Multi-Agent Policies: From Centralized Ones to Decentralized Ones *

Ping Xuan and Victor Lesser
Department of Computer Science
University of Massachusetts at Amherst
Amherst, MA 01003
pxuan@cs.umass.edu, lesser@cs.umass.edu

ABSTRACT

In this paper we divide multi-agent policies into two categories: centralized ones and decentralized ones. They reflect different views of multi-agent systems and different decision-theoretic underpinnings. While the centralized policies specify the decision of the agents according to the global system state, the decentralized policies, which correspond to the decisions of situated agents, must assume only a partial knowledge of the system in each agent and must deal with communication explicitly. In this paper we relate these two types of policies by introducing a formal and systematic methodology for transforming centralized policies into a variety of decentralized policies. We introduce a set of transformation strategies, and provide a representation for discussing decentralized communication decisions. Through our experiments, we show that our methodology enables us to derive a class of interesting policies that have a range of expected utilities and amount of communication, and allows us to gain important insights into decentralized coordination strategies from a decision-theoretic perspective.

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

*Effort sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory Air Force Materiel Command, USAF, under agreement number F30602-99-2-0525 and by the National Science Foundation under Grant number IIS-9812755. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), Air Force Research Laboratory, National Science Foundation, or the U.S. Government.

General Terms

Algorithms, Theory

Keywords

action selection and planning, coordinating multi agents & activities, agent comm languages and protocols, methodologies and tools, Cooperation

1. INTRODUCTION

The problem of generating plans is a key problem in multi-agent collaboration. This is a highly active field of research, and numerous approaches have been proposed in the past. An important task is to develop decision-theoretic methods so that we can have frameworks to describe, analyze, and understand planning strategies [2, 9]. Based on the different views of the system used in the planning approaches, we can categorize them into two classes: one that uses a centralized view, and one that uses a decentralized view. Typically, the centralized view is the view taken by the designer of the system. The behavior of the agents are often considered collectively, in terms of joint intentions [5] and joint actions. There, a plan, which specifies the problem solving strategy of the agents, is often described through a mapping from the global system states to the joint actions of the agents. We call this a centralized multi-agent policy, or CP in short. In contrast, the decentralized view is often the view of a situated agent in the system, which deals with a partial observation of the global system state and makes local decisions. There, a plan is often a mapping from local beliefs to local actions, and we call this a decentralized policy, or DP for short. Clearly, in this decentralized view each agent need its own DP, unlike in the centralized view where one CP specifies the joint actions of the agents.

In decision-theoretic terms, the centralized view of the system can be described by a multi-agent Markov decision process model, as proposed in [2]. This is a standard Markov decision process [7] that consists of a set of global states S, a set of joint actions A (each joint action specifies one action for each agent), a transition probability matrix $Pr(s'|s, a)$ – the probability that the system moves from state s to state s' after joint action a, and a reward function r(s) that specifies the global utility received when the system is state s. This framework corresponds to a computational model where at any stage t, the system is described through its current global state s, which is made up of the current local states of the agents. The system then takes a joint action,

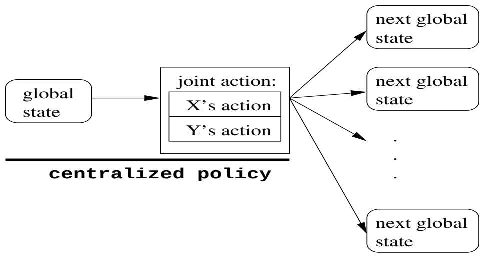

Figure 1: Centralized View and Policy

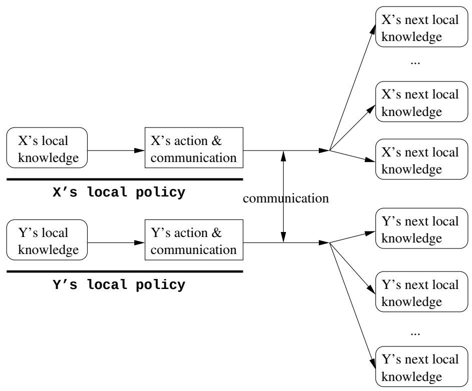

Figure 2: Decentralized View and Policy

and evolves into one of the possible next global states based on the completion of the joint action. In this framework, a centralized policy (CP), which is a mapping from global states to joint actions, is precisely a policy for a Markov decision process. Figure 1 shows how problem solving is carried out under such a policy (in a two-agent system with agents X and Y). The expected utility of the CP can be calculated using a standard policy evaluation algorithms for Markov decision processes.

