Teamwork and Coordination Under Model Uncertainty in DEC-POMDPs

Jun-young Kwak, Rong Yang, Zhengyu Yin, Matthew E. Taylor, Milind Tambe
{junyounk, yangrong, zhengyuy, taylorm, tambe}@usc.edu
University of Southern California
Los Angeles, California 90089

Abstract

Distributed Partially Observable Markov Decision Processes (DEC-POMDPs) are a popular planning framework for multiagent teamwork to compute (near-)optimal plans. However, these methods assume a complete and correct world model, which is often violated in real-world domains. We provide a new algorithm for DEC-POMDPs that is more robust to model uncertainty, with a focus on domains with sparse agent interactions. Our STC algorithm relies on the following key ideas: (1) reduce planning-time computation by shifting some of the burden to execution-time reasoning, (2) exploit sparse interactions between agents, and (3) maintain an approximate model of agents' beliefs. We empirically show that STC is often substantially faster to existing DEC-POMDP methods without sacrificing reward performance.

Introduction

Distributed partially observable Markov decision processes (DEC-POMDPs) have gained considerable popularity for multiagent planning problems. Central to this popularity is the ability of DEC-POMDPs to quantitatively express observational and action uncertainty (Pynadath and Tambe 2002; Goldman and Zilberstein 2003; Seuken and Zilberstein 2007; Spaan, Oliehoek, and Vlassis 2008). DEC-POMDP expressivity and optimality guarantees come at the cost of doubly-exponential (NEXP-complete) policy generation complexity (Bernstein et al. 2002). Additionally, DEC-POMDP planners assume completely accurate models: it may contain uncertainties in action and observation probabilities, but there can be no uncertainty about these uncertainties! Thus, agents can execute pre-made plans without any additional execution-time reasoning.

However, this DEC-POMDP framework conflicts with the older belief-desires-intentions (BDI) models of teamwork (Levesque, Cohen, and Nunes 1990; Grosz and Kraus 1996; Tambe 1997) which were unable to perform such reasoning over uncertainty. BDI research suggests that domain models are often inaccurate for real-world problems and that relying on planners to provide perfectly-coordinated plans is unrealistic. Instead, BDI research on teamwork has focused on robustness via execution-time reasoning about coordination (Kaminka and Frenkel 2007; Tambe 1997). Thus, planners create high-level "team-oriented plans" that ignore coordination; execution-time reasoning fills in required coordination in the abstract plans as required.

Our work focuses on the challenges of multiagent teamwork under model uncertainty. In many domains, we may only have an approximate model of agent observation, or transition functions, which we show causes problems for existing DEC-POMDP planners. We do not downplay the significant progress in DEC-POMDPs, but we do question the wisdom of paying a high computational cost for a promised high-quality DEC-POMDP policy — which may not be realized in practice. Instead, inspired by BDI research, this work proposes using approximate "team-oriented policies" that *ignore* the details of coordination and leave such computations to execution-time reasoning. There has been some past work (Xuan and Lesser 2002; Roth, Simmons, and Veloso 2005; Wu, Zilberstein, and Chen 2009) exploring such execution-time communication reasoning in DEC-POMDPs, but they do not leverage existing BDI research, nor do they account for model errors.

This paper introduces the SELECTIVE-TEAM-COMM (STC) algorithm, which has four key properties. First, STC policy-generation assumes zero-cost communication for the given DEC-POMDP problem, resulting in the DEC-POMDP problem being reduced to a single-agent POMDP problem — an exponential improvement in the worst-case computational complexity (Pynadath and Tambe 2002). Since communication has inherent cost, agents individually reason about communication at execution time and selectively communicate to maximize the expected reward. Second, BDI-inspired "Trigger Points" heuristically choose critical points of communication reasoning rather than reasoning at every time step, saving significant execution-time computation. Third, particularly in domains with interaction-sparseness, agents often benefit from utilizing some form of pre-planning for individual actions (that do not involve inter-agent interaction). This contrasts with previous work in execution reasoning for DEC-POMDPs where every step is planned online — regardless of whether it involves interaction. Fourth, STC agents may maintain an approximate model of joint beliefs, and they do not rely on these beliefs for precise online planning. This not only provides significant speed-ups, but it further improves robustness to model uncertainty without sacrificing the team's reward.

Background and Related Work

A DEC-POMDP, is described by a tuple $\langle I, S, \{A_i\}, \{\Omega_i\}, T, R, O, \mathbf{b}^0 \rangle$, where

- $I = \{1, ..., n\}$ is a finite set of agents.
- $S = \{s_1, ..., s_k\}$ is a finite set of joint states.
- A_i is the finite set of actions of agent i. $A = \prod_{i \in I} A_i$ is the set of joint actions, where $\mathbf{a} = \langle a_1, ..., a_n \rangle$ denotes a particular joint action (one individual action per agent).
- Ω_i is the finite set of observations of agent i. $\Omega = \prod_{i \in I} \Omega_i$ is the set of joint observations, where $\mathbf{o} = \langle o_1, ..., o_n \rangle$ denotes a joint observation.
- $T : S \times A \times S \mapsto \mathbb{R}$ is the transition probability function, where $T(s'|s, \mathbf{a})$ denotes the transition probability from s to s' if joint action $\mathbf{a}$ is executed.
- $O : S \times A \times \Omega \mapsto \mathbb{R}$ is observation probability function, where $O(\mathbf{o}|s', \mathbf{a})$ denotes the probability of receiving the joint observation $\mathbf{o}$ if the end state is s' after $\mathbf{a}$ is taken.
- $R : S \times A \times S \mapsto \mathbb{R}$ is the reward function, where $R(s, \mathbf{a}, s')$ denotes the reward agents get by taking $\mathbf{a}$ from s and reaching s'.
- $\mathbf{b}^0$ is the initial joint belief state.

