Minimizing Communication Cost in a Distributed Bayesian Network Using a Decentralized MDP[*]

Jiaying Shen
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610, USA
jyshen@cs.umass.edu

Victor Lesser
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610, USA
lesser@cs.umass.edu

Norman Carver
Computer Science Department
Southern Illinois University, mailcode 4511
Carbondale, IL 62901, USA
carver@cs.siu.edu

ABSTRACT

In complex distributed applications, a problem is often decomposed into a set of subproblems that are distributed to multiple agents. We formulate this class of problems with a two layer Bayesian Network. Instead of merely providing a statistical view, we propose a satisficing approach to predict the minimum expected communication needed to reach a desired solution quality. The problem is modelled with a decentralized MDP, and two approximate algorithms are developed to find the near optimal communication strategy for a given problem structure and a required solution quality.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Coherence and coordination, Multiagent systems*

General Terms

Algorithms, Design

Keywords

coordination of multiple agents, action selection, decentralized MDPs, decision-theoretic planning, Bayesian Networks

1. INTRODUCTION

In complex distributed applications, such as distributed interpretation, a problem is often decomposed into a set of subproblems and each subproblem is distributed to an agent who will be responsible for solving it. The existence of interactions between subproblems means that the agents cannot simply solve the subproblems individually and then combine local solutions together. In such systems, the amount of communication among agents required to guarantee global optimality or global consistency may be very significant. Thus, "satisficing" approaches have been developed that trade off optimality for reduced communication [4]. One approach is for agents to generate local solutions based on their own data and then transmit these high level solutions to other agents. Based on consistency and credibility of these local solutions, new local solutions may be generated or more detailed data sent until a sufficient level of consistency and credibility has been achieved among the agents. An important characterization of such distributed protocols is how much communication is required and the likelihood that the solution will be the same as what would be generated by an optimal centralized algorithm which uses all available information.

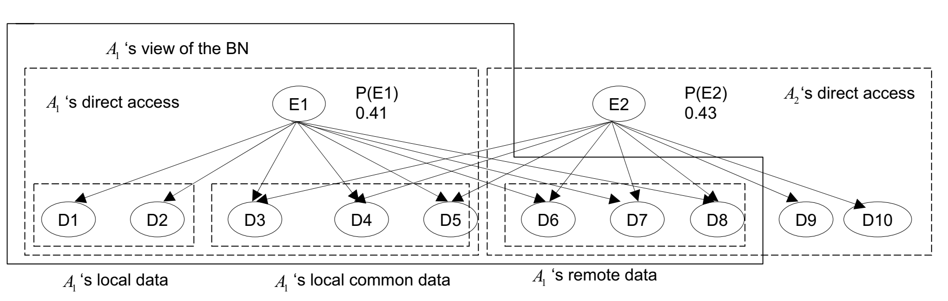
Most approaches to managing communication trade off solution quality for reduced communication, but only from a statistical view. The behavior of the algorithms are often analyzed over a collection of problems to say that p percent of the time they will get the required solution quality q with an average amount of communication c [4].

We would like to take this satisficing approach to the next step by exploring whether we can design a parameterized algorithm where we can predict, for a fixed amount of communication, the maximum expected level of confidence in the final solution. Conversely, given a desired confidence level in the final solution we would like to determine the expected amount of communication the agents need. Finally, the algorithm should produce a communication strategy that will require only the minimum expected amount of communication necessary to achieve the desired solution quality.

We will study these issues in terms of Distributed Bayesian Networks. Recent work includes algorithms such as in [10] that produce the same final solution as is generated by a centralized problem solving system. However, this approach can potentially require significant communication. In contrast, our framework takes the amount of communication into account, as well as the quality of the final solution. The agents do only the necessary amount of communication in order to achieve the required level of solution quality.

In our problem, each agent has its distinct local view, which is related only to its local subproblem and communication history. At the same time, the agents share the same

[*]This material is based upon work supported by the National Science Foundation under Grants No. IIS-0004112 and IIS-0219606.



P(D1 | E1)	
0.1975	1
0.0043	0

P(D2 | E1)	
0.2903	1
0.0559	0

P(D3 | E1	E2)	
0.8020	1	1
0.1000	1	0
0.7800	0	1
0	0	0

P(D4 | E1	E2)	
0.5385	1	1
0.2900	1	0
0.3500	0	1
0	0	0

P(D5 | E1	E2)	
0.1916	1	1
0.0600	1	0
0.1400	0	1
0	0	0

P(D6 | E1	E2)	
0.4342	1	1
0.3100	1	0
0.1800	0	1
0	0	0

P(D7 | E1	E2)	
0.1100	1	1
0.	1	0
0.1100	0	1
0	0	0

P(D8 | E1	E2)	
0.7227	1	1
0.4100	1	0
0.5300	0	1
0	0	0

P(D9 | E2)	
0.2486	1
0.3116	0

P(D10 | E2)	
0.7990	1
0.0164	0

Figure 1: There are two events E_1 and E_2. Data $D_1, D_2, ...D_{10}$ are distributed between two agents. A_1 has access to $D_1, ...D_5$ and is responsible for solving E_1, while A_2 can see only $D_6, ...D_{10}$ and is responsible for solving E_2. The objective is for A_1 and A_2 to figure out what E_1 and E_2 are with required confidence and with minimum expected communication cost

goal of minimizing the total expected communication cost. Therefore, our system is a decentralized and yet cooperative one. Most distributed sensor systems have this feature, and can be represented by a BN structure. Based on these characteristics, we propose a decision-theoretic framework to model this multi-agent coordination decision problem. A decentralized Markov Decision Process (DEC-MDP) is constructed from the BN structure, whose objective is to find a joint communication policy for the two agents that minimizes the expected communication cost. Each agent needs to decide its communication actions based only on its local observations. This lack of complete knowledge of the global state results in difficulty finding an optimal solution for the DEC-MDP. In fact, recent work has shown that solving a DEC-MDP is NEXP-hard [2].