In the decentralized view, an agent cannot see other agents' local states and local actions, and has to decide the next local action on its own. Thus, each agent has only a partial view of the system's global state, and different agents have different partial views. Of course, this does not necessarily mean that the agents are isolated. Rather, an important ability of decentralized cooperative agents is their ability to communicate. We view communication as a way of expanding an agent's partial view by exchanging local information not observed by other agents. For simplicity, we define communication as the action of agents updating each other with local state information and thus collectively discovering the current global state (in other words, the agents synchronize themselves so that they all observe the current global state). Clearly, a DP has to deal with communication explicitly, in particular when communication incurs a cost, or when continuous communication is not feasible. In [9], we proposed a decentralized multi-agent decision process framework that provides a basis for decision-theoretic study of decentralized policies. The system is modeled by each agent having an individual state space, its own local action set, and local state transition probability measure, but uses a global reward function to connect the effects of the actions in the agents. Thus, it is not a standard Markov decision process. The DP under this framework now also explicitly includes communication decisions. Specifically, the DP should define not only what local action to take based on the current local knowledge of the agent, but also whether communication is needed after its local action is completed. Typically, such decisions are not made solely based on current local state information but need to include an agent's history information, including past states and past communications. In general, we are dealing with history-dependent policies when it comes to DP, whereas typically we need only deal with history-independent, or Markovian, policies, when studying CPs.

Figure 2 shows the computation model under this decentralized view: at any stage t, each agent first decides what local action to take (per the DP). Then, decides whether there is need for communication when the action finishes. Next, the agents enter a sub-stage where all communications occur (if any). When the communication sub-stage finishes, each agent enters the next stage with an updated set of local knowledge (one of several possible local knowledge sets based on the outcomes of actions in both agents).

The key difference between a CP and a DP is that a CP assumes the global state as the starting point but in DP the global state is not automatically observed by the agents. As a result we need to explicitly incorporate communication decisions, and deal with history information in the DP. This makes DP much more complex than CP. However, most multi-agent systems are distributed in nature, and agents are generally autonomous — meaning that each agent is a decision maker on its own. Thus, taking a centralized view in such a system would often oversimplify the problem (e.g., assuming the agents see the global view instead of the partial view), and a centralized policy would not be implementable by situated agents without imposing some strong assumptions or special mechanisms to ensure the observability of the global system state. Obviously, DPs are the right form of policies for each agent to carry out the plan. Hence, we need to develop DPs so that the agents with partial views can effectively perform actions and implement cooperative problem-solving strategies.

Of course, this does not mean that centralized policies are invalid. In fact, CPs and DPs are very much related. They simply represent solutions rooted from different perspectives. Decentralized models have an advantage in their representational power, but at the same time suffer from their complexity when obtaining DPs. Typically, solving a standard Markov decision process is of PSPACE complexity [6], but solving a decentralized Markov decision process is of NEXP-time complexity [1], a higher complexity class. As a result, heuristic approaches and approximation methods for developing DPs are extremely important. However, due to the infancy of research into this question, only a number of simple heuristic approaches have been studied so far [9]. These approaches are often quite domain-specific and therefore cannot be easily extended to derive general solution methodologies. On the other hand, CPs are easier to solve and there are systematic methods for obtaining them. Thus, it is quite desirable to find ways to derive DPs directly from CPs. In this paper, we provide the connection between them by providing ways of implementing a centralized policy in a decentralized system. CPs can be transformed to DPs and adopted by the decentralized agents. Also, we argue that the decomposition of a CP should also take into account the cost of communication, since agents may have to consider whether communication (to obtain more information) is worthwhile. Such communication decisions are not

$$\mathrm{CP} \longrightarrow \begin{pmatrix} \mathrm{DP}^X \\ \mathrm{DP}^Y \end{pmatrix}$$

Figure 3: Centralized Policy to Decentralized Policy

considered in CPs. In addition, by solving this plan decomposition problem we may also gain important insights about the nature of plan interdependency, and provide feedback to the research on CPs by studying how feasible or effective a CP is when implemented in decentralized systems.

For a two-agent system containing agents X and Y, this transformation is illustrated in Figure 3. Note that the decentralized policy consists of a separate policy for each agent (what we call *local policies*) - in our case, two local policies, one for agent X (DP^X) and one for agent Y (DP^Y). Together they form a complete DP derived from the CP.

The rest of the paper is organized as the following: in the next section we describe our transformation approach, followed by the choice to communication policies, and how to evaluate a derived DP, and then move on to the issue of non-conforming DPs. In section 6 we discuss our experimental results and section 7 summarizes our work.

2. TRANSFORM CP TO DP

Now we discuss how to transform a given CP to DPs. For clarity, we assume that the each agent has complete knowledge of the CP and also knows the dynamics of the state space. A trivial transformation is to let the agents communicate at all stages, therefore maintaining the observation of the global state all the time in all agents, i.e., establishing a centralized view in each agent. However, our interest is on how to minimize communication. Intuitively, an agent does not need to communicate if it knows precisely what is the next local action it needs to perform. If it knows that the actual global state is one of several possible states *and* all the possible states indicate the same *local* action for this agent (not necessarily the same joint action), then the agent may choose not to communicate. In this case, not knowing the exact global state will not affect its choice of actions.

As such, the agents' problem solving episode can be characterized as a series of non-communicating intervals. At the beginning of each interval, agents are synchronized and know the global state. The agents then remain silent in the next stages until one of them (or both) has insufficient knowledge to decide its local decision. At that time, they communicate (initiated by the agent with insufficient information). After communication, they are again synchronized with the global state. We assume that communication is without error or loss.

