Reasoning about Remote Data in CDPS with Distributed Bayesian Networks *

Jiaying Shen[1], Victor Lesser[1], and Norman Carver[2]

[1] 140 Governor's Drive, Department of Computer Science,
University of Massachusetts, Amherst, MA 01003-4610, USA
{jyshen, lesser}@cs.umass.edu
[2] Computer Science Department, Southern Illinois University, mailcode 4511,
Carbondale, IL 62901, USA
carver@cs.siu.edu

Abstract. Existing Cooperative Distributed Problem Solving systems frequently employ fixed coordination strategies to achieve global consistency or global optimality. However, these strategies generally do not exploit the characteristics of the particular problem they are used on. In this paper we propose an algorithm that given a problem formulated as a Distributed Bayesian Network, finds a coordination strategy which minimizes the communication costs while achieving the desired confidence level of the global solution. We develop a system based on this algorithm which models the communication decision process for any given problem structure as a Markov Decision Process and use dynamic programming to produce the optimal communication strategy.

1 Introduction

Cooperative Distributed Problem Solving (CDPS) is a major focus of research in Multi-Agent Systems (MAS). CDPS studies how large scale problems can be solved using a group of agents working together. Local solutions are merged together to form a global solution by mainly communicating the high level information [2]. However, if the local solutions are not consistent or not sufficiently credible, further communication is needed to resolve the conflict. As a result, one of the key issues is to design an algorithm that manages the agent communication to achieve the desired global solution quality. By solution quality we mean the likelihood that a solution will be the same as what we will get from a centralized system where all information is available.

Most approaches to managing communication trade off solution quality for reducing communication, but only from a statistical view. The behavior of the algorithms are often analyzed over an ensemble 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 [1].

* This material is based upon work supported by the National Science Foundation under Grant No. "NSF/CSS" (w/Carver@SIU); eff. 9/1/00-8/31/03: IIS-0004112

We would like to take this satisfaction approach to the next step by seeing whether we can design a parameterized algorithm where we can predict, for a fixed amount of communication, the maximum level of confidence we expect in the final solution. Or conversely, given a desired confidence level in the final solution how much communication the agents need. Finally, the algorithm should produce a communication strategy that will require only the minimum amount of communication to still achieve the desired solution quality.

We will study these issues in terms of Distributed Bayesian Networks (DBN). Recent work includes algorithms such as in Xiang [3] that produce the same final solution in CDPS as that is generated by a centralized problem solving system. However, this approach can potentially require significant communcation.

In our research, we use two-layer DBN to represent the underlying structure of the problem that needs to be solved [1]. For our problem, we make the following assumptions:

(1) There are two agents in the system.
(2) Every agent has access to the complete DBN.
(3) Evidence is distributed among the agents.
(4) Each agent knows what evidence the other agent has access to.

Bayesian Networks are a powerful tool to calculate conditional probabilities, and we have developed an algorithm that can reason about remote data using DBNs. Without exchanging any information at all, an agent can use our algorithm to compute the likelihood of a hypothesis H being the globally optimal solution based on its local data and direct the agent to ask for critical information from the remote agent in order to reach a higher level of confidence in the global solution. On the simple examples we constructed, the algorithm works well in reducing the communication cost.

We are now implementing a system based on this algorithm. With this system, given any DBN problem structure an agent will be able to dynamically construct a Markov Decision Process (MDP) and learn the optimal policy in terms of what data to ask for from the remote agent and in what order. With the smart conversation thus carried out by the agents, we expect the communication cost of the system to be significantly reduced.

2 Reasoning about Remote Data

In CDPS systems, a problem is decomposed into a set of subproblems and each subproblem is distributed to an agent who will be responsible for solving the subproblem. The existence of interactions between subproblems means that CDPS agents cannot simply solve the subproblems individually and then combine local solutions together. To ensure that the local solutions are globally consistent they must communicate during the problem solving process. A coordination strategy is needed to specify how agents will interact: when they will communicate, and what information they will send or request.