What makes our problem more difficult than some of the others [1] is its tightly coupled nature. An agent's communication action directly changes the other agent's view. The local MDPs of the two agents are largely dependent on each other. This makes it hard to construct algorithms that are guaranteed to find the globally optimal solution. We have designed two algorithms to approximate the globally optimal solution for our DEC-MDP. One is an iterative algorithm that is guaranteed to converge to a local optimal solution, but the quality of the policy it generates largely depends on the starting policy of the iterative process. The other approach is based on a lookup algorithm which is much less computationally expensive and can be easily extended to more than two agents. Though there is no guarantee that can be made about the solution either generates, experimental work described in Section 5 indicates that in the problems studied both approaches lead to policies that are of very good quality. To our knowledge, this is some of the first work providing algorithms that approximate the optimal solution for communication problems in a complex problem solving setting that are formulated in a decision-theoretic model.

2. PROBLEM SETTING

In an interpretation system, there are a set of observable data that are caused by some possible events, which agents do not have the means to directly observe. The agents need to collect the values of the data and pick out the set of events that are most likely the cause of them. In many environments, the problem is inherently distributed and the network is fairly large. In those cases, it is common to distribute not only the data but also the interpretation task among several agents. Inevitably, there is a close interaction between the agents, since the local sub-problems an agent is responsible for are often dependent on some of the data collected by other agents. The existence of subproblem interactions means that the agents will need to communicate during problem solving. Therefore, agent communication strategies can have a major effect on the cost of problem solving.

In our system, we use a two-layer Bayesian Network to represent the problem structure (Figure 1). The top level nodes are the events that are the possible causes of the observed data, while the leaves are the raw data gathered by various agents. There are two agents, each of whom has direct access to only a part of the observable data. The interpretation task is distributed to the two agents as well. Each agent is responsible only for its part of the overall problem and has the knowledge of just the part of the network that is relevant to its task. The agents can either request or send data. The objective is to figure out the most likely interpretation of the causal events with a certain level of confidence using as little communication as possible. For example, in Figure 1, there are two agents A_1 and A_2. A_1 is responsible for interpreting the event E_1 and therefore knows about the part of the causal network that is relevant to E_1. Normally, out of the necessary data $D_1, ...D_8$, it can directly observe only D_1 through D_5.

Definition 1. Based on the nature of different parts of the relevant data to an interpretation task, we divide them into three categories. **Local data** are the data that can be directly observed by the agent and are relevant only to its local task. They do not need to be transmitted to the remote agent at any time since each agent has only a partial view of the network. **Local common data** are the rest of the data that are observable by the local agent. They not only are important for the local task but also would help remote agents in their interpretation tasks. They are the candidates to be sent in the local agent's decision process. **Remote data** are the ones that cannot be directly observed by the local agent, but knowing them might increase the confidence of its local solution. When an agent is considering requesting data, remote data are the natural candidates. Figure 1 gives an example for all three types of data.

Definition 2. The **evidence** ϵ_{A_i} of an agent A_i are the values of the data that the agent has collected so far. They can be the values that are observed directly by that agent or acquired from the remote agent. The **complete evidence** $\epsilon^*_{A_i}$ of A_i are the values of all the relevant data of the agent. ϵ^* are the values of all the low level data in the system. At any time, $\epsilon^* = \cup_i \epsilon^*_{A_i}$. In Figure 1, A_1 observes only the data values of $D_1, \ldots, D_5$, and ϵ_{A_1} can be one of the 32 possible configurations. As communication goes on, an agent will gather more evidence from the remote agents, and the set ϵ_{A_1} will grow. The relevant data of A_1's interpretation task is $D_1, \ldots, D_8$. Therefore, $\epsilon^*_{A_1}$ is one of the 64 possible configurations. ϵ_{A_i} is always a subset of $\epsilon^*_{A_i}$.

Definition 3. **Likelihood** L_{A_i}. Based on the evidence observed, an agent can calculate the conditional probabilities of every possible interpretation of its local events based on the current evidence, i.e., $L_{A_i}(E_{A_i}) = P(E_{A_i}|\epsilon_{A_i})$. For example, in Figure 1 the local event of A_1 is E_1. Therefore, $L_{A_1}(E_1) = P(E_1|\epsilon_{A_1})$. Specifically, before any communication occurs, $L_{A_1}(E_1) = P(E_1|D_1, \ldots, D_5)$.

Definition 4. **MAPI (Maximum A Posteriori Interpretation)**. Based on the current evidence set available to an agent, there is a most likely interpretation of the events. $MAPI(\epsilon_{A_i}) = argmax_h P(E_{A_i} = h|\epsilon_{A_i})$.

An agent A_i in the system is only able to find the most likely interpretation of the local events given its current evidence set $MAPI(\epsilon_{A_i})$. Ideally the decentralized system should generate the interpretation that a centralized system will generate given the complete data values $MAPI(\epsilon^*_{A_i})$. Unfortunately, with only partial knowledge of the relevant data values, an agent cannot always guarantee that the current local MAPI is the global MAPI. On the other hand, with the conditional probability table given by the BN, an agent can predict the probability of different relevant data configurations given its current evidence $P(\epsilon^*_{A_i}|\epsilon_{A_i})$. Hence we have the following definition.

Definition 5. **Confidence** is the likelihood of the local MAPI being the global MAPI of E_A given the current known evidence of A.

$$\begin{aligned} C(MAPI(\epsilon_A)) &= P(MAPI(\epsilon_A) = MAPI(\epsilon^*_A)) \\ &= \sum_{e \in \{\epsilon^*_A | MAPI(\epsilon_A) = MAPI(\epsilon^*_A)\}} P(e|\epsilon_A) \end{aligned}$$

Confidence is not a novel idea, but the way we define it is different from previous work. [4] defined it simply as the probability of an event being the local MAPI, i.e., $P(E_A = MAPI(\epsilon_A)|\epsilon_A)$. As we have stated, what we really care about is not the likelihood of an event being the local MAPI, but whether the local MAPI agrees with the global MAPI, i.e., whether the decentralized system will generate the same result that a centralized system would. That is exactly what our definition of confidence achieves. Using confidence as a measurement of the solution quality of a decentralized system as compared to that of a centralized one, we can make a tradeoff between the quality of the solution and the communication cost. Another interesting and useful property of our definition of confidence is that it is guaranteed to reach 100% when the values of all the relevant data are known. As a result, a given confidence level can always be satisfied.