The set of possible current global states is what we call the *local belief set*, or *LB* in short. The problem is, each agent's local belief may be different. The intersection of the local beliefs in different agents uniquely defines the global state, but an agent cannot observe the local belief of other agents. Therefore, we need to introduce a consensus belief set for all the agents. This is what we call a *common belief state* (or simply *belief state*), a set that consists of possible current global states calculated based on common knowledge only. The details of common belief states follow shortly. Figure 4 shows a common belief state consists of four states when agent X finishes action a and agent Y finishes action b. The current local belief of an agent is then a projection

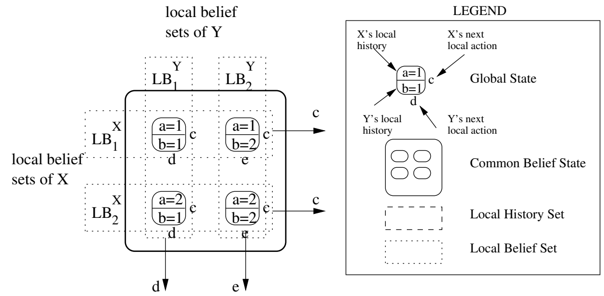

Figure 4: Common Belief State

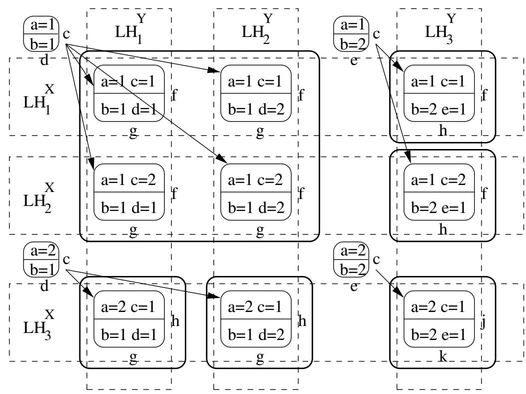

Figure 5: Possible Next Common Belief States

of the common belief state by applying the additional local knowledge of the agent. Figure 4 shows the possible LB sets LB_1^X and LB_2^X for X (according to the outcome of a, i.e., whether a=1 or 2), and similarly LB_1^Y, LB_2^Y for Y.

The local action for each of the LB set is unique: the next action is c for agent X regardless of the outcome of a and b, and action d (if $b = 1$) or e (if $b = 2$) for agent Y regardless of the outcome of action a. Thus, both agents know which action to perform, no matter which the actual LB sets are. The agents then perform their local actions. Now the question is how to update the common belief state (and the LB set too) when the action finishes. Let B be the current belief state, and N be the set of possible next states (when the actions finish) for all states in B. Figure 5 illustrates the N set (9 states) of the B shown in Figure 4, with the arrows showing the next states for each state in B. Note that the possible outcomes of an action may change according to the history, for example c may have outcome 1 or 2 when a=1, but must be 1 when a=2. Now we introduce another concept, the *local history* sets, or LH sets. For each agent, LH sets are projections of N according to local outcome history. In Figure 5, there are 3 LH sets for each agent, shown as LH_i^X and LH_j^Y. Each LH set has a different outcome history (a=? c=? for agent X and b=? d=? for agent Y). For example, the 3 states in LH_1^X all have local state sequence a=1 and c=1 for agent X. Each LH reflects the possible global states when the agent observes the local action outcome (in our example, after c finishes in X and d finishes in Y), but before the communication substage. The states in a LH are locally indistinguishable by agent X (or Y) before communication.

LH sets are the key for defining the communication decisions of the agents. An LH set summarizes all local knowledge that can be used to decide the possible current global state. The communication policy of each agent is a mapping from its partitioned LH sets to the set $\{yes, no\}$. The mapping is not from individual states because all states in an LH set correspond to the same local history and as a result the agent cannot distinguish them when making communication decisions. An agent would communicate if the decision concerning its current LH set is *yes* and not communicate otherwise. If *any* agent communicates, all of the agents will discover the global state after the communication sub-stage. Thus, if the actual global state is s, which is the joint of LH_i^X and LH_j^Y, and at least one of LH_i^X and LH_j^Y maps to *yes*, s will be discovered via communication, and therefore the common belief state — the consensus of all agents regarding the possible current global states — would be updated to a singleton set just containing s. Let K be the subset of N that contains all such s that could be discovered after communication. If the actual global state $s \notin K$ (meaning both LH_i^X and LH_j^Y maps to *no*), then the common belief state at the next stage, by way of elimination, would be the set N-K. In Figure 5, let us assume that the communication policy is to map LH_3^X and LH_3^Y to *yes* and all other LH sets to *no*. Thus, the set K is the union of LH_3^X and LH_3^Y, and the remaining 4 states form the set N-K. As a result, depending on the actual s, there are 6 possible common belief states at the next agent: the set N-K, plus one singleton common belief state for each state in K.

Now that we know how the agents calculate the common belief state in the next stage (given the current common belief state), we have a working definition of the common belief state. Intuitively, the common belief state is the belief of an outside observer who has the knowledge of the CP and the state space, knows the communication policies of the agents, and observes all communication message among the agents, but does not observe any local information of any agent. Such an outside observer will discover the global state once the agents communicate, but can only rule out the set K if the agents are all silent.

