# A Decision-Theoretic Approach for Designing Proactive Communication in Multi-Agent Teamwork

Yu Zhang* Richard A. Volz* Thomas R. Ioerger* John Yen†

*Department of Computer Science
Texas A&M University
College Station, TX 77843-3112
Phone: 01-979-845-8873
{yuzhang, volz, ioerger}@cs.tamu.edu

†School of Information Science and Technology
The Pennsylvania State University
University Park, PA 16802
Phone: 01-814-865-6174
jyen@ist.psu.edu

## ABSTRACT

Techniques that support effective communication during teamwork processes are of particular importance. Psychological study shows that an effective team often can anticipate information exchange among the team and communicate relevant information proactively. Proactive communication is crucial for understanding and sharing common goals and for cooperative actions. Communication can be valuable if it assists agents with new and timely information; it also has cost because it consumes network resources such as bandwidth. To address these issues, we present a new model that uses information production and need to capture the complex multi-agent communication process and a dynamic decision-theoretic determination of communication strategies. We also introduce a generic utility function and an algorithm, DTPC (Decision-Theoretic Proactive Communication), that focuses on representing information production and need of team members and resolving decision interactions among them for making decisions.

### General Terms

Design, Algorithms.

### Keywords

Multi-Agent Systems, Teamwork, Agent Communication, Decision Theory

## 1 INTRODUCTION

Teamwork is a cooperative effort by a team of agents to achieve a joint goal [12]. To date, control paradigms for cooperative teamwork have allowed agents to communicate about their intentions, plans, and the relationships between them [5, 10]. However, this team cooperation behavior is highly complex [8], which weakens teamwork efficiency. Moreover, some researchers have found that communication, while a useful paradigm, is expensive relative to local computation [1]. Therefore, techniques that support effective communication during teamwork processes are of particular importance.



This paper introduces a new model for effective team communication. In contrast to existing agent communication methods, such as [12, 6, 4, 14, 2], we focus on analyzing information production and need of team members to support proactive communication. Proactivity is the ability to take initiative by exhibiting goal-directed behavior [13]. Based on a *shared mental model* [11], which is a major aspect of the psychological underpinnings of teamwork, an effective team often can respond to external stimuli in a timely way, and they can also prepare knowingly for some unexpected future [16, 17]. Hence, the ability to anticipate information needs of teammates and assist them proactively is highly desirable. While an agent can anticipate certain information needs of teammates, it may not always be able to predict all of their needs, especially if the team interacts with a dynamic environment. Therefore, when an agent needs some information, it is also necessary to anticipate information production of teammates and ask for the information actively. Proactive communication allows agents to tell others proactively about a piece of information when producing it or to ask actively for a piece of information when needing it. Proactive communication increases the effectiveness of communication in three ways. First, messages are conveyed to agents when they need an information item, rather than always sending it to them. Second, proactive tell can partially eliminate the need to ask for information. Third, if there is no proactive tell, active ask may eliminate multiple asks, i.e. only ask one provider per need.

We take a decision-theoretic approach, because during multi-agent teamwork, agents need to be able to deal with uncertainties, since they may have only incomplete information about the teamwork, the environment, and the potential value and cost of information delivery. The decision-theoretic approach provides agents an optimal way to fulfill their information needs under uncertainties. Broadly speaking, the decision theory is a means of analyzing a series of strategies in order to decide which should be taken, when it is uncertain exactly what the result of taking the strategy will be [7]. However, departing from the traditional decision-theoretic approach, we use information production and need to capture the complex decision process of information needer and provider. Moreover, we emphasize communication benefiting the team and focus on decision interactions between the needer and provider, i.e. their decisions are interdependent so they must consider the impact of their counterpart's decisions upon their own.

In our model, agents are equipped with a set of communication strategies from which they must choose when making decisions. To quantify agents' decisions, we have developed a generic utility function that focuses on representing the information production and need of team members. After evaluating the utility of a particular strategy, agents will identify the optimal strategy, which maximizes the utility of communication.

Two difficulties exist in agents' decision-making. First, agents cannot compute exact values of the utility since some parameters cannot be known precisely. Hence, they use the expected value of the utility function where they need values, which involve the calculation of probability. Second, agents' decision-making is interdependent, so when evaluating a strategy, they must consider their counterparts' responses, which cannot be known exactly, and so also need to be estimated. An algorithm called DTPC (Decision-Theoretic Proactive Communication) has been developed to deal with these issues.

