) \quad (3)$$

In order to efficiently find a policy, it is initialized randomly, and then each agent improves its local policy by assuming the policies of other agents are fixed. Improvement terminates when an equilibrium among the local policies is found; *i.e.* when no agent can improve its own policy. After each action, all the histories in the pool are updated by considering the corresponding action and every possible observation, and all the joint beliefs are updated. Coordination is guaranteed because each agent will maintain the same set of possible histories, and they use the same seed for a pseudo-random generator to initialize the policies. In MAOP-COMM, agents decide to communicate if they receive an observation that is inconsistent with their current history pool. Formally, given the current history pool H, a local observation o_i, and a small number ϵ, an agent decides to communicate if and only if:

$$\max_{\vec{h} \in H, \vec{o}_{-i}} \left(\sum_{s' \in S} O(\langle o_i, \vec{o}_{-i} \rangle | s', \vec{a}) \sum_{s \in S} P(s|\vec{a}, s') b_{\vec{h}}(s) \right) < \epsilon \quad (4)$$

The rationale for this is that agents should communicate when they receive an unexpected observation.

While MAOP-COMM guarantees coordination, we argue that this comes at a cost. In order to ensure that agents reach the same equilibrium, each agent i must consider all their own possible previous histories, whereas only one history has been observed. This information, in fact, is not generally available to the other agents. Moreover, by taking into account an observation o_i, an agent is often able to infer additional knowledge about the probability of other agents' histories. Even though the current action of an agent depends on its local history, the fact that MAOP finds equilibria among all possible histories, and uses only information that is available to all agents, it results in policies that cannot take full advantage of local observations.

BaGa-S (Subjective Bayesian Game approximation algorithm) [4] attempts to make use of local observations in a more opportunistic way. In BaGa-S agents consider only histories that are consistent with their local observations. Agent i estimates the best action for the other agents in each history $a^*_{-i}(\vec{h})$ (Equation 5) and then finds its best response a_i^* (Equation 6).

$$a^*_{-i}(\vec{h}) = \arg \max_{a_{-i} \in A_{-i}} \left(\max_{a_i} Q(b_{\vec{h}}, a_i) \right) \quad (5)$$

$$a_i^* = \arg \max_{a_i \in A_i} \sum_{\vec{h}} p(\vec{h}) Q(b_{\vec{h}}, \langle a_i, a^*_{-i}(\vec{h}) \rangle) \quad (6)$$

After executing an action and receiving an observation, each agent updates its belief pool by considering, from each possible joint belief, the estimated action of other agents and all the possible joint-observations that are compatible with its local observation. In order

to limit the exponential growth of the belief history, BaGa-S uses weighted k-means clustering to find, after each update, a fixed number of beliefs that represent the distribution over possible histories. k-means clustering divides joint beliefs into k clusters and maintains only the centre of each cluster as a representation of it. The clusters are found such that the sum of the distance of each belief from the corresponding centre is minimized. The centre of each cluster is a belief where the probability of each state is the weighted average among the probabilities of that state given the beliefs in the cluster, with the weights being the probability of each history. Given two possible beliefs b_1, and b_2, held by an agent i, a distance measure for joint beliefs can be defined as:

$$d(b_1, b_2) = \sqrt{\sum_s (b_1(s) - b_2(s))^2} \cdot p(h_2) \tag{7}$$

Intuitively, this estimates the expected loss of information obtained by merging b_2 with b_1. We multiply the distance only by $p(h_2)$ because this merging represents a loss of information only if the true belief is $b2$.

Note that, each agent takes into account its local observation in updating the belief pool, and so the pools maintained by different agents may diverge at runtime. This, in turn, will lead to different policies being computed for each agent and a further divergence in the belief pools. As pointed out by Wu *et al.* [15], this might lead to arbitrarily bad outcomes. On the other hand, taking into consideration local observations in computing a plan might prove beneficial in scenarios where only loose coordination is necessary.

3 OB-MAP

We now present Observation Based Multi-Agent Planning (OB-MAP), an online planning algorithm that provides a good balance between opportunistic exploitation of information and the maintenance of a certain degree of coordination. We argue that BaGa-S fails to do this because, when an agent i is planning, it doesn't consider the fact that agents other than i also have only a partial view of the environment. In Equation 5, for example, they assume that other agents are able to observe the current joint belief. Moreover, the clustering of histories only takes into account the distance among joint beliefs. It has been demonstrated by Oliehoek *et al.* [12] that in order for the clustering of joint-histories to be lossless, one should merge only histories that are equivalent both in terms of joint beliefs, and in terms of the probability distribution over joint histories held by other agents. Merging only equivalent histories only allows for limited reduction in the size of the joint-histories pool, especially in large scenarios where the number of possible histories grows very quickly with the execution horizon. Our aim is to define a distance metric that approximates the lossless criterion and to use it in standard clustering algorithms to perform a more aggressive clustering while still minimizing the loss of information.