Given a problem structure in a two level BN such as the one in Figure 1 and a specified confidence threshold, we need to find a communication strategy. Such a communication strategy should specify what communication action each agent should take based on their current knowledge at each stage. In this paper we are considering only the case of synchronous communication. In other words, the two agents take turns to decide their actions in the sequence of $A_1, A_2, A_1, A_2, ...$ until the confidence threshold is reached.

Definition 6. A **communication strategy** $\pi_{<A_1,A_2>}$ of the system is a pair of policies $< \pi_{A_1}, \pi_{A_2} >$. π_{A_i} is a mapping from a local evidence and communication history pair $< \epsilon_{A_i}, H >$ to local action a_i, where H is the communication history of the system.

There are three classes of actions for each agent: SEND, REQUEST and NULL. The content of the SEND and REQUEST actions at each stage are restricted by the communication history so far. An agent can send only the part of the local common data that the other agent has not acquired yet, and can request only a part of the remote data that it has no knowledge of. When the communication history indicates that it is not A_i's turn to act, the policy of that agent will always be NULL. A REQUEST action and its response by the other agent occur in the same stage. For example, in Figure 1, let us assume that initially $\epsilon_{A_1} = \{D_1 = 1, D_2 = 0, D_3 = 0, D_4 = 1, D_5 = 1\}$ and $H = \emptyset$. A valid action of A_1 at this step can be "SEND D_3 and D_4". It can also be "REQUEST D_8 and D_9". On the other hand, since it is not A_2's turn, the only valid action for A_2 is NULL. The cost model for this framework is very general. Each communication has its own cost, based on its action class and the particular data sent. The only restriction is that the cost of a REQUEST must also include the cost of the immediate reply by the other agent. For example, in our experiments, we use a simplistic cost model. In this model, SEND costs 1 per data sent. A REQUEST action costs 1 regardless of the data requested plus a reply cost equal to a SEND action of the same data. "REQUEST D_8 and D_9" would then cost 3.

Now let us summarize our problem. Given a problem structure in a two level BN and a specified confidence threshold t, a communication strategy $\pi_{<A_1,A_2>}$ needs to be found. The agents will follow this strategy until their confidence in the interpretation of the local events is higher than t, i.e., $C(MAPI(\epsilon_{A_1})) \geq t$ and $C(MAPI(\epsilon_{A_2})) \geq t$. The expected cost of a most desirable communication strategy should be minimized.

3. DECENTRALIZED MDP MODEL

As we have mentioned before, the system we described in the last section is both decentralized and cooperative at the same time. The communication actions of each agent are entirely dependent on its current local evidence and the communication history. Based on these characteristics, we are modelling this problem as a decentralized MDP[1].

Unlike centralized MDPs, in a decentralized MDP the process is controlled by multiple agents, each with possibly dif-

[1]For a more centralized approach to our problem, please refer to our previous work [9].

ferent information about the global state. At each time step the agents' observations together will uniquely determine the global state, though possibly none of them have the complete information of the global state. A local policy for an agent is a mapping from its local histories of observations to an action. The task of a decentralized MDP is normally to find an optimal joint policy of all the agents that maximizes the expected total return.

First let us give a formal definition of decentralized MDP, adapted from [2].

Definition 7. A **decentralized MDP** is a tuple $< S, Ac_1, Ac_2, P, R, \Omega_1, \Omega_2, O >$, where

- S is a finite set of global states, with distinguished initial state s_0.
- Ac_1 and Ac_2 are action sets of the two agents.
- P is a transition probability table. $P(s'|s, a_1, a_2)$ is the probability of transitioning from s to s' on taking actions a_1 and a_2, where $s, s' \in S$, $a_1 \in Ac_1$, and $a_2 \in Ac_2$.
- R is a reward function. $R(s, a_1, a_2, s')$ is the reward obtained from taking actions a_1 and a_2 from state s and reaching state s', where $s, s' \in S$, $a_1 \in Ac_1$ and $a_2 \in Ac_2$.
- Ω_1 and Ω_2 are observations of the two agents.
- O is an observation probability table. $O(s, a_1, a_2, s', o_1, o_2) = P(o_1, o_2|s, a_1, a_2, s')$ is the probability of A_1 observing o_1 and A_2 observing o_2 when taking actions a_1 and a_2 respectively in state s and reaching state s'. Here $s, s' \in S$, $a_1 \in Ac_1$, $a_2 \in Ac_2$, $o_1 \in \Omega_1$ and $o_2 \in \Omega_2$.

Definition 8. A **local policy** π_i for agent A_i is a mapping from local histories of observations $\overline{o}_i$ to actions in Ac_i. A **joint policy** $\pi =< \pi_1, \pi_2 >$ is a pair of local policies, one for each agent.

Definition 9. The value $V_\pi(s)$ of a state s following policy $\pi =< \pi_1, \pi_2 >$ is:

$$\begin{aligned} V_\pi(s) \;=\; & \sum_{<\overline{o}_1,\overline{o}_2>} \sum_{q\in S} \sum_{s'\in S} P_\pi(\overline{o}_1, \overline{o}_2, q|s) \cdot \\ & P(s'|q, \pi_1(\overline{o}_1), \pi_2(\overline{o}_2)) \cdot \\ & R(q, \pi_1(\overline{o}_1), \pi_2(\overline{o}_2), s'), \end{aligned} \tag{1}$$

where $P_\pi(\overline{o}_1, \overline{o}_2, q|s)$ is the probability of observing $\overline{o}_1, \overline{o}_2$ and reaching state q from state s following policy π. The value of the initial state $V_\pi(s_0)$ is the expected total reward following the policy π from state s_0. To solve a decentralized MDP is to find a joint policy $< \pi_1, \pi_2 >$ that maximizes $V_\pi(s_0)$.

Definition 10. An **underlying centralized MDP** of a decentralized MDP is a tuple $< S, Ac_1, Ac_2, P, R >$, where S, Ac_i, P and R are the same as its corresponding part in the DEC-MDP. A **centralized policy** $\pi^c =< \pi^c_1, \pi^c_2 >$ for this MDP is a mapping from the global state $s \in S$ to a pair of local actions $< a_1, a_2 >\in Ac_1 \times Ac_2$. The value of the state is denoted as $V^c(s)$. Solving an underlying centralized MDP is to find a centralized policy that maximizes the value of the start state. We denote the value of a state for the optimal policy as $V^{c*}(s)$.

