Goal-oriented Dec-MDPs with Direct Communication

Claudia V. Goldman Shlomo Zilberstein
Department of Computer Science
University of Massachusetts Amherst, Amherst, MA 01003
{clag,shlomo}@cs.umass.edu

UMass Computer Science Technical Report #04-44

Abstract

Multi-agent planning in stochastic environments can be framed formally as a decentralized Markov decision problem. Finding the optimal joint solution in the general case is hard, limiting the applicability of recently developed algorithms. This paper provides a more practical approach for situations in which communication among the decision makers is possible. We specifically focus on goal-oriented decentralized MDPs, where the agents have the ability to communicate with each other. These problems are particularly difficult to solve because global goal-oriented behavior does not necessarily induce local goal-oriented behavior when communication is allowed. No efficient algorithms are known to date that solve optimally this class of problems, which are common in manufacturing, multi-robot coordination, and information gathering scenarios.

We develop the notion of mechanism design for communication that allows us to decompose a decentralized MDP into multiple single-agent problems. In this framework, referred to as Dec-SMDP-Com, agents operate separately between communications. We show that finding an optimal mechanism is equivalent to solving optimally a Dec-SMDP-Com. We also provide a heuristic search algorithm that converges on the optimal solution. Restricting mechanisms to specific types of local behaviors reduces significantly the complexity leading to a polynomial algorithm. The paper concludes with two additional polynomial algorithms: the first computes a myopic-greedy policy of communication and the second one, based on backward induction, computes the optimal policy of communication for monotonic decentralized MDPs when the policies of action are given. Empirical results show that these approaches provide good approximate answers.

1 Introduction

Decentralized Markov decision processes are becoming a popular formal tool to study multi-agent planning and control problems [3, 14, 15, 22, 26]. Reinforcement learning and heuristic approaches were also pursued to study the problem of coordinating distributed decision makers (e.g., [38, 31, 39]). In [13], we briefly described the connections between these works and our approach. It is already known [4, 28] that solving optimally a general decentralized control problem is very hard. This difficulty is due to two main reasons: 1) none of the decision-makers has full-observability of the global system and 2) the global performance of the system depends on a global reward, which is affected by the agents' behaviors. Solving a decentralized Markov decision problem (Dec-MDP) optimally,[1] without restricting the problem to any specific class requires a

[1] Dec-POMDP problems refer to decentralized partially observable Markov decision processes.

double-exponential algorithm in the worst case. In our previous work [13], we have studied the complexity of solving optimally certain classes of Dec-MDPs and Dec-POMDPs. We showed that decentralized problems with independent transitions and observations are considerably easier to solve, i.e., they are NP-complete. Moreover, we show that adding goal-oriented behavior can even reduce the complexity to polynomial when information sharing is not possible. However, no efficient algorithm is known that finds the optimal decentralized solution to a goal-oriented Dec-MDP-Com with direct communication and many global goal states (we showed that in the worst case this algorithm time is NP). This problem is complex because global goal-directed behavior does not induce local goal-directed behavior. Information that can be acquired by the agents during execution may affect the choice of the most beneficial global goal state.

The contribution of this paper is a tractable approximation method, based on mechanisms for communication. This approach provides a practical way to solve these goal-oriented decentralized problems when exchange of messages is allowed (GO-Dec-MDP-Com). Our approach is to allow the agents to exchange information from time to time attaining knowledge of their current global state. Between communications agents will act independently of each other, and therefore may not follow the optimal policy of action. This paper presents the first algorithms that solve this type of problems, including one that computes the optimal mechanism: the algorithms presented compute the mappings from global states to individual behaviors and policies of communication with the highest values. The algorithms presented differ in the space of behaviors they search: our solutions range from the most general search space available to more restricted set of behaviors. We also discuss how to extend this approach to other hard decentralized control problems.

The problem of decentralized control with direct communication was also studied by Ghavamzadeh and Mahadevan [10] and Nair et al. [21]. The first authors assumed an action dependency graph similar to the one used by Guestrin et al. [14] and added a cost to each information exchange. They suggested an approximate algorithm that is based on an on-line learning procedure assuming that the centralized solution is known ahead of time. The policy of communication is determined by the given hierarchy of tasks. That is, an agent will communicate if it is beneficial to incur the communication cost at a known time. Therefore, communication can not prevent an agent from doing a certain task if it has already started to perform it. We take an off-line approach to compute the optimal policy of communication considering that information exchanges incur some cost and that the temporal abstracted actions can be interrupted at any time. The second group of researchers compute a local optimal joint policy that includes both domain and communication actions. It is not clear how close this approximation is to the optimal joint solution. We know [13] that solving a decentralized MDP with direct communication and independent transitions and observations is NP-complete. We show first an algorithm that computes the optimal mechanism, and later a polynomial algorithm, which finds the optimal local goal-oriented behaviors. In addition, we present two polynomial algorithms that compute the policy of communication for a given set of temporal abstracted actions. We characterize the class of problems for which this policy of communication is optimal. The last two algorithms take advantage of possible existing human knowledge that could be combined into the decentralized model to solve decentralized Markov decision problems optimally. The closer the human-designed local plans are to local optimal behaviors, the closer our mechanism approach solution will be to the optimal joint solution. Balch and Arkin's [2] approach to communication between robots is inspired by biological models and refers to specific tasks such as foraging, consumption and grazing. Their empirical study was performed in the context of reactive systems and communication was free. Our general aim is to find optimal policies of communication and action off-line taking into account information that agents can acquire on-line. Game theory researchers [19, 1, 37, 5] have also looked at communication although the approaches and questions

are somewhat different from ours. We study sequential decision making problems where all the agents can send various types of messages, which incur some cost.

Section 3 introduces the notion of mechanism for communication as a method to decompose decentralized problems into temporary single-agent problems, which are assigned based on information exchanged from time to time (also determined by the mechanism). We formally frame this approach as a decentralized semi-Markov decision process with direct communication (Dec-SMDP-Com) in Section 4. We are interested in the mechanism approach as a way to solve decentralized processes where direct communication is allowed to overcome the lack of global information. Single-agent semi-Markov decision processes with concurrent actions were studied when a set of temporal abstracted actions were given as part of the model [29]. Communication was not relevant in that model since there was only one agent controlling a single process. We are also interested in computing the temporal abstractions out from the primitive domain actions and communication acts. Section 5 presents an algorithm that computes the optimal mechanism. The decentralized multi-step backup policy-iteration algorithm solves for the optimal communication actions and domain actions altogether. Unfortunately, searching all possible trees of primitive actions is hard. Section 6 presents the first practical solution, assuming that each agent can be assigned local goal states as local planning problems. For each global state, the algorithm finds the optimal pair of local goals and the period of time given to the agents to work on these goals. Agents communicate at the end of the period of time allowed for acting even though they may not have reached their local goals. Assuming local goal-oriented behavior reduces the complexity of the problem to polynomial in the number of states. In the first two solutions presented in the paper, the policy of communication is found together with the policy of action. Empirical results are shown for a manufacturing line process in Section 6.2.

Another way to approach the decentralized control problem with a mechanism approximation is to assume that the decomposition of a global problem into single agent problems is given. However, the policy of communication needs to be computed. We have implemented a myopic-greedy solution (Section 7) and a backward-induction solution (Section 8) to compute the greedy and the optimal policies of communication, respectively. We prove that the backward-induction algorithm finds indeed the optimal solution for monotonic Dec-MDPs with a given mechanism. Both methods run in polynomial time. These approximations require that each agent's local policy be composed of two separate policies, one for communication and one for action. Empirical results are shown for the Meeting under Uncertainty scenario presented in Appendix B. We view the mechanism for communication as a general approach to approximate the optimal joint solution of a decentralized control problem. Section 9 discusses a similar mapping from global states to individual behaviors that could be computed in processes which are not necessarily goal oriented. Section 10 concludes this research.

2 The Dec-MDP model

In this paper, we concentrate on goal-oriented decentralized Markov processes.[2] The general underlying process, which allows the agents to exchange messages directly with each other is a decentralized POMDP with direct communication:

[2]All the definitions are taken from Goldman and Zilberstein [13]. The definitions relevant to this paper are brought here to make the paper self-explanatory. For simplicity of exposition, we present our definitions for a system composed of two agents.

Definition 1 (Dec-POMDP-Com) *A decentralized partially-observable Markov decision process with direct communication, Dec-POMDP-Com is given by the following tuple:*
$\overline{M} =< S, A_1, A_2, \Sigma, C_\Sigma, P, R, \Omega_1, \Omega_2, O, T >$, *where*

- S *is a finite set of world states with a distinguished initial state* s^0.
- A_1 *and* A_2 *are finite sets of control actions.* a_i *denotes the action performed by agent* i.
- Σ *denotes the alphabet of messages and* $\sigma_i \in \Sigma$ *represents an atomic message sent by agent* i *(i.e.,* σ_i *is a letter in the language).*
- C_Σ *is the cost of transmitting an atomic message:* $C_\Sigma : \Sigma \rightarrow \Re$. *The cost of transmitting a null message is zero.*
- P *is the transition probability function.* $P(s'|s, a_1, a_2)$ *is the probability of moving from state* $s \in S$ *to state* $s' \in S$ *when agents* 1 *and* 2 *perform actions* a_1 *and* a_2 *respectively. We note that the transition model is stationary, i.e., it is independent of time.*
- R *is the global reward function.* $R(s, a_1, a_2, s')$ *represents the reward obtained by the system as a whole, when agent* 1 *executes action* a_1 *and agent* 2 *executes action* a_2 *in state* s *resulting in a transition to state* s'.
- Ω_1 *and* Ω_2 *are finite sets of observations.*
- O *is the observation function.* $O(o_1, o_2|s, a_1, a_2, s')$ *is the probability of observing* o_1 *and* o_2 *(respectively by the two agents) when in state* s *agent* 1 *takes action* a_1 *and agent* 2 *takes action* a_2, *resulting is state* s'.
- *If the Dec-POMDP has a finite horizon, it is represented by a positive integer* T.

We assume that the system has independent observations and transitions. Assuming that $s = (s_1, s_2) \in S$ are the factored states of the system, a_i the domain actions and o_i the agents' observations, the formal definitions for decentralized processes with independent transitions, and observations follow.

Definition 2 (A Dec-POMDP with Independent Transitions) *A Dec-POMDP has independent transitions if the set* S *of states can be factored into two components* S_1 *and* S_2 *such that:*

$$\forall s_1, s_1' \in S_1, \forall s_2, s_2' \in S_2, \forall a_1 \in A_1, \forall a_2 \in A_2,$$

$$Pr(s_1'|(s_1, s_2), a_1, a_2, s_2') = Pr(s_1'|s_1, a_1) \wedge$$
$$Pr(s_2'|(s_1, s_2), a_1, a_2, s_1') = Pr(s_2'|s_2, a_2).$$

In other words, the transition probability P *of the Dec-POMDP can be represented as* $P = P_1 \times P_2$, *where* $P_1 = Pr(s_1'|s_1, a_1)$ *and* $P_2 = Pr(s_2'|s_2, a_2)$.

Definition 3 (A Dec-POMDP with Independent Observations) *A Dec-POMDP has independent observations if the set* S *of states can be factored into two components* S_1 *and* S_2 *such that:*

$$\forall o_1 \in \Omega_1, \forall o_2 \in \Omega_2, \forall s = (s_1, s_2), s' = (s_1', s_2') \in S, \forall a_1 \in A_1, \forall a_2 \in A_2,$$

$$Pr(o_1|(s_1, s_2), a_1, a_2, (s_1', s_2'), o_2) = Pr(o_1|s_1, a_1, s_1') \wedge$$

$$Pr(o_2|(s_1,s_2),a_1,a_2,(s'_1,s'_2),o_1) = Pr(o_2|s_2,a_2,s'_2)$$

$$O(o_1,o_2|(s_1,s_2),a_1,a_2,(s'_1,s'_2)) = Pr(o_1|(s_1,s_2),a_1,a_2,(s'_1,s'_2),o_2)\times Pr(o_2|(s_1,s_2),a_1,a_2,(s'_1,s'_2),o_1).$$

In other words, the observation probability O of the Dec-POMDP can be decomposed into two observation probabilities O_1 and O_2, such that $O_1 = Pr(o_1|(s_1,s_2),a_1,a_2,(s'_1,s'_2),o_2)$ and $O_2 = Pr(o_2|(s_1,s_2),a_1,a_2,(s'_1,s'_2),o_1)$.

Definition 4 (Dec-MDP) *A decentralized Markov decision process (Dec-MDP) is a Dec-POMDP, which is jointly fully observable, i.e., the combination of both agents' observations determine the global state of the system (note that none of the agents can see the global state).*

We have proved that Dec-MDPs with independent transitions and observations are locally fully-observable. In particular, exchanging the last observation is sufficient to obtain complete information about the current global state and guarantees optimality of the solution [13].

In this paper, we concentrate on a particular hard class of Dec-MDP problems comprised of goal-oriented decentralized problems with independent transitions and observations. A goal-oriented Dec-MDP $\overline{GM}$ has the following characteristics:

Definition 5 (Finite-horizon Goal-oriented Dec-MDPs (GO-Dec-MDP)) *A finite-horizon Dec-MDP is* goal-oriented *if the following conditions hold:*

1. *There exists a special subset G of S of global goal states. At least, one of the global goal states $g \in G$ is reachable by some joint policy.*
2. *The process ends at time T (the finite horizon of the problem).*
3. *All actions in A incur a cost, $C(a_i) < 0$. For simplicity, we assume in this paper that the cost of an action depends only on the action. In general, this cost may also depend on the state.*
4. *The global reward is $R(s,a_1,a_2,s') = C(a_1)+C(a_2)$.*
5. *If at time T, the system is in a state $s \in G$ there is an additional reward $JR(s) \in \Re$ that is awarded to the system for reaching a global goal state.*

We present the mechanism for communication approach for goal-oriented Dec-MDPs with direct communication and independent transitions and observations. Section 9 extends this approach to more general Dec-MDP problems.

3 Mechanism Design for Communication

We introduce the notion of mechanisms for communication as a practical approach for approximating the optimal joint policy of a decentralized control problem.We borrow from game theory and economics the notion of mechanism design [20].[3] In order to reduce the complexity of solving optimally the general control problem, we propose to design mechanisms for decentralizing the control, allowing the agents to synchronize their partial views from time to time through communication. That is, a mechanism reduces the decentralized optimization problem to a series of temporary and individual behaviors. Suggesting a mechanism comprises also a policy of communication that instructs the agents when to obtain global information. The mechanism is applied on

[3]Our approach is different from other studies done on algorithmic mechanism design (See related work in Appendix A).

each global state revealed, each time the agents communicate. Between communications, agents follow individual behaviors assigned by the mechanism.

A decentralizing control mechanism DCM is a function from any global state of the decentralized problem to two single agent behaviors or policies:[4] $DCM : S \rightarrow \{opt_1, opt_2\}$. In order to study mechanisms for communication, we draw an analogy between temporary and local policies of actions of each agent and options. Options were defined by Sutton et al. [34] as temporal abstracted actions, formalized as triplets including a stochastic single-agent policy, a termination condition, and a set of states in which they can be initiated: $Opt =< \pi : S \times A \rightarrow [0,1], \beta : S^+ \rightarrow [0,1], I \subseteq S >$. An option is available in a state s if $s \in I$.

In our approach, we consider options with *terminal actions* (instead of terminal states). Terminal actions were also considered by Hansen and Zhou in the framework of indefinite POMDPs [17]. We denote the domain actions of agent i as A_i. The set of terminal actions includes only the messages that could be exchanged, i.e., Σ. For one agent, an option is given by the following tuple: $Opt_i =< \pi : S_i \times T \rightarrow A_i \bigcup \Sigma, I \subseteq S_i >$, i.e., an option is a non-stochastic policy from the agent's partial view (local states) and time to the set of its primitive domain actions and terminal actions. The local states S_i are given by the factored representation of the Dec-MDP with independent transitions and observations. Similarly, the transitions between local states are known since $P(s'|s, a_1, a_2) = P_1(s'_1|s_1, a_1)P_2(s'_2|s_2, a_2)$.

In this paper, we concentrate on terminal actions that are necessarily communication actions. We assume that all options are terminated whenever at least one of the agents initiates communication (i.e., the option of the message sender terminates when it communicates and the hearer's option terminates due to this external event). We also assume that there is joint exchange of messages, i.e., whenever one agent initiates communication, the global state of the system is revealed to all the agents: when agent 1 sends its observation o_1, it will also receive agent 2's observation o_2. This exchange of messages will cost the system only once. Here, we focus on finite-horizon processes, so the options may also be artificially terminated if the time limit of the problem is over. The cost of communication C_Σ may include, in addition to the actual transmission cost, the cost resulting from the complexity of computing the agents' local policies.