In order to do that, while updating the belief pool we also keep track of the local belief of each agent in each history. Formally we define a belief-node n^k as a tuple:

$$n^k = \langle \vec{h}^k, b_0^k, b_1^k, \ldots, b_n^k, p^k \rangle \tag{8}$$

where $\vec{h}^k$ is a joint-history, b_0^k is the joint belief associated with the joint history, b_i^k with $1 \leq i \leq n$ is the local belief for agent i and p^k is the probability associated with the node. Given a local belief, b_i^t, for agent i at time t, the joint-action $\vec{a}^t$ and the local observation o_i^t, the local belief for i at time $t+1$ can be computed as follows:

$$b_i^{t+1}(s) = p(s|b_i^t, \vec{a}^t, o_i^t) = \frac{\sum_{s', o_{-i}} b^t(s') p(s|s', \vec{a}^t) O(\langle o_i^t, o_{-i} \rangle | s, \vec{a}^t)}{\sum_{s', s'', o_{-i}} b^t(s') p(s''|s', \vec{a}^t) O(\langle o_i^t, o_{-i} \rangle | s'', \vec{a}^t)} \tag{9}$$

where o_{-i} is a tuple consisting of local observations for all agents other than i. The update procedure is similar to the one for a joint belief, but considers all possible joint-observations that have o_i as a local component. We refer to the updated local belief as $p(*|b_i^t, \vec{a}^t, o_i^t)$. Note that, while the update considers only the local component of an observation, the complete joint-action is needed.

3.1 Planning

In common with BaGa-S, when planning, each agent estimates the local action that will be taken by the other agents in each joint history and finds the best response. In doing so, however, it takes into account the fact that if a set of joint histories is associated with the same local history for agent j, the agent will not be able to distinguish among them, and will choose the same action for all of them. A local policy π is defined in the same way as in MAOP-COMM; *i.e.* a mapping from local histories to local actions. Let $h(n^k, i)$ denote the local history for agent i in the node n^k. Given the current set of belief nodes N_i held by an agent i, and a history h_j for agent j different from i, agent i estimates the local action performed by j by finding the joint action that maximizes the following:

$$\pi_j(h_j) = \arg \max_{a_j \in A_j} \left(\max_{a_{-j}} \sum_{\substack{n^k \in N_i \ s.t. \\ h(n^k, j) = h_j}} Q(b_0^k, \langle a_j, \vec{a}_{-j} \rangle) \cdot p^k \right) \tag{10}$$

For each agent, j, we consider together all the nodes that are associated with the same local history for j. These nodes, therefore, represent joint beliefs that are indistinguishable from j's perspective. We assume that j will select, for all these nodes, the action that maximises the expected reward over all the associated joint beliefs.

After an agent has estimated all the actions of other agents, it finds the local action that maximizes its expected reward over all possible nodes. Suppose we take π_{-i} to denote the joint policy that maps each node n^k to a joint action that is left unspecified for agent i and that associates the action $\pi_j(h(n^k, j))$ to each agent j other than i. Now, we can estimate the best action for agent i, thus:

$$a_i^* = \arg \max_{a_i \in A_i} \sum_{n^k \in N_i} q(b_0^k, \langle a_i, \pi_{-i}(n^k) \rangle) \cdot p^k \tag{11}$$

The expected value of local action a_i^* corresponds to $\sum_{n^k \in N_i} q(b_0^k \langle a^*, \pi{-}i(n^k) \rangle) \cdot p^k$. When describing the reasoning of agent i we will use $\pi_i(n^k)$ to denote the policy that assign a_i^* to every node n^k and π to denote the joint policy that assigns to each node n^k the joint action $\langle a_i^*, \pi_{-i}(n^k) \rangle$.

There is an important point of comparison to note here regarding the OB-MAP and BaGa-S models. If there are only two agents, and assuming the same clustering technique is used by them both, the plan computed by our algorithm will be identical to that computed by BaGa-S. Consider, for example, the point of view of agent 1. Since agent 1 only maintains beliefs that are compatible with its local history, each belief node in the pool must be associated with

Algorithm 1 Belief propagation

Input: N_i^t, π_{-i}, o_i
Output: N_i^{t+1}
1: $N_i^{t+1} = \emptyset$
2: **for all** $n^k \in N_i^t$ **do**
3: $\quad \vec{a} = \pi(n^k)$
4: $\quad$ **for all** $o_{-i} \in \Omega_{-i}$ **do**
5: $\quad\quad \vec{o} = \langle o_i, o_{-i} \rangle$
6: $\quad\quad b_0(*) = p(*|b_0^k, \vec{a}, \vec{o})$
7: $\quad\quad p = p(\vec{o}|b_0^k, \vec{a})$
8: $\quad\quad$ **for all** $0 \leq j \leq n$ **do**
9: $\quad\quad\quad b_j(*) = p(*|b_j^k, \vec{a}, \vec{o}[j])$
10: $\quad\quad\quad h_j = (h_j^k, \pi_j(n^k), \vec{o}[j])$
11: $\quad\quad$ **end for**
12: $\quad\quad \vec{h} = \langle h_1, \ldots, h_n \rangle$
13: $\quad\quad N_i^{t+1} = N_i^{t+1} \cup \langle \vec{h}, b_0, b_1, \ldots, b_n, p \rangle$
14: $\quad$ **end for**
15: **end for**

a different history for agent 2, otherwise the two nodes would be equivalent. When estimating the action of agent 2, each belief-node will be considered separately and Equations 10 and 11 (OB-MAP) are equivalent to Equations 5 and 6 (BaGa-S). When there are more than 2 agents, two beliefs in the pool might have the same history for agent 2, but a different one for, for example, agent 3.