The computation model for each agent can now be described as the following: at any stage t, the agent first calculates its local belief set by projecting its local knowledge on the current common belief state B. Then, the agent checks the CP to decide the next local action for its LB set. Next, when the action finishes, the agent calculates the N set and also the LH sets, and decides which LH set it is in. Then, the agent makes the communication decision according to the communication policy. If there is communication, the new B in the next stage would simply be the discovered global state (a set containing just s). Otherwise, the new B would be the set N-K, and the agent's LB set in the next would be the joint of N-K and its LH set.

In summary, the transformed DP consists of two mappings (for each agent): first, a mapping from the LB set to its local action, and second, a mapping from the LH set to {yes,no}. The common belief state, although not directly involved in the mappings, is the key for the calculation of the LB and LH sets.

3. COMMUNICATION POLICIES

The mapping from LH sets to {yes,no} cannot be arbitrary. The mapping from LB set to the agent's local action is decided by the CP, but we need to ensure that such a mapping is valid: it is trivial to decide the local action if the common belief state is a singleton set, but if the common belief state is N-K, the mapping is valid only when local component of the joint action (according to the CP) is the same for all the states in the LB set. In other words, each LB^X (or LB^Y) set in B must have a common local action for agent X (or Y), otherwise there is ambiguity toward the choice of local actions. Since the set N-K is calculated based on the communication policy (recall that K is simply the union of all communicated LH sets), and LB^X sets are simply $LH_i^X \cap N{-}K$ (if LH_i^X is not communicated), the no-ambiguity rule on LB sets translates to a constraint on the choice of communication policy: if LH_i^X is not communicated, all states in $LH_i^X \cap (N{-}K)$ must have the same local action for X. A similar constraint applies to LH_j^Y.

$$\begin{array}{cc} & \begin{array}{cccc} LH_1^Y & LH_2^Y & \dots & LH_n^Y \end{array} \\ \begin{array}{c} LH_1^X \\ LH_2^X \\ \vdots \\ LH_m^X \end{array} & \begin{pmatrix} s_{1,1} & s_{1,2} & \dots & s_{1,n} \\ s_{2,1} & s_{2,2} & \dots & s_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m,1} & s_{m,2} & \dots & s_{m,n} \end{pmatrix} \end{array}$$

Figure 6: Local History Set Matrix

A matrix representation can be used if the system consists of two agents X and Y: let M be a $|LH^X| \times |LH^Y|$ matrix whose elements are the states $s_{i,j} = LH_i^X \cap LH_j^Y$, illustrated in Figure 6. Then, choosing to communicate on LH_i^X (or LH_j^Y) can be symbolized by crossing out all matrix elements on row i (or column j). So, after applying all communicating LH_i^X's and LH_j^Y's, the remaining matrix is the N-K set. Thus, the constraint is to make sure that for each row (and column) of the remaining matrix, there is no ambiguity about the next local action for X (or for Y).

Thus, if all LH_i^X and LH_j^Y have unambiguous local actions, the constraint is automatically satisfied, and no communication is necessary. Otherwise, we should choose to communicate on some LH_i^X and/or LH_j^Y sets so that the remaining matrix meet the constraint. Clearly, there are several strategies for achieving this. In this paper we discuss only two: the *default* strategy and the *hill-climbing* strategy.

The default strategy simply maps all ambiguous LH_i^X and LH_j^Y sets to *yes*. Let $\mathrm{Com}^X(LH_i^X)$ be X's communication decision on LH_i^X and $\mathrm{Com}^Y(LH_j^Y)$ for Y, the default strategy assigns $\mathrm{Com}^X(LH_i^X) = yes$ if and only if LH_i^X has ambiguous actions for X, and $\mathrm{Com}^Y(LH_j^Y) = yes$ if and only if LH_j^Y has ambiguous actions for Y. This is the strategy illustrated in Figure 5, where both LH_3^X and LH_3^Y have ambiguous next local actions and are mapped to *yes*.

The hill-climbing strategy further reduces communication by performing a heuristic search process that greedily minimizes the total number of *yes* decisions, so that the N-K set contains as many states as possible. Due to space limitations we omit the details of our search process (also, the choice of heuristic search methods is an open one). Using the same example in Figure 5, although both LH_3^X and LH_3^Y have ambiguous next local actions, the hill-climbing strategy may just choose to communicate for LH_3^X (or LH_3^Y) only, since the remaining 6 states would have no ambiguous actions.

4. EVALUATING A DP

Based on different communication policies, there are different DPs. The key criteria for DPs are the expected utility

(EU) and the expect total amount of communication (AoC). (If an explicit communication cost is given, we may combine the two criteria into one overall utility measure.) Evidently, the EU of a derived DP is the same as the EU of the CP, which can be evaluated by the policy evaluation algorithm of Markov decision processes. The only remark we want to add is that we can define $v(B)$, the value of a common belief state B (seen by the outside observer), as the weighted value of all states in B:

$$v(B) = \sum_{s \in B} p(s)V(s)/p(B). \tag{1}$$

where $p(s)$ is the probability of reaching state s during problem solving, and $p(B) = \sum p(s)$ for all $s \in B$. $p(s)$ can be calculated through the transition probability $p(s'|s)$ since

$$p(s'|s) = p(s')/p(s). \tag{2}$$

The calculation of AoC is also straightforward: Assuming that each synchronization counts as 1, then for each reachable state s, if there is communication, its contribution toward the expected total amount of communication is the probability of reaching that state, $p(s)$. Clearly, for the trivial transformation mentioned at the beginning of section 2, the total AoC is simply $\sum p(s)$ for all state s reachable via the CP but excluding the starting state and terminal states (since they do not need synchronization). We can also define $c(s)$, the contribution to the total expected AoC at and after state s:

$$c(s) = p(s) + \sum_{s' \in \mathrm{next}(s)} c(s'), \tag{3}$$

where $\mathrm{next}(s)$ is the set of possible next states of s.