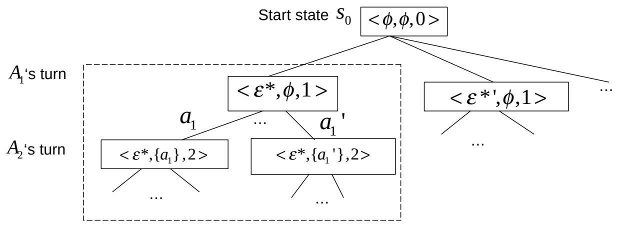


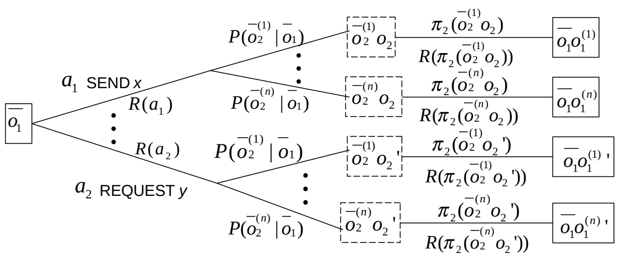
Figure 2: A portion of a typical decentralized MDP.

The key difference between a decentralized MDP and its underlying centralized MDP are their policies. The joint policy of a DEC-MDP is a pair of local policies each of which is dependent on the local observation sequence of the corresponding agent, while the centralized policy for the centralized MDP is itself dependent only on the global state, and is not directly related to the local policies. When the two agents have access to the global state at all times, a DEC-MDP is reduced to its underlying centralized MDP. Therefore an underlying centralized MDP is equivalent to a Multi-agent Markov decision-process (MMDP) [3].

Now let us model our problem in the framework of a decentralized MDP as follows.

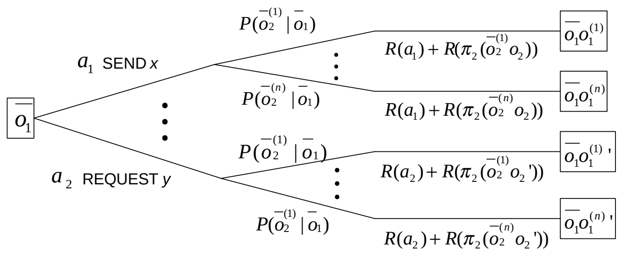
- Every global state $s \in S$ is a tuple $< \epsilon^*, H, i >$, where ϵ^* is all of the low level data values observed by the system, H is the communication history, and i indicates that it is A_i's turn to communicate. s_0 is a dummy start state $< \emptyset, \emptyset, 0 >$, which transitions to all the possible real start states $< \epsilon^*, \emptyset, 1 >$ before any communication takes place. If at state s, both agents have reached confidence level t for its local interpretation task, i.e., when $C(MAPI(\epsilon_{A_i})) \geq t$, $i = 1, 2$, then s is a final state.
- Ac_1 and Ac_2 are action sets of the two agents. $Ac_i \in \{$ NULL, SEND x, REQUEST $y\}$, where x is a subset of the local common data of A_i and y is a subset of the remote data. When it is not A_i's turn to communicate, its action is the NULL action.
- P is a transition probability table. $P(s'|s, a_1, a_2) = P(< \epsilon^{*\prime}, H', i > | < \epsilon^*, H, j >, a_1, a_2) = 1$ if and only if $\epsilon^* = \epsilon^{*\prime}$, $H' = H \cup \{a_1, a_2\}$ and $i \neq j$. Otherwise, $P(s'|s, a_1, a_2) = 0$. As a special case, the transition probability from s_0 to $s =< \epsilon^*, \emptyset, 1 >$ after both agents execute NULL is $P(s|s_0, NULL, NULL) = P(\epsilon^*)$. It can be calculated given the BN structure.
- R is a reward function. $R(s, a_1, a_2, s') = R(a_1, a_2) = -c(a_1) - c(a_2)$, where $c(a_i)$ is the cost of the communication action a_i, $i = 1, 2$.
- Ω_1 and Ω_2 are the sets of observations for the two agents. An observation $o_i \in \Omega_i$ is the data value just sent or received by A_i. As a special case, after s_0, each agent observes the values of its local data. An observation sequence $\overline{o}_i =< \epsilon_{A_i}, H >$. ϵ_{A_i} can be mapped from the global state s using ϵ^* and H.
- $O(s, a_1, a_2, s', o_1, o_2) = 1$ if and only if o_1 and o_2 are the new observations seen after the agents take actions a_1 and a_2 in state s. Otherwise it equals to 0.

For example, take the BN structure in Figure 1. A possible global state s might be $\epsilon^* =< \{D_1 = 1, D_2 = 0, D_3 =$



$\overline{o}_1$
a_1 SEND x
$R(a_1)$
$R(a_2)$
a_2 REQUEST y
$P(\overline{o}_2^{(1)} \mid \overline{o}_1)$
$P(\overline{o}_2^{(n)} \mid \overline{o}_1)$
$\overline{o}_2^{(1)} o_2$
$\overline{o}_2^{(n)} o_2$
$\overline{o}_2^{(1)} o_2'$
$\overline{o}_2^{(n)} o_2'$
$\pi_2(\overline{o}_2^{(1)} o_2)$
$R(\pi_2(\overline{o}_2^{(1)} o_2))$
$\pi_2(\overline{o}_2^{(n)} o_2)$
$R(\pi_2(\overline{o}_2^{(n)} o_2))$
$\pi_2(\overline{o}_2^{(1)} o_2')$
$R(\pi_2(\overline{o}_2^{(1)} o_2'))$
$\pi_2(\overline{o}_2^{(n)} o_2')$
$R(\pi_2(\overline{o}_2^{(n)} o_2'))$
$\overline{o}_1 o_1^{(1)}$
$\overline{o}_1 o_1^{(n)}$
$\overline{o}_1 o_1^{(1)}{}'$
$\overline{o}_1 o_1^{(n)}{}'$

Figure 3: A snapshot of the local belief MDP of A_1 with the internal states.