We denote the joint observation history at time step t with $\mathbf{h}^t = \{\mathbf{o}^1, \mathbf{o}^2, \ldots, \mathbf{o}^t\}$ and the set of $\mathbf{h}^t$ with H^t. $H = \bigcup_t H^t$ is the set of all possible joint observation histories at all time steps. A joint policy $\pi : H \mapsto A$ is a mapping from joint observation history to joint action. Let $h_i^t = \{o_i^1, o_i^2, \ldots, o_i^t\}$ be the individual observation history of agent i at time step t. The set of all possible h_i^t is denoted by H_i^t. Let $H_i = \bigcup_t H_i^t$ be the set of all possible individual observation histories at all time steps for agent i. The individual policy for agent i, $\pi_i : H_i \mapsto A_i$, is a mapping from agent i's individual observation history to its individual action. We use h_{-i}^t to denote an observation history of all agents except i and H_{-i}^t to denote the set of all possible h_{-i}^t. π_{-i} represents the policy for all agents except i.

We represent uncertainty in our model with a Dirichlet distribution similar to previous work on single-agent POMDP learning (Jaulmes, Pineau, and Precup 2007), which is not applicable to multiagent domains since they did not account for additional coordination among agents while handling model errors (i.e., only focused on a single-agent problem). A separate Dirichlet distribution for the observation and transition function is used for each joint state, action, and observation. An L-dimensional Dirichlet distribution is a multinomial distribution parameterized by positive hyper-parameters $\alpha = \langle \alpha_1, \ldots, \alpha_L \rangle$ that represents the degree of model uncertainty. The probability density function is given by

$$f(x_1, ..., x_L; \alpha) = \frac{\prod_{i=1}^{L} x_i^{\alpha_i - 1}}{B(\alpha)}, B(\alpha) = \frac{\prod_{i=1}^{L} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{L} \alpha_i)},$$

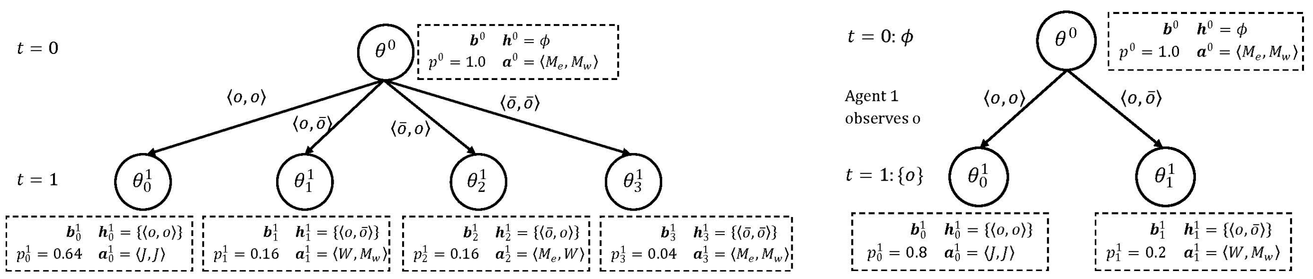
and $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$ is the standard gamma function. The maximum likelihood can be computed: $x_i^* = \frac{\alpha_i}{\sum_{j=1}^{L} \alpha_j}$, for $i = 1, ..., L$. Let $\mathbf{T}_{s,\mathbf{a}}$ be the vector of transition probabilities from s to other states when $\mathbf{a}$ is taken and $\mathbf{O}_{s',\mathbf{a}}$ be the vector of observation probabilities when $\mathbf{a}$ is taken and s' is reached. Then $\mathbf{T}_{s,\mathbf{a}} \sim Dir(\alpha)$ and $\mathbf{O}_{s',\mathbf{a}} \sim Dir(\alpha')$, where α and α' are two different hyper-parameters.

Existing DEC-POMDP Solution Methods

We are interested in teamwork when agents are allowed to communicate. Such communication reasoning may be handled at plan-time or at execution-time. COM-MTDP (Pynadath and Tambe 2002) and DEC-POMDP-COM (Goldman and Zilberstein 2003) provide theoretical frameworks to generate an optimal policy for communication at plan-time. However, these models assume correct domain models. Furthermore, given general communication costs, the policy generation problem remains NEXP-complete. Although execution-time approaches have also assumed model correctness, they assume at least some degree of communication while planning to reduce its complexity and, therefore, have been shown to be more scalable. We now discuss this literature on execution-time reasoning approaches, and specifically algorithms are most relevant to our problem.