The rest of this paper is organized as follows. Section 2 provides an example of the proactive communication problem in teamwork. Section 3 gives basic contextual information. Section 4 introduces a generic utility function. Section 5 presents multi-agent communication process. Section 6 discusses DTPC algorithm. Section 7 concludes this paper.

## 2 AN EXAMPLE

In order to understand the proactive communication problem in teamwork, such as which kind of information will be communicated; who needs it; who provides it; how to determine whether or not an agent having information should tell another agent; and whether to ask some specific agent for needed information, we introduce an extended, classical example of a multi-agent team.

As an example, we have extended the Wumpus World problem [9] into a multi-agent version [18]. The goals of the team, four agents, one carrier and three fighters, are to kill wumpuses and get gold. Wumpuses live in their small kingdoms that have boundaries. They can move randomly to adjacent locations, or choose to stay at the current locations. The carrier is capable of finding wumpuses and picking up gold. The fighters are capable of shooting wumpuses. When a wumpus is killed, agents can determine that the wumpus is dead only by getting the message from the one who killed it.

Plans are at the center of activity. They describe how individuals or teams can go about achieving various goals. Our plans are represented in MALLET (Multi-Agent Logic Language for Encoding Teamwork), which provides descriptors for encoding knowledge about teamwork processes (i.e. individual/team plans and operations), as well as specifications of team structures (e.g., team members and roles) [15]. Plans are classified into individual plans and team plans. Each individual plan has a process consisting of a set of operations, each of which is either a primitive operator, or a composite operation (e.g., a sub-plan). Team plans are similar to individual plans, but they allow multiple agents or agent variables to be assigned to carry out operations or plans (some of them requiring a team). A DO statement is used to assign one or several agents to carry out specific operators or sub-plans. CAST (Collaborative Agents for Simulating Teamwork) is a multi-agent architecture that simulates and supports teamwork [15], and is used as the basis for simulating our example system.

The following is an example team plan for the multi-agent version of Wumpus World:

```
(tplan killwumpuses()
  (foreach (cond (wumpus ?w))
    (killwumpus(?w))))

(tplan killwumpus(?w)
  (process
    (par
      (seq
        (agent-bind ?ca (constraint (play-role ?ca carrier)))
        (DO ?ca (findwumpus ?w))) ;the carrier is assigned
      (seq
        (agent-bind ?fi (constraint ((play-role ?fi fighter)
                                     (location ?w ?x ?y))))
            ; the fighter assigned must know wumpus' location
        (DO ?fi (movetowumpus ?w))
        (DO ?fi (shootwumpus ?w))))))
```

where findwumpus and movetowumpus are individual plans, and shootwumpus is an individual operator specified as follows:

```
(ioper shootwumpus (?w)
    (pre-cond (location ?w ?x ?y) (dead ?w false))
    (effect (dead ?w true)))
```

Note that the plan does not explicitly state the communication that is to take place. Rather, the agents are to infer the necessary communication from their knowledge of the plan and the environment.

As one can infer from this plan, the key problems are which kind of information will be communicated, how agents can *know* who will need or produce the information and *when* it will be produced or needed. The answer to the first problem is that there are two kinds of information communicated in the team: 1) an unknown conjunct that is part of a constraint (e.g., "wumpus location"); 2) an unknown conjunct that is part of the precondition of a plan or an operator (e.g., "wumpus location" and "wumpus is dead"). To determine who needs and who produces a given item of information, agents analyze the preconditions and effects of operators and plans and generate a list of needers and a list of providers for every piece of information. The needers are agents who might need to know the information (e.g. the fighters), and the providers are agents who might know the information (e.g. the carrier and other fighters).

Challenges come from the third problem. We can see that the fighters need to know a "wumpus' location" and whether the "wumpus is dead", which can be obtained by communicating with the carrier and other fighters. However, knowing *when* the information is needed or produced is sometimes impossible because it requires agents to have comprehensive knowledge about both parts of the communication. To explain this, consider the case of getting "wumpus' location." The fighters can obtain the "wumpus' location" information either by the carrier proactively telling them, or by actively requesting it from the carrier. The more up-to-date information the fighter receives, the better its chance of locating the wumpus, and since wumpuses can only move inside certain boundaries, a piece of old information may be useful to the fighters, in the case that they do not have the most recent one. However, since wumpuses move freely in their kingdoms, the carrier cannot know the precise time interval between a fighter moving to a found wumpus and killing it. For the same reason, the fighters cannot know the exact time when the carrier will find a wumpus.