$\overline{o}_1$
a_1 SEND x
a_2 REQUEST y
$P(\overline{o}_2^{(1)} \mid \overline{o}_1)$
$P(\overline{o}_2^{(n)} \mid \overline{o}_1)$
$R(a_1) + R(\pi_2(\overline{o}_2^{(1)} o_2))$
$R(a_1) + R(\pi_2(\overline{o}_2^{(n)} o_2))$
$R(a_2) + R(\pi_2(\overline{o}_2^{(1)} o_2'))$
$R(a_2) + R(\pi_2(\overline{o}_2^{(n)} o_2'))$
$\overline{o}_1 o_1^{(1)}$
$\overline{o}_1 o_1^{(n)}$
$\overline{o}_1 o_1^{(1)}{}'$
$\overline{o}_1 o_1^{(n)}{}'$

Figure 4: A snapshot of the local MDP of A_1 without the internal states

$0, D_4 = 1, D_5 = 1, D_6 = 0, D_7 = 1, D_8 = 1, D_9 = 0, D_{10} = 0\}, \emptyset, 1 >$. The possible actions for A_1 are NULL, SEND x, and REQUEST y, where $x \subseteq \{D_3 = 0, D_4 = 1, D_5 = 1\}$ and $y \subseteq \{D_6, D_7, D_8\}$. Since it is not A_2's turn to communicate, its only action is NULL. If A_1 chooses REQUEST $\{D_7, D_8\}$, then the next state is $< \epsilon^*, \{D_7 = 1, D_8 = 1\}, 2 >$ and both agents observe $\{D_7 = 1, D_8 = 1\}$.

Figure 2 is a portion of a typical DEC-MDP built for our problem. It is interesting to note that the structure of the DEC-MDP is stochastic only in the first transition, from s_0. The remaining transitions are all deterministic. In future work, we would like to exploit this structure to develop better approximation algorithms or possibly a tractable optimal algorithm.

4. SOLVING THE DEC-MDP

4.1 Local MDP

In a general DEC-MDP, there is one transition function, reward function and observation function, and they are defined over two global states and a joint action. This is why a DEC-MDP cannot simply be treated as two separate POMDPs, one for each agent. It also cannot be treated as a single centralized POMDP. In the case of the DEC-MDP, an agent must choose its action based only on the sequence of observations it has seen. In the POMDP case, it chooses its actions based on the sequence of pairs of observations seen by both agents. It has a centralized view inside the model instead of a distributed view. However, if the policy of one of the agents is fixed, then the DEC-MDP reduces to a POMDP, and the POMDP can be reduced to a belief MDP (the belief state of the POMDP is the state of the MDP). Since this MDP corresponds to the local decision problem for one of the agents, we refer to it as the local MDP. By solving the local MDP, we can find the optimal local policy for one agent given the fixed local policy for the other.

The interesting element of such a local MDP is its transition probability table. To make it more understandable we generate a group of internal states for each local state. Each internal state represents the possible observation sequences of the remote agent after the agent executes a certain action at the current local state. Since the remote policy is fixed for each local MDP, for each pair of current local state and internal state there is one and only one next local state. For example, Figure 3 shows a snapshot of A_1's local MDP. Here, before A_1 chooses its next action, its observation is $\overline{o}_1$ and A_2 may have observation sequence of $\overline{o}_2^{(1)}, \ldots, \overline{o}_2^{(n)}$ which are dependent on A_1's current local evidence ϵ_{A_1} and the past communication history of the system H, which is exactly $\overline{o}_1$. Hence, the probabilities of A_2 observing $\overline{o}_2^{(1)}, \ldots, \overline{o}_2^{(n)}$ are $P(\overline{o}_2^{(1)}|\overline{o}_1), \ldots, P(\overline{o}_2^{(n)}|\overline{o}_1)$ respectively. If A_1 sends data to A_2, no matter what A_2's current observation sequence is, it will have the same new observation o_2, i.e., A_1's action a_1 and the new data from A_1. This updates A_2's observation history to $\overline{o}_2^{(1)} o_2, \ldots, \overline{o}_2^{(n)} o_2$, which are the n internal states for the local state $\overline{o}_1$ after action a_1. Since A_2 has a fixed policy π_2, A_2's action when observing $\overline{o}_2^{(1)} o_2, \ldots, \overline{o}_2^{(n)} o_2$ are determined, namely $\pi(\overline{o}_2^{(1)} o_2), \ldots, \pi(\overline{o}_2^{(n)} o_2)$, which will change A_1's new observation sequence to be $\overline{o}_1 o_1^{(1)}, \ldots, \overline{o}_1 o_1^{(n)}$. Therefore, the next states of $\overline{o}_1$ after the action are $\overline{o}_1 o_1^{(1)}, \ldots, \overline{o}_1 o_1^{(n)}$ with the transitional probability of $P(\overline{o}_2^{(1)}|\overline{o}_1), \ldots, P(\overline{o}_2^{(n)}|\overline{o}_1)$ respectively. Similarly, we get the next states from $\overline{o}_1$ by executing REQUEST y actions. Figure 4 shows the corresponding snapshot of the actual local MDP of A_1 without the internal states.

Since our system is a cooperative one, the goal of the agents is to maximize the global utility instead of the local one. This means that in a local MDP, when calculating the reward received from an action, we not only need to include the reward from the local action alone but also that gained by the remote action. For example, in Figures 3 and 4, the reward for executing action a_1 at state $\overline{o}_1$ and reaching state $\overline{o}_1 o_1^{(1)}$ is $R(\overline{o}_1, a_1, \overline{o}_1 o_1^{(1)}) = R(a_1) + R(\pi_2(\overline{o}_2^{(n)} o_2))$.

We can summarize the local MDP of an agent A_i when fixing the other agent's policy π_j as follows:

- S is a finite local state set. Each state is the observation sequence of the local agent $\overline{o}_i =< \epsilon_{A_i}, H >$. To resolve the uncertainty of the initial observation, a dummy start state o_i^0 is added. When the confidence level is reached at a state, it is a final state.
- Ac is the action set. For A_i, it is Ac_i in the DEC-MDP.
- P is the transition probability table, the calculation of which has been illustrated.
- $R(s, a, s')$ is the reward function. It is a sum of the reward gained directly by A_i executing action a and that gained by the remote agent's action as a result of a.
- $V_{<\pi_i, \pi_j>}(\overline{o}_i)$ is the value of the state $\overline{o}_i$ for a local policy π_i of the local MDP. To solve the local MDP is to find the policy π_i that maximizes the state value of the start state o_i^0.