After an action is taken and an observation received, the nodes in the belief pool need to be propagated. We consider each node in the pool, with the estimated joint action and, for each joint-observation compatible with the local observation received, we update the joint-history and all the joint and local beliefs. We refer to the set of all possible joint observations as Ω_{-i}, with the i-th component left unspecified, and we refer to the local component of a joint observation $\vec{o}$ associated with agent j as $\vec{o}[j]$. Algorithm 1 specifies how this pool of belief nodes is updated.

3.2 Clustering

After propagating the beliefs, we perform clustering in order to maintain a bounded number of beliefs. The distance metric for belief nodes is defined as:

$$d(n^k, n^l) = \sqrt{\sum_s \left(\max_{0 \leq i \leq n} (b_i^k(s) - b_i^l(s))^2 \right)} \cdot p^l \qquad (12)$$

This captures the idea that, for each state, we take the maximum distance among all pairs of corresponding (joint or local) beliefs. Moreover, instead of using weighted k-means clustering, we use a modified k-medoid clustering. This algorithm partitions N_i into k clusters and finds, for each cluster $C_j = \{n^k, \ldots\}$, the node $\bar{n}^{C_j}$ that minimizes the sum of the distances of each other element of the cluster from $\bar{n}^{C_j}$. Formally:

$$\bar{n}^{C_j} = \arg \min_{n^k \in C_j} \sum_{n^l \in N^k} d(n^l, n^k) \qquad (13)$$

We refer to node $\bar{n}^{C_j}$ as the medoid of the cluster. Since the medoid is an actual data-point, k-medoids is more robust to outliers and noise, which are important to consider in planning problems. For each cluster, we retain the one node at the medoid of the cluster, defined as follows:

$$n_*^{C_j} = \left\langle \vec{h}^{C_j}, \bar{b}_0, \ldots, \bar{b}_n, p_*^{C_j} \right\rangle \qquad (14)$$

where:

- $\bar{b}_0, \ldots, \bar{b}_n$ are the joint and local beliefs of the medoid node $\bar{n}^{C_j}$.
- $p_*^{C_j} = \sum_{n^k \in C_j} p^k$ is the sum of all the probabilities of the nodes in the cluster.
- $\vec{h}^{C_j} = \langle h_1^{C_j}, \ldots, h_n^{C_j} \rangle$ consists of, for each agent, the local history that appears with highest probability in the cluster. Formally, if H_i denotes the set of possible local histories for agent i:

$$h_i^{C_j} = \arg \max_{h_i \in H_i} \sum_{\substack{n^k \in C_j \ s.t. \\ h(n^k, i) = h_i}} p^k \qquad (15)$$

3.3 Communication Heuristics

We consider agents that can communicate in order to share their local histories and synchronize on a common joint belief. This enables agents to obtain more precise information about the current state of the environment and to restore coordination when this is lost due to misaligned belief pools. We assume that communication comes at a cost (a negative reward), and we propose a heuristic technique to adaptively make decisions on whether or not to communicate, based on the current level of uncertainty and the expected value obtained from communication. Given a communication cost R_c, and the current belief node pool N_i, agent i can estimate the expected value obtained by communicating as follows:

$$V_c = -R_c + \sum_{n^k \in N_i} p^k \cdot \max_{\vec{a} \in \vec{A}} Q(b_0^k, \vec{a}) \qquad (16)$$

Informally, since after communicating each agent will have full knowledge about the current joint belief, we assume that the agents can choose a different joint-action for each joint belief. Each agent finds the best action for each belief-node and averages over their expected values. The agent then subtracts the cost of communication and, if the resulting V_c is greater than the estimated value without communication, it chooses to communicate. Note that, since the action chosen for other agents when updating the belief pool are only an estimate, and because of the clustering, the actual joint belief might not be present in the belief-node pool. Moreover, from the point of view of agent i, the heuristic does not take into account the value obtained by other agents when they receive i's local history, but only the value obtained by i when it receives other agent's observations. We will demonstrate experimentally that this heuristic works well in practice and provides an efficient way to estimate the added value of communication.

Algorithm 2 specifies the main function of the OB-MAP planner. The belief-node pool is initialized with a single node corresponding to the empty history and that is assigned the initial beliefs b^0 of all the agents (Line 1). The **computePolicy** function (Line 3) estimates the actions of other agents and finds the best response according to Equations 10 and 11. **ComputeCommValue** finds the expected value after communication by applying Equation 16. If this value is higher than the estimated value without communication the agent will communicate with its team-mates in order to **sync** their true histories and find a common joint belief. If communication occurs, the agents must recompute their policies taking into account the true joint belief (Lines 4-9). After executing their part of the policy (the local action found as best response) and receiving an observation, the agents will propagate the current belief-nodes and use k-medoid clustering to find k belief nodes that best represent all the possible histories.