Decentralizing control mechanisms enable the agents to operate separately for certain periods of time. The question, then, is how to design mechanisms that will approximate best the optimal joint policy of the decentralized problem. The best approximation is obtained when we compute the optimal mechanism among all possible mechanisms (i.e., we need to search over all possible pairs of local single-agent policies and communication policies). Mechanisms with lower complexity can be computed by restricting the characteristics of the options allowed. We can talk, then, about the optimal mechanism for a certain set of options (e.g., goal-oriented options). Sections 4-6 study this approach. Sometimes, a restricted set of options may be available to the designer of a mechanism. For example, knowledge about individual procedures may already exist. The mechanism approach allows us to combine such human-designed knowledge into the solution of the decentralized problem. Assuming that a mapping is given for every global state to single-agent behaviors, the computation of a mechanism will involve computing the policy of communication that will synchronize the agents' partial information. In such case, the local policy of communication is computed at the meta-level of control. In Sections 7 and 8, we study a greedy approach and an optimal algorithm for computing a policy of communication when knowledge about local behaviors is given.

Practical concerns lead us to the study of mechanisms for communication. In order to design applicable mechanisms, three desirable properties need to be considered:

[4] In general, a mechanism can be applied to systems with n agents, in which case the decomposition of the decentralized process will be into n individual behaviors.

- **Computational complexity** — The whole motivation behind the mechanism approach is based on the idea that the mechanism itself has low computational complexity. Therefore, the computation of the DCM mapping should be practical in the sense that the two single-agent individual behaviors will have complexity that is lower than the complexity of the decentralized problem with communication at free cost. There is a trade-off between the complexity of computing a mechanism and the global reward of the system. There may not be a simple way to split the decentralized process into two separate local behaviors. The complexity characteristic should be taken into account when designing a mechanism; different mechanisms can be computed at different levels of difficulty.

- **Complete** — A mechanism is complete if there exists a communication policy that guarantees that the agents reach one of the global goals whenever it is possible.

- **Efficient** — A mechanism DCM_1 is more efficient than another mechanism DCM_2 if the global reward attained by DCM_1 with some policy of communication is larger than the global reward attained by DCM_2 with any communication policy. A mechanism is *optimal* for a certain problem if there is no mechanism that is more efficient.

4 Decentralized Semi-Markov Decision Problems

The problem of solving a GO-Dec-MDP-Com with a mechanism can be stated as a decentralized semi-Markov problem with direct communication over temporal abstracted actions. Formally, a GO-Dec-SMDP-Com is given as follows:

Definition 6 (GO-Dec-SMDP-Com) *A factored, finite-horizon goal-oriented Dec-SMDP-Com over an underlying goal-oriented Dec-MDP-Com $\overline{GM}$ is a tuple $< \overline{GM}, Opt_1, Opt_2, P^N, R^N >$ where:*

- *S, Σ, C_Σ, Ω_1, Ω_2, O and T are components of the underlying process $\overline{GM}$ defined in definitions 5, 4 and 1.*

- *Opt_i is the set of actions of agent i. It comprises the possible options that an agent can choose to perform, which terminate necessarily with a communication act: $Opt_i =< \pi : S_i \times T \rightarrow A_i \bigcup \Sigma, I \subseteq S_i >$.*

- *$P^N(s', t+N|s, t, opt_1, opt_2)$ is the probability of the system reaching state s' after exactly N time units ($t+N < T$), when at least one option terminates (necessarily with a communication act). We are assuming that after N time steps at least one agent initiates communication (for the first time since time t) and this interrupts the option of the hearer agent. Then, both agents get full observability of the synchronized state. Assuming that the decentralized process has independent transitions and observations, P^N is the probability that either agent has communicated or both of them have. The probability that agent i terminated its option exactly at time $t+N$, P_i^N, is given as follows:*

$$P_i^N(s_i', t+N|s_i, t, opt_i) = \begin{cases} 1 & \text{if } (\pi_{opt_i}(s_i,t) \in \Sigma) \wedge (N=1) \wedge (s_i'=s_i)) \\ 0 & \text{if } (\pi_{opt_i}(s_i,t) \in \Sigma) \wedge (N=1) \wedge (s_i' \neq s_i)) \\ 0 & \text{if } (\pi_{opt_i}(s_i,t) \in A) \wedge (N=1)) \\ 0 & \text{if } (\pi_{opt_i}(s_i,t) \in \Sigma) \wedge (N>1)) \\ & \text{if } (\pi_{opt_i}(s_i,t) \in A) \wedge (N>1)) \\ \Sigma_{q_i} Pr(q_i|s_i, \pi_{opt_i}(s_i,t)) P_i^N(s_i', t+N|q_i, t+1, opt_i) \end{cases}$$

The single-agent probability is one when the policy of the option instructs the agent to communicate (i.e., $\pi_{opt_i}(s_i,t) \in \Sigma$) and the local process remains in the same local state. Finally, we obtain that:

$$P^N(s',t{+}N|s,t,opt_1,opt_2) = P_1^N(s'_1,t{+}N|s_1,t,opt_1)(1 - P_2^N(s'_2,t{+}N|s_2,t,opt_2))+$$
$$P_2^N(s'_2,t{+}N|s_2,t,opt_2)(1 - P_1^N(s'_1,t{+}N|s_1,t,opt_1))+$$
$$P_1^N(s'_1,t{+}N|s_1,t,opt_1)P_2^N(s'_2,t{+}N|s_2,t,opt_2)] =$$
$$P_1^N(s'_1,t{+}N|s_1,t,opt_1))+P_2^N(s'_2,t{+}N|s_2,t,opt_2)-P_1^N(s'_1,t{+}N|s_1,t,opt_1))P_2^N(s'_2,t{+}N|s_2,t,opt_2)$$

- *$R^N(s,t,opt_1,opt_2,s',t{+}N)$ is the expected reward obtained by the system N time steps after the agents started options opt_1 and opt_2 respectively in state s at time t, when at least one of them has terminated its option with a communication act (resulting in the termination of the other agent's option).*

$$R^N(s,t,opt_1,opt_2,s',t{+}N) = \begin{cases} JR(s') & \text{if } s' \in G \text{ and } t{+}N = T \\ 2Cost(a)(t{+}N) + C_\Sigma & \text{if } t{+}N < T \end{cases}$$

The dynamics of a semi-Markov decentralized process are as follows: Each agent performs its option starting in some global state s that is fully-observed. Each agent's option is a mapping from local states to actions, so agent i starts the option in state s_i at time t until it terminates in some state s'_i, k time steps later. There are two cases when this option may terminate earlier than $t{+}k$: 1) either $t{+}k{>}T$, where T is the finite horizon; then, even though the action chosen by the option at time $T-1$ was not terminal the process will end. 2)Or the other agent communicates before $t{+}k$ terminating the hearer's option. Whenever the options are terminated, the agents get full observability of their global state. If they reach state s' at time $t{+}k{<}T$, then the joint policy chooses a possible different pair of options at state s' at time $t{+}k$ and the process continues.

Since we assume that communication leads to joint exchange of messages, all the agents observe the global state of the system once information is exchanged. This exchange occurs at every global state and therefore all the states are *fully-observable*(as opposed to jointly fully-observable states as in the classical Dec-MDP-Com). In general, this may not always be the case. For example, we may have different models of communication, where partial information may be transferred, or where the flow of information is unidirectional (i.e., only the receiver of the message knows information about the sender, but the sender does not get any information from the receiver).

The local policy for an agent i in the GO-Dec-SMDP-Com is a mapping from the global states to its options (as opposed to 1) a mapping from sequences of observations as in the general Dec-POMDP case, and 2) a mapping from a local state as in the Dec-MDPs with independent transitions and observations, which are locally fully observable).

$$\mu_i : S \times T \rightarrow Opt_i$$

A joint policy is a tuple of local policies, one for each agent, i.e., a joint policy instructs each agent to choose an option in each global state. In other words, solving for an optimal mechanism is equivalent to solving optimally a decentralized semi-Markov decision problems with temporal abstracted actions.

Lemma 1 *A GO-Dec-SMDP-Com is equivalent to a multi-agent MDP.*

Proof. Multiagent MDPs (MMDPs [6]) are tuples of the form $< \alpha, \{A_i\}_{i\in\alpha}, S, Pr, R >$. α is a finite collection of n agents. In particular, these are the agents that are controlling the decentralized process in our scenario. $\{A_i\}_{i\in\alpha}$ represents the joint action space. This corresponds to the options chosen in each global state in our case (e.g., if $n = 2$ then $A_i = \{Opt_1, Opt_2\}$). S is a finite set of system states. In the GO-Dec-SMDP-Com case, the set S is the set of global states, where the agents attained this global information as a result of communication. Pr is the transition probability from states and actions to next states. In the GO-Dec-SMDP-Com, this corresponds to the probability function P^N. Finally, R is the reward function that assigns a real number to states. The GO-Dec-SMDP-Com model includes the function R^N that assigns these values to transitions between states (in the case of global goal-oriented behavior, the system only incurs some negative cost while acting and yields some joint reward JR at time T if the system is at a global goal state then). □

Solving a decentralized semi-Markov process with communication is P-complete because of Lemma 1 and the complexity result known for deciding single agent MDPs [25]. However, the input to this problem not only includes the states but also a double exponential number of domain actions for each agent. The naive solution to the GO-Dec-SMDP-Com problem is to search the space of all possible pairs of options, and find the pair that maximizes the value of each global state. The multi-step policy-iteration algorithm, presented in Section 5, implements a heuristic version of this search that converges to the optimal mechanism. Since the resulting search space becomes intractable for even very simple and small problems, we constrain the mechanisms in Sections 6-8 so that they necessarily assign local goals to each one of the agents, allowing them to communicate before having reached their local goals. Section 6 presents the algorithm that computes the optimal mechanism assuming that the agents can be assigned local goal-oriented behaviors. Another way to simplify this solution is to restrict ourselves to a certain set of options that does not include all the possible actions. The solution to the decentralized semi-Markov problem when the options are restricted can be found in a manner similar to the one shown for the single agent semi-Markov process when the options were restricted (see [34] or equivalently see semi-Markov models [27]). Following Sutton et al., a corresponding optimal policy $\mu^*_{\mathcal{O}}$, given a set of restricted options $\mathcal{O}$ is any policy that achieves $V^*_{\mathcal{O}}$, i.e., for which $V^{\mu^*_{\mathcal{O}}} = V^*_{\mathcal{O}}(s_i)$ in all states $s_i \in S_i$. $V^*_{\mathcal{O}}(s_i)$ is the maximum value that can be obtained over all possible options in the restricted set.

5 Multi-step Backup Policy-Iteration for GO-Dec-MDP-Com

Solving a Dec-SMDP problem optimally means computing the optimal pair of options for each fully-observable global state. These options instruct the agents how to act independently of each other and terminate with exchange of information. In order to find these options for each global state, we apply an adapted version of the multi-step backup policy-iteration algorithm with heuristic search [16]. We prove that this algorithm converges to the optimal policy of the decentralized case with temporal abstracted actions.

We extend the model of the single-agent POMDP with costs [16] to our GO-Dec-SMDP-Com model. From the global perspective, each agent that follows its own option without knowing the global state of the system, is following an open-loop. However, locally, each agent is following an option, which does depend on the agent's local observations. We first define a *multi-step backup* for options, when s and s' are global states of the decentralized problem: $V(s, t, T) =$

$$max_{opt_1,opt_2\in OPT_b}\{\Sigma_{k=1}^{min\{b,T-t\}}\Sigma_{s'}P^N(s', t+k|s, t, opt_1, opt_2)[R^N(s, t, opt_1, opt_2, s', t+k)+V(s', t+k, T)]\}$$

OPT_b is the set of options of length at most b. This length is defined as follows:

Definition 7 (The length of an Option) *The length of an option is k if the option can perform at most k domain actions in one execution.*

Similarly to Hansen's work, b is a bound on the length of the options ($k \leq b$). We analyze here the finite horizon GO-Dec-SMDP-Com case, therefore $b \leq T$. $P^N(s', t+k|s, t, opt_1, opt_2)$ and $R^N(s, t, opt_1, opt_2, s', t+k)$ are taken from the GO-Dec-SMDP-Com model (Definition 6).

We apply the multi-step backup policy-iteration algorithm (see Figure 2) using Hansen's pruning rule [16] that was adapted to work on pairs of policies instead of single linear sequences of actions. The resulting optimal multi-step backup policy is equivalent to the optimal policy of the MMDP (Lemma 1), i.e., it is equivalent to the optimal decentralized policy of a Dec-MDP-Com with temporal abstracted actions. In order to explain the pruning rule for the decentralized case with temporal abstracted actions, we define what policy-trees structures are.

Definition 8 (Policy-tree) *A* policy-tree *is a tree structure, composed of local state nodes and corresponding action nodes at each level. Communication actions can only be assigned to leaves of the tree. The edges connecting an action a (taken at the parent state s_i) with a resulting state s'_i have the transition probability $P(s'_i|s_i, a)$ assigned to them.*

Figure 1 shows a possible policy-tree. An option is represented by a policy-tree with all its leaves assigned communication actions. We denote a policy-tree $s_{root}\alpha$, by the state assigned to

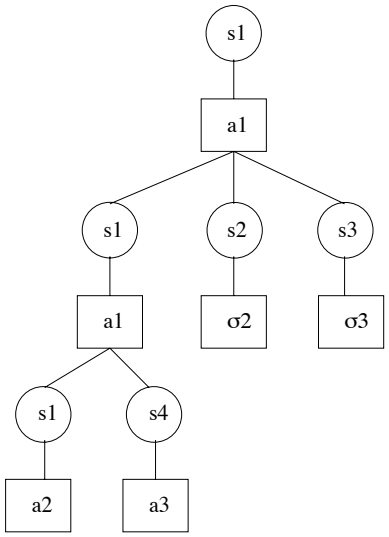

Figure 1: A Policy Tree of Size $k = 3$.

its root (e.g., s_{root}), and an assignment of domain actions and local states to the rest of the nodes (e.g., α). The size of a policy-tree is defined as follows:

Definition 9 (Size of a Policy-tree) *The size of a policy-tree is k if the longest branch of the tree, starting from its root is composed of $k-1$ edges.*[5] *A policy-tree of size one includes the root state and the action taken at that state.*

$\pi(\alpha_k)$ is the policy induced by the assignment α with at most k actions in its implementation. The *expected cost g of a policy-tree* $s_{root}\alpha_k$ is the expected cost that will be incurred by an agent

[5]We count the edges between actions and resulting states.

when it implements the policy $\pi(\alpha_k)$. We denote the set of nodes in a tree that do not correspond to leaves as NL and the set of states assigned to them S_{NL}. We use the notation $\alpha \setminus n$ to refer to the α assignment excluding node n. The expected cost of a tree, $g(s_{root}\alpha_k)$, is given as follows:

$$g(s_{root}\alpha_k) = \begin{cases} Cost(a_{root}) & \text{if } k = 1 \\ Cost(a_{root}) + \Sigma_{s'_i \in S_{NL}}[Pr(s'_i|s_{root}, a_{root})g(s'_i(\alpha \setminus root)_{k-1})] & \text{if } 1 \leq k \leq T \end{cases}$$

Since the decentralized process has factored states, we can write a global state s as a pair (s_1, s_2). Assuming that each agent can act independently of each other for some period of time k, we can refer to the information global state of the system after k time steps as $s_1\alpha_k s_2\beta_k$, where $s_1\alpha_k$ and $s_2\beta_k$ correspond to each agent's policy tree of size k.[6]

The heuristic function that will be used in the search for the optimal decentralized joint policy of the GO-Dec-SMDP-Com follows the traditional notation, i.e., $f(s) = g(s) + h(s)$. In our case, these functions will be defined over pairs of policy-trees, i.e., $f(s\alpha_k\beta_k) = G(s\alpha_k\beta_k) + H(s\alpha_k\beta_k)$. The f value denotes the backed-up value for implementing policies $\pi(\alpha_k)$ and $\pi(\beta_k)$, respectively by the two agents, starting in state s at time t. The expected value of a state s at time t when the horizon is T is given by the multi-step backup for state s as follows:

$$V(s, t, T) = max_{|\alpha|,|\beta| \leq\ b}\{f(s\alpha\beta)\}.$$

(The policy-trees corresponding to the assignments α and β are of size *at most* $b \leq T$).