The CPT of the BN is used to calculate the transitional probability tables and decide whether a local state is a final state of the local MDP. By constructing such an MDP, we are utilizing the information provided by the BN to model the local agent's belief in the current global state based on its local state. Its belief in the observation sequence of the remote agent directly influences the transitional probability of the states. Therefore, by finding an optimal policy for the local MDP, the agent is implicitly using the knowledge obtained from both the current evidence set and the communication history of the system.

In our DEC-MDP, the two agents interact through actions, which directly affect each other's local observations. The local MDPs of the two agents are tightly coupled. The rewards, transitional probability table of the MDPs are both dependent on each other. This means that it is very hard to solve our DEC-MDP without exhaustive search. In the next two subsections we will present two approximate algorithms which utilize the property of the local MDPs as expressed by Theorem 1.

THEOREM 1. *Maximizing the utility of the local MDP of A_i maximizes the utility of the decentralized MDP if the other agent's policy π_j is fixed.*

PROOF. First let us simplify the utility function (1) due to the alternating action behavior of our decentralized MDP.

$$\begin{aligned}
& V_{<\pi_1,\pi_2>}(s_0) \\
= & \sum_{<\bar{o}_1,\bar{o}_2>} \sum_{q\in S} \sum_{S'\in S} P_{<\pi_1,\pi_2>}(\bar{o}_1,\bar{o}_2,q|s) \cdot \\
& P(s'|q,\pi_1(\bar{o}_1),\pi_2(\bar{o}_2))R(q,\pi_1(\bar{o}_1),\pi_2(\bar{o}_2),s') \\
= & \sum_{<\bar{o}_1,\bar{o}_2>} \sum_{q\in S_1} \sum_{S'\in S_2} P_{\pi_1,\pi_2>}(\bar{o}_1,\bar{o}_2,q) \cdot \\
& P(s'|q,\pi_1(\bar{o}_1))R(\pi_1(\bar{o}_1)) \\
& + \sum_{<\bar{o}_1,\bar{o}_2>} \sum_{q\in S_2} \sum_{S'\in S_1} P_{\pi_1,\pi_2>}(\bar{o}_1,\bar{o}_2,q) \cdot \\
& P(s'|q,\pi_2(\bar{o}_2))R(\pi_2(\bar{o}_2)) \\
= & \sum_{\bar{o}_1} R(\pi_1(\bar{o}_1))P_{<\pi_1,\pi_2>}(\bar{o}_1) \\
& + \sum_{\bar{o}_2} R(\pi_2(\bar{o}_2))P_{<\pi_1,\pi_2>}(\bar{o}_2)
\end{aligned}$$

Next, let us see what the global utility that the local MDP is trying to maximize is. Here we are taking advantage of the independence relationships due to our MDP setup.

$$\begin{aligned}
& V_{<\pi_1,\pi_2>}(o_1^0) \\
= & \sum_{\bar{o}_1} \sum_{o_1} P_{<\pi_1,\pi_2>}(\bar{o}_1|o_1^0)P(o_1|\bar{o}_1,\pi_1(\bar{o}_1)) \cdot \\
& R(\bar{o}_1,\pi_1(\bar{o}_1),\bar{o}_1 o_1) \\
= & \sum_{\bar{o}_1} \sum_{\bar{o}_2} \sum_{o_2} P_{<\pi_1,\pi_2>}(\bar{o}_1)P(\bar{o}_2|\bar{o}_1) \cdot \\
& P(o_2|\bar{o}_1,\pi_1(\bar{o}_1))R(\pi(\bar{o}_1)) \\
& + \sum_{\bar{o}_1} \sum_{\bar{o}_2} \sum_{o_2} P_{<\pi_1,\pi_2>}(\bar{o}_1)P(\bar{o}_2|\bar{o}_1) \cdot \\
& P(o_2|\bar{o}_1,\pi_1(\bar{o}_1))R(\pi_2(\bar{o}_2 o_2)) \\
= & \sum_{\bar{o}_1} R(\pi_1(\bar{o}_1))P_{<\pi_1,\pi_2>}(\bar{o}_1) \\
& + \sum_{\bar{o}_2} R(\pi_2(\bar{o}_2))P_{<\pi_1,\pi_2>}(\bar{o}_2)
\end{aligned}$$

Hence, we get

$$V_{<\pi_1,\pi_2>}(o_1^0) = V_{<\pi_1,\pi_2>}(s_0) \tag{2}$$

The same equation holds for o_2^0. □

4.2 Iterative Algorithm

ALGORITHM 1. **Iterative Algorithm**

```
1. Start from $A_1$, choose a random policy $\pi_2$ for $A_2$ (the simplest being that $A_2$ will not do any action at any state), generate the local MDP for $A_1$. Solve the MDP and get the local policy $\pi_1$ for $A_1$.
2. Repeat
    (a) Fix $\pi_1$ for $A_1$, generate the local MDP for $A_2$. Generate the local optimal policy $\pi'_2$ for $A_2$. If $V_{<\pi_1,\pi_2>}(o_2^0) == V_{<\pi_1,\pi'_2>}(o_2^0)$, go to step 3; else, update $\pi_2$ to $\pi'_2$.
    (b) Fix $\pi_2$ for $A_2$, generate the local MDP for $A_1$. Generate the local optimal policy $\pi'_1$ for $A_1$. If $V_{<\pi_1,\pi_2>}(o_1^0) == V_{<\pi'_1,\pi_2>}(o_1^0)$, go to step 3; else, update $\pi_1$ to $\pi'_1$.
3. Return $< \pi_1, \pi_2 >$.
```

THEOREM 2. *The Iterative Algorithm converges.*

PROOF. Let us first prove that the global utility of the local MDP's is monotonically increasing at each iteration, i.e., $V_{<\pi_1,\pi'_2>}(o_2^0) \geq V_{<\pi_1,\pi_2>}(o_1^0)$ and $V_{<\pi'_1,\pi_2>}(o_1^0) \geq V_{<\pi_1,\pi_2>}(o_2^0)$.