One of the common coordination strategies used in CDPS systems is Consistent Local Solutions Strategy (CLSS) [2]. According to this strategy, agents first independently solve their local subproblems and then transmit their local solutions to all other agents. If these agents' local solutions are consistent with each other, they are merged without further verification of what the globally optimal solution is. If the local solutions are not consistent, lower level results/data is transferred to ensure that the local solution chosen are consistent.

There are several key problems with CLSS. First, there is no way to ensure that the solution chosen is the globally optimal solution or that it has reached the desired confidence level. Secondly, there can be significant delays in problem solving when agents require substantial amounts of raw data from other agents. Hence, we propose the idea of transmitting data incrementally until sufficient confidence in the current best solution is reached. By "incrementally" we mean that not all of the local raw data is transferred at once. Instead, it is transmitted as needed until global consistency is achieved. This raises another question: in what order should the raw data be transmitted when it is necessary?

Instead of giving a one-size-fits-all strategy, we are trying to design an algorithm that can produce a strategy for any given problem structure that requires as little communication cost as possible to achieve the desired confidence level of the final global solution.

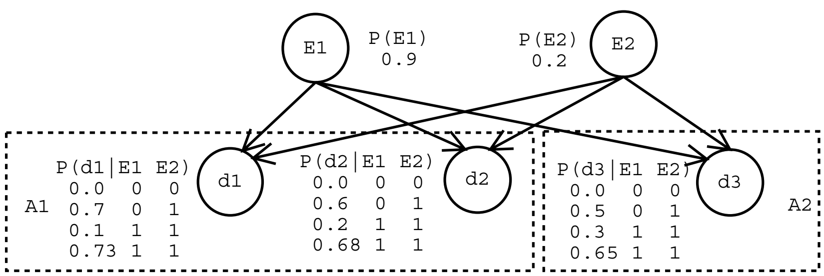

Fig. 1. An example of DBN problem structure. There are two events E_1 and E_2. Data d_1, d_2 and d_3 are distributed to two agents. A_1 has access to d_1 and d_2, while A_2 can only see d_3. The objective is for A_1 and A_2 to figure out what E_1 and E_2 are without too much communication.

We use a two-layer Bayesian Network to represent the problem structure (Figure 1). The top level nodes are the events that are the cause of the observed data, while the leaves are the raw data gathered, which are distributed to various agents. The objective of the CDPS system is to figure out the likelihood of events E_1 and E_2 without significant communication.

Notation 1 *Agent A_i. In Figure 1, there are two agents $\{A_i|i = 1, 2\}$.*

Notation 2 *Event E_i. The possible events in the environment which caused the observed data. For example, in Figure 1 there are two possible events $\{E_i|i = 1,2\}$.*

Notation 3 *Data d_i. The data observed by agents. In Figure 1, we have 3 data $\{d_i|1 \le i \le 3\}$.*

Notation 4 *Hypothesis H_i. The possible hypotheses the agents might draw from the data. Normally they are possible configurations of the events. For example, in Figure 1 we have four possible hypotheses $\{H_i|1 \le i \le 4\}$, as shown in (1).*

$$\begin{array}{ccccc} & H_1 & H_2 & H_3 & H_4 \\ E_1 & 1 & 1 & 0 & 0 \\ E_2 & 1 & 0 & 1 & 0 \end{array} \tag{1}$$

Notation 5 *Evidence ϵ_i. The data set possibly observed by one agent. They are possible configurations of the data set of the agent. For example, in Figure 1 agent A_1 has four possible evidence configuration $\{\epsilon_i|1 \le i \le 4\}$, as shown in (2).*

$$\begin{array}{ccccc} & \epsilon_1 & \epsilon_2 & \epsilon_3 & \epsilon_4 \\ d_1 & 1 & 1 & 0 & 0 \\ d_2 & 1 & 0 & 1 & 0 \end{array} \tag{2}$$

Notation 6 *Actual evidence ϵ_{A_i}. Every agent can only observe one data value configuration from the possible evidence set. In Figure 1, A_1 can only observe $\epsilon_{A_1} \in \{\epsilon_i|1 \le i \le 4\}$.*

Notation 7 *Confidence C_{A_i}. Based on the evidence observed, an agent can calculate the conditional probabilities of every hypothesis based on the evidence, i.e., $C_{A_i}(H) = P(H|\epsilon_{A_i})$.*

Let us now assume that A_2 has not received any information from A_1. With the knowledge of its own data and the Bayesian Network structure, A_2 can still do some reasoning to decide what information it needs to determine the globally best hypothesis. We illustrate this with an example.