The ACE-PJB-COMM (Roth, Simmons, and Veloso 2005) algorithm assumes full/free communication when planning and collapses the multi-agent problem to a single agent POMDP. At execution time, if an agent detects an expected utility improvement from communication that is greater than the communication cost, it will share its history with the rest of the team. Similarly, MAOP-COMM (Wu, Zilberstein, and Chen 2009) communicates whenever it detects a "history inconsistency" that might cause miscoordination. STC, instead, employs a fundamentally different mechanism of detecting critical points of communication and deciding whether to communicate by relying on BDI-inspired trigger points and cost-utility analysis. Both MAOP-COMM and ACE-PJB-COMM reason about how the belief space of an individual agent changes when it does not know its teammate's actions or observations. These possible beliefs are represented by a tree, where each path through the tree represents a joint observation history. Roth (Roth, Simmons, and Veloso 2005) and Wu (Wu, Zilberstein, and Chen 2009) respectively define *GrowTree* and *JointHistoryPool* at depth t to be the set of possible joint beliefs of the team at time t. Each node (i.e., joint belief) in *GrowTree* has a tuple consisting of $\langle \mathbf{b}^t, p^t, \mathbf{h}^t \rangle$, and *JointHistoryPool* has a tuple $\langle \mathbf{b}^t, \mathbf{h}^t \rangle$, where $\mathbf{b^t}$ is the joint belief given that observation history, $\mathbf{h}^t$ is the joint observation history, and p^t is the likelihood of observing $\mathbf{h}^t$.

For example, consider a 1-by-3 grid world with two agents. The agents can wait (W) or move in 2 directions: east (M_e), or west (M_w). The team's goal is to execute a joint action (J) at a pre-specified location to achieve a high positive reward. Each agent can obtain two individual observations: they see that they are at the location of joint task (o) or not ($\bar{o}$). This domain can be represented as a tree shown in Figure 1(a). The initial distribution of possible joint beliefs is composed of a single leaf at belief $\mathbf{b}^0$, the starting belief of the team, with probability 1 and an empty observation history. Suppose that the team executes the action $\langle M_e, M_w \rangle$. The agents must consider the likelihood of all four possible joint observations: $\langle o, o \rangle$, $\langle o, \bar{o} \rangle$, $\langle \bar{o}, o \rangle$, and $\langle \bar{o}, \bar{o} \rangle$. Then, *GrowTree* and *JointHistoryPool* grow to contain four leaves.



$t = 0$
θ^0
$\mathbf{b}^0$ $\mathbf{h}^0 = \phi$
$p^0 = 1.0$ $\mathbf{a}^0 = \langle M_e, M_w \rangle$
$\langle o,o \rangle$
$\langle o,\bar{o} \rangle$
$\langle \bar{o},o \rangle$
$\langle \bar{o},\bar{o} \rangle$
$t = 1$
θ_0^1
θ_1^1
θ_2^1
θ_3^1
$\mathbf{b}_0^1$ $\mathbf{h}_0^1 = \{\langle o,o \rangle\}$
$p_0^1 = 0.64$ $\mathbf{a}_0^1 = \langle J,J \rangle$
$\mathbf{b}_1^1$ $\mathbf{h}_1^1 = \{\langle o,\bar{o} \rangle\}$
$p_1^1 = 0.16$ $\mathbf{a}_1^1 = \langle W,M_w \rangle$
$\mathbf{b}_2^1$ $\mathbf{h}_2^1 = \{\langle \bar{o},o \rangle\}$
$p_2^1 = 0.16$ $\mathbf{a}_2^1 = \langle M_e,W \rangle$
$\mathbf{b}_3^1$ $\mathbf{h}_3^1 = \{\langle \bar{o},\bar{o} \rangle\}$
$p_3^1 = 0.04$ $\mathbf{a}_3^1 = \langle M_e,M_w \rangle$
$t = 0: \phi$
θ^0
$\mathbf{b}^0$ $\mathbf{h}^0 = \phi$
$p^0 = 1.0$ $\mathbf{a}^0 = \langle M_e, M_w \rangle$
Agent 1
observes o
$\langle o,o \rangle$
$\langle o,\bar{o} \rangle$
$t = 1: \{o\}$
θ_0^1
θ_1^1
$\mathbf{b}_0^1$ $\mathbf{h}_0^1 = \{\langle o,o \rangle\}$
$p_0^1 = 0.8$ $\mathbf{a}_0^1 = \langle J,J \rangle$
$\mathbf{b}_1^1$ $\mathbf{h}_1^1 = \{\langle o,\bar{o} \rangle\}$
$p_1^1 = 0.2$ $\mathbf{a}_1^1 = \langle W,M_w \rangle$

(a) JB: Individually maintained by both agents

(b) IB_1: Maintained by agent 1

Figure 1: Example of joint beliefs: IB_1 is subset of JB

BDI teamwork

BDI approaches emphasize the need for execution-time teamwork reasoning as relying on pre-planned team coordination may lead to failures when unanticipated events occur. We draw upon four key ideas from BDI teamwork approaches (Levesque, Cohen, and Nunes 1990; Grosz and Kraus 1996; Tambe 1997).