We define the expected cost of implementing a pair of policy-trees, G, as the sum of the expected costs of each one separately. If the leaves had communication actions, the cost of communication is taken into account in the g functions. If the leaves do not have communication actions assigned to them, similarly to Hansen, we assume for the computation of the f function that the agents can sense and we do not consider the cost then.

$$G(s_1\alpha_k s_2\beta_k) = g(s_1\alpha_k) + g(s_2\beta_k).$$

An option is a policy-tree with communication actions assigned to all its leaves. We denote that option by $opt_1(\alpha_k)$ (or $opt_2(\beta_k)$). The message associated with a leaf corresponds to the local state that is assigned to that leaf by α (or β). We define the expected value of perfect information of the information state $s\alpha\beta$ after k time steps:

$$H(s\alpha_k\beta_k) = \Sigma_{s'} P^N(s', t+k|s, t, opt_1(\alpha_k), opt_2(\beta_k))V(s', t+k, T)$$

If s' is a goal state and $t+k$ equals T then the term $V(s', t+k, T)$ should be replaced with JR(s') that is a special joint reward that the system receives when a global goal is reached at time T. A penalty is awarded if at time T no global goal state is reached. Since we assume that the transitions and observations are independent,

$$H(s\alpha_k\beta_k) = H(s_1\alpha_k s_2\beta_k) = \Sigma_{(s'_1,s'_2)}[(P_1^N(s'_1, t+k|s_1, t, opt_1(\alpha_k)) + P_2^N(s'_2, t+k|s_2, t, opt_2(\beta_k))$$

$$-P_1^N(s'_1, t+k|s_1, t, opt_1(\alpha_k))P_2^N(s'_2, t+k|s_2, t, opt_2(\beta_k)))V((s'_1, s'_2), t+k, T)].$$

The multi-step backup policy-iteration algorithm adapted from Hansen to the decentralized control case appears in Figure 2. Intuitively, the heuristic search over all possible options unfolds as follows: Each node in the search space is composed of two policy-trees, each representing a

[6] We assume that at least one agent communicates at time $t+k$. This will necessarily interrupt the other agent's option at the same time $t+k$. Therefore, it is sufficient to look at pairs of trees of the same size k.

local policy for one agent. The search advances through nodes whose f value (considering both trees) is greater than the value of the global root state (composed of the roots of both policy-trees). All nodes whose f value does not follow this inequality are actually pruned and are not used for updating the joint policy. The policy is updated when a node, composed of two options is found for which $f > V$. All the leaves in these options (at all possible depths) include communication acts. The updated policy δ' maps the global state s to these two options. When all the leaves in one policy-tree at current depth i have communication actions assigned, the algorithm assigns communication acts to all the leaves in the other policy-tree at this same depth. This change in the policies is correct since we assumed joint exchange of information (i.e., all the actions are interrupted when at least one agent communicates). We notice, though, that there may be leaves in these policy-trees at depths lower than i that may still have domain actions assigned. Therefore, these policy-trees cannot be considered options yet and they remain in the stack. Any leaves that remain assigned to domain actions will be expanded by the algorithm. This expansion requires the addition of all the possible next states, that are reachable by performing the domain-actions in the leaves, and the addition of a possible action for each such state. If all the leaves at depth i of one policy-tree are already assigned communication acts, then the algorithm expands only the leaves with domain actions at lower levels in both policy-trees. No leaf will be expanded beyond level i because at the corresponding time one agent is going to initiate communication and this option is going to be interrupted anyways.

Hansen [16] proved that multi-step backup policy-iteration with heuristic pruning converges to the optimal policy in the single-agent case with linear sequences of actions and infinite horizon. In the next section, we show the convergence of the MSBPI algorithm presented in Figure 2 to the optimal decentralized solution of the GO-Dec-SMDP-Com when agents follow temporal abstracted actions and the horizon is finite.

5.1 Optimal Decentralized Solution with Multi-step Backups

In this section, we prove that the MSBPI algorithm presented in Figure 2 converges to the optimal decentralized control joint policy with temporal abstracted actions and direct communication. We first show that the policy improvement step in the algorithm based on heuristic multi-step backups improves the value of the current policy if it is sub-optimal. Finally, the policy iteration algorithm iterates over improving policies and it is known to converge.

Theorem 1 *When the current joint policy is not optimal, the policy improvement step in the multi-step backup policy-iteration algorithm always finds an improved joint policy.*

Proof. We adapt Hansen's proof to our decentralized control problem, when policies are represented by policy-trees. Algorithm MSBPI in Figure 2 updates the current policy when the new policy assigns a pair of options that yield a greater value for a certain global state. We show by induction on the size of the options, that at least for one state, a new option is found in the improvement step (step *3.b.ii*).

If the value of any state can be improved by two policy-trees of size one, then an improved joint policy is found because all the policy-trees of size one are evaluated. We initialized δ with such policy-trees. We assume that an improved joint policy can be found with policy-trees of size at most k. We show that an improved joint policy is found with policy-trees of size k. Lets assume that α_k is a policy tree of size k, such that $f(s\alpha_k\beta) > V(s)$ with communication actions assigned to its leaves. We notice that if this is the case then the policy followed by agent 2 will be interrupted at time k at the latest. One possibility is that $s\alpha_k\beta$ is evaluated by the algorithm.

1. **Initialization:** Start with an initial joint policy δ that assigns a pair of options to each global state s.
2. **Policy Evaluation:** $\forall s \in S, V^{\delta}(s,t,T) =$
$\Sigma_{k=1}^{T-t}\Sigma_{s'}P^N(s',t+k|s,t,\pi_{opt_1}(s_1,t),\pi_{opt_2}(s_2,t))[R^N(s,t,\pi_{opt_1}(s_1,t),\pi_{opt_2}(s_2,t),s',t+k)+V^{\delta}(s',t+k,T)]$
3. **Policy Improvement:** For each state $s=(s_1,s_2) \in S$:
   a. **Set-up:**
      Create a search node for each possible pair of policy-trees with length 1 $(s_1\alpha_1, s_2\beta_1)$.
      Compute $f(s\alpha_1\beta_1) = G(s\alpha_1\beta_1) + H(s\alpha_1\beta_1)$.
      Push the search node onto a stack.
   b. **While** the search stack is **not empty**, **do**:
      i. **Get the next pair of policy-trees:**
      Pop a search node off the stack and let it be $(s_1\alpha_i, s_2\beta_i)$
      (the policy-trees of length i starting in state $s=(s_1,s_2)$)
      Let $f(s\alpha_i\beta_i)$ be its estimated value.
      ii. **Possibly update policy:**
         if $(f(s\alpha_i\beta_i) = G(s\alpha_i\beta_i) + H(s\alpha_i\beta_i)) > V(s,t,T)$, then
            if all leaves at depth i in either α_i or β_i have a communication action assigned, then
               Assign a communication action to all the leaves in the other policy-tree at depth i
            if all leaves in depths $\leq i$ in both α_i and β_i have a communication action assigned, then
               Denote these new two options opt_1^i and opt_2^i.
               Let $\delta'(s) = (opt_1^i, opt_2^i)$ and $V(s,t,T) = f(s\alpha_i\beta_i)$.
      iii. **Possibly expand node:**
         If $(f(s\alpha_i\beta_i) = G(s\alpha_i\beta_i) + H(s\alpha_i\beta_i)) > V(s,t,T)$, then
            if ((some of the leaves in either α_i or β_i have domain actions assigned) and
               $((i+2) \leq T)$) then
            /*At $t+1$ the new action is taken and there is a transition to another state at $t+2$*/
            Create the successor node of the two policy-trees of length i,
            by adding all possible transition states and actions to each leaf of each tree
            that does not have a communication action assigned to it.
            Calculate the f value for the new node (i.e., either $f(s\alpha_{i+1}\beta_{i+1})$ if both policy
            trees were expanded, and recalculate $f(s\alpha_i\beta_i)$ if one of them has communication
            actions in all the leaves at depth i)
            Push the node onto the stack.
   /*All nodes with $f < V$ are pruned and are not pushed to the stack.*/
4. **Convergence test:**
   if $\delta = \delta'$ then
      return δ'
   else set $\delta = \delta'$, **GOTO** 2.

Figure 2: Multi-step Backup Policy-iteration Using Depth-first Branch and Bound Search (MSBPI).

Then, an improved joint policy is indeed found. If this pair of policy-trees was not evaluated by the algorithm, it means that α was pruned earlier. We assume that this happened at level i. This means that $f(s\alpha_i\beta) < V(s)$. We assumed that $f(s\alpha_k\beta) > V(s)$ so we obtain that:

$$f(s\alpha_k\beta) > f(s\alpha_i\beta).$$

If we expand the f values in this inequality, we obtain the following:

$$g(s\alpha_i)+g(s\beta)+\Sigma_{s'}P^N(s',t+i|s,opt_1(\alpha_i),opt_2(\beta))[g(s'\alpha(i,k))+g(s'\beta)+\Sigma_{s''}P^N(s'',t+i+k-i)V(s'')] >$$

$$g(s\alpha_i) + g(s\beta) + \Sigma_{s'}P^N(s',t+i|s,opt_1(\alpha_k),opt_2(\beta))V(s',t+i)$$

$\alpha(i,k)$ is the subtree starting at level i and ending at level k starting from s'.
After simplification we obtain:

$$\Sigma_{s'}P^N(s',t+i|s,opt_1(\alpha_i),opt_2(\beta))[g(s'\alpha(i,k)) + g(s'\beta) + \Sigma_{s''}P^N(s'',t+i+k-i)V(s'')] >$$

$$\Sigma_{s'}P^N(s',t+i|s,opt_1(\alpha_k),opt_2(\beta))V(s',t+i)$$

That is, there exists some state s' for which $f(s'\alpha(i,k)\beta) > V(s')$. Since the policy-tree $\alpha(i,k)$ has size less than k, by the induction assumption we obtain that there exists some state s' for which the multi-step backed-up value is increased. Therefore, the policy found in step 3.*b.ii* is indeed an improved policy. □

Lemma 2 *The complexity of computing the optimal mechanism by the MSBPI algorithm is* $O(((|A_1| + |\Sigma|)(|A_2| + |\Sigma|))^{|S|^{T-1}})$.

Proof. Each agent can perform any of the primitive domain actions in A_i and can communicate any possible message in Σ. There can be at most $|S|^{T-1}$ leaves in a policy tree with horizon T and $|S|$ possible resulting states from each transition. Therefore, each time the MSBPI algorithm expands a policy tree (step 3.b.iii in Figure 2), the number of resulting trees is $((|A_1| + |\Sigma|)(|A_2| + |\Sigma|))^{|S|^{T-1}}$. In the worst case, this is the number of trees that the algorithm will develop in one iteration. Therefore, the size of the search space is a function of this number times the number of iterations until convergence. □

As we proved in [13], solving optimally a GO-Dec-MDP-Com is NP in the worst case assuming a naive search algorithm. As we show here, solving for the optimal mechanism is harder although the solution may not be the optimal joint solution of the decentralized process. This is due to the main difference between these two problems. In the GO-Dec-MDP-Com, we know that due to the independent transitions and observations a local state is a sufficient statistic for the history of observations. However, in order to compute an optimal mechanism we need to search in the space of options, that is, no single local state is a sufficient statistic. The search space is larger since each possible option that needs to be considered can be arbitrarily large (with each branch length being bound by T). In the Meeting under Uncertainty example (presented in Appendix B) there are six primitive actions (four move actions, one stay action and a communication action) and 100 states (when $0 \leq x, y < 10$). We need to expand all the possible combinations of pairs of policy-trees leading to a possible addition of $36^{100^{T-1}}$ nodes to the search space at each iteration. Restricting the mechanism to a certain set of possible options, for example goal-oriented options leads to a significant reduction in the complexity of the algorithm as it is shown in the next sections.

6 GO-Dec-SMDP-Com With Local Goal-oriented Behavior

The previous section provided an algorithm that computes the optimal mechanism, searching over all possible combinations of domain and communication actions for each agent. Since this approach is very complex and time consuming, we take the approximation one step further: we look for the optimal mechanism that assigns *goal-oriented options* to each global state.

Definition 10 (Goal-oriented Options) *Let $\pi_{opt_i}(s_i, t)$ be the policy of an option opt_i. Let k be some time limit. If there exists an optimal policy $\delta : S_i \rightarrow A_i$ that minimizes the cost to some goal state component g_i then opt_i is goal-oriented if for every state s_i and $t \leq k$, $\pi_{opt_i}(s_i, t) = \delta(s_i)$ and $\pi_{opt_i}(s_i, k{+}1) \in \Sigma$. We assume that the agents have the capability of executing a NOP action when they are in some goal state component g_i, which incurs zero cost and has no transition effect, i.e., $\delta(g_i) = NOP$.*

We study locally goal-oriented mechanisms, which map each global state to a pair of goal-oriented options. When the mechanism is applied, each agent follows its policy to the corresponding local goal for k time steps. At time $k+1$, the agents exchange information and stop acting (even though they may not have reached their local goal). The agents, then, become synchronized and they are assigned possibly different local goals and a working period k'.

It is important to notice that we are assuming that a set of local goal states is provided. This set must include the components of the global goal states in G. However, this set may include additional local states from S which may be considered temporary and local goals states. We denote this larger set of local goals $\hat{G}_i$

The mechanism approach assumes that agents can operate independent of each other for some period of time. However, if the decentralized process has some kind of dependency in its observations or transitions, this assumption will be violated, i.e., the plans to reach the local goals can interfere with each other (the local goals may not be compatible). We define $\Delta-$*independent* decentralized processes to refer to *nearly-independent* processes whose dependency can be quantified by the cost of their marginal interactions.

Definition 11 ($\Delta-$*independent* Process) *Let $\overline{Cost}_{A_i}(s \rightarrow \hat{g}_k|\hat{g}_j)$ be the expected cost incurred by agent i when following its optimal local policy to reach local goal state $\hat{g}_k$ from state s, while the other agent is following its optimal policy to reach $\hat{g}_j$. A decentralized control process is $\Delta-$independent if $\Delta = max\{\Delta_1, \Delta_2\}$, where Δ_1 and Δ_2 are defined as follows: $\forall \hat{g}_1, \hat{g}'_1 \in \hat{G}_1 \in S_1$, $\hat{g}_2, \hat{g}'_2 \in \hat{G}_2 \in S_2$ and $s \in S$:*

$$\Delta_1 = max_s\{max_{\hat{g}_1}\{max_{\hat{g}_2,\hat{g}'_2}\{\overline{Cost}_{A_1}(s^0 \rightarrow \hat{g}_1|\hat{g}'_2) - \overline{Cost}_{A_1}(s^0 \rightarrow \hat{g}_1|\hat{g}_2)\}\}\}$$

$$\Delta_2 = max_s\{max_{\hat{g}_2}\{max_{\hat{g}_1,\hat{g}'_1}\{\overline{Cost}_{A_2}(s^0 \rightarrow \hat{g}_2|\hat{g}'_1) - \overline{Cost}_{A_2}(s^0 \rightarrow \hat{g}_2|\hat{g}_1)\}\}\}$$

That is, Δ is the maximal difference in cost that an agent may incur when trying to reach one local goal state that interferes with any other possible local goal being reached by the other agent.

When the Dec-MDP has independent transitions and observations the value of Δ is zero. Otherwise, the Δ value denotes the amount of interference that might occur between the agents' locally goal-oriented behaviors. The algorithm proposed in this section computes the mechanism for each global state as a mapping from states to pairs of local goal states ignoring the potential interference. Therefore, the difference between the actual cost that will be incurred by the options found by the algorithm and the optimal options can be at most Δ. Since the mechanism is applied for each

global state for T time steps and this loss in cost can occur in the worst case for both agents, the algorithm presented here is $2T\Delta - optimal$ in the general case.