Since π'_2 is the optimal policy of A_2 when fixing A_1's policy as π_1, we have $V_{<\pi_1,\pi'_2>}(o_2^0) \geq V_{<\pi_1,\pi_2>}(o_2^0)$. According to Theorem 1, $V_{<\pi_1,\pi_2>}(o_2^0) = V_{<\pi_1,\pi_2>}(s_0) = V_{<\pi_1,\pi_2>}(o_1^0)$. Therefore, we have $V_{<\pi_1,\pi'_2>}(o_2^0) \geq V_{<\pi_1,\pi_2>}(o_1^0)$. Similarly we can prove $V_{<\pi'_1,\pi_2>}(o_1^0) \geq V_{<\pi_1,\pi_2>}(o_2^0)$.

On the other hand, since there is a finite joint policy space, there exists a globally optimal solution. The iteration is bound to stop when the utility hits the upper bound, i.e., that of the optimal solution if it does not stop before that. Therefore, the algorithm converges. □

A natural corollary of this theorem is that the global optimal solution is at one of the local convergence points generated by the Iterative Algorithm. From the proof of Theorem 2, we can see that for an iterative algorithm like ours, the sufficient condition for it to converge to a local optimal solution is that the local MDP is maximizing the same global utility as the decentralized MDP. Therefore, for any decentralized MDP, if we can construct a local MDP for the local agents which maximizes the same global utility as that of the DEC-MDP, an iterative algorithm will converge to a local optimal solution. There is other work using the same general idea, such as described in [5].

Although the Iterative Algorithm is guaranteed to converge to a local optimal solution, it needs to dynamically regenerate the local MDPs at each iteration. As a result, it is suitable to be run off-line. The other main disadvantage is that it depends on the starting policy chosen; it may be stuck at some fairly low quality local optimal solution without being able to reach the globally optimal one. One solution to the later problem is to randomly restart with a new initial policy and pick the best joint policy generated after a few time steps. In the experimental work that we present later, we did not use this variation since the original algorithm already performed very well.

4.3 Lookup Algorithm

THEOREM 3. *For a local MDP of A_i, when the policy of the remote agent is fixed as π_j ($i \neq j$) its state value $V_{<\pi_1,\pi_2>}(\overline{o}_i)$ for a specific local policy π_i has the property:*

$$\begin{aligned} V_{<\pi_1,\pi_2>}(\overline{o}_i) &= \sum_{s\in S} P(s|\overline{o}_i)V_{<\pi_1,\pi_2>}(s) \\ &= R(<\pi_1,\pi_2>(\overline{o}_i)) + \sum_{s\in S} P(s|\overline{o}_i)V_{<\pi_1,\pi_2>}(s') \end{aligned}$$

where s' is the next global state in the DEC-MDP after executing the action $<\pi_1,\pi_2>(\overline{o}_i)$ at state s.

This theorem is an extension of Theorem 2 and the proof is similar. It establishes the relationship between the value of an observation sequence of a local MDP and the value of the states in the decentralized MDP given a joint policy. As we have said, the local MDPs of our problem are very tightly coupled and the unknown policy of the other agent directly decides the shape of a local MDP.

The Lookup Algorithm presented below is an attempt to take advantage of Theorem 3 to approximate the values of the local MDP without knowing the remote policy.

ALGORITHM 2. **Lookup Algorithm**

1. *Solve the underlying centralized MDP for the given DEC-MDP, build a lookup table of the optimal state value of each state $V^{c*}(s)$ (see Definition 10). The state values are indexed by the value MDP the states belong to.*
2. *When reaching a new observation sequence $\overline{o}_i$, choose the action according to the following equation, where s' is the next state of s in the decentralized MDP after taking action a.*

$$\pi_i(\overline{o}_i) = argmax_a(R(a) + \sum_{s\in S} P(s|\overline{o}_i)V^{c*}(s')) \quad (3)$$

This algorithm makes use of the simple structure of the decentralized MDP. As we discovered before, the decentralized MDP is mostly deterministic, which makes the underlying centralized MDP very easy to solve. The Lookup Algorithm utilizes the optimal state values of the underlying centralized MDP as a guidance to approximate the globally optimal solution. According to Theorem 3, (3) is trying to maximize a very optimistic approximation of $V_{<\pi_1,\pi_2>^*}(\overline{o}_i)$, since $V^{c*}(s')$ is an optimistic estimate of $V_{<\pi_1,\pi_2>^*}(s')$. Nevertheless, without knowing what the optimal solution is, it is still a fairly good estimate. The main merit of the algorithm is the simplicity of the heuristic. If along with the state value lookup table, we also store the prior probability of each value combination of the data set, then even $P(s|\overline{o}_i)$ can be easily calculated without resorting back to the original BN. This algorithm can be executed online efficiently after the lookup table is constructed. A simple algorithm described in [7] uses the same idea to solve POMDP.

5. EXPERIMENTAL RESULTS

We implemented both approximate algorithms described above. We have run experiments on 100 problem structures with 2 high level events and 10 low level data (5 local to each agent) for different confidence levels. All the networks are fully connected, which means that for both agents to have the complete evidence, 10 pieces data need to be transmitted. We also implemented the exhaustive search algorithm to generate the optimal joint policy for the problem. Figure 5 shows the comparison of the expected communication cost of the communication strategies generated by the three algorithms. On the problem sets we ran experiments on, the two approximate algorithms performed surprisingly well, with an expected communication cost only slightly higher than that of the optimal solution. In fact, for about 95% of the problems, both of the two algorithms generated the optimal solution. We have done further experiments on larger networks with 4 high level events and 20 low level data and found similar results.