First, BDI teamwork frameworks simplify planning by focusing on team-oriented programs that abstract away from "low-level coordination," instead shifting coordination reasoning (i.e., communication) to execution time. Second, agents differentiate between individual actions and actions that require interaction with others; indeed, in many teamwork domains, agents act individually for most tasks and only occasionally perform tightly coupled actions. Third, an agent distinguishes between its own individual beliefs and the team's joint beliefs. Fourth, agents only reason about communication when the plan requires interactions with others; agents do not continually reason about communication at each step. For example, key teamwork execution systems have been based on Joint Commitments (Levesque, Cohen, and Nunes 1990). A joint commitment of two agents to a joint goal P leads to two types of communications:

- In order to form a joint commitment, an agent requests others to commit to its goal, P. We refer to this communication as "asking," and an agent's action changes based on response from the other agent.
- Once jointly committed to P, if an agent privately comes to believe that P is achieved, unachievable, or irrelevant, it communicates this information to its teammates. We refer to this form of communication as "telling," and the other agent's action changes due to the communication.

We will use these in our algorithm by having a single-agent POMDP planner plan a policy for the team by: (1) assuming full/free communication and then have agents engage in execution-time reasoning to decide when to actually communicate with other agents, (2) having agents directly execute individual actions in the policy but trigger communication when actions involve interactions with others (as discussed in the following section), (3) allowing agents to maintain an individual estimate of the team's beliefs based on its observations, and maintain an estimate of the entire team's estimated joint beliefs, and (4) adapting "asking" and "telling" to the POMDP framework, as described next.

SELECTIVE-TEAM-COMM

Before explaining the details of our teamwork-inspired planner, we first introduce the concepts of "Trigger Points" and then how belief updates are performed.

Trigger Points: In order to avoid reasoning about communication on every time step, agents use "trigger points," heuristically-chosen time steps at which agents should reason about communication to improve team performance. We later show empirically that this heuristic allows for significantly faster execution times while maintaining superior solution quality compared to other methods.

Definition Time step t is a trigger point for agent i if either of the following conditions are satisfied.

Asking Let h_i^t be the actual individual observation history of agent i. Time step t is an Asking trigger point for agent i if there exist two different $h_{-i}^t, \tilde{h}_{-i}^t \in H_{-i}^t$ such that $\pi_i(\mathbf{h}^t) \neq \pi_i(\tilde{\mathbf{h}}^t)$, where $\mathbf{h}^t = h_i^t \otimes h_{-i}^t$ and $\tilde{\mathbf{h}}^t = h_i^t \otimes \tilde{h}_{-i}^t$.

Telling Time step t is a Telling trigger point for agent i if there exists at least one $h_{-i}^t \in H_{-i}^t$, and two different $h_i^t, \tilde{h}_i^t \in H_i^t$, such that $\pi_{-i}(\mathbf{h}^t) \neq \pi_{-i}(\tilde{\mathbf{h}}^t)$, where $\mathbf{h}^t = h_i^t \otimes h_{-i}^t$ and $\tilde{\mathbf{h}}^t = \tilde{h}_i^t \otimes h_{-i}^t$.

Joint/Individual Estimates of Joint Beliefs: *Joint estimate of joint Beliefs (JB)* and *Individual estimate of joint Beliefs (IB)* are concepts used in STC to estimate the most likely actions of other agents and decide whether or not communication would be beneficial. JB (as shown in Figure 1(a)) and IB (as shown in Figure 1(b)) are represented by trees, which are similar to GrowTree (discussed in the Background Section). JB^t and IB^t describe the set of nodes of the possible belief trees of depth t. IB^t can be conceptualized as a subset of JB^t that depends on an agent's local history. Each node θ in $\mathrm{JB}^t/\mathrm{IB}^t$ has a tuple consisting of $\langle \mathbf{b}(\theta), \mathbf{h}(\theta), \mathbf{a}(\theta), p(\theta) \rangle$, where $\mathbf{b}(\theta)$ is the joint belief given that observation history, $\mathbf{h}(\theta)$ is the joint observation history, $\mathbf{a}(\theta)$ is the joint action obtained from a given policy tree, and $p(\theta)$ is the likelihood of observing $\mathbf{h}(\theta)$. Details to compute JB/IB are given in the following section.

SELECTIVE-TEAM-COMM Algorithm

The STC algorithm takes a joint policy as input. During policy execution, each node in JB/IB is expanded using possible observations and a joint action from the given policy, and then STC detects trigger points based on the possible beliefs in the belief tree. Once an agent detects a trigger point, it reasons about whether communication would be beneficial. If it does communicate, all the agents use their shared histories to agree on a common belief node and can thus choose a joint action for the team. Otherwise, each agent chooses the best locally optimal action that corresponds to the estimated most likely actions of the other agents. If the agents do not detect trigger points, they take individual actions from the given policy, without any additional online planning.

STC is detailed in Algorithm 1. The joint policy π is an input to STC and is a policy of the collapsed POMDP obtained from a standard POMDP planner (Kaelbling, Littman, and Cassandra 1998). In line 1, the initial distribution of possible beliefs, $\mathrm{JB}^0/\mathrm{IB}^0$, is composed of a single node at belief $\mathbf{b}^0$ (the starting belief of the team), which has probability 1, an empty observation history, and a joint action $\mathbf{a}^0$, which is described by the root of π. In line 8, JB and IB are updated according to Algorithm 2. Belief updates follow the standard Bayes update rule. Once the belief tree is updated, STC decides whether or not a trigger point exists on the current time step (line 9). If a trigger point is detected, STC reasons about whether the expected utility of coordination justifies communication (lines 11-23). Specifically, when a trigger point is detected, agent i calculates the expected utility assuming communication happens (line 11). Then, it gets a set of possible actions from IB_i^t, estimates most likely actions for other agents via JB, and gets an optimal individual action of agent i assuming other agents take the most likely action a_{-i}^* (lines 12-15). Agent i communicates when the expected utility gain by communication is higher than a given communication cost σ: $U_C(i) - U_{\mathrm{NC}}(i) > \sigma$ (line 17). Agents can then know the actual joint observation histories and execute the optimal joint action given by the policy (line 21). Otherwise, they execute the estimated best joint action (line 23). We assume communication is instantaneous and lossless.