It can be seen that the distributed nature of the agent team and the dynamic nature of the world often make it infeasible for an agent to have complete and up-to-date information about other teammates and the world. The resultant uncertainty, which may seriously affect the quality of communication among agents, and the proposed solution to that uncertainty is what we will address in the rest of this paper.

# 3 BASICS

Communication generally involves two parts: needer and provider. In order to obtain an information item, messages may be conveyed from either part to its counterpart. For clarity, we will assume that the system consists of two agents, *a,* a provider, and *b,* a needer. The ideas can be extended readily to larger numbers of agents if desired. For the needer, we need to know what to do when it either needs or receives an item of information, while a provider must decide what to do when it produces an item of information or receives a request for information. In the following, we delineate the different strategies that each might employ.

## 3.1 Strategies

The needer has two situations to consider. In situation NA, it needs a piece of information, *I*. In situation NB, it receives a piece of information, which may be either a reply from the provider whom it has actively asked or a proactive tell from the provider. In situation NA, the needer has the following strategies from which to choose:

- *Silence* : The agent does not ask for *I* and uses the most recent value it has;
- *ActiveAsk* : The agent actively asks for *I*;
- *Wait* : The agent waits to be told *I* proactively.

In situation NB, the needer has following strategies from which to choose:

- *Accept* : The agent notifies the acceptance of *I*;
- *RejectNeed* : The agent notifies the rejection of *I*, because it does not need *I* soon;

In either situation, the needer must select a single strategy and act accordingly.

The provider will also face two situations when making the decision. In situation PA, it produces a new piece of information. In situation PB, it receives a request for a piece of information. In situation PA, the provider has the following strategies from which to choose:

- *ProactiveTell*: The agent proactively provides *I*;
- *Silence* : The agent does not provide *I*.

In situation PB, the provider has the following strategies from which to choose:

- *Reply* : The agent provides most recent value for *I*;
- *WaitUntilNext*:The agent waits until next production of *I* and then provides *I*;
- *Reject* :The agent notifies the rejection of providing *I*, because the agent neither produced *I* recently, nor expects to produce *I* soon.

## 3.2 Time

In order to make their communication decisions, agents need to consider the relationship between the time at which information is needed and the time at which it is produced. The various strategies involve using the information produced at different times or satisfying needs at different times. Thus, to describe the range of possibilities encompassed by the different strategies adequately, several different points in time must be defined. For clarity, we define two sets of relevant points in time, one set for the needer and another for the provider.

### *Time points for needer*

We first consider situation NA. Let $T_{b,N}^{0}$ be the time at which agent *b*'s most recent need for *I* arises; we consider this to be the current time of decision-making. We assume that agent *a* produces values for a piece of information *I* from time to time. The inter-production time interval is assumed to be random according to an unknown distribution. Let $T_{a,P}^{a0}$ be the time at which agent *a* most recently produced *I*. Further, let $\{T_{a,P}^{-an}, ..., T_{a,P}^{-a1}\}$ be the (ordered) set of previous times when agent *a* produced *I*. And $\{T_{a,P}^{a1}, T_{a,P}^{a2}, \ldots\}$ denote the (ordered) set of times at which agent *a* will produce *I* in the future, which is unknown at the current time. Let $T_{s,r}$ be the time at which agent *b* most recently received a value for *I*. We assume the communication delay is negligible, thus $T_{s,r}$ is also the time agent *a* most recently sent out a value for *I*. Hence $T_{s,r}$ is one of time points among $\{T_{a,P}^{-an}, ..., T_{a,P}^{-a1}, T_{a,P}^{a0}\}$. Based on their definitions, these time points have the following relations:

$$T_{a,P}^{-an} < \ldots < T_{a,P}^{-a2} < T_{a,P}^{-a1} < T_{a,P}^{a0} < T_{a,P}^{a1} < T_{a,P}^{a2} < \ldots,$$

$$T_{a,P}^{a0} \le T_{b,N}^{0} < T_{a,P}^{a1},$$

$$T_{s,r} < T_{b,N}^{0}.$$

In situation NB, let $T_{b,g}$ be the time at which agent *b* gets (receives) *I*; we consider this to be the current time. Let $T_{b,N}^{g_0}$ be the most recent need time, prior to the receiving time $T_{b,g}$, and $T_{b,N}^{g_1}$ be the next need time, subsequent to the receiving time $T_{b,g}$. These time points have following relations:

$$T_{b,N}^{g_0} < T_{b,g} < T_{b,N}^{g_1}.$$

### *Time points for provider*

In situation PA, let $T_{a,P}^{0}$ be the time at which agent *a* produces a value for *I*; we consider this to be the current time. We assume that agent *b* needs values for *I* from time to time. The inter-need time interval is assumed to be random according to an unknown distribution. We assume that the needer will not proceed until a need is fulfilled, that is, a second need for *I* does not occur until after the current need has been fulfilled. Let $T_{b,N}^{uf}$ be the time of the unfulfilled need after the most recent fulfilled need. A need can be fulfilled either by a value for *I* from the provider or by the most recent value for *I* that agent *b* already has. $T_{b,N}^{uf}$ could be either before or after the current time, $T^{0}_{a,P}$. These time points have following relations:

$T_{s,r} < T^{0}_{a,P}$,

$T_{s,r} < T^{uf}_{b,N}$.

In situation PB, let $T_{b,r}$ be the time at which agent *b* requests *I*; we consider this to be the current time. Let $T^{r_0}_{a,P}$ be the most recent production time, prior to the request time $T_{b,r}$, and $T^{r_1}_{a,P}$ be the next production time, subsequent to the request time $T_{b,r}$. These time points have following relations:

$T^{r_0}_{a,P} < T_{b,r} < T^{r_1}_{a,P}$.

# 4 UTILITY

We assume information needer and provider have the same utility function; because as they are cooperative in a team, we consider the utility function to represent the utility gained by the team when a particular needer uses a particular item of information at a particular time. Consequently, needers and providers have the same utility function when referencing the use of an item of information by the same user at the same time. However, because their knowledge of the various points in time differs, they must approximate the value function by its expected value, and their evaluation of the utility function will typically vary.

## 4.1 Defining the utility function

We first analyze parameters that should be included in the utility function U. Information is the center of communication. A piece of information *I*, which the communication may convey, is a parameter in U. The purpose of communication is to assist the needer. The needer cares about when it needs information *I* and the time at which value for *I* it will use was produced. Hence, the times at which *I* is needed and at which the value used for *I* is produced, called time $t_1$ and $t_2$ respectively[1], are two other parameters in U. Depending upon the strategy for which the utility function is being evaluated, different time points from the previous definitions will be appropriate to use. Because *I* can be obtained by a set of messages M, these messages are also necessary parameters in U. To sum up, the utility function U should have four parameters $\{I, t_1, t_2, M\}$, where I is the information about which the communication occurs, $t_1$ and $t_2$ are times related to the need and providing the information, and M is the set of messages involved in obtaining that information. These parameters will be filled by specific values based on the strategy chosen by a needer or provider when making decisions.

Utility is the difference between the value gained by having information and the cost of sending the messages:

$$U(I, t_1, t_2, M) = V(I, t_1, t_2) - C(M).$$

The cost of sending a message $M_i$ is assumed to be:

$$C(M_i) = \begin{cases} 0 & \text{if } M = \varphi \\ k_0 + k_1 \times \text{len}(M) & \text{otherwise} \end{cases}$$

[1] When the utility function is evaluated for the different possible strategies, $t_1$ and $t_2$ will take on various time points defined in the previous section.

where $\text{len}(M_i)$ is length of message $M_i$, and $k_0$ and $k_1$ are coefficients.

We recognize two effects, timeliness and relevance of using the information, in determining the form of the value function. If the needer uses the most recent value it has at the time the need arises, the value is available immediately but may be obsolete. Otherwise, it may use a new value that may be produced only after some delay (it might have to wait for the provider). The provider faces similar choices; the value it has at the time of a need may have been produced at some earlier time and hence may not be timely for the needer, whereas the production of a new value may be an unknown time in the future.

We assume that once the needer receives the information, it is consumed immediately. We assume that the value gained by having timely information can be represented by a function $f_t$ of the time difference between $t_2$ and $t_1$, and we will consider value functions that decrease as a function of the time difference. First, we define a function:

$$d(t_1, t_2) = t_2 - t_1.$$

We then define a non-increasing function:

$$f_t(x) \text{ s.t. } 0 < f_t(y) \leq f_t(x) \text{ if } y \geq x,$$

and use

$$k_2 \bullet f_t(d(t_1, t_2))$$

as value function V, where $k_2$ is a coefficient. $f_t$ may has various forms. For example, it might decrease exponentially, or it might be constant for a length of time and zero thereafter, indicating that the information must be consumed in a finite length of time or it is useless.