We provide the details of the algorithm assuming that the control process is goal-oriented, S is factored and $\hat{G}_i \subseteq S_i$ is the set of local goals that can be assigned to agent i. The algorithm that solves the decentralized control problem with communication finds the optimal mapping between global states to local goals and periods of time. We denote this algorithm LGO-MSBPI. We start with an arbitrary joint policy that assigns one pair of local goal states and a number k to each global state. The current joint policy is evaluated and set as the current best known mechanism. Given a joint policy $\delta : S \times T \rightarrow \hat{G}_1 \times \hat{G}_2 \times T$, the value of a state s at a time t, when T is the finite horizon is given in Equation 1: (this value is only computed for states in which $t+k \leq T$).

$$(1) \qquad V^{\delta}(s,t,T) = \begin{cases} JR(s) & \text{if } t=T \text{ and } s \in G \subseteq S \\ Penalty & \text{if } t=T \text{ and } s \notin G \subseteq S \\ 0 & \text{if } t<T \text{ and } s \in \hat{G} \\ & \text{if } t<T \text{ and } s \notin \hat{G} \\ \Sigma_{s'} P_g^N(s', t+k_\delta | s, t, \pi_{\hat{g_1}_\delta}(s_1), \pi_{\hat{g_2}_\delta}(s_2))[2k_\delta Cost(a) + C_\Sigma + V^{\delta}(s', t+k_\delta, T)] \end{cases}$$

To make the notation simpler, we use k_δ and $\hat{g}_{i_\delta}$ to denote the time period k and local goals $\hat{g}_i$ assigned by the policy δ to the relevant state s and current time t. The local policy $\pi_{\hat{g}_{i_\delta}}$ denotes the optimal policy followed by agent i aimed at reaching local goal state $\hat{g}_i$ assigned by policy δ to state s and time t. When the time is over and the local goals assigned by the mechanism are not consistent with the global goals of the system, a constant penalty is awarded. $\mathrm{JR}(s) \in \Re$ denotes an arbitrary joint reward that the system may obtain if the agents reach certain global states $s \in G \subseteq S$ at time T and zero when s is not such a special state and $t < T$. There is a one-to-one mapping between goals and goal-oriented options. That is, the policy $\pi_{g_{i_\delta}}$ can be found by each agent independently by solving optimally each agent's local process $MDP_i = (S_i, P_i, R_i, \hat{G}_i, T)$: The set of global states S is factored and the process has independent transitions, so each agent has its own set of local states and P_i is the primitive transition probability assumed known when we described the options framework. R_i is the cost incurred by an agent when it performs a primitive action a and zero if the agent reaches a goal state in $\hat{G}_i$. T is the finite horizon of the global problem.

P_g^N (with the goal g subscript) is different from the probability function P^N that appears in the former section. P_g^N is the probability of reaching a global state s' after k time steps, while trying to reach $\hat{g}_1$ and $\hat{g}_2$ respectively following the corresponding optimal local policies.

$$P_g^N(s', t+k_\delta | s, t+i, \pi_{\hat{g_1}_\delta}(s_1), \pi_{\hat{g_2}_\delta}(s_2)) = \begin{cases} 1 & \text{if } i=k \text{ and } s=s' \\ 0 & \text{if } i=k \text{ and } s \neq s' \\ \Sigma_{s^*} Pr(s^* | s, \pi_{\hat{g_1}_\delta}(s_1), \pi_{\hat{g_2}_\delta}(s_2)) & \\ \quad P_g^N(s', t+k_\delta | s^*, t+i+1, \pi_{\hat{g_1}_\delta}(s_1^*), \pi_{\hat{g_2}_\delta}(s_2^*)) & \text{if } i<k \end{cases}$$

Each iteration of the LGO-MSBPI algorithm in Figure 3 tries to improve the value of each state by testing all the possible pairs of local goal states with increasing number of time steps allowed until communication. The value of f is computed for each mapping from states to assignments of local goals and periods of time. The f function for a given global state, current time, pair of local goals and a given period of time k expresses the cost incurred by the agents after having acted for k time steps and having communicated at time $k+1$, and the expected value of the reachable states after k time steps (these states are those reached by the agents while following their corresponding optimal local policies towards $\hat{g}_1$ and $\hat{g}_2$ respectively). Formally:

1. **Initialization:** Start with an initial joint policy δ that assigns local goals
   $\hat{g}_i \in \hat{G}_i$ and time periods $k \in N$
   $\forall s \in S, t: \ \delta(s,t) = (\hat{g}_1, \hat{g}_2, k)$
2. **Policy Evaluation:** $\forall s \in S$, Compute $V^{\delta}(s,t,T)$ based on Equation 1.
3. **Policy Improvement:**
   a. $k = 1$
   b. **While** $(k < T)$ **do**
      i. $\forall s, t, \hat{g}_1, \hat{g}_2$: **Compute** $f(s,t,\hat{g}_1,\hat{g}_2,k)$ based on Equations 2,3 and 4.
      ii. **Possible update policy**
         if $f(s,t,\hat{g}_1,\hat{g}_2,k) > V^{\delta}(s,t,T)$ **then**
            $\delta(s,t) \leftarrow (\hat{g}_1, \hat{g}_2, k)$ $/* Communicate\ at\ k+1 */$
            $V^{\delta}(s,t,T) \leftarrow f(s,t,\hat{g}_1,\hat{g}_2,k)$
      iii. **Test joint policies for next extended period of time**
         $k \leftarrow k+1$
4. **Convergence test:**
   if δ did not change in **Step 3 then**
      return δ
   else **GOTO** 2.

Figure 3: Multi-step Backup Policy-iteration With Local Goal-oriented Behavior (LGO-MSBPI).

$$f(s,t,\hat{g}_1,\hat{g}_2,k) = G(s,t,\hat{g}_1,\hat{g}_2,k) + H(s,t,\hat{g}_1,\hat{g}_2,k) \quad (2)$$

$$G(s,t,\hat{g}_1,\hat{g}_2,k) = 2kCost(a) + C_{\Sigma} \quad (3)$$

$$H(s,t,\hat{g}_1,\hat{g}_2,k) = \begin{cases} JR(s) & \text{if } t{=}T \text{ and } s \in G \\ Penalty & \text{if } t{=}T \text{ and } s \notin G \\ 0 & \text{if } t{<}T \text{ and } s \in \hat{G} \\ \Sigma_{s'} P_g^N(s', t+k | s, t, \pi_{\hat{g}_1}(s_1), \pi_{\hat{g}_2}(s_2)) V^{\delta}(s', t+k, T) & \text{otherwise} \end{cases} \quad (4)$$

We notice that the goals referred to in the computation of f are the goals being evaluated by the algorithm. The policy is evaluated (step 2) with the goals *assigned* by the current best policy ($\hat{g}_i$ and k are $\hat{g}_{i_\delta}$ and k_δ).

The LGO-MSBPI algorithm evaluates the f value of *all* the states, times and possible local goals for a certain time period k, and iterates on this computation while increasing the value of k. The current joint policy is updated when the f value for some state s, time t, local goals $\hat{g}_1$ and $\hat{g}_2$ and period k is greater than the value $V^{\delta}(s,t)$ computed for the current best known assignment of local goals and period of time.

6.1 Convergence of the Algorithm and Its Complexity

Lemma 3 *The algorithm LGO-MSBPI in Figure 3 converges to the optimal solution.*

Proof. The set of global states S and the set of local goal states $\hat{G}_i \subseteq S$ are finite. The horizon T is also finite. Therefore, step 3 in the algorithm will terminate. Like the classical policy-iteration, this algorithm also will converge after a finite numbers of calls to step 3 where the policy can only improve its value from iteration to another. □

Lemma 4 *The complexity of computing the optimal mechanism based on local goal behavior by the algorithm LGO-MSBPI is polynomial.*

Proof. Step 2 of the LGO-MSBPI algorithm can be computed with dynamic programming which is polynomial (we compute the value of a state in a backwards manner from a finite horizon T). The complexity of improving a policy in Step 3 is polynomial in the time, number of states and number of goal states, i.e., $O(T^2|S||\hat{G}|)$. In the worst case, every component of a global state can be a local goal state. However, in other cases, $|\hat{G}_i|$ can be much smaller than $|S_i|$ when $\hat{G}_i$ is a strict subset of S_i, decreasing even more the complexity of the algorithm. □

6.2 Experiments - Goal-oriented Options

We exemplify the mechanism approach in a production control scenario. We assume that there are two machines, which can control the production of boxes and cereals: machine M_1 can produce two types of boxes denoted as BA and BB, and machine M_2 can produce two types of cereals, CA and CB. The boxes differ in their presentation to advertise the different cereals included, i.e., either of type A or type B. The process is stochastic in the sense that the machines are not perfect: with probability P_{M_1}, machine one succeeds in producing the intended box (either A or B) and with probability $1 - P_{M_1}$, the machine does not produce any box in that particular time unit. Similarly, we assume P_{M_2} expresses the probability of machine two producing the amount of cereals of type A or B that is required to sell in one box. We study a process with a finite-horizon T, where the joint reward (JR) attained by the system at T is equal to $min\{BA, CA\} + min\{BB, CB\}$ (BA,BB,CA and CB stand for the amount of boxes and cereals required for one box).

An option opt_i in this scenario is described by a pair of numbers (XA, XB), i.e., machine i is instructed to produce XA items (either boxes or cereals) of type A, followed by XB items of type B, followed by XA items of type A and so forth until either the time limit is over or anyone of the machines decides to communicate. Once the machines exchange information, the global state is revealed, i.e., the current number of boxes and cereals produced so far is known. Given a set of goal-oriented options (i.e., each option instructs a machine to produce a certain number of items as a local goal), we found the optimal joint policy of action and communication that solves this problem. The cost of an action was set to -1 to express a time unit that elapsed. We compare the locally goal oriented multi-step backup policy iteration algorithm (LGO-MSBPI) with two other approaches: the *Ideal* case when exchanging information is free and the machines can adapt their production at every step, and the *Full observation* case, when the machines incur a cost when they exchange information and they do it at every step. Tables 1, 2, and 3 present the average utility obtained by the production system when the cost of communication was set to -0.1, -1 and -10 respectively and the joint utility was averaged over 1000 experiments. The initial state was (0,0,0,8), there were no boxes produced and only 8 items of cereals of type B. The finite horizon T was set to 10 and there were seven possible options given by the ratios (0,1),(1,4),(2,3),(1,1),(3,2),(4,1) and (1,0).[7]

Following the mechanism computed by the LGO-MSBPI algorithm, the average number of products produced, averaged over 1000 runs is summarized in Tables 4, 5 and 6. Although the machines incur a higher cost when the mechanism is applied compared to the ideal case (due to the cost of communication), the number of final products ready to sell were almost the same amount. That is, it will take some more time in order to produce the right amount of products

[7]The machines do not need to communicate if they start from (0,0,0,0). The options chosen then are (0,1) and $k = T$.

P_{M_1}, P_{M_2}	Average Utility		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	-17.012	-18.017	-17.7949
$0.2, 0.8$	-16.999	-17.94	-18.0026
$0.8, 0.8$	-11.003	-12.01	-12.446

Table 1: $C_\Sigma = -0.10$,$R_a = -1.0$.

P_{M_1}, P_{M_2}	Average Utility		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	-17.012	-26.99	-19.584
$0.2, 0.8$	-16.999	-26.985	-25.294
$0.8, 0.8$	-11.003	-20.995	-17.908

Table 2: $C_\Sigma = -1.0$,$R_a = -1.0$.

when the policies implemented are those computed by the locally goal oriented multi-step backup policy iteration algorithm. The cost of communication in this scenario can capture the cost of changing the setting of one machine from one production program to another. Therefore, our result is significant when this cost of communication is very high compared to the time that the whole process takes.

7 A Myopic-greedy Approach to Direct Communication

In some cases, it is reasonable to assume that the mapping from any global state to a pair of single-agent behaviors is known and fixed, ahead of time. For example, in settings where each agent is designed ahead of the coordination time (e.g., agents in a manufacturing line represent machines, which may be built ahead of time to implement certain procedures). In this section, we present a polynomial-time algorithm based on the mechanism approach that computes the policy of communication for a given goal-oriented Dec-MDP with independent transitions and observations and given local goal-oriented behaviors. We first present a myopic-greedy approximation, i.e., each time an agent makes a decision, it chooses the action with maximal expected accumulated reward assuming that agents are only able to communicate once along the whole process. Each time the agents exchange information, the mechanism is applied inducing two individual behaviors. In this section, we assume that these are the optimal local policies to reach some local goal. In the former section, the LGO-MSBPI algorithm computed what these local goals should be. In this section, we assume a set of restricted options is given, i.e., these local goals are known and fixed ahead of time for each global state. We denote the given optimal policies of action (with no communication actions) by δ_1^{A*} and δ_2^{A*} respectively. The complexity of computing these policies of action is polynomial (dynamic programming).

The expected global reward of the system, given that the agents do not communicate at all and each follows its corresponding optimal policy δ_i^{A*} is given by the value of the initial state s^0: $\Theta_{nc}^{\delta}(s^0, \delta_1^{A*}, \delta_2^{A*})$. This value can be computed by summing over all possible next states and computing the probability of each agent reaching it, the reward obtained then and the recursive value computed for the next states.

$$\Theta_{nc}^{\delta}(s^0, \delta_1^{A*}, \delta_2^{A*}) = \Sigma_{(s'_1, s'_2)} P_1(s'_1|s_1^0, \delta_1^{A*}(s_1^0)) P_2(s'_2|s_2^0, \delta_2^{A*}(s_2^0))(R(s'|s^0, \delta_1^{A*}(s_1^0), \delta_2^{A*}(s_2^0)) + \Theta_{nc}^{\delta}(s', \delta_1^{A*}, \delta_2^{A*}))$$

P_{M_1}, P_{M_2}	Average Utility		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	-17.012	-117	-17.262
$0.2, 0.8$	-16.999	-117.028	-87.27
$0.8, 0.8$	-11.003	-110.961	-81.798

Table 3: $C_\Sigma = -10.0$,$R_a = -1.0$.

P_{M_1}, P_{M_2}	# Products		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	2.988	2.983	2.771
$0.2, 0.8$	3.001	3.06	2.784
$0.8, 0.8$	8.997	8.99	8.097

Table 4: $C_\Sigma = -0.10$,$R_a = -1.0$.

At each state, each agent decides whether to communicate its partial view or not based on whether the expected cost from following the policies of action, and having communicated is larger or smaller than the expected cost from following these policies of action and not having communicated. We denote the expected cost of the system computed by agent i, when the last synchronized state is s^0, and when the agents communicate once at state s and continue without any communication, $\Theta_c(s^0, s_i, \delta_1^{A*}, \delta_2^{A*})$:

$$\Theta_c(s^0, s_1, \delta_1^{A*}, \delta_2^{A*}) = \Sigma_{s_2}\overline{P_2}(s_2|s_2^0, \delta_2^{A*})(\overline{R}((s_1, s_2)|s^0, \delta_1^{A*}(s_1^0), \delta_2^{A*}(s_2^0)) + \Theta_{nc}^{\delta}((s_1, s_2), \delta_1^{A*}, \delta_2^{A*}) + C_\Sigma * Flag)$$

Flag is zero if the agents reached the global goal state before they reached state s. We denote by $t(s)$ the time stamp in state s. $\overline{P}(s|, s^0, \delta_1^{A*}, \delta_2^{A*})$ is the probability of reaching state s from state s^0, following the given policies of action.

$$\overline{P}(s'|s, \delta_1^{A*}, \delta_2^{A*}) = \begin{cases} 1 & if\ s = s' \\ P(s'|s, \delta_1^{A*}(s_1), \delta_2^{A*}(s_2)) & if\ t(s') = t(s) + 1 \\ 0 & if\ t(s') < t(s) + 1 \\ \Sigma_{s''}\overline{P}(s'|s'', \delta_1^{A*}, \delta_2^{A*})P(s''|s, \delta_1^{A*}, \delta_2^{A*}) & else \end{cases}$$

Similarly, $\overline{P_1}$ ($\overline{P_2}$) can be defined for the probability of reaching s_1' (s_2'), given agent 1 (2)'s current partial view s_1 (s_2) and its policy of action δ_1^{A*} (δ_2^{A*}).

The accumulated reward attained while the agents move from state s^0 to state s is given as follows:

$$\overline{R}(s^0, \delta_1^{A*}, \delta_2^{A*}, s) = \begin{cases} R(s^0, \delta_1^{A*}(s_1), \delta_2^{A*}(s_2), s) & if\ t(s) = t(s^0) + 1 \\ \Sigma_{s''}\overline{P}(s''|\delta_1^{A*}, \delta_2^{A*}, s^0)P(s|\delta_1^{A*}, \delta_2^{A*}, s'') & \\ (\overline{R}(s^0, \delta_1^{A*}, \delta_2^{A*}, s'') + R(s'', \delta_1^{A*}(s_1''), \delta_2^{A*}(s_2''), s)) & if\ t(s) > t(s^0) + 1 \end{cases}$$

Lemma 5 *Deciding a Dec-MDP-Com with the myopic-greedy approach to direct communication is in the P class.*

Proof. Each agent executes its known δ_i^{A*} when the mechanism is applied. When the provided input included only the mapping from states to local goal states, then, finding the optimal single-agent policy that reaches that goal state can be done in polynomial time. The complexity of finding

P_{M_1}, P_{M_2}	# Products		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	2.988	3.01	2.726
$0.2, 0.8$	3.001	3.015	2.678
$0.8, 0.8$	8.997	9.005	8.103

Table 5: $C_\Sigma = -1.0$,$R_a = -1.0$.