Algorithm 1 STC(JOINTPOLICY π, AGENTINDEX i)

```
 1: Initialize joint estimate $\mathrm{JB}^0$ and individual estimate $\mathrm{IB}_i^0$
 2: $\mathbf{a}^0 \leftarrow \pi(\mathrm{IB}_i^0)$
 3: Execute action $a_i^0$
 4: $\tau \leftarrow false$
 5: for $t = 1, \ldots, T-1$ do
 6:     $o_i^t \leftarrow$ Get the observation from the environment
 7:     $h_i^t \leftarrow$ Update agent $i$'s own local history with $o_i^t$
 8:     $\{\mathrm{JB}^t, \mathrm{IB}_i^t\} \leftarrow$ EXPAND($\mathrm{JB}^{t-1}, \mathrm{IB}_i^{t-1}, o_i^t$)
 9:     $\tau \leftarrow$ DETECTTRIGGERPOINT($\pi, \mathrm{IB}_i^t, \mathrm{JB}^t$)
10:     IF $\tau = true$ THEN
11:         $U_C \leftarrow \sum_{\theta \in \mathrm{IB}_i^t} p(\theta) \cdot V(\mathbf{b}(\theta), \mathbf{a}(\theta))$
12:         $A_i^t \leftarrow \{\pi_i(\mathbf{h}^t(\theta)) | \theta \in \mathrm{IB}_i^t\}$
13:         $\theta^* = \arg\max_{\theta \in \mathrm{JB}^t} p(\theta)$
14:         $a_{-i}^* = a_{-i}(\theta^*)$
15:         $a_i^* \leftarrow \arg\max_{a_i \in A_i^t} \sum_{\theta \in \mathrm{IB}_i^t} p(\theta) V(\mathbf{b}(\theta), \langle a_i, a_{-i}^* \rangle)$
16:         $U_{\mathrm{NC}} \leftarrow \sum_{\theta \in \mathrm{IB}_i^t} p(\theta) \cdot V(\mathbf{b}(\theta), \langle a_i^*, a_{-i}^* \rangle)$
17:         IF $U_C - U_{\mathrm{NC}} > \sigma$ THEN
18:             SYNC $h_i^t$ WITH OTHER AGENTS
19:             $\tau \leftarrow false$
20:             $\{\mathrm{JB}^t, \mathrm{IB}_i^t\} \leftarrow$ UPDATE BELIEF VIA COMMUNICATED JOINT HISTORY $\mathbf{h}^t$
21:             $a_i^t \leftarrow \pi_i(\mathrm{IB}_i^t)$
22:         ELSE
23:             $a_i^t \leftarrow a_i^*$
24:     ELSE
25:         $a_i^t \leftarrow \pi_i(\mathrm{IB}_i^t)$
26:     EXECUTE THE ACTION $a_i^t$
```

Pruning

The number of possible beliefs in JB/IB grows rapidly over time, particularly when agents choose not to communicate for long periods of time. To address this problem, we use a new pruning algorithm that keeps a fixed number of "most likely" beliefs in JB and IB, which are used to reason about communication. This is similar to memory-bounded policy search (Seuken and Zilberstein 2007) and low probability pruning (Emery-Montemerlo et al. 2005), but different from particle filters (Roth, Simmons, and Veloso 2005) and joint history merging (Wu, Zilberstein, and Chen 2009).

Our pruning method first expands beliefs using the Bayes update rule and then selects the most likely belief, θ, at each time step until the selected number of beliefs reaches a predefined upper-limit. This reduced belief set is used to detect trigger points and reason about communication in STC. The expanded JB and IB are obtained by Algorithm 2. Then, the pruning method retrieves the most likely node from the propagated belief trees. We also provide a process to recover a belief node that has been pruned while bounding the belief nodes at each time step. Specifically, when agents communicate, there is a possibility that they cannot find the matching belief with a communicated local history in JB and IB. Since agents sync and exchange their action and observation history in STC, they can exactly construct the correct belief node across the team members. This provides a way to ascertain the team joint status and avoid miscoordination.

Empirical Validation

In this section, we show that STC successfully plans for DEC-POMDPs with model uncertainty. We evaluate the performance of STC on several domains and compare it with two previous techniques: ACE-PJB-COMM (APC) (Roth, Simmons, and Veloso 2005) and MAOP-COMM (Wu, Zilberstein, and Chen 2009). Since APC and MAOP have only limited abilities to scale up to large domains, we first show results in small domains with state spaces of up to 72 states and a fixed time horizon (T=3). We then scale up STC using pruning and show results for larger domains. The planning time for all algorithms is identical and thus we only measure the wall clock execution-reasoning time. The algorithms are tested in a grid domain and a DEC-Tiger domain. Noise in the transition and observation matrices follow a Dirichlet distribution (which is not known by the planner or the agents). The level of model error is represented by a parameter α in the distribution: error *increases* as α *decreases*. We evaluate the performance under four levels of error by varying α from 10 to 10000. The experiments were run on Intel Core2 Quadcore 2.4GHz CPU with 3GB main memory. All results were averaged over 600 independent trials.