For a DP, we can define the $c(B)$, the contribution toward the total expected AoC at and after belief state B. Of all the possible next common belief states of B, only N-K is not communicated:

$$c(B) = \begin{cases} p(B) + \sum_{B' \in \mathrm{next}(B)} c(B'), & \text{if B is communicated;} \\ \sum_{B' \in \mathrm{next}(B)} c(B'), & \text{otherwise.} \end{cases} \tag{4}$$

Thus, the total expected AoC of a DP is simply $c(B_0)$, where B_0 is the starting belief state. and is common knowledge to the agents and hence not communicated. Also, if every state $s \in B$ is a terminal state, $c(B) = 0$, since the agents would realize that the global state is a terminal state without communication. Such a belief state B is called a terminal belief state.

5. NON-CONFORMING DPS

So far, a DP derived from a given CP would adhere to the exact CP in the eyes of an outsider observer who sees the global state and the joint actions of the agents. We call these DPs *conforming* DPs. Conforming DPs offer no loss of EU compared to the CP while decreasing the AoC.

The problem is that it limits the kinds of DPs that might be derived from a given CP. Later we will show some ways to create non-conforming DPs from a CP. These non-conforming DPs may not follow the exact CP as seen by an observer. As a result, the EU of these DPs may change. In other words, they may degrades the EU of the CP. However, they also have the potential of further reducing AoC to the extent not possible by conforming DPs. This offers a tradeoff

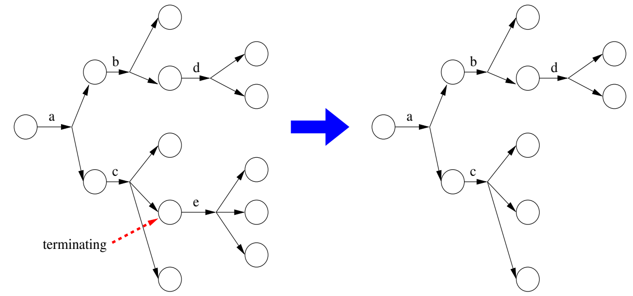

Figure 7: Terminating a State

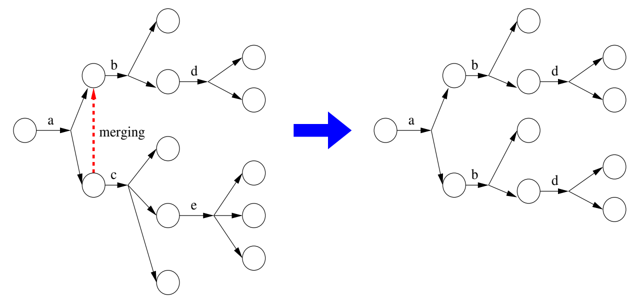

Figure 8: Merging a State

between EU and AoC when selecting DPs, which the conforming DPs lack. This is why it is interesting to study non-conforming DPs: it offers a wider selection of DPs and offers tradeoff choices.

In order to derive non-conforming DPs from a given CP, we first examine how to derive new CPs from a given one. We are mostly interested in domain-independent techniques, which modifies the given CP based on its structure, not on domain knowledge. Specifically, since a policy can be viewed as a directed tree with each node representing a state, and the outgoing edges representing possible transitions to the next states, as shown in the left half of Figure 7. Note that the label of the outgoing edge represents the action to be taken at that state, and there are multiple states from a single action due to non-deterministic action outcomes.

One domain-independent technique for deriving a new CP from an existing one is *terminating*: to mark one or more non-terminal states in the original policy to be terminal states in the new policy. This technique is illustrated in Figure 7. The resulting policy would terminate when reaching the marked states. Clearly, the new policy may not receive any additional reward beyond the marked states, therefore the expected utility of the new policy is different from the original policy's. This technique is well-suited for typical planning problems, where the agents make sequential decisions regarding the execution of tasks, and the problem solving may terminate at any stage and the utility is calculated at that time. Thus, the new terminal states in the new CP is valid.

Another domain-independent technique is *merging*. Illustrated in Figure 8, this technique marks one state to be merged: grafting the subtree beyond another state (target state) onto this state, replacing the original subtree of the merged state. Such a merging operation requires that the merged state is compatible to the target states, so that the subtree grafted correctly reflects the actual structure of the state space: the actions and states in the grafted subtree must exist in the state space, and correctly reflects the transition relationships.