Secondly, we consider the relevance of the information. In this case, the information used is somewhat stale and it may be the case that using stale information degrades the value of using the information. We represent this as a function $f_r$ of $t_2$-$t_1$. There are again many forms that $f_r$ could take. For example, if the information used changes value at discrete points in time and the use of incorrect information has zero value, one might describe $f_r$ statistically and use probabilities that the information has changed in the interval $[t_2, t_1]$. On the other hand, in other domains, it might be the case that the there is a value to using stale information but that for some reason (perhaps it takes longer to find a target) the value of using the stale information decreases with the age of the information.

We then combine the timeliness and relevance aspects into a single function V such that

$$V(I, t_1, t_2) = f_t(t_2 - t_1) + f_r(t_2 - t_1).$$

## 4.2 Identifying parameters in the utility function

In the utility function, parameters *I* and M are relatively fixed when agents make decisions, while parameters $t_1$ and $t_2$ may vary according to different strategies.

In Table 1, we identify $t_1$ and $t_2$ for different strategies. N and P indicate strategies for needer or provider; for example, *N.Silence* denotes needer's strategy *Silence*. We use time cutoff T to prevent the needer waiting too long if the provider uses some strategy, such as *P.Silence* or *P.Reject* (refer to section 6 for more details).

For some strategies, the time point used for a parameter depends upon the counterpart's response, which is unknown to the decision maker. For example, if a needer *ActiveAsk*s for I, it does not know whether the provider will *Reject*, *Reply* or *WaitUntilNext*; if the needer *Wait*s, it does not know whether the provider will keep *Silence* or *ProactiveTell*. The needer, therefore, needs to find a way to estimate which strategy the provider will choose and how this choice impacts the needer's decision. To solve this problem, agents are asked to think as their counterparts do. We assume that both needer and provider know the other's possible strategies and estimation process. When making a decision, each will go through the estimation process of its counterpart to identify the strategy the counterpart will choose. This process, of course, will be based on each one's own information. The time point of the estimated strategy will be used to fill in the parameter.

| Parameter / Strategy | $t_1$ | $t_2$ |
|---|---|---|
| *N.Silence* | $T^{0}_{b,N}$ | $T_{s,r}$ |
| *N.ActiveAsk* | $T^{0}_{b,N}$ | $T^{a0}_{a,P}$, if *a Reply* |
| | | $T^{a1}_{a,P}$, if *a WaitUntilNext* |
| | | $T^{0}_{b,N}$ +T, if *a Reject* |
| *N.Wait* | $T^{0}_{b,N}$ | $T^{a1}_{a,P}$, if *a ProactiveTell* |
| | | $T^{0}_{b,N}$ +T, if *a Silence* |
| *N.Accept* | $T^{g_0}_{b,N}$ | $T_{b,g}$ |
| *N.RejectNeed* | $T^{g_1}_{b,N}$ | $T_{b,g}$ |
| *P.ProactiveTell* | $T^{uf}_{b,N}$ | $T^{0}_{a,P}$ |
| *P.Silence* | $T^{uf}_{b,N}$ | $T^{0}_{a,P}$ +T |
| *P.Reply* | $T_{b,r}$ | $T^{r_0}_{a,P}$ |
| *P.WaitUntilNext* | $T_{b,r}$ | $T^{r_1}_{a,P}$ |
| *P.Reject* | $T_{b,r}$ | $T^{r_0}_{a,P}$ and $T^{r_1}_{a,P}$ |

Table 1. Identifying parameters for strategies

## 4.3 Calculating expected utility

In our study, utility is a function that maps from states of information need and production to real numbers. The states of information need and production are represented by parameters {I, $t_1$, $t_2$, M} in the utility function. Among them, at least one (and perhaps both) of $t_1$ and $t_2$ are unknown when an agent evaluates the utility of a strategy. Hence, we will use the expected value of the utility function as the basis for decision-making.