Algorithm 2 Main OB-MAP execution function

Input: b^0, R_c, k
1: $N = \{\langle (), b^0, b^0, \ldots, b^0, 1\rangle\}$
2: **for** t = 0 to T **do**
3: $\pi =$ **computePolicy**(N)
4: $V_C =$ **computeCommValue**(N, R_c)
5: **if** $V_C \geq V(\pi)$ **then**
6: $\langle b^t, h^t\rangle =$ **sync**()
7: $N = \{\langle h^t, b^t, b^t, \ldots, b^t, 1\rangle\}$
8: $\pi =$ **computePolicy**(N)
9: **end if**
10: execute best local response.
11: o = received observation
12: $N =$ **propagateBeliefs**(N, π, o)
13: $N =$ **cluster**(N, k)
14: **end for**

3.4 OB-MAP Complexity

Before presenting an empirical analysis of our model, we analyse its runtime complexity for both belief propagation and belief node clustering. At each step, the size of the set of beliefs N is bounded by k, the number of clusters specified for the k-medoids algorithm. When expanding the beliefs we take each of these k beliefs, consider the action given by the policy $\pi(n^k)$ and, for each observation that is compatible with the observed o_i, we update the joint belief and all the local beliefs. This gives a time complexity of

$$O(|\Omega_{-i}| \cdot k \cdot |I| \cdot |S|^2)$$

where S^2 is the belief update, and $|\Omega_i| = \prod_{j \in I \setminus \{i\}} |\Omega_i|$.

For k-medoids, we use the Partitioning Around Medoid (PAM) [8] algorithm, which performs $O(I \cdot (k \cdot (N-k)^2))$ comparisons, where I is the number of iterations performed by the algorithm, N is the initial number of beliefs, and k the number of clusters. Since, as discussed above, the number of data points is $|\Omega_{-i}| \cdot k$, and each comparison (Equation 12) takes $|I| \cdot |S|$, the complexity of the clustering phase is:

$$O(I \cdot |\Omega_{-i}|^2 \cdot k^3 \cdot |S| \cdot |I|)$$

The complexity of the process of computing a policy will depend on the heuristics employed, but, in general, we evaluate $|\vec{A}| \cdot k$ actions at each step. For the large majority of problems, runtime is dominated by the clustering phase (which we confirmed experimentally), and PAM is not the most efficient algorithm that could be employed. CLARANS, for example, is an efficient clustering model designed for mining large data sets. Although our problem is to solve a large number of small clustering problems, it was analysed to be "a few times faster than PAM" [11] for small data sets, and hence could be employed to further optimise OB-MAP.

4 Evaluation

In order to effectively evaluate the OB-MAP model, we consider a number of widely-employed benchmark problems. We pitch OB-MAP against two state-of-the-art algorithms: one that uses communication to synchronise agents' beliefs to maintain coordination (MAOP-COMM), and one that exploits only local observations (BaGa-S). Furthermore, in order to provide a fair comparison, in each benchmark we assess relative performance where communication is cost-free (OB-MAP versus MAOP-COMM under the same assumptions) and where no communication is permitted (OB-MAP versus BaGa-S under the same assumptions, and also MAOP-COMM with no communication).

In all the benchmark problems, we used a multi-step lookahead MDP-based Q heuristic, Q_{MDP}. The heuristic is defined as follows, where $V_{MDP}(s)$ is the expected value of the optimal policy for the underlying MDP, starting from state s:

$$Q_{MDP}(b, \vec{a}_j, l) = \begin{cases} \sum_{s,s'} b(s)(R(s,\vec{a}) + P(s'|s,\vec{a}_j)V_{MDP}(s')) & \text{if } l \leq 1 \\ \sum_{s'}[b(s')R(s',\vec{a})] + \sum_{\vec{o}_k}[p(\vec{o}_k|b_i^t,\vec{a}_j) & \\ \qquad \max_{\vec{a}_l}\{Q(b_{\vec{a}_i,\vec{o}_k},\vec{a}_l,l-1)\}] & \text{if } l > 1 \end{cases} \quad (17)$$

Essentially the heuristic assumes full communication for l steps and full observability after that. A better approximation could be obtained by using a POMDP policy, which assumes full communication, but partial observability, over the whole problem horizon.

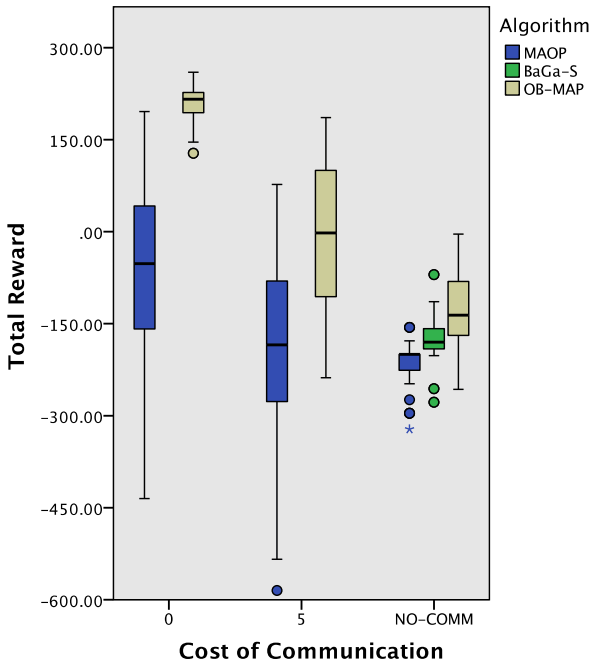
The benchmark problems that we used are:

- The decentralized tiger problem [10]. In this scenario, two agents are in a corridor facing two doors: "left" and "right". Behind one door lies a hungry tiger and behind the other lies a treasure. Each agent can either open one of the doors or listen for the presence of the tiger. After each step, each agent receives a noisy observation about the position of the tiger. By listening, agents increase their probability of receiving the correct observation. The agents minimize their loss if they jointly open the door with the tiger and maximize their reward if they jointly open the treasure door. After either door is opened the problem is reset.
- The variant of 3x3 grid problem presented by Amato *et al.* [2]. In this problem two agents can move along 4 directions in a 3x3 grid world. Each agent only receives noisy observations about neighbouring walls. The objective of the agents is to meet in either the top-left or bottom-right cell.
- The stochastic Mars rover problem [3]. In this scenario, two agents must perform different experiments at certain research sites. Some of these sites may require multiple agents performing an experiment together in order to get the most scientific value, while other sites may require a specific tool to be used by a single agent. Positive rewards are given for successfully performing experiments at each site and the task is completed when all experiments have been conducted.