P_{M_1}, P_{M_2}	# Products		
	Ideal $C_\Sigma = 0$	Full Observation	LGO-MSBPI
$0.2, 0.2$	2.988	3	2.738
$0.2, 0.8$	3.001	2.972	2.64
$0.8, 0.8$	8.997	9.039	8.132

Table 6: $C_\Sigma = -10.0$,$R_a = -1.0$.

the communication policy is the same as dynamic programming (based on the formulas above), therefore computing the policy of communication is also in P. There are $|S|$ states for which Θ^{δ}_{nc} and Θ_c need to be computed, and each one of these formulas can be solved in time polynomial in $|S|$. □

Lemma 6 $\Theta^{\delta}_{nc}(s^0, \delta_1^{A*}, \delta_2^{A*}) \leq \Theta_c(s^0, s_i, \delta_1^{A*}, \delta_2^{A*})$

Proof. Θ^{δ}_{nc} is the expected joint cost incurred by the joint policy assuming that the agents set a certain global goal state they are planning to reach at time 0, and they do not communicate until they achieve this goal. If the world were deterministic then the value of a joint policy computed by Θ^{δ}_{nc} will be equal to the value of a joint policy computed by Θ_c. In our case, there exists uncertainty in the outcome of the actions, i.e., the transition probability of the Dec-MDP can be larger than zero. Myopic-greedy agents may synchronize their information from time to time. Thus, they may change the global goal state based on the information exchanged. When the agents do not communicate they do not have the chance to correct their policy with respect to another global goal state that may incur a lower cost based on their current local states (had them both known this information). Therefore, the value of a joint policy computed with the myopic-greedy approach is at least as large as the value of the joint policy computed without any communication. □

7.1 Experiments - Myopic-greedy Approach

We present empirical results obtained when the myopic-greedy approach was implemented for the Meeting under Uncertainty example (explained in Appendix B).[8] We showed [13] that exchanging the last observation guarantees optimality in a Dec-MDP-Com process with constant message cost. In the example we tested, the messages exchanged correspond to the agents' own observations, i.e., their location coordinates. In all the experiments run, we assumed that $P_1 = P_2$ and we refer to these uncertainties as P_u. The mechanism that is applied whenever the agents communicate at time t results in each agent adopting a local goal state, that is set at the location in the middle of the Manhattan path connecting the agents (the Manhattan distance between the agents is revealed at time t). We compare the joint utility attained by the system in the following four different scenarios:

[8] Some of the empirical results obtained in this section appeared in [12].

1. No-Communication — The meeting point is fixed at time t_0 and remains fixed along the simulation. It is located in the middle of the Manhattan path that connects between the agents, known at time t_0. Each agent follows its optimal policy of action without communication to this location.

2. Ideal — Assuming that C_Σ is zero, and that the agents communicate at every time step, this is the highest global utility that both agents can attain. Notice, though, that this is not the optimal solution we are looking for, because we do assume that communication is not free. Nevertheless, the difference in the utility obtained in these first two cases shed light on the trade-off that can be achieved by implementing non-free communication policies.

3. Communicate SubGoals — A heuristic solution to the problem, which assumes that the agents have a notion of sub-goals. They notify each other when these sub-goals are achieved, eventually leading the agents to meet.

4. Myopic-greedy Approach — Agents act myopically optimizing the choice of when to send a message, assuming no additional communication is possible. For each possible distance between the agents, a policy of communication is computed such that it stipulates when it is the best time to send that message. By iterating on this policy agents are able to communicate more than once and thus approximate the optimal solution to the decentralized control problem with direct communication. The agents continue moving until they meet.

The solution to the No-Communication case can be solved analytically for the Meeting under Uncertainty example, by computing the expected cost $\Theta_{nc}(d_1, d_2)$ incurred by two agents located at distances d_1 and d_2 respectively from the goal state at time t_0 (the complete mathematical solution appears in Appendix B).

In the Ideal case, a set of 1000 experiments was run in which the cost of communication was assumed to be zero. Agents communicate their locations at every time instance, and update the location of the meeting place accordingly. Agents move optimally to the last synchronized meeting location.

For the third case tested (Communicate SubGoals) a sub-goal was defined by the cells of the grid with distance equal to $p * d/2$ from the fixed current meeting point. p is a parameter of the problem that determines the radius of the circle that will be considered a sub-goal. Each time an agent reaches a cell inside the area defined as a sub-goal, it initiates exchange of information (therefore, p induces the communication strategy). d expresses the Manhattan distance between the two agents, this value is accurate only when the agents synchronize their knowledge. That is, at time t_0 the agents determine the first sub-goal as the area bounded by a radius of $p * d_0/2$ and, which center is located at $d_0/2$ from each one of the agents. Each time t that the agents synchronize their information through communication, a new sub-goal is determined at $p * d_t/2$. Figure 4 shows how new sub-goals are set when the agents transmit their actual location once they reached a sub-goal area. The meeting point is dynamically set at the center of the sub-goal area.

Experiments were run for the Communicate SubGoals case for different uncertainty values, values of the parameter p and costs of communication (for each case, 1000 experiments were run and averaged). These results show that agents can obtain higher utility by adjusting the meeting point dynamically rather than having set one fixed meeting point. Agents can synchronize their knowledge and thus they can set a new meeting location instead of acting as two independent MDPs that do not communicate and move towards a fixed meeting point (see Figure 5). Nevertheless, for certain values of p, the joint utility of the agents is actually smaller than the joint utility achieved in the No-Communication case (2 MDPs in the figure). This points out the need to empirically

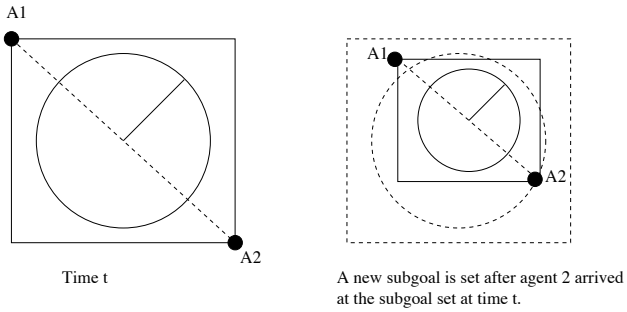

Figure 4: Goal Decomposition Into Sub-goal Areas.

tune up the parameters needed in the implemented heuristic, as opposed to a formal approach to approximating the solution to the problem as is shown in the Myopic-greedy case.

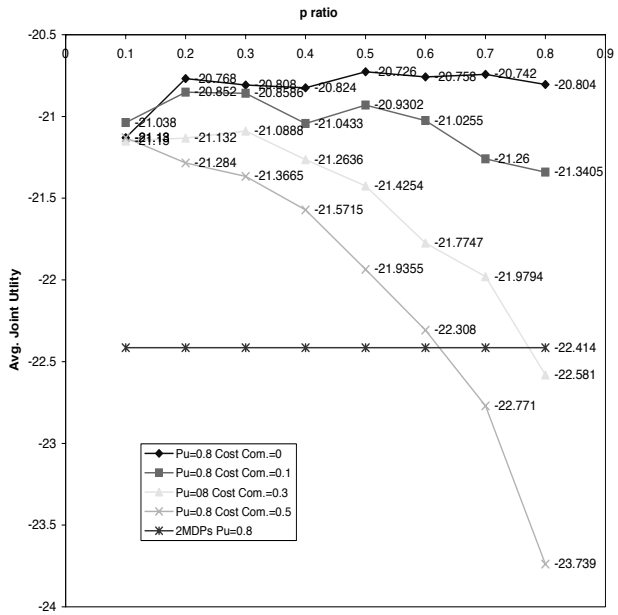

Figure 5: The Average Joint Utility Obtained When Sub-goals Are Communicated.

In the Myopic-greedy case, we design the agents to optimize the time when they should send a message, assuming that they can communicate only once. At the off-line planning stage, the agents compute their expected joint cost to meet for any possible state of the system (s^0) and time t (included in the local state s_i), $\Theta_c(s^0, s_i, \delta_1^{A*}, \delta_2^{A*})$. The global states revealed through communication correspond to the possible distances between the agents. Each time the agents get synchronized, the mechanism is applied assigning local goals and instructing the agents to follow the optimal local policies to achieve them. In the Meeting under Uncertainty scenario we study, Θ_c is the expected joint cost incurred by taking control actions during t time steps, communicating then at time $t+1$ if the agents have not met so far, and continuing with the optimal policy of control actions without communicating towards the goal state (the meeting location agreed upon at $t+1$) at an expected cost of $\Theta_{nc}(d_1, d_2)$ as computed for the No-Communication case. When the agents meet before the t time steps have elapsed, they only incur a cost for the time they act before they met.

At each time t, each one of the agents knows a meeting location, that is the goal location computed from the last exchange of information. Consequently, each agent moves optimally towards this goal state. In addition, the myopic-greedy policy of communication is found by computing the earliest time t, for which $\Theta_c(d_1 + d_2, s_1, \delta_1^{A*}, \delta_2^{A*}) < \Theta_{nc}(d_1, d_2)$, that is, what is the best time to communicate such that the expected cost to meet is the least. The myopic-greedy policy of communication is a vector that states the time to communicate for each possible distance between the agents.

We found the myopic-greedy communication policies for the Meeting under Uncertainty problem where P_u takes any of the following values: $\{0.2, 0.4, 0.6, 0.8\}$, the cost of taking a control action is $R_a = -1.0$ and the costs of communicating C_Σ tested were $\{-0.1, -1.0, -10.0\}$. The resulting policies of communication are presented in Appendix C. For the smallest cost tested, it is always beneficial to communicate rather early, no matter the uncertainty in the environment, and almost no matter what d_0 is (the differences in time are between 2 and 4). For larger costs of communication and a given P_u, the larger the distance between the agents, the later they will communicate (e.g., when $P_u = 0.4$, $C_\Sigma = -1$ and $d = 5$, agents should communicate at time 4, but if $C_\Sigma = -10$, they should communicate at time 9). For a given C_Σ, the larger the distance between the agents is, the later the agents will communicate (e.g., when $P_u = 0.4$, $C_\Sigma = -10$ and $d = 5$, agents should communicate at time 9, but if $d = 12$, they should communicate at time 16). The results from averaging over 1000 runs show that for a given cost C_Σ as long as P_u decreases (the agent is more uncertain about its actions' outcomes), the agents communicate more times.

In the 1000 experiments run, the agents exchange information about their actual locations at the best time that was myopically found for d_0 (known to both at time t_0). After they communicate, they know the actual distance d_t, between them. The agents follow the same myopic-greedy communication policy to find the next time when they should communicate if they did not meet already. This time is the best time found by the myopic-greedy algorithm given that the distance between the agents was d_t. Iteratively, the agents approximate the optimal solution to the decentralized control problem with direct communication by following their independent optimal policies of action, and the myopic-greedy policy for communication. Results obtained from averaging the global utility attained after 1000 experiments show that these myopic-greedy agents can perform better than agents who communicate sub-goals (that is a more efficient approach than no communicating at all). The results for $C_\Sigma = -0.1$ are presented in Tables 7 and 8. Additional results obtained for other costs of communication appear in Appendix D.

	Average Joint Utility			
P_u	No-Comm.	Ideal $C_\Sigma = 0$	SubGoals[9]	Myopic-Greedy
0.2	-104.925	-62.872	-64.7399	-63.76
0.4	-51.4522	-37.33	-38.172	-37.338
0.6	-33.4955	-26.444	-27.232	-26.666
0.8	-24.3202	-20.584	-20.852	-20.704

Table 7: $C_\Sigma = -0.10$,$R_a = -1.0$.

The Myopic-greedy approach attained utilities statistically significantly greater than those obtained by the heuristic case when $C_\Sigma = -0.1$.[10] Ideal always attained higher utilities than Myopic-greedy, but when $C_\Sigma = -0.1$ and $P_u = 0.4$ both values were not significantly different with prob-

[9]The results are presented for the best p, found empirically.

[10]Statistical significance has been established with t-test.

ability 98%. When $C_\Sigma = -1$ the utilities attained for the Myopic-greedy approach when $P_u < 0.8$ are significantly greater than the results obtained in the heuristic case and for $P_u = 0.8$, the heuristic case for the best p was found to be better than Myopic-greedy (Myopic-greedy obtained -21.3, and the SubGoals with $p = 0.1$ attained -21.05 (variance=2.18)). The utilities attained by the Myopic-greedy agents, when $C_\Sigma = -10$ and P_u in $\{0.2, 0.4\}$, were not significantly different from the SubGoals case for the best p with probabilities 61% and 82%, respectively. However, the heuristic case yielded smaller costs for the other values of $P_u = 0.6, 0.8$. One important point to notice is that these results consider the best p found for the heuristic, but in general a designer may not know this value. In all the settings tested, Myopic-greedy always attain utilities higher than those attained in the SubGoals case with the worst p.

	Average Communication Acts Performed			
P_u	No-Comm.	Ideal $C_\Sigma = 0$	SubGoals	Myopic-greedy
0.2	0	31.436	5.4	21.096
0.4	0	18.665	1	11.962
0.6	0	13.426	1	8.323
0.8	0	10.292	1	4.579

Table 8: $C_\Sigma = -0.10$,$R_a = -1.0$.

For the same parameters tested so far, experiments were run with two deadlines, T in $\{8, 15\}$ (i.e., if the agents did not meet by T, a penalty was given to the system). We tested how the policies of communication change as a result of these penalties. Examples of the communication policies computed when the cost of communication was set to -10 are presented in Appendix E. In general, the myopic-greedy policy found may instruct the agent not to communicate if $\Theta_{nc} < \Theta_c$, i.e., had the agents communicated, unnecessary information had been exchanged (because the agents are going to meet anyway incurring less cost if they do not communicate). On the other hand, this policy may instruct an agent not to communicate, if given a deadline, the agent is not going to be able to reach the goal (i.e., communication is not beneficial, it cannot help). In the first case, limiting the deadline to be earlier, results in policies of communication, which instruct the agents to communicate earlier, compared to the case when there are no deadlines (for large values of d_0 with low uncertainties P_u). When no deadlines are assumed, the agents may benefit from exchanging information later. When a short deadline is assumed, if the agents have the chance to meet without communication given a later deadline, they will need to communicate earlier if the time stipulated in the policy with no deadlines is larger than the deadline T. If the deadline T is large enough for these agents to meet, they do not need to communicate at all. For shorter d values if the policy with no deadline allows the agent to communicate at a time smaller than the deadline T, the same policy holds.

In the second case, the agents may not communicate if they may not meet at all by the stipulated deadline. The empirical results show that by extending the deadline T, agents benefit from communicating at a time that is later than the time found by the myopic-greedy policy when no deadlines were assumed. Since, in this case there is a chance of not meeting at all, agents need to wait more time until it becomes beneficial to communicate.

8 An Optimal Communication Policy

As in the previous section, we assume that a set of individual behaviors are provided for each global states. In this section, we characterize the set of *monotonic* goal-oriented Dec-MDPs for which we provide an algorithm that finds the optimal policy of communication. First, we define the *rank* of a global state to be a function $\rho : S \rightarrow \mathcal{N}$ such that when s is a global goal state ($s \in G$), $\rho(s) = 0$. For example, the rank can express the expected cost of the optimal policy to reach the global goal state.

Definition 12 (Monotonic GO-Dec-MDPs) *A goal-oriented Dec-MDP is* monotonic *with respect to a given mechanism if there exists a ranking function ρ such that for all global states s and all global states s' reachable from s ($s' \neq s$), following the joint policy induced by the mechanism, $\rho(s') < \rho(s)$.*

Although the transitions are between states with non-increasing rank, the uncertainty about the outcomes of the agents' actions does exist, i.e., the agents' actions can fail. The algorithm presented in this section finds the optimal policy of communication at the meta-level of control. This policy instructs the agents to synchronize the information in their partial views at the most beneficial time.