Typically, this requirement means that the part of state space beyond the merged state should be topologically identical to the part of state space beyond the target state. Usually, this is satisfied between *sibling* states, meaning that the two states are both possible resulting states from the same state and action (see Figure 8). This requirement is met when dealing with sequential task planning problems, where the resulting sibling states represent the different possible outcomes of an action (in this case, the same as a task). There, two outcomes are often identical to each other except that they correspond to different utility values. The task interrelationships also remain the same for both outcomes. This means that, if a problem solving episode contains one outcome in its outcome sequence, there must also exist a possible episode that is the same episode except it replaces this outcome with the other one.

For example, an action produces two possible outcomes o_1 and o_2 (and ends in state s_1 and s_2), but the only difference between them is that o_1 produces utility 10 and o_2 produces utility 5, and otherwise they are exactly the same in the problem solving. In this case, the structures of the state spaces beyond s_1 and s_2 are the same.

Since utility value is typically a function of the outcomes, the calculation of the value of the merged state is straightforward: simply use the same process of calculating the utility value of the targeted state, but replace any occurrence of outcome o_2 with o_1 (assuming s_1 is merged to s_2). This technique is used in our calculation when dealing with merged states. The calculation of AoC is even simpler: it is simply the AoC of s_2 multiplied by the factor $p(s_1)/p(s_2)$ — to account for the fact that the probabilities of getting o_1 and o_2 are different.

5.1 Generating Non-Conforming DPs

Given the above discussion, one method of generating non-conforming DPs would be to generate new CPs by applying various *combinations* of the aforementioned techniques, and then to create conforming DPs for the new CPs. However, this method of generating non-conforming DPs is not tied to the communication issue, and therefore may not be efficient in our search for DPs that reduce communication. Therefore, we introduce a method that performs the techniques on common belief states rather than global states in the CP, therefore ties the choice of techniques with communication by putting the selection of the non-conforming technique into the process of generating alternative next belief states.

Specifically, during the process of deciding the mapping of Com^X and Com^Y according to the local history matrix M, we can create alternative strategies of generating next belief states. Previously we have defined the *default* communication strategy, and an alternative, the *hill-climbing* strategy, which produces a different set of possible next common belief states. Now we would like to perform the *terminating* and *merging* operations on these alternatives and thus produce new alternatives, i.e., new sets of possible next common belief states. Similarly, we will use the matrix representation to help the understanding of these operations.

First we discuss the terminating operations. We identify two strategies for applying this operations:

Terminating 1 (T1): For a given belief state B, one strategy is to mark all of its next states (i.e. any state in N) terminal. In this case all next belief states are terminal, and as a result no more communication is needed. This corresponds to marking all states in local history matrix M terminal and therefore the next joint action (in the form of $(a_X|a_Y)$) for each state in M is $(\lambda|\lambda)$, where λ means no action. In this case M automatically satisfies the no-ambiguity condition, thus K is empty (by default).

Terminating 2 (T2): We notice that the above method may result in drastic changes — it eliminates all further communications, but also eliminates all further activities. As a result, the expected utility may suffer. So an alternative method is to keep the $N - K$ belief state intact (based on the conforming alternatives) and mark only a subset of the rest of the next states. Thus, the communication decision remains the same as the conforming alternatives, but some of the synchronized next belief states would be terminal. This may reduces communication if communication is needed in the further stages for those belief states according to the conforming strategies. And since it only marks a subset of the next states, the degradation of utility is more limited compared to T1.

Now we study the merging operations. Since communication decisions are based on local history sets (LH^X and LH^Y) rather than individual states, we focus on merging between LH sets. The goal is to merge communicating LH sets to non-communicating LH sets, so that the communication on the merged sets can be saved. Thus, the merging happens in the local history level rather than the state level: it replaces one local history with another one.

Merging: for a given belief state B, examine the LH sets (the rows and columns of M). For example, if a row is ambiguous (not counting elements already marked or merged), we try to find another row which is compatible to it but is unambiguous. If such a row exists, we can then merge each element in the original row to the same column element in the target row. The same can be applied to columns as well.

Note that there might be many other non-conforming techniques that could be used to explore new CPs, and also other strategies for applying the terminating and merging techniques discussed here. These are excellent areas for future developments, but not the focus of this paper. Next we shall see how our approach actually works in our experiments.

6. EXPERIMENTAL RESULTS

To evaluate our approach, we implemented our CP to DP method and performed some experiments using an example problem. Figure 9 shows a multi-agent task for agents A and B, with both local task interrelationships (such as A1 *enables* A2), and nonlocal interrelationships (such as A2 *enables* B2) between the tasks in different agents. The task has a deadline of 160. Here we are using TAEMS [3, 4, 8], a well-known hierarchical task modeling framework, as our task specification.

First, we mapped the TAEMS task structure into a multi-agent Markov decision process, and then used a standard dynamic programming algorithm to obtain the optimal centralized policy (CP).