In order to verify our claim that the approximate planning model described by Equations 10 and 11 improves upon BaGa-S for scenarios with more than 2 agents, we compared our approach against standard BaGa-S and BaGa-S with our medoids-based clustering approach in the following scenarios:

- A 3 agent version of the decentralized tiger problem.
- A 3 agent version of the broadcast channel problem [7]. In this problem, 3 agents attempt to send messages over a shared channel. Each agent has a buffer of at most one message. The channel can deliver only one message at a time. Moreover the channel randomly switches between a functional and a non-functional state. If only one agent attempts to send a message, the channel is functioning, and the buffer of that agent has one message, the agents receive a reward of $+1$. If no message is delivered because the channel is not functioning, because of a collision, or because the agent trying to send a message has an empty buffer, the agents obtain a reward of $-s$, where s is the number of agents that attempted to send a message. At each step each agent receiving a noisy observation signalling whether there has been a collision, a successfully delivered message, or the state of the channel.

Figure 1. The 2-Agent Decentralized Tiger Scenario



For all the scenarios we used an horizon of 100 steps.

Table 1. The 2-Agent Decentralized Tiger Scenario

	MAOP-COMM			OB-MAP			BaGa-S
$\mathbf{R_C}$	0	5	NO	0	5	NO	NO
R	−62.9 ±138.2	−185.3 ±136.6	−207.0 ±30.7	207.3 ±30.8	−6.7 ±117.6	−128.8 ±54.9	−174.1 ±28.7
C	24.5 ±4.1	24.5 ±4.1	0 ±0	36.3 ±2.4	7.6 ±3.3	0 ±0	0 ±0
T[ms]	0.65	0.65	0.80	0.53	0.57	0.61	1.07

Table 1 presents results for the 2-agent decentralized tiger scenario. The table summarises the average and standard deviation over 100 trials of the reward (**R**), the number of communication steps (**C**) and the average execution time per agent per step (**T[ms]**) for the three algorithms with varying cost of communication ($\mathbf{R_c}$). For both BaGa-S and OB-MAP we maintained 20 clusters. Since BaGa-S does not consider communication, we report only the results in the absence of communication (NO). In Figure 1 we present the distribution of reward for each algorithm and for varying cost of communication. Notice that the frequency of communication does not change with the communication cost in MAOP-COMM. This is due to the fact that MAOP-COMM only allows us to specify whether agents can or cannot communicate, and communication cost is not taken into account. In order to simulate different communication costs we simply subtracted from the obtained reward the total cost of communication. Comparing the results of BaGa-S and MAOP (*i.e.* MAOP-COMM with no communication) we can see that this scenario favours a more opportunistic approach over guaranteed coordination. The OB-MAP planner has a better average than either of the other algorithms, and it makes better use of communication when available in comparison with MAOP-COMM. Since the results are not normally distributed, we tested them for significance using the Kruskall Wallis test with post-hoc analysis consisting of Bonferroni corrected Mann-Whitney tests. We obtained an asymptotic p-value of 0.000 both for the comparison with MAOP-COMM and with BaGa-S[2]. The execution times for OB-MAP are comparable, but slightly lower than those for MAOP-COMM and almost half of those for BaGa-S.

Table 2. The 3x3 Grid Scenario

	MAOP-COMM			OB-MAP			BaGa-S
$\mathbf{R_C}$	0	0.1	NO	0	0.1	NO	NO
R	25.4 ±1.4	22.2 ±1.8	15.6 ±3.2	25.3 ±1.2	21.7 ±2.7	21.5 ±3.9	6.88 ±5.7
C	32.2 ±4.6	32.2 ±4.6	0 ±0	68.3 ±3.0	0.6 ±0.7	0 ±0	0 ±0
T[ms]	0.77	0.77	72	7.09	90.0	71.19	86.45

Table 2 and Figure 2 summarize the results for the 3x3 Grid scenario. For this scenario we used the Q_{MDP} heuristic with lookahead equal to 1 and we set the number of clusters to 20 for both BaGa-S and OB-MAP. In this scenario MAOP-COMM performs better than BaGa-S. We believe that this is because this scenario requires tighter coordination among team-mates; see discussion of the strict coordination argument above. Our approach performs significantly better (asymptotic p-value of 0.000) than either MAOP-COMM or BaGa-S in the absence of communication. When communication is available, however, the results for OB-MAP are not significantly different from MAOP-COMM (asymptotic p-value of 0.508).

[2] The p-value of 0.000 denotes 0 to the precision available from the statistical analysis tool used.