We assume a goal-oriented Dec-MDP with independent transitions and observations and a finite horizon T (time is discrete). Each agent i can choose an action a_i^j, $1 \leq j \leq m$, from its set of actions A_i. The notation we use in the algorithm is as follows: a_i^{j*} denotes the optimal action that agent i chooses given δ_i^A, i.e., the provided mapping from global states to individual behaviors (assigned to i by the mechanism). After successfully performing the optimal control action a_i^{j*}, agent i moves to a state s_i' that is denoted by $a_i^{j*}(s_i, 1)$. $a_i^{j*}(s_i, 0)$ represents the resulting state when agent i fails to perform action a_i^{j*}.[11] $EU^i(s, s_i, t)$ denotes the expected joint utility of the multi-agent system, computed by agent i at time t, when the synchronized global-state is s and agent i's partial view is s_i. The set of global-states are ordered by the rank assumed for monotonic Dec-MDPs. The states with rank k are represented by S^k (K is the largest rank that a state can have). For example, if the agents' goal is to meet, then the state of the Dec-MDP-Com may be given by the Manhattan distance between the agents, and the goal state is reached when this distance is zero. The rank is given, then, by the possible Manhattan distances between the agents given a 2D grid. The algorithm for computing the optimal policy of communication is based on backward induction. It is shown in Figure 6. EU_C and EU_{NC} are two temporary variables that denote the expected joint utility when agent i decides to communicate or when it does not. *Penalty* is the reward obtained when the agents do not achieve their goal by the time limit of the problem. $\Phi_C(P_1, P_2, R, C_\Sigma, s^0, s_i, t)$ computes the expected joint utility when agent i communicates its partial view s_i at time $t+1$, and the current synchronized global-state is s^0. This function computes the possible synchronized global-states in which the system could be (given that agent i communicates), the expected costs incurred to arrive at these states, and the joint expected utility of these new states. Notice that since we deal with monotonic Dec-MDPs, the ranks of these new states are at most as high as the rank of the last synchronized state.

Theorem 2 *OptCom computes the optimal communication policy for a given monotonic goal-oriented Dec-MDP-Com with independent transitions and observations and a given mechanism.*

[11] In general, an action a_i may have multiple outcomes. Then, the algorithm presented in this section can be adapted to include all the possible next states reachable when executing a_i with the corresponding probability.

function OptCom(DCM, δ_i^A, Dec-MDP-Com)
returns the optimal communication policy $Policy(s, s_i, t)$,
where s is the last synchronized state and s_i is the current partial view and
t is the time when the decision to communicate at $t{+}1$ is made.
The value of zero means that the agent will not communicate at $t{+}1$.
inputs: DCM is the mechanism for communication, that induces an option opt_i
for each global state s (i.e., a policy of actions and communications).
δ_i^A is the policy of domain actions.
Dec-MDP-Com=$< S, A_1, A_2, \Sigma, C_\Sigma, P, R, T >$

For each state $s \in S^k$ (for $k \leftarrow 0$ to K)
For time $t \leftarrow T{-}1$ **to** 0
For each $s_i \in S_i$
if $(k = 0)$ and $(t{=}T{-}1)$ **then** /*agents reached the global goal state*/
$Policy(s, s_i, t) \leftarrow 0$
$EU^i(s, s_i, t) \leftarrow 0$
else if $(t = T{-}1)$ **then** /*time is over*/
$Policy(s, s_i, t) \leftarrow 0$
$EU^i(s, s_i, t) \leftarrow Penalty$/*agents did not reach the global goal state*/
else if $(t = 0)$ **then** /*agents are synchronized*/
$Policy(s, s_i, t) \leftarrow 0$
$EU^i(s, s_i, t) \leftarrow ComputeEU_{NC}(\delta_i^A, s, s_i, R, t)$
else
$EU_{NC} \leftarrow ComputeEU_{NC}(\delta_i^A, s, s_i, R, t)$
$EU_C \leftarrow \Phi_C(P, R, C_\Sigma, s^0, s_i, t)$
if $(EU_{NC} > EU_C)$ **then**
$Policy(s, s_i, t) \leftarrow 0$
$EU^i(s, s_i, t) \leftarrow EU_{NC}$
else /*communicate at t+1*/
$Policy(s, s_i, t) \leftarrow t + 1$
$EU^i(s, s_i, t) \leftarrow EU_C$
return Policy

function $ComputeEU_{NC}(\delta_i^A, s, s_i, R, t)$
returns the expected joint utility given that the agent does not communicate.
inputs: δ_i^A, the given local policy of action for agent i.
s, the last Dec-MDP-Com synchronized state.
s_i, the current partial view of agent i
R, the Dec-MDP-Com reward function.
t, the current time.

$EU_{Succ} \leftarrow EU^i(s, a_i^{j*}(s_i, 1), t + 1)$
$EU_{Fail} \leftarrow EU^i(s, a_i^{j*}(s_i, 0), t + 1)$
return $(1{-}P_i)(R + EU_{Fail}){+}P_i(R + EU_{Succ})$

Figure 6: The OptCom Algorithm for Monotonic Dec-MDP-Com With Independent Transitions and Observations.

Proof. The correctness proof of the algorithm is given by induction. The induction is both on the time t that elapses and on the rank k of the global states.

Basis: Based on the Dec-MDP-Com model, the agents *decide* to communicate at time t, but the actual communication act occurs at time $t+1$. If the time limit is T then it is not beneficial to decide to communicate at time $T-1$. If the synchronized state that is known by all the agents is a global goal state ($S^k = S^0$) and time is $T-1$ then all agents are aware of having achieved this global goal state. Therefore, it is optimal not to communicate then. If time is $t = T-1$, and no global goal state is reached, then a penalty will be awarded.

We assume that the algorithm OptCom computes the optimal time to communicate for any state $s \in S^k$ for any $0 \leq k \leq K'$ (for some $K' < K$, K is the largest rank of a global state), and for any time $0 \leq t < T$.

By induction on k and t, we prove that the *OptCom* algorithm presented in Figure 6 finds the optimal time to communicate for any state $s \in S^{K'+1}$ and time t. Following the algorithm, when the agent decides whether to communicate or not in state $s \in S^{K'+1}$, it compares its utility when it does not communicate (EU_{NC}) with its utility when it does communicate (EU_C). If the agent does not communicate, then it chooses the optimal control action a_i^{j*} based on the δ_i^A (i.e., the given individual behavior). The outcome of this action is given by the transition probability P_i, i.e., with probability P_i agent i moves to state $s_i' = a_i^{j*}(s_i, 1)$ and with probability $1 - P_i$ it moves to a state $s_i'' = a_i^{j*}(s_i, 0)$. Therefore, $EU_{NC} = (1 - P_i)EU^i(s, s_i'', t+1) + P_i EU^i(s, s_i', t+1)$. In general, an action could have multiple outcomes. In any case, due to the monotonicity of the process, we know that $\rho(s_i'') < \rho(s_i)$ and $\rho(s_i') < \rho(s_i)$, therefore $s_i'' \wedge s_i' \in S^k$ for some $k < K' + 1$. This is true for any possible next state. Based on the assumption of the induction, these values are optimal and have taken into account the optimal decision when to communicate.

The expected utility if the agent decides to communicate is $EU_C = EU(s''', 0)$. A communication act always succeeds because we assume messages are reliable. Time becomes 0 because after communicating the agents become synchronized (thus they are reset). Since the Dec-MDP is monotonic, $\rho(s''') < \rho(s)$. Therefore, the expected utility of this state at time 0 is known and has been computed optimally. Therefore, the algorithm presented finds the optimal policy of communication given a monotonic Dec-MDP with independent transitions and observations. □

Lemma 7 *Deciding a monotonic Dec-MDP-Com with OptCom is in P.*

Proof. Each agent implements a local policy of action when the mechanism is applied. The complexity of finding the optimal communication policy by running the *OptCom* algorithm is the same as dynamic programming, therefore computing the resulting policy of communication is also in P. □

8.1 Experiments - Monotonic Goal-oriented Dec-MDPs

The performance of the *OptCom* algorithm is exemplified on the Meeting under Uncertainty example presented in Appendix B. We compare here the *OptCom* results to the No-Communication, Ideal and Myopic-greedy cases, described in Section 7.1.

The results from experimenting with different communication costs (and averaging over 1000 runs) appear in Table 9 and in Appendix F.[12] The cost of taking a moving action was set to -1.0. In the Ideal case, C_Σ is zero. Note that the setup of these experiments differs from the setting we

[12] Although we do have a program that can precisely compute the solution for the No-Communication case, the results presented were obtained from averaging over 1000 empirical tests, which result less time consuming than the analytical solution for the finite-horizon case.

studied in Section 7.1 because here the agents may not meet by the time limit and consequently they are penalized. In the former experiments (without deadlines) the time limit was large enough so that the agents always met.

P	Average Joint Utility			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	-71.138	-62.968	-62.834	-63.226
0.4	-42.112	-37.372	-37.778	-37.734
0.6	-29.078	-26.518	-26.782	-26.642
0.8	-22.344	-20.52	-20.714	-20.574

Table 9: $C_\Sigma = -0.10$.

The results obtained by *OptCom* in Table 9 for $P = 0.2$ are not significantly different neither from Ideal (with probability 65%) nor from Myopic-greedy (with probability 48%). Table 10 shows the average number of messages exchanged in each one of the tested cases when the cost of communication was -0.1. Since the Meeting under Uncertainty example is indeed a monotonic Dec-MDP, the results obtained by the backward-induction algorithm (*OptCom*) are optimal. Even in this simple example, we notice that when $C_\Sigma = -0.1$ and P is either 0.6 or 0.8, the results in the backward-induction column are significantly greater than in the greedy column. For smaller uncertainties (0.2 and 0.4) the results in this case are not significantly different with probabilities 48% and 88% respectively. The results obtained for increasing costs of communication (see Appendix F) lead to significantly different utility values between the backward-induction and the greedy algorithm. When the price of communication is higher, it is more important to decide optimally when to communicate. The greedy implementation does not compute the optimal policy of communication, therefore, it cannot consider how crucial these decisions are. Therefore, the greedy agents do not communicate as much as necessary to find the optimal joint solution (see Table 10).

P	Average Communication Acts Performed			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	0	31.484	20.778	30.613
0.4	0	18.686	12.171	17.867
0.6	0	13.259	8.252	12.321
0.8	0	10.26	4.588	9.287

Table 10: $C_\Sigma = -0.10$.

9 Beyond Goal-oriented Behavior

The models presented in this paper assumed that actions incur a negative cost and that a positive joint reward (JR) is awarded to the system at time T (the finite horizon) if the system is at a global goal state. That is, no positive intermediate rewards can be obtained before the horizon is reached. We can think of a mechanism as an approach to approximate the optimal joint solutions of a decentralized process by contemplating the decentralized problem as a Dec-SMDP-Com (not necessarily goal-oriented as defined in Definition 6). We propose the mechanism approach as a general approximation method to solve decentralized processes, by computing temporal abstracted actions (either goal-oriented or not). Agents controlling such processes will follow these local

behaviors between communications. The only difference between a Dec-SMDP-Com over an underlying Dec-MDP-Com and the goal-oriented version relies in the definition of the reward function: $R^N(s, t, opt_1, opt_2, s', t{+}N)$. In the goal oriented case this reward was given by $2Cost(a)(t{+}N){+}C_\Sigma$ since each primitive domain action taken when following the options incurs some cost $Cost(a)$ and JR(s) if s is a global goal state and $t{+}N$ is T. In the general case, primitive actions may yield some positive reward given by $R_i(s_i, a_i, s'_i)$ which should be considered in the computation of R^N. Moreover, if the options are not totally independent of each other (e.g., they can be $\Delta - dependent$ (see Definition 11)) then the computation of R^N will take these additional dependent values into account. Designing algorithms for computing mechanisms for communication for the general Dec-SMDP-Com case remains for future work.

10 Conclusions

Solving decentralized control problems optimally requires the optimization of a global reward function computed over global states with stochastic transitions. Our previous work [13] has analyzed the complexity of solving optimally certain classes of decentralized problems when direct or indirect communication may occur. We showed that this complexity ranges between NEXP and Polynomial. In particular, we studied goal-oriented Dec-MDPs when no information sharing is allowed. We proved that solving optimally these cases is equivalent to solving a combination of single agent MDP problems. This paper focuses on goal-oriented Dec-MDPs with direct communication and many global goal states. This is a particular hard class of problems because due to the possible information exchange, the global goal-oriented behavior may not be decomposed into individual goal-oriented processes. Therefore, in this paper, we approximate this optimization problem by decomposing the global reward function into local and temporal problems. Communication between the decision makers serves as a synchronization point where local information is exchanged in order to assign to each controller an individual behavior. In real and practical applications this communication has a cost associated with the complexity of computing the messages, the bandwidth of the communication or the risk of revealing information to competitive parties operating in the same environment. We presented the mechanism for communication approach, which decomposes the global problem into temporal and individual behaviors and combines these with communication acts to overcome the lack of global information from time to time. All the complexity results known for Dec-MDPs are summarized in Figure 7.

We, first, formalized our mechanisms for communication for a decentralized problem with goal-oriented behavior approach as a decentralized semi-Markov process with communication (GO-Dec-SMDP-Com). We proved that solving optimally such problem with temporal abstracted actions is equivalent to solving optimally a multi-agent MDP (MMDP) known to be P-complete. However, in our case, the input of the GO-Dec-SMDP-Com problem is now double-exponential in the number of actions. We presented the multi-step backup policy iteration algorithm adapted to the decentralized case, which solves optimally the GO-Dec-SMDP-Com problem. It is based on heuristic search and converges to the optimal solution. This algorithm provides us with the optimal mechanism for communication, i.e., the optimal joint solution over temporal abstracted actions. This is the first algorithm to tackle a general GO-Dec-MDP-Com problem.

Then, we presented a refinement of this algorithm to provide a practical approximation. The idea behind this mechanism is that agents can adopt local goals. That is, the optimal algorithm that we presented assigns the best pair of local goals found for each global state (which is fully observable since communication occurs at these states). The algorithm also finds the optimal period of time to work towards these goals. The policy of communication instructs the agent to exchange

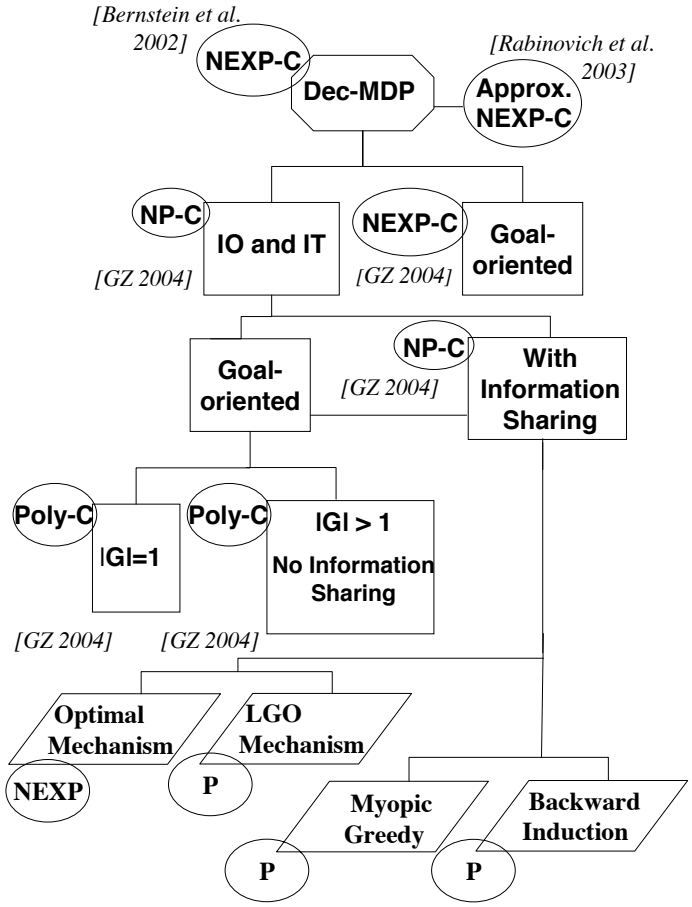

Figure 7: The Complexity of Solving Dec-MDPs.

information when this period of time is over, even though none of them may have reached their assigned local goals. We prove that this algorithm takes only polynomial time.

The paper ends by presenting a simpler approximation method. It assumes that a certain mechanism is given, i.e., human-designed knowledge is combined into the model to provide agents with individual policies of actions (not including communication acts). We showed two approaches to compute a policy of communication, given a mechanism. First, we showed a greedy-approach that computes the best time to communicate assuming there is only one opportunity for exchanging information. Then, we presented an optimal algorithm that computes when to communicate assuming that the decentralized process is monotonic. We support our approaches with empirical results. Future work will look into generalizing the mechanism technique to situations that may not be factored as assumed in this paper. Such a generalization may require a non-trivial decomposition of the set of global states, the system transition function and the system reward function.