In order to calculate the expected value of the utility function, it is necessary to have the distributions of the information production and need times. Since these distributions can be arbitrarily complex, if we have no other information, the entire sample will be the best estimate of the population as long as the current samples are randomly generated according to the actual distribution. This feature allows us to take advantage of some previous data and to use the Empirical Distribution Function (EDF) [3] to estimate the distributions. We assume that the needer has a list of the time intervals between the times at which the provider produced values for a piece of information, and the provider has a similar list of the time intervals between the needer needs for a piece of information. Given the list of time intervals, the needer can estimate the distribution of time intervals between production of an item of information by a given provider. Similarly, the provider can estimate the distribution of time intervals between successive needs an item of information by a given needer. In order to make the interval lists known, each request by a needer will be accompanied by the change in the need list since the last request and each tell of the information will be accompanied by the change in the production list since the last tell.

For a given *I* and strategy S, M is known. Thus, for a given decision point involving *I*, U becomes a function of three variables, $t_1$, $t_2$ and the strategy S. The time points to be used for $t_1$ and $t_2$ are given in Table 1. Then, the expected value of the utility function for a given strategy S, may be computed as

$$E(U)=\int_{t_{1s}}^{\infty}\int_{t_{2s}}^{\infty}(p(t_1,t_2)\times U(S,t_1,t_2))dt_2dt_1$$

where for each of the possible strategies, $t_1$ and $t_2$ are replaced by the variables of Table 1 and $t_{1s}$ and $t_{2s}$ are lower limits of these variables, based on the time relationship described in section 3.2. Since we are only able to determine a discrete approximation to the probability density function using EDF, we use a discrete approximation to this integral function. The details of this approximation, however, are beyond the scope of this paper.

## 5 MULTI-AGENT COMMUNICATION PROCESS

Figures 1 and 2 show finite state diagrams representing the communication process of getting and telling (respectively) an information item. Each node represents a decision point. As one proceeds through the graph, the nodes represent alternating decisions by the needer and the provider. Since some of the decisions that can be made involving waiting an arbitrary length of time for another agent to do something, the possibility of infinite waits arises. To circumvent this, we use a simple time-out rule that cancels the strategy that was selected and returns to node 0 for a new decision, but one made with the knowledge that the time-out has occurred. The nodes marked "e" are special in the sense that they represent the receipt of the information or a rejection. If the information was received, the agent just proceeds to use it, while if there was a rejection, it returns to its node 0 of the pertinent figure.

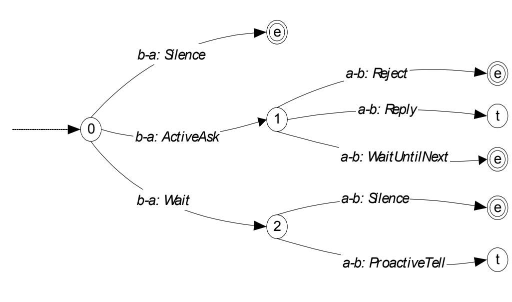


Figure 1. Situation NA: Needer needs a piece of information

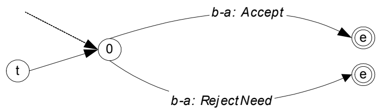


Figure 1. Situation NB : Needer receives an item from a provider

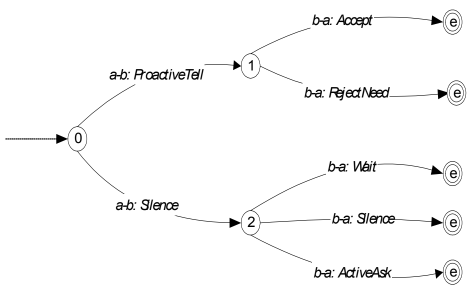


Figure 2. Situation PA: Provider produces a new piece of information

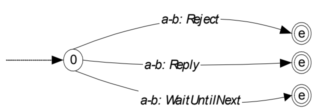


Figure 2. Situation PB: Provider receives a request for a piece of information

For example, in situation NA of Figure 1, *b* needs an information item *I*. If *b ActiveAsk*s *a* and *a Replie*s, the state will transfer to the start state of situation NB, the situation where *b* receives *I* from *a*. In this case, *b* will decide to accept the information and so inform *a*. If *b ActiveAsk*s *a*, *a* may *Reject* this request. In such a case, *b* needs to update its data about *a*'s production time and reconsider its decision; in the more general setting, it might ask a different provider. Similar sequences can occur for the provider.