Figure 2. The 3x3 Grid Scenario

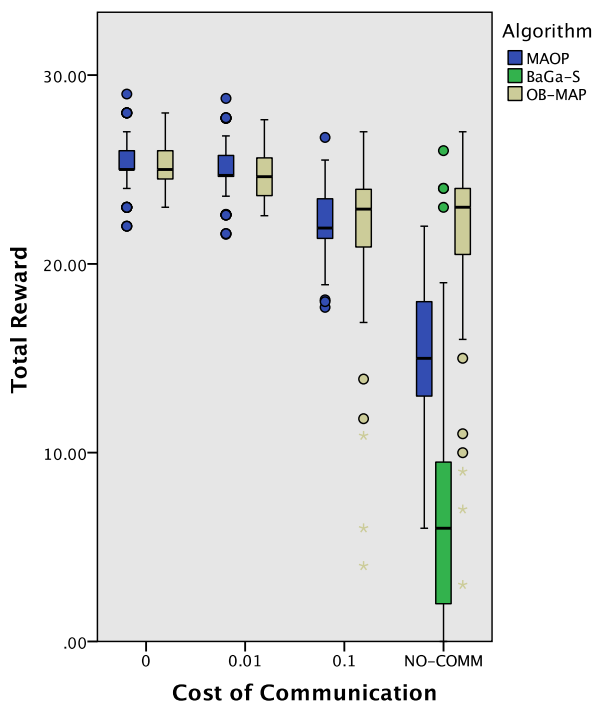


Table 3. The Stochastic Mars Rover Scenario

	MAOP-COMM			OB-MAP			BaGa-S
R_C	0	2	NO	0	2	NO	NO
R	150.9 ±8.8	119.7 ±10.2	43.5 ±15.8	284.6 ±12.01	222.6 ±29.8	141.9 ±61.9	110.0 62.1
C	15.5 ±3.3	15.5 ±3.3	0 ±0	30.9 ±3.2	15.9 ±1.7	0 ±0	0 ±0
T[ms]	0.7	0.7	434.3	3.5	12.8	31.5	27.9

Table 3 and Figure 3 report results for the Stochastic Mars Rover Scenario. This represents another situation in which BaGa-S significantly outperforms MAOP-COMM in the absence of communication. OB-MAP, however, performs better than either MAOP (asymptotic p-value of 0.000) or BaGa-S (asymptotic p-value of 0.005) in the absence of communication. When communication is available, OB-MAP significantly out-performs MAOP-COMM (asymptotic p-value of 0.000) and adapts well to more costly communication. Note that, since all the scenarios discussed so far include only two agents, the differences between OB-MAP and BaGa-S are only due to the different clustering methods used.

In Table 4 we present the results for the comparison between BaGa-S and OB-MAP algorithms on the 3-agent version of the tiger scenario. We could not test MAOP-COMM in this scenario because the implementation of MAOP-COMM used (kindly provided by the authors) does not support scenarios with more than two agents. In order to separate the effects of the clustering algorithm used, and the different action selection mechanisms, we also analysed the behaviour of a version of BaGa-S that uses medoid clustering (**BaGa-S-medoids** in Table 5). In our experiments, BaGa-S with k-medoids agents always listened, without ever attempting to open any door. This is the reason why their reward is always -303 (where -1 is the cost of one agent listening). BaGa-S agents deviate only slightly from this behaviour. OB-MAP behaves significantly better (asymptotic p-value of 0.000) than either BaGa-S or BaGa-S-medoids, while the results of BaGa-S with medoids are not different from those of BaGa-S (p-value of 1.000). We argue that this difference is due to the fact that agents using the BaGa-S algorithm fail to correctly estimate the other agents' actions, because they assume that the true joint belief is known to them.

Figure 3. The Stochastic Mars Rover Scenario

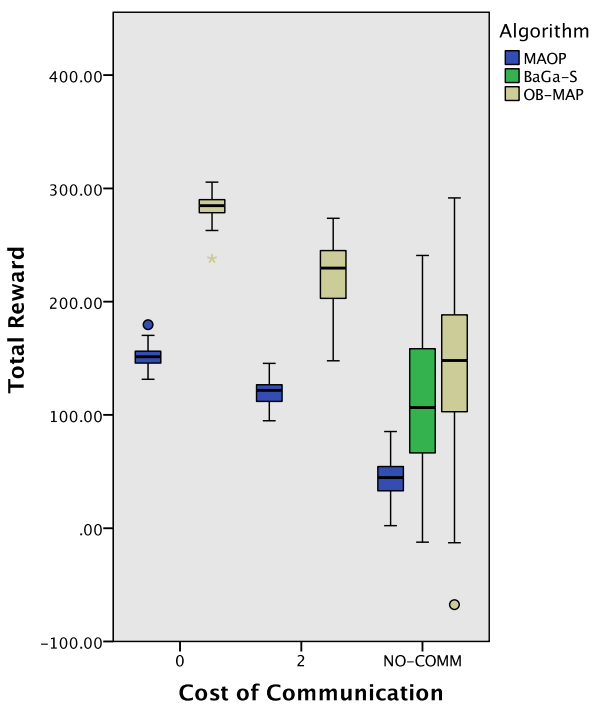


Table 4. The 3-Agent Decentralized Tiger Scenario

	OB-MAP			BaGa-S	BaGa-S-medoids
R_c	0	5	NO	NO	NO
R	577.6 ±58.0	200.3 ±39.9	-267.5 ±133.9	−299.6 ±11.1	−303.0 ±0
C	61.0 ±2.5	34.0 ±1.1	0 ±0	0 ±0	0 ±0
T[ms]	7.16	10.29	62.3	157.6	97.5

In Table 5 and Figure 5 we report results for the BaGa-S and OB-MAP algorithms on the 3-agent Broadcast Scenario. In this scenario BaGa-S-medoids performs considerably better than standard BaGa-S. OB-MAP obtains better rewards than either standard BaGa-S (p value 0.000) or BaGa-S-medoids (p value 0.040). The significance of this difference between OB-MAP and BaGa-S-medoids confirms that our planning algorithm better captures scenarios where there are more than two agents.