11 Acknowledgments

This work was supported in part by the National Science Foundation under grants IIS-0219606, by the Air Force Office of Scientific Research under grant F49620-03-1-0090 and by NASA under grant NCC 2-1311. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the NSF, AFOSR or NASA.

References

[1] R. J. Aumann and S. Hart, editors. *Handbook of Game Theory with Economic Applications*, volume 2. Elsevier, North Holland, 1994.

[2] Tucker Balch and Ronald C. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1:1—25, 1994.

[3] Raphen Becker, Shlomo Zilberstein, Victor Lesser, and Claudia V. Goldman. Solving transition independent decentralized MDPs. *Journal of Artificial Intelligence Research*, 2004, To appear.

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

[5] Andreas Blume and Joel Sobel. Communication-proof equilibria in cheap-talk games. *Journal of Economic Theory*, 65:359—382, 1995.

[6] Craig Boutilier. Sequential optimality and coordination in multiagent systems. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 478–485, Stockholm, Sweden, 1999.

[7] Will Briggs and Diane Cook. Flexible social laws. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Quebec, 1995.

[8] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17—34, 1971.

[9] David Fitoussi and Moshe Tennenholtz. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119, 2000.

[10] Mohammad Ghavamzadeh and Sridhar Mahadevan. Learning to act and communicate in cooperative multiagent systems using hierarchical reinforcement learning. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1114–1121, New York City, NY, 2004.

[11] Claudia V. Goldman and Jeffrey S. Rosenschein. Emergent coordination through the use of cooperative state-changing rules. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 408–413, Seattle, Washington, 1994.

[12] 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 Multi-Agent Systems*, pages 137—144, Melbourne, Australia, 2003.

[13] Claudia V. Goldman and Shlomo Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research*, 2004, to appear.

[14] Carlos Guestrin and Geoffrey Gordon. Distributed planning in hierarchical factored MDPs. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pages 197–206, Edmonton, Canada, 2002.

[15] Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems (NIPS-14)*, pages 1523–1530, Vancouver, British Columbia, 2001.

[16] Eric A. Hansen. Markov decision processes with observation costs. Technical Report 97-01, University of Massachusetts at Amherst, Computer Science, 1997.

[17] Eric A. Hansen and Rong Zhou. Synthesis of hierarchical finite-state controllers for POMDPs. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS-03)*, Trento, Italy, June 2003.

[18] Noa Kfir-Dahav, Dov Monderer, and Moshe Tenenholtz. Mechanism design for resource-bounded agents. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, 2000.

[19] R. Duncan Luce and Howard Raiffa. *Games and Decisions*. John Wiley and Sons, Inc., 1957.

[20] John Moore. Implementation, contracts, and renegotiation in environments with complete information. In Jean-Jacques Laffont, editor, *Advances in economic theory Sixth World Congress Volume 1*, pages 182–282. Cambridge University Press, 1992.

[21] R. Nair, M. Tambe, M. Roth, and M. Yokoo. Communication for improving policy computation in distributed POMDPs. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1098–1105, New York City, NY, 2004.

[22] 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 Eighteenth International Joint Conference on Artificial Intelligence*, pages 705–711, Acapulco, Mexico, 2003.

[23] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Proceedings of the Thirty First Annual ACM Symposium in Theory of Computing (STOC)*, 1999.

[24] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[25] Christos H. Papadimitriou and John Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[26] Leonid Peshkin, Kee-Eung Kim, Nicolas Meuleau, and Leslie Pack Kaelbling. Learning to cooperate via policy search. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI00)*, pages 489–496, Stanford, CA, 2000.

[27] Martin L. Puterman. *Markov Decision Processes - Discrete Stochastic Dynamic Programming*. Wiley & Sons, Inc., New York, 1994.

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

[29] Khashayar Rohanimanesh and Sridhar Mahadevan. Decision-theoretic planning with concurrent temporally extended actions. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, 2001.

[30] Jeffrey S. Rosenschein and Gilad Zlotkin. *Rules of Encounter*. MIT Press, Cambridge, Massachusetts, 1994.

[31] Jeff Schneider, Weng-Keen Wong, Andrew Moore, and Martin Riedmiller. Distributed value functions. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 371—378, 1999.

[32] Yoav Shoham and Moshe Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artificial Intelligence*, 73:231–252, 1995.

[33] Yoav Shoham and Moshe Tennenholtz. On the emergence of social conventions: modeling, analysis and simulations. *Artificial Intelligence*, 94, 1997.

[34] Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.

[35] Adam Walker and Michael Wooldridge. Understanding the emergence of conventions in multi-agent systems. In *Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95*, 1995.

[36] Michael Wellman. Multiagent systems. In R. Wilson and F. Kiel, editors, *MIT Encyclopedia of Cognitive Sciences*, pages 573—574. MIT Press, 1999.

[37] Karl Wärneryd. Cheap talk, coordination, and evolutionary stability. *Games and Economic Behavior*, 5:532—546, 1993.

[38] David H. Wolpert, Kevin R. Wheeler, and Kagan Tumer. General principles of learning-based multi-agent systems. In *Proceedings of the Third International Conference on Autonomous Agents (Agents '99)*, pages 77—83, Seattle, Washington, 1999.

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

A Mechanism Design - Related Work

Mechanism design was originally studied in Game Theory to design games that yield outcomes with certain characteristics. Later, research in Computer Science has looked at adapting this approach to achieve social coordination and optimization of social welfare in distributed systems. We are interested in mechanisms that result in near-term behaviors that produce good approximations to the optimal control of a decentralized cooperative system. Here, we review the classic approach to mechanism design, followed by studies done for computational systems.

A.1 The Economic Approach

Mechanism design or implementation theory is studied in Game Theory [24] in order to find rules for a game with certain characteristics. The players in this game, have each a preference function over the outcomes of the game. Given a choice rule from profiles of preferences to a subset of feasible outcomes, the question is whether a game can implement this choice rule in a such a way that a certain solution concept is attained (e.g., the Nash equilibrium is reached). The players are self-interested and therefore information about their own preferences is kept private. A designer of the game looks for a mechanism that will produce the desired outcome (e.g., a Nash equilibrium) when the players reveal some part of their information as input to the designer. Notice that following our approach, each time that the agents apply the mechanism for decentralized control DCM, they are faced with a behavior to follow that is only a temporary step in the complete solution to the decentralized problem. In the economic approach the mechanism itself solves the problem.

An algorithmic view to mechanism design is found in [23]. The mechanism designer sets the algorithm for interaction among the agents and a payment structure that motivates the agents to participate in the interaction. This literature is concerned with agents that are self-interested and may hold privately known information about their preferences. Thus, the main question handled by a designer of a mechanism is to combine the private preferences of the players into an outcome state that corresponds to the "social choice". Since Nisan and Ronen took an algorithmic approach to mechanism design, they were interested in *poly-time computable mechanisms*, i.e., mechanisms whose output and payment functions are computable in polynomial time.

Following this economic approach, agents are self-interested. Therefore, they are not willing to reveal private information to prevent competitors from taking advantage of their actions. Therefore, an important notion in this approach is whether the mechanism is *truthfully-implementable*, i.e., whether the implementation will induce the agents to report their true types and preferences. An example of a truthful implementable mechanism is the known Clarke mechanism [8], where a set of compensation rules is given as an incentive structure, leading the agents to reveal their true preferences as their optimal strategies. In cooperative decentralized systems, it is clear that any mechanism for communication is truthfully-implementable. However, an interesting feature of these mechanisms is that the whole system may benefit if one of the agents does not send its actual observation, but a function of it. An agent may take an action as a result of receiving a message, and move eventually to a new state. Observing this new state may lead the agent to take an additional action that was not the original aim of the message sent, but it is the result of an effect the sender of the message had on the resulting state. So, even though agents are truthful they may benefit by not changing exactly what they observe. Notice that the contents of the messages are not set by the mechanism. The mechanism assumes that the agents are programmed, knowing which are the messages they can exchange. Another interesting research area is to study the design of languages of communications for decentralized control.

Based on Wellman's definition of multi-agent systems [36], the role of a mechanism in a cooperative system (with global objectives) is "to coordinate local decisions and disseminate local information in order to promote these global objectives." We study mechanisms that lead each agent to face possibly new local policies of behavior, which are simpler to compute. The mechanism also comprises a policy of communication that enables the agents to coordinate their local information. Achieving the global objectives of the system results from interleaving near-term local behaviors with exchanging information when the mechanism is applied.

A.2 Social Laws

The areas related to mechanism design in distributed artificial intelligence are social laws and negotiation mechanisms. Social laws were defined as mechanisms of coordination. Two approaches were studied: Shoham and Tennenholtz [32] define social laws as constraints on the agents' actions. Goldman and Rosenschein [11] define social laws as extensions to the agents' local plans of actions.

Shoham and Tennenholtz study social laws as mechanisms for coordination that will induce agents to avoid conflicts between their actions. A social law is a predicate over a local state prohibiting some of the actions that an agent is capable of performing. Once the social law is imposed on a multi-agent scenario, its effects are transparent to the agents. A modified multi-agent system is created in which only the permitted actions and the corresponding transitions are allowed. When designing mechanisms for communication, the agents are actively applying the convention at each global state they reach. In our case, agents' plans of actions can be affected by the messages received on-line. In the social-laws analysis, once the law is imposed, the agents find a plan of action that is not going to change anymore because of the law. More recently [33], a *rational social-law* was defined assuming that the agents play a game g, and that a social law sl induces a sub-game g_{sl} of g that includes only the actions that are not prohibited by sl.[13] In a game theoretical sense, rational agents are captured as utility maximizers. As such, the solutions that will be preferred by the agents in such settings will be either the maximin strategies, Nash equilibrium, or Pareto Optimal strategies(more details about these solutions can be found in [33]).

Goldman and Rosenschein [11] define social laws as extensions to the agents' local plans of actions. Social laws are intended to transform the global world state for the benefit of the whole system. Each agent is assigned a level of cooperation value that determines how much effort the agent will invest in extending its own plan in order to follow the law (this effort could be null). The social law in this case was studied as a simpler (less complex) means to reach coordination rather than computing the optimal multi-agent joint plan. Mechanisms for communication are also introduced to reduce the complexity of solving the complete decentralized control problem with communication. In our case, the information exchanged by the agents together with the mechanisms imply two local individual behaviors. Social laws are more strict in the sense that either they prohibit the execution of certain actions, or demand the execution of longer plans in order to follow the social rule.

The implementation of social laws in the two approaches aforementioned is aimed at improving the coordination level of the multi-agent system. Flexible social laws were studied by Briggs and Cook [7]. Agents are allowed to choose from laws with various levels of strictness starting with the most strict and moving to more lenient laws when they cannot succeed in finding a plan. These social laws follow the approach taken by Shoham and Tennenholtz as restrictions to the agents' actions, reducing the chance of interaction between the agents. Mechanisms for communication are not intended to avoid interactions, the assumption is that for certain costs of communication,

[13] A *social convention* [33] is a social law that restricts the agents' behaviors to one particular strategy.

the exchange of information is indeed beneficial. The mechanisms for communication will induce the agents to better coordinate and thus attain increased joint utility. The works explained so far assume that the social laws are designed off-line. Another line of research study the emergence of these conventions [35, 33].

In the cases described above, the agents are motivated to follow the social laws implicitly because the agents comprise a cooperative system. Therefore, it is in their benefit to implement the law. In the next section, we describe work done on self-interested agents who need to be motivated to follow a social mechanism.

A.3 Negotiation Mechanisms

Self-interested agents need to be motivated to follow a certain interaction mechanism (similarly to the economic approach). Negotiation mechanisms were developed by Zlotkin and Rosenschein [30]. They suggested a negotiation protocol over possible joint deals. This protocol can either end in reaching the conflict deal (i.e., no cooperation is beneficial and each agent ends up performing its initially locally assigned deals) or the negotiation ends with an agreement that corresponds to some division of the set of both agents' deals. In this process the goal-oriented agents are interested in achieving their *pre-set local* goals at the minimum cost through possible cooperation and resolution of conflicts if they exist. The goal of this line of research is to find distributed consensus mechanisms such that agents that follow simple and stable strategies will obtain efficient (Pareto Optimal [19][14]) outcomes. They studied monotonic negotiation mechanisms that are guaranteed to converge to a deal [30]. They also studied incentive-compatible mechanisms, and show that whether an agent will benefit from lying or not depends on the domain characteristics (e.g., concave/sub-additive/modular Task Oriented Domains). For example, it was proved that no lie (i.e., hidden, phantom or decoy) is beneficial in any encounter of two agents in concave[15] Task Oriented Domains with any optimal negotiation mechanisms over all-or-nothing deals.

This work does not deal with sequential decision making. The agreements could be over a set of many tasks that eventually will be performed in a sequence. However, the negotiation process is over all the possible deals as one decision. All the possible deals are already known when the negotiation mechanism is applied. In our case, we are interested in applying the communication convention as part of the control process in order to optimally behave and communicate.

A.4 Evaluation Criteria

Mechanism designers in Economics (e.g., [20, 24]) are interested in *stable* mechanisms so that they cannot be manipulated by self-interested agents. Thus, mechanisms are sought to implement a solution concept such as dominant strategies or Nash equilibrium for example. Economists are also interested in *truthfully-implementable* mechanisms that will induce the players to report their true preferences to the system designer. A good mechanism should have the following characteristics: *strategy-proof* , *efficient* and *budget-balanced* [18]. A mechanism is strategy-proof if the agents are motivated to participate in it and will reveal their true preferences. A mechanism is efficient if its output state maximizes the utility of the system (i.e., the social-welfare is optimized taking into account the individual selfish utilities of the agents). A mechanism is budget-balanced if the total monetary transfer from the agents to the center (the system designer) is non-negative. Kfir-Dahav

[14] A deal is Pareto optimal if it cannot be improved for one agent without decreasing the utility of another agent from the same deal.

[15] A Task Oriented Domain is concave if for all finite sets of tasks $X \subseteq Y, Z \subseteq T, c(Y \bigcup Z) - c(Y) \leq c(X \bigcup Z) - c(X)$. c is the cost function.

et al. show [18] that the procedures of the Clarke mechanism used to optimize the social welfare are NP-hard and they suggest a heuristic that will maintain the strategy-proof and budget-balanced features at the expense of social-welfare efficiency.

According to the economic approach, an optimal social-law is one that attains maximal utility at the system level. Based on [9], social laws should impose the minimal possible number of constraints on the agent's actions and they should be easy to implement. The features discussed in [11] for cooperative state-changing rules include the following. A rule is: 1) *guaranteed* if it will not increase global work with certainty. 2) *reversible* if its effects can be undone. 3) *redundant* if performing the extra work will cause the agent to remain in the same state. 4) *resource-dependent* if following the rule implies the use of consumable resources. 5) *state-dependent* if the rule can be applied only in a certain state.

Negotiation mechanisms [30] were evaluated based on the following criteria: 1) *Symmetric distribution*, i.e., no agent is to have a special role in the negotiation mechanism. In our case, since the agents are cooperative we do not risk having manipulating malevolent agents. However, the designer may want a more capable agent to have more influence on another agent when it communicates (e.g., by sending an instruction message). 2) *Efficiency*, i.e., the solution arrived at through negotiation should be efficient (e.g., satisfy the criterion of Pareto Optimality). 3) The strategies should be *stable* (e.g., strict Nash equilibrium where no single agent can benefit by changing its strategy, though a group might). 4) *Simplicity* , i.e., a good negotiation mechanism should have low computational complexity. The mechanism for communication is a means to interpret messages received and translate them into near-term problems that can be solved locally and optimally. Notice that the communication in our case is not in the form of KQML commands where the reaction to a message received is the clear and expected response action to the performative command.