## 6 DECISION-THEORETIC PROACTIVE COMMUNICATION

```
/*self is an agent who makes the decision;
  counterpart is an agent about whom the decision is;
  I is information that communication conveyed.*/
SelectStrategy(self, counterpart){
   for each strategy S
      Identify(self, counterpart, S);
      EU(S)=Evaluate(self, counterpart, S);
   select one S with maximum EU(S);
   return S;}
```

Figure 3. An algorithm of selecting a strategy

DTPC (Decision-Theoretic Proactive Communication) is the overall process for managing communication. It has three parts: an algorithm for selecting a strategy, algorithms for getting needed information and algorithms for providing information. Figure 3 shows the algorithm for selecting a strategy. The Identify function identifies parameters ($t_1$ and $t_2$) that will be used appropriately, based on the values listed in Table 1. A strategy with maximum expected utility will be chosen.

Figures 4 and 5 show algorithms for getting a piece of information from a provider, or providing a piece of information to a needer. Generally, agents select a strategy that has maximum expected utility and act corresponding to that and their counterpart's response.

```
/*Executed when needer is in situation NA.*/
GetNeededInfo(needer, provider, I){
   set time cutoff T;
   Boolean obtained=FALSE;

   while ((!obtained)&&(waitTime<T))
      S=SelectStrategy(needer, provider, I);
      switch(S){
         case Silence:
            Silence; //use most recent value it has
            obtained=TRUE;
            break;
         case ActiveAsk:
            ActiveAsk;
            if provider Reject to provide I
               AdjustData(needer, provider, I);
            elseif provider sends Reply
               Update(KB_needer, I);
               obtained=TRUE;
            else      //provider chose WaitUntilNext
               obtained=TRUE;
            break;
         case Wait:
            Wait;
            if provider Proactivetell I
               Update(KB_needer, I);
               obtained=TRUE;
            break;
   }}
   If (!obtained)
      Silence;
}

/*Executed when needer is in situation B.*/
ReceiveInfo(needer, provider, I){
   S=SelectStrategy(needer, provider, I);
   switch(S){
   case Accept:
      Accept;
      Update(KB_needer, I);
      break;
   case RejectNeed:
      RejectNeed;
      AdjustData(needer, provider, I);
      break;
}}
```

Figure 4. Algorithms about getting needed information

Function AdjustData adjusts the data about counterpart's information production or need. If the needer *ActiveAsk*s the provider, the provider may *Reject* this request. This means that the needer's estimation of the provider's information production time is not precise. Consequently, the needer must adjust its data about the provider's production time and reconsider its decision. Similarly, if the needer *RejectNeed*s the *proactiveTell*ed *I*, the provider needs to do the same thing.

```
/*Executed when provider is in situation PA.*/
ProvideNeededInfo(provider, needer, I){
  req=FALSE;
  done=FALSE;
  While (!done){
    S=SelectStrategy(provider, needer, I);
    switch(S){
      case ProactiveTell:
        ProactiveTell;
        if needer RejectNeed to I
          AdjustData(provider, needer, I);
        else done=TRUE;
        break;
      case Silence:
        Silence;
        while ((req=FALSE)&&(waitTime<T))
          wait;
        if (req=FALSE)
          AdjustData(provider, needer, I);
        else done=TRUE;
        break;
}}}

/*Executed when provider is in situation PB.*/
ReceiveRequest(provider, needer, I){
  S=SelectStrategy(provider, needer, I);
  switch(S){
    case Reply:
      Reply;
      break;
    case WaitUntilNext:
      WaitUntilNext;
      break;
    case Reject:
      Reject;
      break;
}}
```

Figure 5. Algorithms about providing needed information

We set a time cutoff T[2], to guarantee that the system does not go into a waiting forever state. Thus, we need secondary decisions if the delay has expired. The algorithms simply loop back to the strategy selection point in such cases, but with the additional information that the timeout occurred. Since time passed and data has been updated, each strategy may generate different expected utility. One could then re-evaluate the strategies, again selecting the one with the maximum utility function value. Alternatively one could apply heuristically chosen loop breaking algorithms. For example, if a needer does not get the information during T, the needer could adopt strategy *Silence*, thus using the most recent value it has, or, in the more general case of multiple providers, it could ask a different provider.

Function Update is used to update agents' knowledge bases with the received information and maintain the consistency of the knowledge base. Since the number of values for an item of information increases with time, an agent keeps only the most recent value.