Figure 4. The 3-Agent Decentralized Tiger Scenario

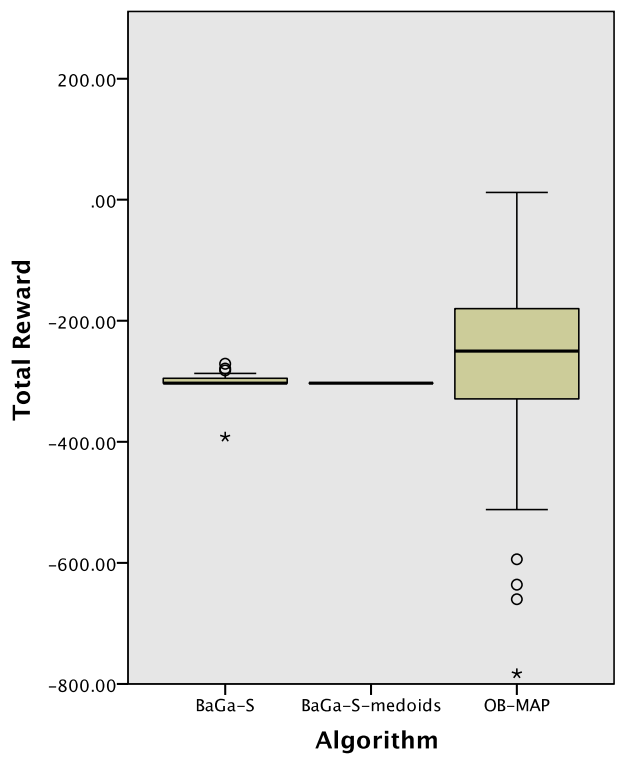
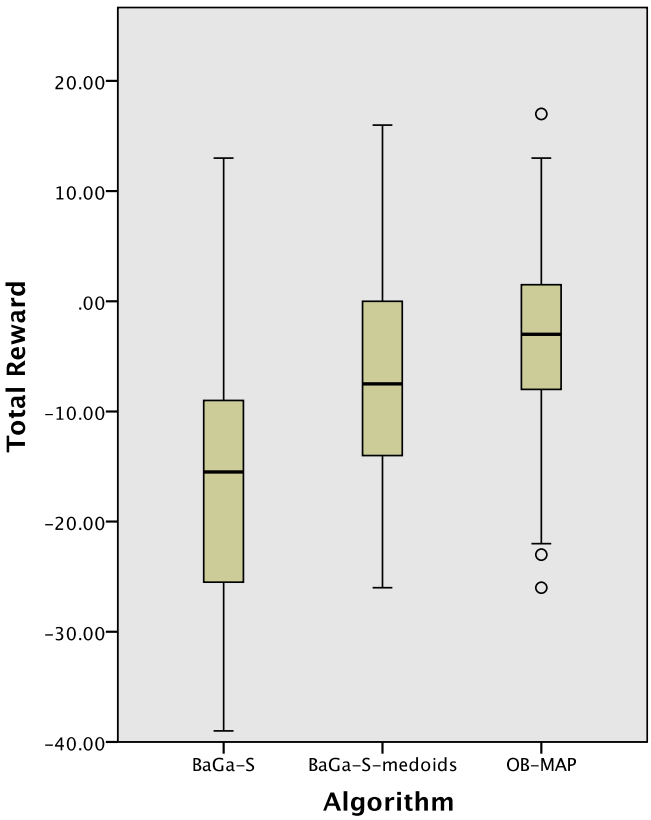


Table 5. The 3-Agent Broadcast Scenario

	OB-MAP	BaGa-S	BaGa-S-medoids
$\mathbf{R_c}$	NO	NO	NO
R	-3.72 ± 7.88	-16.55 ± 10.74	-6.75 ± 9.12
C	0 ± 0	0 ± 0	0 ± 0
T[ms]	520.7	300.0	514.0

Figure 5. The 3-Agent Broadcast Scenario



5 Discussion

Online decentralized planning under uncertainty has been the subject of a number of prior studies. Roth *et al.* [14], for example, proposed Dec-COMM, an approach to online planning that provides guaranteed coordination at the cost of completely ignoring observations unless these have been shared with all the team-mates. In common with OB-MAP, agents keep track of the possible joint beliefs of the team and use a Q_{MDP} (or Q_{POMDP}) heuristic to find an action that maximizes the expected reward over all possible beliefs. In order to guarantee that all agents will agree on the same action, they do not take into account their local observation when propagating the possible beliefs. Agents decide to communicate their observations when including these would result in choosing a different action.