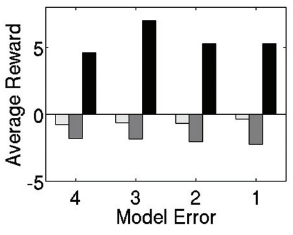
Algorithm 2 EXPAND (JOINTESTIMATE JB^t, INDIVIDUALESTIMATE IB_i^t, LOCALOBSERVATION o_i^t)

```
1: $\{\mathrm{JB}^{t+1}, \mathrm{IB}_i^{t+1}\} \leftarrow \Phi$
2: for all nodes $\theta \in \mathrm{JB}^t$ do
3:     $\mathbf{b}^t \leftarrow \mathbf{b}^t(\theta)$
4:     $\mathbf{a}^t \leftarrow \mathbf{a}^t(\theta)$
5:     for all $s' \in S$ do
6:         $b'(s') \leftarrow \sum_{s \in S} T(s, \mathbf{a}^t, s') b^t(s)$
7:     for all $\mathbf{o} \in \Omega$ do
8:         $p \leftarrow \sum_{s \in S} O(s, \mathbf{a}^t, \mathbf{o}) b'(s)$
9:         $\mathbf{h}^{t+1} \leftarrow \mathbf{h}^t \cup \{\mathbf{o}\}$
10:        for all $s' \in S$ do
11:            $b'(s') \leftarrow O(s', \mathbf{a}^t, \mathbf{o}) \sum_{s \in S} b^t(s)$
12:        $\mathbf{a}^{t+1} \leftarrow \pi(\mathbf{h}^{t+1})$
13:        $\mathrm{JB}^{t+1} \leftarrow \mathrm{JB}^{t+1} \cup \langle \mathbf{b}', \mathbf{h}^{t+1}, \mathbf{a}^{t+1}, p \rangle$
14: for all nodes $\theta$ from $\mathrm{IB}_i^t$ do
15:     $\mathbf{b}^t \leftarrow \mathbf{b}^t(\theta)$
16:     $\mathbf{a}^t \leftarrow \mathbf{a}^t(\theta)$
17:     for all $s' \in S$ do
18:         $b'(s') \leftarrow \sum_{s \in S} T(s, \mathbf{a}^t, s') b^t(s)$
19:     for all $o_{-i} \in \Omega_{-i}$ do
20:         $\mathbf{o} \leftarrow \langle o_i^t, o_{-i} \rangle$
21:         $p \leftarrow \sum_{s \in S} O(s, \mathbf{a}^t, \mathbf{o}) b'(s)$
22:         $\mathbf{h}^{t+1} \leftarrow \mathbf{h}^t \cup \{\mathbf{o}\}$
23:         for all $s' \in S$ do
24:             $b'(s') \leftarrow O(s', \mathbf{a}^t, \mathbf{o}) \sum_{s \in S} b^t(s)$
25:         $\mathbf{a}^{t+1} \leftarrow \pi(\mathbf{h}^{t+1})$
26:         $\mathrm{IB}_i^{t+1} \leftarrow \mathrm{IB}_i^{t+1} \cup \langle \mathbf{b}', \mathbf{h}^{t+1}, \mathbf{a}^{t+1}, p \rangle$
27: normalize $\mathrm{JB}^{t+1}$ s.t. $\sum_{\theta \in \mathrm{JB}^{t+1}} p(\theta) = 1$
28: normalize $\mathrm{IB}_i^{t+1}$ s.t. $\sum_{\theta \in \mathrm{IB}_i^{t+1}} p(\theta) = 1$
29: return $\{\mathrm{JB}^{t+1}, \mathrm{IB}_i^{t+1}\}$
```

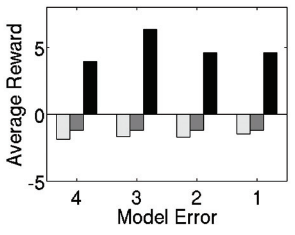
Small Domains, No Pruning

Grid Domain: A 2-by-3 grid was used for evaluation. In this domain, there are two agents trying to perform one joint task. We denote the location of a cell as (i, j), where i is the row index and j is column index. Two agents are initially located in cells $(2, 1)$ and $(2, 3)$, and the joint task is located in cell $(1, 2)$. Each agent has five actions: *move east*, *move west*, *move north*, *move south* and *perform joint task*. Each agent can obtain two observations: *at-location-of-joint-task* and *not-at-location-of-joint-task*. There are 72 joint states, 25 joint actions and 4 joint observations in this configuration. Every movement action incurs a small penalty of -0.2. The joint task requires that both agents perform the task together at the joint location. If the joint task is successfully performed, a reward of +10 is obtained. If only one agent performs the joint task, a mis-coordination penalty of -5 is given to the team. A penalty of -2 is given if the joint task action is performed by both agents but at least one of the agents is at the wrong location. The time horizon is set to 3, as this is sufficient to complete the joint task.