In our example, A_2 has access only to $d_3 = 0$ and the BN structure shown in Figure 1. Does A_1 need to send all the data? What information should A_2 request from A_1? Can A_2 save communication cost by requesting only the necessary data? Naturally, A_2 will put itself in the place of A_1 to reason about A_1's data. It calculates the probabilities of the four possible evidence of A_1 given $d_3 = 0$ as follows:

$$\begin{aligned} &P(\epsilon_1|d_3 = 0) = 0.0679,\ P(\epsilon_2|d_3 = 0) = 0.0987 \\ &P(\epsilon_3|d_3 = 0) = 0.1632,\ P(\epsilon_4|d_3 = 0) = 0.6701. \end{aligned} \tag{3}$$

A_2 can also calculate the probabilities of the four hypotheses assuming the evidence of A_1, i.e. $P(H_i|\epsilon_j \wedge d_3 = 0), 1 \le i \le 4, 1 \le j \le 4$ as in Table 1.

	ϵ_1	ϵ_2	ϵ_3	ϵ_4
H_1	0.872	0.576	0.346	0.061
H_2	0.065	0.364	0.625	0.932
H_3	0.063	0.059	0.029	0.007
H_4	0	0	0	0

Table 1. Reasoning about what to request: $P(H_i|\epsilon_j \wedge d_3 = 0)$

Notation 8 *Compound Probability $CP(\epsilon_i|\epsilon_A)$ is a pair $(P(\epsilon_i|\epsilon_A), H_j)$, where $j = maxarg_j P(H_j|\epsilon_i \wedge \epsilon_A)$. It indicates that given the local evidence ϵ_A, with probability of $P(\epsilon_i|\epsilon_A)$, H_j is the best hypothesis, where ϵ_i are all the possible evidence of the remote agent.*

In our example, from (3) and table 1, A_2 has the compound probabilities of the four possible remote evidence as follows:

$$\begin{aligned} &CP(\epsilon_1|d_3 = 0) = (0.0679, H_1),\ CP(\epsilon_2|d_3 = 0) = (0.0987, H_1) \\ &CP(\epsilon_3|d_3 = 0) = (0.1632, H_2),\ CP(\epsilon_4|d_3 = 0) = (0.6701, H_2). \end{aligned} \tag{4}$$

It is obvious that based on A_2's data, H_2 is the most probable hypothesis. However, from (4), we can see that for ϵ_1 and ϵ_2, H_1 is the best hypothesis, while for ϵ_3 and ϵ_4, H_2 is. The conclusion is thus: with probability of $P(\epsilon_1|d_3 = 0) + P(\epsilon_2|d_3 = 0) = 0.1666$, H_1 is the globally optimal hypothesis, and with probability of $P(\epsilon_3|d_3 = 0) + P(\epsilon_4|d_3 = 0) = 0.8333$, H_2 is the globally optimal hypothesis. So, we are able to collapse (4) into:

$$\begin{aligned} &CP(\epsilon_1 \vee \epsilon_2|d_3 = 0) = (0.1666, H_1) \\ &CP(\epsilon_3 \vee \epsilon_4|d_3 = 0) = (0.8333, H_2). \end{aligned} \tag{5}$$

If we collapse (2) into:

$$\begin{array}{cc} & d_1 \\ \epsilon_1 \vee \epsilon_2 & 1 \\ \epsilon_3 \vee \epsilon_4 & 0 \end{array}, \tag{6}$$

it is easy to see that what makes the difference of choosing H_1 or H_2 as the globally optimal hypothesis is d_1. Consequently, A_2 will be able to determine the globally optimal hypothesis if A_1 sends it the observed data d_1. Based on this knowledge, A_2 only needs to ask A_1 for d_1 instead of both d_1 and d_2. This saves communication cost.