BaGa-S builds upon BaGa [5]. In the BaGa algorithm, each agent keeps track of all possible types (local histories) of all agents and finds a policy mapping, for each agent type to actions. A policy represents an equilibrium; *i.e.* a situation where no agent can improve the expected value of the policy by unilaterally changing their local policy. Montemerlo proposed a number of heuristics for deciding whether or not to communicate, some of which take into account the cost of communication. They consider a model of communication where agents can decide to broadcast their own history to other agents, without requiring other agents to also synchronize their history. In order to estimate the value of communication, each agent prunes all the possible histories that are incompatible with its local histories and find a policy for that belief pool.

In MAOP-COMM [15], agents perform one-step planning in a decentralized manner and estimate the long-term value from a belief using a Q_{MDP} heuristic. In order to find a locally optimal equilibrium among agents' policies, they use alternative maximization, where each agent iteratively improves its policies assuming other agents' policies are fixed. Agents communicate when they receive a local observation that is inconsistent with their beliefs. While these algorithms represent improvements in the way observations are taken into account, the fact that the policy must represent an equilibrium between all histories, even those that are not compatible with an agent's local history, results in policies that are sometimes too conservative. Moreover, by taking into account only joint histories that are compatible with an agent's observations, we can decrease the number of candidate histories and obtain better clusters.

As mentioned by Wu *et al.* [15], BaGa is not able to deal with scenarios as large as those used to evaluate our approach. This is due to the fact that the clustering technique used in BaGa does not ensure that only a bounded number of beliefs is retained at each step. Moreover, the results reported by Wu *et al.* [15] shows that MAOP-COMM out-performs Dec-COMM in most situations. For these reasons we used MAOP-COMM to compare OB-MAP against; it represents the best available state-of-the-art algorithm.

Kaufman and Roberts [9] analyse the trade-off between using local information and guaranteed coordination in multi-agent planning. Their analysis, however, considered limited scenarios where the transition probabilities are uniform and therefore the value of observations is limited.

6 Conclusion

In this paper we have proposed OB-MAP, a novel algorithm for online planning in Dec-POMDPs. The algorithm is inspired by BaGa-S in terms of the use of local observations, but also enables value-aware communication between agents to maintain coordination in domains in which local observations are insufficient. This provides a balance between approaches that guarantee strict coordination but fail to take into account local information during planning, and approaches that use local information but fail to maintain acceptable levels of coordination in many scenarios. We propose a heuristic to decide when, given the cost of communication, agents should communicate in order to synchronize their histories and agree on a common joint belief. We evaluated our approach on a number of benchmark scenarios and we have shown that it performs significantly better than two algorithms that best represent the state-of-the-art.

Acknowledgments

This research has been sponsored by SELEX ES. We thank Feng Wu for providing the source code of the MAOP-COMM planner.

REFERENCES

[1] C. Amato, 'Cooperative decision making.', in *Decision Making Under Uncertainty: Theory and Application*, ed., Mykel J. Kochenderfer, chapter 7, MIT Press, (2014).

[2] C. Amato, J. S. Dibangoye, and S. Zilberstein, 'Incremental policy generation for finite-horizon Dec-POMDPs', in *Proceedings of the 19th International Conference on Automated Planning and Scheduling*, pp. 2–9, (2009).

[3] C. Amato and S. Zilberstein, 'Achieving goals in decentralized POMDPs', in *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pp. 593–600, (2009).

[4] A. Chechetka and K. Sycara, 'Subjective approximate solutions for decentralized POMDPs', in *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, p. 224, (2007).

[5] R. Emery-Montemerlo, *Game-Theoretic Control for Robot Teams*, Ph.D. dissertation, Rutgers, The State University of New Jersey, 2005.

[6] C. V. Goldman and S. Zilberstein, 'Decentralized control of cooperative systems: Categorization and complexity analysis', *Journal of Artificial Intelligence Research*, **22**, 143–174, (2004).

[7] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, 'Dynamic programming for partially observable stochastic games', in *Proceedings of the 19th National Conference on Artificial Intelligence*, pp. 709–715, (2004).

[8] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: An introduction to cluster analysis*, volume 344, John Wiley & Sons, 2009.

[9] M. Kaufman and S. Roberts, 'Coordination vs. information in multi-agent decision processes', in *Proceedings of the 5th Workshop on Multi-agent Sequential Decision Making in Uncertain Domains*, pp. 1–6, (2010).

[10] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella, 'Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings', in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pp. 705–711, (2003).

[11] R. T. Ng and J. Han, 'CLARANS: A method for clustering objects for spatial data mining', *IEEE Transactions on Knowledge and Data Engineering*, **14**(5), 1003–1016, (2002).

[12] F. A. Oliehoek, S. Whiteson, and M. T. J. Spaan, 'Lossless clustering of histories in decentralized POMDPs', in *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, pp. 577–584, (2009).

[13] M. Roth, R. Simmons, and M. Veloso, 'Reasoning about joint beliefs for execution-time communication decisions', in *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 786–793, (2005).

[14] M. Roth, R. Simmons, and M. Veloso, 'What to communicate? execution-time decision in multi-agent POMDPs', in *Distributed Autonomous Robotic Systems 7*, 177–186, Springer, (2006).

[15] F. Wu, S. Zilberstein, and X. Chen, 'Online planning for multi-agent systems with bounded communication', *Artificial Intelligence*, **175**(2), 487–511, (2011).