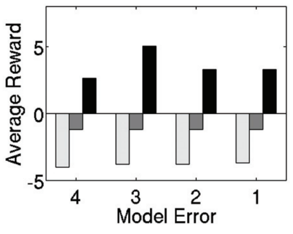
Dec-Tiger Domain: The second domain used for our evaluation is the Dec-Tiger domain (Nair et al. 2004). We included this domain because APC was built specifically for this domain. While STC focuses on domains where interactions among agents are sparse, tiger domain has highly coupled actions among agents. In these experiments, the time horizon is set to 3 as well.



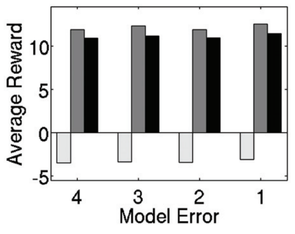
(a) 2-by-3, Cc 5%



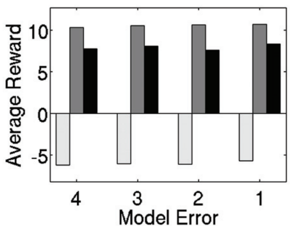
(b) 2-by-3, Cc 20%



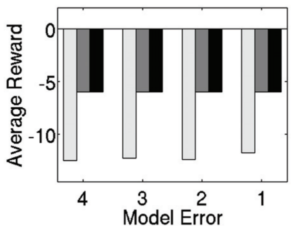
(c) 2-by-3, Cc 50%



(d) Tiger, Cc 5%



(e) Tiger, Cc 20%



(f) Tiger, Cc 50%

Figure 2: Comparing STC with APC and MAOP: Average Performance in Small Domains

Solution Quality Comparison: We compared the average rewards achieved by all algorithms for three different communication costs in all three domain settings. The communication costs are selected proportional to the expected value of the policies: $5\%, 20\%$, and 50%. The reward comparisons are shown in Figure 2. In Figure 2, the x-axis shows α, the level of model error and y-axis shows the average reward of each algorithm. Figures 2(a)–2(c) show the results in the 2-by-3 grid domain. Figures 2(d)–2(f) are for the multi-agent tiger domain.

STC received much higher reward than both APC and MAOP in the grid domain, whereas APC has slightly higher reward than STC in the tiger domain. As communication cost increases, the reward obtained by all three algorithms decrease. With high communication cost (50%), STC is still able to achieve positive reward but rewards obtained by APC and MAOP become negative. The average reward of MAOP in particular is affected by communication cost since it does not reason about communication cost during execution. The Dec-Tiger domain is qualitatively different from the grid domain because every action is a tightly-coupled action, allowing APC to outperform STC.

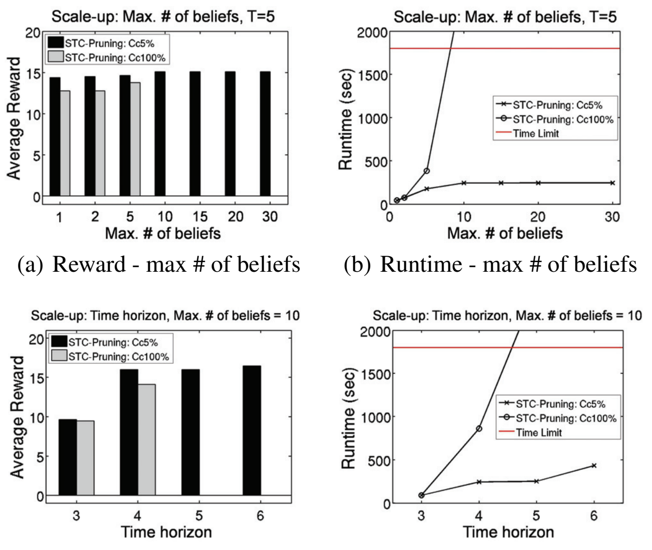
Runtime Comparison: We also compared the average (execution) runtime of the algorithms. In all domains, STC and APC take similar amounts of time, but STC accrues significantly higher rewards than APC. The runtime of MAOP is 1.3–1.5 times that of STC's runtime in both domains. A more interesting question is whether we could scale up the algorithms to run in even larger domains or with longer time horizons. As we will show in the next section, pruning techniques are necessary for STC to handle larger domains.

Large Domains with Pruning

Grid Domain with Individual Task: In this section, we consider the following modification to the 2-by-3 domain. Two individual tasks are added to the grid, which require only one agent to perform. In this new domain, each agent has one additional observation: *location-of-individual-task* and two additional actions: *perform-individual-task* and *stay*. The number of joint states is 288, the number of joint actions is 49, and the number of joint observations is 9. If any agent performs the individual task action at the correct location, the team receives a reward of +5. If both agents try to perform the same individual task, the team reward remains +5. On the other hand, if an agent attempts to performs the individual task at any incorrect location, a penalty of -1 will be assessed. If an agent chooses the action *stay*, there will be no penalty or reward. Other than the above modifications, the new domain is the same as the 2-by-3 grid domain defined in the previous section.