What is more, from (5), we can see that with probability of 0.8333, H_2 is the globally best hypothesis. As a result, if we only need to reach the confidence level of 80%, A_2 does not even need to request A_1 to send it data d_1. Only when the confidence requirement is above 83%, data request is needed. The compound probabilities enables us to see what data to request to reach the desired confidence level and only communicate as little as needed.

Now we can summarize the algorithm as follows:

Algorithm 1 1. Calculate the probabilities of the possible evidence of the remote agent A_1 based on the local data, i.e. $P(\epsilon_i|\epsilon_{A_2})$.
2. Calculate the probabilities of the hypothesis of assuming the remote evidence, $P(H_i|\epsilon_j \wedge \epsilon_{A_2})$.
3. Calculate and collapse the compound probabilities.
4. Group the evidence according to the most probable optimal hypothesis, and collapse the evidence table.
5. Request data according to the collapsed evidence table and the required confidence level of the final solution.

The advantage of Algorithm 1 is evident. With CLSS, A_1 will have to transmit both raw data d_1 and d_2 to get a globally consistent solution, while using Algorithm 1, we need to transmit only d_1 to ensure the globally optimal solution and there is no communication at all to reach the confidence level of 80%.

One thing worth noting is that we are assuming that only A_2 is doing the reasoning in our algorithm. This is reasonable since there will often be one agent who is responsible for assembling the global solution. A more interesting case is when A_1 is simutaneously reasoning about what data it should provide to A_2.

3 Communication Strategy System

Algorithm 1 has answered the question of what to request (communicate). Now, we have another equally important question to answer: if there is more than one piece of critical data, in what order and combination the data should be transmitted to minimize the communication cost. This is essentially a solution to step 5 of the algorithm.

To answer this question we frame the problem as an MDP and use dynamic programming to find the optimal communication strategy. Each state of the MDP includes the current known remote data set, the current best solution and its compound probability, i.e.,

$$\begin{aligned} S \; : \; & D_{known-remote}, CP(\epsilon_{unknown-remote}|\epsilon_{known-remote} \wedge \epsilon_{local}) \\ & = D_{known-remote}, (P(\epsilon_{unknown-remote}|\epsilon_{known-remote} \wedge \epsilon_{local}), H) \end{aligned} \quad (7)$$

where $D_{known-remote}$ is the known remote data set, $\epsilon_{unknown-remote}$ is the unknown remote evidence with the highest $P(\epsilon_{unknown-remote}|\epsilon_{known-remote} \wedge \epsilon_{local})$, and H is the corresponding hypothesis and hence the current best solution. The action set of the MDP is all the possible combinations of the critical data. The cost of each state-action pair is the amount of communication needed to take this action in this state. We assume that the cost of a request message is 1 no matter how many data are requested, and each data transmitted costs 1. The MDP starts at the state where no remote data is known and the best global solution is based on its own local information. It stops when the desired confidence level is reached.

As an example, we will construct an MDP (Fig 2) for (8). Please note that (8) is different from (4). In (8) the best hypothesis for ϵ_4 is H_3 instead of H_2.

All the other assumptions are essentially the same as the previous example. The existence of H_3 makes our example sufficiently complex to illustrate the algorithm.

$$\begin{aligned} &CP(\epsilon_1|d_3=0)=(0.0679,H_1),\ \ CP(\epsilon_2|d_3=0)=(0.0987,H_1)\\ &CP(\epsilon_3|d_3=0)=(0.1632,H_2),\ \ CP(\epsilon_4|d_3=0)=(0.6701,H_3). \end{aligned} \tag{8}$$

Without knowing any remote data, A_2 may determine that the best global solution is H_3 with the confidence of 0.67. Going through Algorithm 1, it decides that the critical remote data set is $\{d_1, d_2\}$. As a result, the action set for A_2 is $\{(d_1), (d_2), (d_1, d_2)\}$. If it takes the action (d_1, d_2), it will get the globally optimal solution no matter what the reply is. If it asks for d_1, with a probability of $0.0679 + 0.0987 = 0.1666$, $d_1 = 1$ and H_1 is the globally optimal solution with confidence of 1. If $d_1 = 0$, the remote data can be either ϵ_3 or ϵ_4. As a result, A_2 can decide only that H_3 is the best solution with confidence of $0.6701/(0.1632 + 0.6701) = 0.8042$. If it still wants to improve its confidence level, it will need to take further action, asking for d_2, after which it can draw the best conclusion with full confidence. The cost for path $S0 \rightarrow S3$ is 3 and for path $S0 \rightarrow S1$ is 2, while the cost for path $S0 \rightarrow S2 \rightarrow S3$ is 4. Applying Dynamic Programming to this MDP, to achieve confidence level of 80%, the best strategy is for A_2 to ask for only d_1, while to achieve confidence level of 100%, A_2 should ask for both d_1 and d_2 at once.