## 7 SUMMARY AND CONCLUSIONS

In this paper, we have presented a method for achieving proactive information exchange using decision theory for determining the communication strategy to be used. We have each situation that might (or might not) involve the exchange of information; we have identified the strategies that could be selected. We have then introduced the general form of a utility function that can be used for the decision theoretic selection of the best strategy. In order to do this, it is necessary to estimate the value of the utility function, as some of the independent variables cannot be precisely known by the evaluating agent. In order to provide the probability density functions necessary for estimating the values of the utility function, we suggest transmitting data on the times of information production or need along with any messages that are send among agents, and then using empirical distribution function methods to approximate the needed density functions.

In the future, we plan a simulation-based evaluation of our approach. We believe our approach can lead to more effective communication, in terms of lower message count, higher team utility gained, and better team performance than other methods.

## ACKNOWLEDGEMENTS

This work was supported in part by DoD MURI grant F49620-00-I-326 administered through AFOSR.

## REFERENCES

[1] Balch, T. and R. C. Arkin, 1994. Communication in Reactive Multi-agent Robotic Systems. *Autonomous Robots*, 1(1): 27-53.

[2] Bui, H., D. Kieronska, and S. Venkatesh, 1997. Optimal Communication Among Team Members. *Lecture Notes in Artificial Intelligence*, 1342: 116-126.

[3] Casella, G., and R. L Berger, *Statistical Inference*, Duxbury Press, 1990.

[4] Gmytrasiewicz, P. J. and E. H. Durfee, 2000. Rational Communication in Multi-Agent Environments. *Autonomous Agents and Multi-Agent Systems*, 3(4):233-272.

[5] Jennings, N. R., 1995. Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems Using Joint Intentions. *Artificial Intelligence,* 75(2).

[6] Kaminka, G. A. and M. Tambe, 2002. Monitoring Teams by Overhearing: A Multi-Agent Plan-Recognition Approach. *Journal of Artificial Intelligence Research*.

[2] Of course, if desired, one could use a different cutoff for each situation.

[7] Parsons, S. and M. Wooldridge, 2002. Game Theory and Decision Theory in Multi-Agent Systems. *Autonomous Agents and Multi-Agent Systems*, 5: 243-254.

[8] Pynadath, D., and M. Tambe, 2002. Multi-Agent Teamwork: Analyzing the Optimality and Complexity of Key Theories and Models. *Proceedings of the 1st Autonomous Agents and Multi-Agent System Conference* (AAMAS'02).

[9] Russell, S. and P. Norvig, 1995. *Artificial Intelligence: A Modern Approach*, NJ: Prentice Hall.

[10] Stone, P. and M. Veloso, 1999. Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork. *Artificial Intelligence*, 110(2): 241-273.

[11] Sycara, K. P. and M. C. Lewis, 1991. Forming Shared Mental Models. *Proceedings of 13th Annual Meeting of the Cognitive Science Society*, Chicago, pp. 400-405.

[12] Tambe, M., 1997. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research*, 7: 83-124.

[13] Wooldridge, M. J., and N. R. Jennings, 1995. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2).

[14] Xuan, P., V. Lesser, and S. Zilberstein, 2001. Communication Decisions in Multi-Agent Cooperation: Model and Experiments. *Proceedings of the 5th International Conference on Autonomous Agents (Agents'01)*, pp. 616-623.

[15] Yen, J., J. Yin, T. R. Ioerger, M. S. Miller, D. Xu, and R. A. Volz, 2001. CAST: Collaborative Agents for Simulating Teamwork. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI'01)*.

[16] Yen, J., X. Fan and R. A. Volz, "Proactive Information Exchanges Based on the Awareness of Teammates' Information Needs", *Working Notes of AAMAS 2003 Workshop on Agent Communication Languages and Communication Policies*, Melbourne, Australia, July 15, 2003.

[17] Yen, J. X. Fan and R. A. Volz, "On Need-driven Proactive Information Exchanges in Agent Teams," *IEEE Computer Society and Web Intelligence Consortium Intelligent Agent Technology 2003*, Beijing, China, Oct., 2003.

[18] Zhang, Y., R.A. Volz, T.R. Ioerger, S. Cao, and J. Yen, 2002. Proactive Information Exchange During Team Cooperation. *International Conference on Artificial Intelligence (IC-AI'02)*, pp. 341-346.