In Figure 10 we sketch the sequences of joint actions that might occur in an episode when this policy is used, by analyzing the policy graph. As listed, there are five possible sequences. Sequences 2-4 differ from sequence 1 after the end of the fourth stage (the first 4 stages of sequences 2-4

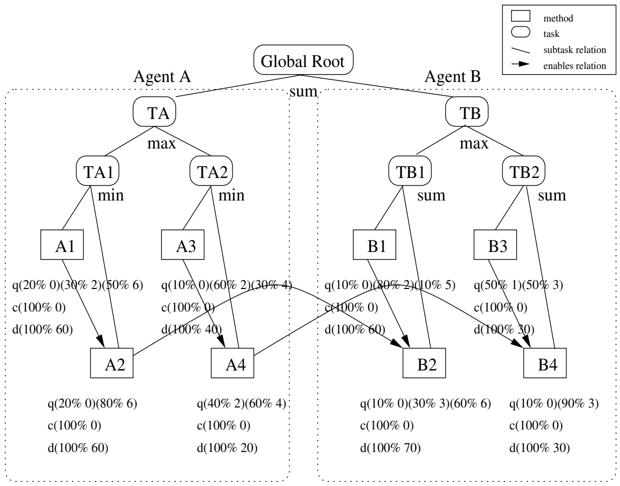

Figure 9: A Multi-Agent Task

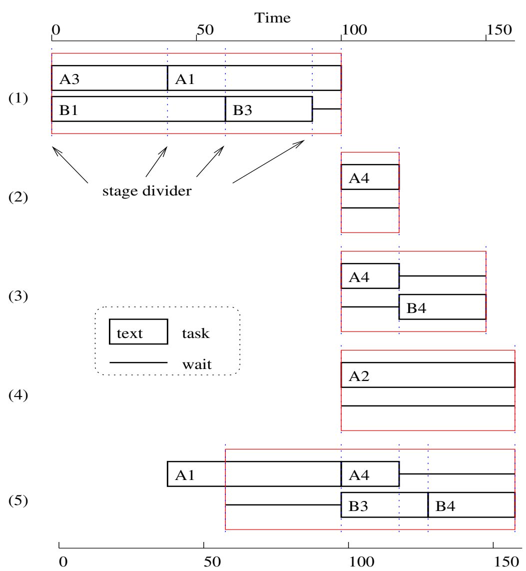

Figure 10: Joint Action Sequences of CP

are the same as those of sequence 1, therefore are omitted in the figure). Similarly, sequence 5 differs from sequence 1 after the end of the second stage. Note that sequence 3 is identical to sequence 2 except the last stage. Based on the specifications given in Figure 10, this CP has a EU of 7.27425 and an AoC of 5.5425 (according to the trivial transformation).

Next, to generate the DPs, we start from the initial common belief state, and calculate the local history matrix M. If all LH sets (rows and columns of H) have unambiguous local actions, there is no communication needed and the next common belief state is simply N, and we move to the next stage. Otherwise, we first apply the default communication strategy, then the hill-climbing strategy (if it produces different communication decisions). Also, the T1 operation is applied to the common belief state *if* the difference between the current (terminating) reward and the default value is small then the default AoC times a constant (the cost of communication). Intuitively, this criteria means that it is not worthwhile to continue the problem solving when the cost of communication outweighs the potential utility gain

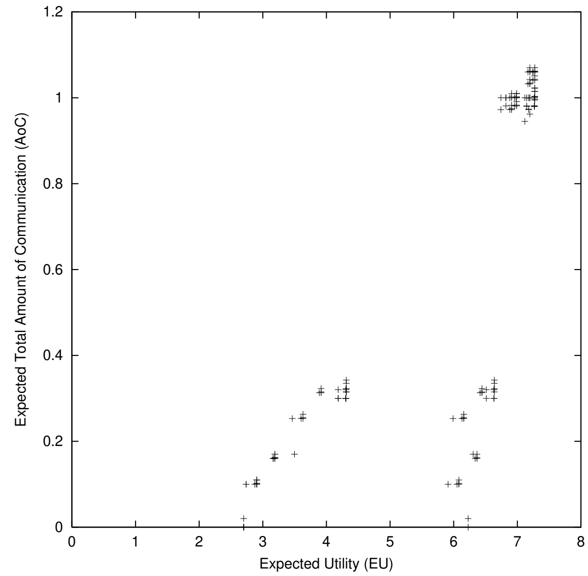

Figure 11: EU/AoC of Generated DPs

by further actions. Then we apply T2 operations on the next common belief states produced by the default strategy and the hill-climbing strategy: marking a communicating next belief state terminal *if* its default AoC is greater than the probability of reaching it, and its default value is below the default value of the current belief state. And finally we try the merging operations on the sets of next belief states generated by the default strategy and the hill-climbing strategy. In our case, for each communicating LH row (or column), merge the row (column) to the compatible row (column) with the best default value, if such a compatible row (column) exists. Thus, we have several alternative sets of possible next common belief states in the next stage. This process continues and therefore the combinations of alternatives produce different DPs.

As a result, we obtained 160 different DPs for the optimal CP mentioned before, with 12 DPs containing only combinations of default strategy and hill-climbing strategy - these are the conforming DPs. We evaluated the EU and AoC of each DP, and 11 shows how their EU and AoC values are scattered.