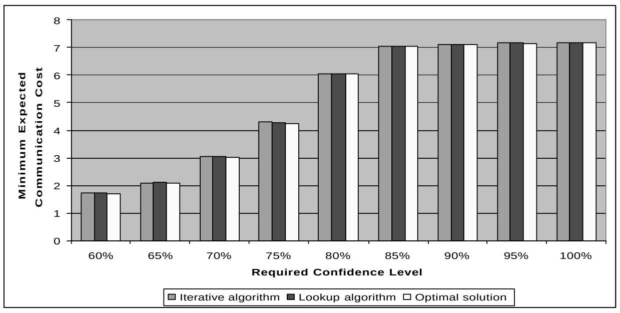


Figure 5: The comparison of the expected communication cost of the joint policies generated by the two approximate algorithms and the optimal solution.

It is interesting to observe how the minimum expected communication cost changes when the required confidence level increases. At first, increasing the confidence does not require much increase in the communication cost, while later the same amount of improvement in the confidence needs a lot more communication. When the confidence reaches a certain level, no further communication is required for improvement. This is understandable because the critical data are chosen to be transferred first, which has the most impact on increasing the confidence. Later, more non-critical data needs to be communicated to gain the same amount of improvement. Finally, the comparatively irrelavant data will not contribute a lot to improving the solution quality.

The framework we have presented can also be used to analyze the suitability of a problem to be solved by a distributed system. Figure 6 shows the minimum expected communication cost curve of two different problems. Problem structure 1 is much more suitable for a distributed solution than problem structure 2 since it needs comparatively little communication to achieve high confidence level. Further experiments and theoretic work need to be done to help design suitable systems for different problems with different communication properties.

6. CONCLUSION

There is beginning to be research applying the decision-theoretic framework to multi-agent systems. The Multi-agent Markov decision process (MMDP) defined in [3] is a centralized multi-agent extension of an MDP. In that work, each agent observes the global state directly and takes an joint action. [11] proposes a decentralized extension, in which an agent has its local state in addition to the global

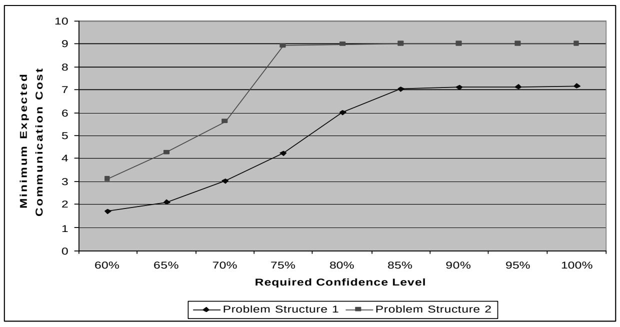


Figure 6: The minimum expected communication cost required by two different problem structures.

state. When an agent has ambiguity over which action it should take based on its local state, it needs to communicate with the other agent to synchronize the global view. Our work is unique in the way that the agents cannot simply observe the global state directly or even synchronize to it occasionally. In fact, this inability to observe the global state at any time is the key factor that leads to the difficulty of the problem.

Modeling the problem structure as a BN is also unique. By utilizing the information provided by the BN structure, we are able to build a complete model of the DEC-MDP and calculate the local agent's belief in the global state and the other agent's local state. In this respect, our model of the local MDPs resembles a POMDP. In fact, the techniques of solving a DEC-MDP presented in this paper can be used for general DEC-MDPs as long as they have a complete model. That is exactly what the BN representation achieves.

[8] proposes a general framework (COM-MTDP) which extends the DEC-MDP to incorporate communication actions as well as domain actions. This framework provides an excellent foundation for theoretical analysis of different coordination strategies in the teamwork context, but does not provide a practical algorithm for approximate or optimal solutions. In contrast, our framework is designed for the distributed systems that can be represented by a BN structure and we are modelling only the communication of the system. This enables us to find the optimal or near-optimal solutions for a given problem.

The current BN structure we are working with is comparatively small and simple. To effectively model real world applications we need to scale along several dimensions: number of agents, number of events, number of sensor data elements, and the layers of intermediate results between the events and the raw data. In order to incorporate these changes, we will need to extend the framework presented in this paper. In addition, scaling will inevitably increase the state and action space of the DEC-MDP under the current mapping, and therefore we plan to investigate techniques like heuristic search and state abstraction, which have proven useful in large single-agent problems [6].

Another way to handle scaling is to change the way we map the BN problem structure to DEC-MDP. In our current mapping, the state space and action space of the DEC-MDP are exponential to the amount of data in the BN. It is computationally infeasible to generate a large DEC-MDP when the quantity of raw data grows. One approach we are looking at is to extend the BN to multiple layers, where the intermediate nodes represent the intermediate causes. Instead of transmitting raw data directly, the agents will abstract the information contained in the raw data into the intermediate nodes and communicate them.

7. REFERENCES

[1] R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman. Transition-independent decentralized markov decision processes. In *Proceedings of the Second International Joint Conference on Autonomous Agents and MultiAgent Systems*, 2003, to appear.

[2] D. 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, November 2002.

[3] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 1999.

[4] N. Carver and V. Lesser. Domain monotonicity and the performance of local solutions strategies for CDPS-based distributed sensor interpretation and distributed diagnosis. *International Journal of Autonomous Agents and Multi-Agent Systems*, 6:35–76., 2003.

[5] I. Chades, B. Scherrer, and F. Charpillet. A heuristic approach for solving decentralized-POMDP: Assessment on the pursuit problem. In *Proceedings of the 17th ACM Symposium on Applied Computing (SAC 2002)*, pages 57–62, Madrid, 2002.

[6] Z. Feng and E. Hansen. Symbolic heuristic search for factored markov decision processes. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 455–460, Edmonton, Alberta, Canada, July 2002.

[7] M. Littman, A. Cassandra, and L. Kaelbling. Learning policies for partially observable environments: Scaling up. In A. Prieditis and S. Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning*, pages 362–370, San Francisco, CA, 1995. Morgan Kaufmann.

[8] D. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of AI Research (JAIR)*, 2002.

[9] J. Shen, V. Lesser, and N. Carver. Controlling information exchange in distributed bayesian networks. Technical Report 02-22, University of Massachusetts, 2002.

[10] Y. Xiang. A probabilistic framework for multi-agent distributed interpretation and optimization of communication. *Artificial Intelligence*, 87(1–2):295–342, 1996.

[11] P. Xuan and V. Lesser. Multi-agent policies: from centralized ones to decentralized ones. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, Bologna, Italy, 2002.