Scale-up in Maximum Number of Beliefs: We show how the algorithm scales with respect to the maximum number of possible beliefs in JB and IB. We tested the algorithms with two different communication costs, 5% and 100% of the expected value of the policy. The maximum runtime per trial was set to 1,800 seconds. MAOP-COMM was not able to finish running within the time limit. APC uses a particle filtering technique to improve speed, but even with only one particle, APC takes more than one hour to finish a trial, exceeding the time limit. The final reward obtained by APC using one particle is only 2.6. Figure 3(a) shows the reward of STC: the x-axis is the number of belief nodes used in the pruning technique and the y-axis is reward, averaged over 100 independent executions. Note that results are averaged over four different model errors from Figure 2. Figure 3(b) displays the runtime on the y-axis over the same set of experiments. Experiments in the new domain use a time horizon of 5. As can be seen in the figure, with small communication cost (5%), STC takes less than 250 seconds to run and is roughly constant after the maximum belief nodes exceeds 10. The performance of STC also improves as the number of belief nodes increases, up to a total of 10 belief nodes. With very large communication cost (100%), the runtime of STC is much longer since the number of beliefs that agents maintain grows quickly. After the number of maximum beliefs becomes larger than 5, the runtime exceeds the time limit. Thus, we do not report the reward values of the cases that exceed the time limit in Figure 3(a).

Scale-up in Time Horizon: We then ran experiments with increased time horizons. Figure 3(c) shows the reward on the y-axis and the time horizon on the x-axis. Results are averaged over four different model errors from Figure 2. Figure 3(d) shows the runtime on the y-axis over the same set of experiment as in Figure 3(c). We again set an upper



(a) Reward - max # of beliefs (b) Runtime - max # of beliefs

(c) Reward - time horizon (d) Runtime - time horizon

Figure 3: Scale-up results

bound on the execution time to 1,800 seconds and test the algorithm with two different communication costs. STC used 10 belief nodes. MAOP and APC (with 1 particle) could not solve the problem within the given time limit for even the shortest time horizon (i.e., T=3). As the time horizon increases, STC obtains better rewards, since there is more time for agents to recover from any failed actions. With high communication cost, STC takes much more time to run, but the rewards obtained in the two communication settings are similar. We also do not report the reward values of the cases that exceed the time limit in Figure 3(c).

Discussion and Conclusion

This work details a new algorithm for DEC-POMDPs that is more robust to model uncertainty than previous work, with a focus on domains that have sparse inter-agent interactions. Our work is inspired by the belief-desires-intentions (BDI) framework, which, when faced with model uncertainty, shifts teamwork-related computation from plan-time to execution-time. STC relies on the following key ideas: (1) reduce planning-time computation by shifting some of the burden to execution-time reasoning, (2) exploit sparse interactions between agents to reduce the required amount of reasoning about coordination, and (3) maintain an approximate model of agents' beliefs. We empirically compared STC to existing methods for execution-time DEC-POMDP coordination reasoning and showed that STC can provide solutions much faster than existing algorithms while achieving comparable, or even superior, solution quality.

Acknowledgments

We thank the anonymous reviewers for their comments and suggestions, and Maayan Roth for providing us with the source code for ACE-PJB-COMM. This work was supported in part by US. Army SBIR contract number W15P7T-09-C-S601, DARPA SBIR contract number W31P4Q-06-

0286, and Perceptronics Solutions, Inc.

References

Bernstein, D.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research, 27(4), 819 - 840.*

Emery-Montemerlo, R.; Gordon, G. J.; Schneider, J. G.; and Thrun, S. 2005. Game theoretic control for robot teams. In *ICRA*.

Goldman, C. V., and Zilberstein, S. 2003. Optimizing information exchange in cooperative multi-agent systems. In *AAMAS*, 137–144.

Grosz, B., and Kraus, S. 1996. Collaborative plans for complex group actions. *Artificial lntelligence 86:269-358.*

Jaulmes, R.; Pineau, J.; and Precup, D. 2007. A formal framework for robot learning and control under model uncertainty. In *ICRA*.

Kaelbling, L.; Littman, M.; and Cassandra, A. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence, 101.*

Kaminka, G. A., and Frenkel, I. 2007. Integration of coordination mechanisms in the BITE multi-robot architecture. In *ICRA*.

Levesque, H. J.; Cohen, P. R.; and Nunes, J. H. T. 1990. On acting together. In *AAAI*, 94–99.

Nair, R.; Yokoo, M.; Roth, M.; and Tambe, M. 2004. Communication for improving policy computation in distributed POMDPs. In *AAMAS*.

Pynadath, D. V., and Tambe, M. 2002. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *JAIR, Volume 16, pages 389 - 423.*

Roth, M.; Simmons, R.; and Veloso, M. 2005. Reasoning about joint beliefs for execution-time communication decisions. In *AAMAS*.

Seuken, S., and Zilberstein, S. 2007. Memory-bounded dynamic programming for DEC-POMDPs. In *IJCAI*.

Spaan, M. T. J.; Oliehoek, F. A.; and Vlassis, N. 2008. Multiagent planning under uncertainty with stochastic communication delays. In *ICAPS*, 338–345.

Tambe, M. 1997. Towards flexible teamwork. *JAIR, Volume 7, Pages 83-124.*

Wu, F.; Zilberstein, S.; and Chen, X. 2009. Multi-agent online planning with communication. In *ICAPS*.

Xuan, P., and Lesser, V. 2002. Multi-agent policies: from centralized ones to decentralized ones. In *AAMAS*.