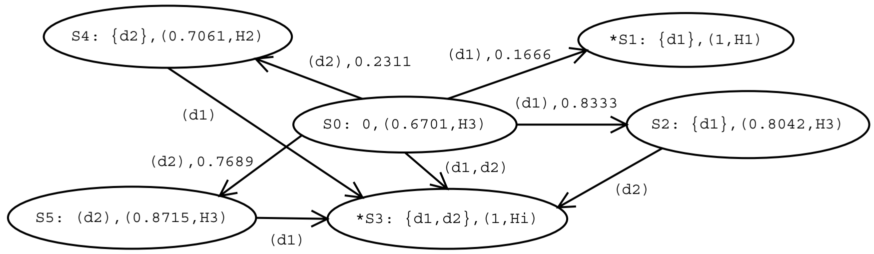

Fig. 2. Framework of the Communication Strategy System

We are now implementing a system to dynamically construct an MDP for any given problem structure based on this algorithm. The general framework of this system is illustrated in Figure 3. The input of the Communication Strategy System is the problem structure represented in the form of a DBN, and the output is the optimal communication strategy the agent should deploy. The BN toolkit is used to calculate all the necessary conditional probabilities (steps 1–3 in Algorithm 1) and the decision tree data structure is employed to collapse the truth table and find data critical to the globally optimal solution according to Algorithm 1 (steps 3 and 4). The system then uses dynamic programming to produce the optimal communication action sequence for the MDP constructed this way(step 5).

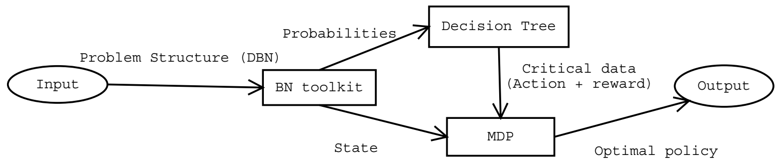

Fig. 3. Framework of the Communication Strategy System

4 Future Work

Once the system is implemented and data sets are collected from experiments, we plan to compare the results with the data collected from a previous system [1] without communication planning. Some formalization based on the statistics result will also be done to predict the amount of information that needs to be exchanged to reach a certain level of confidence. Furthermore, we will try to apply some approximation techniques to reduce the computational complexity inherent to Bayesian Networks. Dynamic programming will guarantee the optimality of the solution, but it is also time consuming and computationally expensive. We are considering applying some reinforcement learning techniques such as Q-learning to approximate the optimal policy. We will also try to scale our algorithm to larger DBNs, to more agents and to multi-level networks.

The reasoning about what data to request gives us some insight into the relation between the confidence level in the hypothesis and the communication needed. In Carver [1], we have seen some experimental results on this relation in the context of different near-monotonicity levels. This work may help us explain those results. We are currently looking into the relation between the near monotonicity measures used by Carver [1] and the method used here, in hopes of finding such explanations.

References

1. Norman Carver, Victor Lesser, Domain Monotonicity and the Performance of Local Solutions Strategies for CDPS-based Distributed Sensor Interpretation and Distributed Diagnosis, *to appear in International Journal of Autonomous Agents and Multi-Agent Systems.*
2. Norman Carver, Victor Lesser, and Robert Whitehair, Nearly Monotonic Problems: A Key to Effective FA/C Distributed Sensor Interpretation?, *Proceedings of AAAI-96*, 88–95, 1996.
3. Yang Xiang, A Probabilistic Framework for Multi-agent Distributed Interpretation and Optimization of Communication, *Artificial Intelligence*, vol. 87, no. 1–2, 295–342, 1996.