We summarize the differences among these approaches in Figure 8:

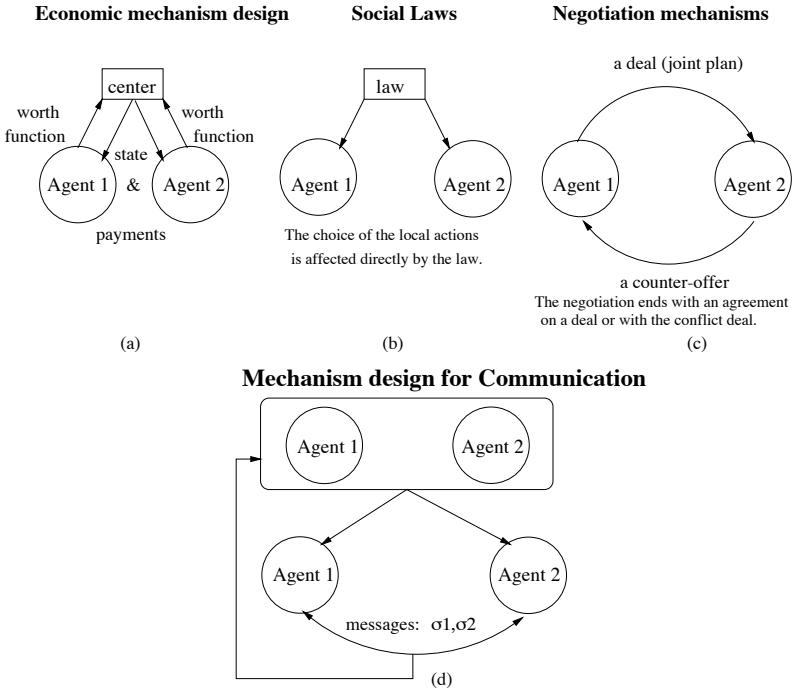

Figure 8: A Comparison Between Mechanism Design Approaches.

Cases (a) and (d) — Economic agents are self-interested. The agents send their state-worth

functions to the center who outputs a state and a payment structure. The agents' input to the center states how much they are willing to pay for a certain output-state (i.e., how desirable that state is with regard to the goal the agent intends to achieve). A mechanism sets the rules for a game; in our approach, a mechanism decomposes a global problem into two temporary local behaviors. The agents may communicate based on their policy of communication, in which case, the decomposition into two individual behaviors is applied again.

Cases (b)(Goldman and Rosenschein approach) and (d) — Although the mechanism induces the agents to change a global state, by adopting a temporary local subgoal, the parameters of the law are based on the features of the agent's own observations, and not as a result of another agent sending its observations.

Cases (b)(Shoham and Tennenholtz approach) and (d) — This approach imposes a social law on the multi-agent system, which is then transformed into another system including only those actions that are allowed by the law. From then on, the agents plan as usual. In our case the mechanism is actively applied by the agents while they compute their optimal policies of action and communication.

Cases (c) and (d) — The mechanism is applied once and it provides the agents with the solution of a joint plan. Agents are self-interested and they negotiate assuming that both agents have fully-observable information (although it may not be reliable).

B Meeting under Uncertainty Example

The testbed we consider is a sample problem of a GO-Dec-MDP-Com involving two agents that have to meet at some location as early as possible. The environment is represented by a 2D grid with discrete locations. In this example, any global state in which the agents are at the same location can be considered a global goal state. The observations and the transitions are independent. The set of control actions includes moving North, South, East and West, and staying at the same location. The agents can initiate direct communication. Each agent's partial view (which is locally fully-observable) corresponds to the agent's location coordinates. There is uncertainty regarding the outcomes of the agents' actions. That is, with probability P_i, agent i arrives at the desired location after having taken a move action, but with probability $1 - P_i$ the agent remains at the same location. Due to this uncertainty in the effects of the agents' actions, it is not clear that setting a predetermined meeting point is the best strategy for designing these agents. Agents may be able to meet faster if they change their meeting place after realizing their actual locations. This can be achieved by exchanging information on the locations of the agents, that otherwise are not observable.

Adding direct communication to this setting allows the agents to attain full observability of the global state of the system. Each time the agents exchange information, a mechanism is applied to the decentralized process resulting in two single-agent goal-oriented behaviors. We have implemented the locally goal-oriented mechanism that assigns a single local goal to each agent at each synchronized state, i.e., it instructs each agent to reach the location in the middle of the shortest Manhattan path between the agents' locations (this distance is revealed when information was exchanged). This mechanism has low computational complexity (i.e., each agent computes the location in the middle of the Manhattan path with the information acquired by communication in constant time and the policy of communication can be found in polynomial time). Section 7 presents a myopic-greedy policy of communication for which this mechanism is complete, eventually the agents achieve their global goal and meet.

Intuitively, it is desirable for a mechanism to set a meeting place in the middle of the shortest

Manhattan path that connects the two agents because in the absence of communication, the cost to meet at that point is minimal. This can be shown by computing the joint expected time to meet, Θ_{nc}, for any pair of possible distances between the two agents and any location in the grid, when no communication is possible. The minimal value is attained when these distances are equal. To simplify the exposition, we use a function that takes advantage of the specific characteristics of the example. In Section 7, we return to the notation of the general case. The notation is as follows: agent 1 is at distance d_1 from the meeting location, agent 2 is at distance d_2 from that location, the system incurs a cost of one at each time period if the agents have not met yet. and P is the transition probability of the Dec-MDP-Com. If both agents are at the meeting location, the joint expected time to meet is zero, $\Theta_{nc}(0,0) = 0$. If only agent 2 is at the meeting location, but agent 1 has not reached that location yet, then the joint expected time to meet is given by

$$\Theta_{nc}(d_1, 0) = P_1(-1 + \Theta_{nc}(d_1 - 1, 0)) + (1 - P_1)(-1 + \Theta_{nc}(d_1, 0)) =$$

$$= P_1\Theta_{nc}(d_1 - 1, 0)) + (1 - P_1)\Theta_{nc}(d_1, 0)) - 1$$

i.e., with probability P_1 agent 1 succeeds in decreasing its distance to the meeting location by one, and with probability $1 - P_1$ it fails and remains at the same location. Recursively, we can compute the remaining joint expected time to meet with the updated parameters. Similarly for agent 2: $\Theta_{nc}(0, d_2) = P_2(-1 + \Theta_{nc}(0, d_2 - 1)) + (1 - P_2)(-1 + \Theta_{nc}(0, d_2))$. If none of the agents has reached the meeting place yet, then there are four different cases in which either both, only one, or none succeeded in moving in the right direction and either or not decreased their distances to the meeting location respectively:

$$\Theta_{nc}(d_1, d_2) = P_1P_2(-1 + \Theta_{nc}(d_1 - 1, d_2 - 1)) + P_1(1 - P_2)(-1 + \Theta_{nc}(d_1 - 1, d_2)) +$$

$$+(1 - P_1)P_2(-1 + \Theta_{nc}(d_1, d_2 - 1)) + (1 - P_1)(1 - P_2)(-1 + \Theta_{nc}(d_1, d_2)) =$$

$$= P_1P_2\Theta_{nc}(d_1–1, d_2–1)+P_1(1–P_2)\Theta_{nc}(d_1–1, d_2)+(1–P_1)P_2\Theta_{nc}(d_1, d_2–1)+(1–P_1)(1–P_2)\Theta_{nc}(d_1, d_2)-1$$

We computed $\Theta_{nc}(d_1, d_2)$ for all possible distances d_1 and d_2 in a 2D grid of size 10×10. The minimal expected time to meet was obtained when $d_1 = d_2 = 9$ and the expected cost was -12.16.

In summary, approximating the optimal solution to the Meeting under Uncertainty example when direct communication is possible and the mechanism applied is the one described above will unfold as follows: At time t_0, the initial state of the system s^0 is fully observable by both agents. The agents set a meeting point in the middle of a Manhattan path that connects them. Denote by d_0 the distance between the agents at t_0 and $g_{t_0} = (g^1_{t_0}, g^2_{t_0})$ the goal state set at t_0. Each one of the agents can move optimally towards its corresponding component of g_{t_0}. Each agent moves independently in the environment because the transitions and observations are independent. Each time t, when the policy of communication instructs an agent to initiate exchange of information, the current Manhattan distance between the agents d_t is revealed to both. Then, the mechanism is applied, setting a possibly new goal state g_t, which decomposes into two components one for each agent. This goal state g_t is in the middle of the Manhattan path that connects the agents with length d_t revealed through communication.

C Policies of Communication - Myopic-greedy Approach

Tables 11, 12, and 13 present the complete policies of communication for agents acting in the Meeting under Uncertainty scenario (see Appendix B). Each row corresponds to a configuration

tested with different state-transition uncertainties. Each column corresponds to a synchronized state, given by the possible Manhattan distance between the agents moving in a 2D grid of size 10x10. Given a certain value for P_u and a certain global distance, each agent interprets the value in any entry as the next time to communicate its position. Time is reset to zero when the agents exchange information. As long as the distance between the agents is larger and the communication cost increases the policy instructs the agents to communicate later, i.e., the agents should keep operating until the information exchanged will have a better effect on the rescheduling of the meeting place.

	d0=distance between agents when last synchronized, g located at $d0/2$																	
P_u	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3
0.4	2	2	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3
0.6	2	2	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3
0.8	2	2	2	3	2	4	2	4	2	4	2	4	2	4	2	4	2	4

Table 11: Myopic-greedy Policy of Communication: $C_\Sigma = -0.1$,$R_a = -1.0$.

	d0=distance between agents when last synchronized, g located at $d0/2$																	
P_u	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	3	4	3	5	3	6	4	7	4	7	5	7	5	8	5	8	6	9
0.4	2	3	3	4	4	5	4	6	5	7	5	7	6	8	6	8	7	9
0.6	2	2	3	4	4	5	5	6	6	7	6	8	7	8	7	9	8	10
0.8	2	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10

Table 12: Myopic-greedy Policy of Communication: $C_\Sigma = -1.0$,$R_a = -1.0$.

	d0=distance between agents when last synchronized, g located at $d0/2$																	
P_u	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	9	9	11	13	14	17	18	20	21	23	25	27	28	30	32	34	35	37
0.4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0.6	4	4	5	6	6	7	8	9	9	10	11	12	12	13	14	15	15	16
0.8	3	3	4	4	5	5	6	7	7	8	8	9	10	10	11	11	12	12

Table 13: Myopic-greedy Policy of Communication: $C_\Sigma = -10.0$,$R_a = -1.0$.

D The Average Performance of the Myopic-greedy Approach

Tables 14 and 16 present the results obtained after running 1000 experiments when the cost of communication was zero (Ideal case), when sub-goals could be communicated and when the myopic-greedy policy of communication was computed. When communication is not allowed, the results were computed based on the analytical functions explained in Appendix B (a meeting point was set in the middle of the grid at time 0). Tables 15 and 17 present the average number of communication acts performed in each one of these cases. When the cost of communication was -1, in all the cases when the transition probability was 0.2,0.4, or 0.6 the results obtained by the greedy approach were significantly greater than the results obtained by communicating sub-goals. When $P = 0.8$, the greedy approach produced poorer results than the communicate subgoals did. When the communication cost was increased to -10, the results obtained for the greedy implementation and the communication of sub-goals (ad-hoc rule) were insignificantly different (with probability 61% and 82% when $P = 0.2$ and $P = 0.4$ correspondingly).

	Average Joint Utility				
P_u	No-Comm.	Ideal $C_\Sigma = 0$	Comm. SubGoals – Best p		Myopic-greedy
0.2	-104.925	**-62.872**	-65.906	0.3	-63.84
0.4	-51.4522	**-37.33**	-39.558	0.2	-37.774
0.6	-33.4955	**-26.444**	-27.996	0.2	-27.156
0.8	-24.3202	**-20.584**	-21.05	0.1	-21.3

Table 14: $C_\Sigma = -1.0$ in SubGoals and Myopic-greedy,$R_a = -1.0$.

	Average Communication Acts Performed			
P_u	No-Comm.	Ideal $C_\Sigma = 0$	Comm. SubGoals	Myopic-greedy
0.2	0	31.436	1.194	6.717
0.4	0	18.665	1	3.904
0.6	0	13.426	1	2.036
0.8	0	10.292	0	1.296

Table 15: $C_\Sigma = -1.0$ in Myopic-greedy and SubGoals,$R_a = -1.0$.

	Average Joint Utility				
P_u	No-Comm.	Ideal $C_\Sigma = 0$	Comm. SubGoals – Best p		Myopic-greedy
0.2	-104.925	**-62.872**	-69.286	0.1	-68.948
0.4	-51.4522	**-37.33**	-40.516	0.1	-40.594
0.6	-33.4955	**-26.444**	-28.192	0.1	-28.908
0.8	-24.3202	**-20.584**	-21.118	0.1	-22.166

Table 16: $C_\Sigma = -10.0$ in SubGoals and Myopic-greedy,$R_a = -1.0$.

P_u	Average Communication Acts Performed			
	No-Comm.	Ideal $C_\Sigma = 0$	Comm. SubGoals	Myopic-greedy
0.2	0	31.436	0	0.416
0.4	0	18.665	0	0.417
0.6	0	13.426	0	0.338
0.8	0	10.292	0	0.329

Table 17: $C_\Sigma = -10.0$ in Myopic-greedy and SubGoals,$R_a = -1.0$.

E Myopic-greedy Policies of Communication with Deadlines

Table 18 presents the policy of communication computed following the myopic-greedy approach when the agents continue acting until they meet (no deadlines). Tables 19 and 20 show how this policy changes if different deadlines are added to the system, i.e., agents are penalized if they do not meet by the deadline.

P_u	d0=distance between agents when last synchronized, g located at $d0/2$																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	9	9	11	13	14	17	18	20	21	23	25	27	28	30	32	34	35	37
0.4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0.6	4	4	5	6	6	7	8	9	9	10	11	12	12	13	14	15	15	16
0.8	3	3	4	4	5	5	6	7	7	8	8	9	10	10	11	11	12	12

Table 18: Myopic-greedy Policy of Communication: $C_\Sigma = -10.0$,$R_a = -1.0$, No Deadline.

P_u	d0=distance between agents when last synchronized, g located at $d0/2$																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	0	0	0	0	0	0	0	0	0	0	0	0	5	5	4	4	4	4
0.4	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.6	4	4	5	6	6	7	0	0	0	0	0	0	0	0	0	0	0	0
0.8	3	3	4	4	5	5	6	7	7	8	0	0	0	0	0	0	0	0

Table 19: Myopic-greedy Policy of Communication: $C_\Sigma = -10.0$,$R_a = -1.0$, Deadline at T=8, Penalty=-100.0.

P_u	d0=distance between agents when last synchronized, g located at $d0/2$																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.2	9	9	11	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.4	5	6	7	8	9	10	11	12	13	0	0	0	0	0	0	0	0	0
0.6	4	4	5	6	6	7	8	9	9	10	11	12	12	13	0	0	0	0
0.8	3	3	4	4	5	5	6	7	7	8	8	9	10	10	11	11	12	12

Table 20: Myopic-greedy Policy of Communication: $C_\Sigma = -10.0$,$R_a = -1.0$, Deadline at T=15, Penalty=-100.0.

F The Average Performance of the *OptCom* Algorithm

Tables 21 and 23 present the joint utilities attained by a monotonic goal-oriented Dec-MDP, implemented in the Meeting under Uncertainty example when C_Σ took the values -1 and -10. We compare between the No-Communication case (where the meeting point is fixed at time 0 in the middle of the grid), the Ideal case with communication cost zero, the myopic-greedy case that punishes the agents if they did not meet by the finite-horizon, and the results obtained from running the *OptCom* algorithm (see Section 8). As can be seen from these results, even in this simple example, backward-induction attains results significant higher than the greedy approach. Tables 22 and 24 present the corresponding average number of communication acts in each case.

P	Average Joint Utility			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	-71.138	-62.55	-63.298	-62.776
0.4	-42.112	-37.292	-38.014	-37.622
0.6	-29.078	-26.716	-27.178	-26.692
0.8	-22.344	-20.57	-21.23	-20.622

Table 21: $C_\Sigma = -1.0$.

P	Average Communication Acts Performed			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	0	31.275	6.687	30.388
0.4	0	18.646	3.99	17.811
0.6	0	13.358	2.115	12.346
0.8	0	10.285	1.233	9.311

Table 22: $C_\Sigma = -1.0$.

P	Average Joint Utility			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	-71.138	-62.7	-69.516	-63.4
0.4	-42.112	-37.788	-40.994	-37.678
0.6	-29.078	-26.644	-28.974	-26.89
0.8	-22.344	-20.606	-22.09	-20.59

Table 23: $C_\Sigma = -10.0$.

P	Average Communication Acts Performed			
	No-Comm.	Ideal $C_\Sigma = 0$	Myopic-greedy	OptCom
0.2	0	31.35	0.444	28.957
0.4	0	18.894	0.428	16.032
0.6	0	13.322	0.33	11.474
0.8	0	10.303	0.301	9.2

Table 24: $C_\Sigma = -10.0$.