To give a comparison and also some details about the results, the following table lists the EU and AoC of various policies:

	EU	AoC
Optimal CP	7.27425	5.5425
Default DP	7.27425	1.07
Conforming (12 DPs)	7.27425	0.996 – 1.07
All (160 DPs)	2.7 – 7.27425	0 – 1.07

Immediately, we notice that even the default DP (i.e., the DP uses the default strategy only) reduces communication greatly compared to the AoC of the centralized policy (5.5425). This is done without any loss to EU. But the AoC can be further reduced, even to 0, while still having good EU (the rightmost data point on the EU axis has a EU of 6.228). However, the conforming DPs offer only a very limited range of AoC compared to nonconforming DPs, which reduce AoC but degrade the EU.

An interesting pattern shown in Figure 11 is that the data points are somewhat "clustered". The points can be roughly divided into 3 clusters: top-right, lower-right, and lower-left. Note the AoC gap from 0.4 to 0.9, and the EU gap from 4.5 to 5.5. By looking at the details of the DPs we notice that gaps are largely due to the effect of some non-conforming operations, such as marking some belief states terminal, i.e., to stop the problem solving at an earlier stage and therefore abandon all further communications, or the merging of one local outcome to another, therefore causes a significant change of AoC and EU. In other words, some operations are the deciding factors of EU and AoC changes. Thus, by looking at these operations we can detect the "critical points" in DP, and therefore help us in designing coordination strategies for the agents.

As an example, let us examine the DP corresponding to the data point (EU=6.228, AoC=0):

1. At time 0, A starts task A3 and B starts task B1. The next stage begins at time 40 when A3 finishes, and has 3 possible states. Both agents do not have ambiguity toward the next local action, so the next belief state contains these 3 states.

2. At time 40, A starts task A1 and B continues B1. The next stage is when B1 finishes, with 9 possible next states, and merging is applied to B1's q=0 outcome (which contains 3 states). The remaining 6 states form the next common belief state, and no communication is needed.

3. B1 finishes (at time 60), A continues A1 and B starts B3. In the next stage (when B3 finishes), A continues A1 and B waits, so the 12 next states become the next belief state.

4. B3 finishes now (time 90). Next stage, when A1 finishes, there are 36 possible next states based on the 12-state current common belief state. The merging operation is applied to 8 of the next states, and the rest of 28 states become the next belief state.

5. A1 finishes now, and agent A chooses either A2 or A4, depending on the previous outcome of A3 and A1: if the q(A1) = 6 or if q(A1) = 2 and q(A3) = 0, choose A2, otherwise A4. Clearly this is a local decision, so there is no ambiguity. B simply waits (idle). The next stage has 56 states, and 50 of them are merged, so the next belief state contains only 6 states, all of them terminal states. So the problem solving finishes at the end of the stage.

Essentially, this DP contains 3 merging operations, and as a result the agents have unambiguous local action throughout the problem solving. In typical planning language, it means that the agents simply ignore unexpected local outcomes (such as errors) and stick to a local plan, thus save communication. This divides the cooperation into two seemingly independent local processes, but in fact it is an indication of possible constraint relaxations in the plan: the non-local interrelationships are silently established and therefore can be relaxed in each agent's planning.

7. SUMMARY

In this paper we propose a method for deriving decentralized multi-agent policies from centralized ones. In the past, the design of decentralized policies has been limited to the use of *ad hoc* heuristic methods, and the results are often domain-specific. By using this method, we now have a domain-independent, systematic way of developing decentralized policies. Furthermore, this method provides a bridge between centralized policies and decentralized policies, thus allows us to connect the research in these areas and find more insights. The use of nonconforming operations in generating new CPs is particularly interesting. We are developing new and more complex operations [8] and are trying to generalize them in terms of searching in the policy space.

Also, this method explores the possibility of making trade-offs between the expected total utility and the amount of communication to be used in multi-agent cooperation. Communication in multi-agent systems can be viewed as the dynamic process of obtaining (exchanging) information, which reduces uncertainty in the system but may incur a cost. Thus, it belongs to one of the fundamental domain in artificial intelligence — the question of the value of information. Since communication is often associated with the dynamics of agent commitments, communication policies gives important hints on how to deal with commitments in multi-agent planning.

Designing good decentralized policies is a very challenging task. Yet, it is very important to understand the reasoning, planning, and decision-making in a decentralized, situated agent. One important direction of our future work is to further enforce the assumption of decentralization. Specifically, in this paper we assume that the knowledge of the global state space is available to every agent in the system. In the future, we plan to relax this assumption and examine how to generate decentralized policies when each agent only has its own, partial view of the global state space – apparently a much more realistic reflection of actual systems.

8. REFERENCES

[1] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI-2000)*, 2000.

[2] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proceedings of the Sixteenth International Joint Conferences on Artificial Intelligence (IJCAI-99)*, July 1999.

[3] K. S. Decker and V. R. Lesser. Quantitative modeling of complex computational task environments. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 214–217, 1993.

[4] V. Lesser, B. Horling, R. Vincent, A. Raja, and S. Zhang. *The TAEMS White Paper*. http://mas.cs.umass.edu/research/taems/white/.

[5] H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 1990.

[6] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[7] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

[8] P. Xuan. *Uncertainty Handling and Decision Making in Multi-Agent Cooperation*. PhD thesis, University of Massachusetts at Amherst, 2002.

[9] P. Xuan, V. Lesser, and S. Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the Fifth International Conference on Autonomous Agent (AGENTS 01)*, 2001.