MDPs for Adjustable Autonomy in Real-World Multi-Agent Environments

David Pynadath and **Paul Scerri** and **Milind Tambe**
Information Sciences Institute and Computer Science Department
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
{pynadath, scerri, tambe}@isi.edu

Abstract

Research on adjustable autonomy (AA) is critical if we are to deploy multiagent systems in support of important human activities. Through AA, an agent can dynamically vary its level of autonomy — harnessing human abilities when needed, but also limiting such interaction. While most previous AA work has focused on individual agent-human interactions, this paper focuses on agent teams embedded in human organizations in the context of real-world applications. The need for agent teamwork and coordination in such environments introduces novel AA challenges. In particular, transferring control to human users becomes more difficult, as a lack of human response can cause agent team miscoordination, yet not transferring control causes agents to take enormous risks. Furthermore, despite appropriate individual agent decisions, the agent teams may reach decisions that are completely unacceptable to the human team.

We address these challenges by pursuing a two-part decision-theoretic approach. First, to avoid team miscoordination due to transfer of control decisions, an agent: (i) considers the cost of potential miscoordination with teammates; (ii) does not rigidly commit to a transfer of control decision; (iii) if forced into a risky autonomous action to avoid miscoordination, considers changes in the team's coordination that mitigate the risk. Second, to ensure effective team decisions, not only individual agents, but also subteams and teams can dynamically adjust their own autonomy. We implement these ideas using Markov Decision Processes, providing a decision-theoretic basis for reasoning about costs and uncertainty of individual and team actions. This approach is central to our *deployed* multi-agent system, called *Electric Elves*, that assists our research group in rescheduling meetings, choosing presenters, tracking people's locations and ordering meals.

Introduction

Recent exciting, ambitious applications in agent technology involve agents acting individually or in teams in support of critical activities of individual humans or even entire human organizations, in arenas such as intelligent homes (Lesser *et al.* 1999), "routine" organizational coordination (Pynadath *et al.* 2000), electronic commerce (Collins *et al.* 2000), and long-term space missions (Kortenkamp *et al.* 1999; Dorais *et al.* 1998). These new applications have raised interest in the development of agents with *adjustable autonomy*, i.e., agents that dynamically adjust their own level of autonomy based on their situation (Call for Papers 1999). An agent may act with full autonomy, or it may act with reduced or no autonomy, instead transferring decision-making control to a person. The agent must decide to transfer such control at appropriate times, without overly burdening people, while simultaneously harnessing their possibly superior skills or knowledge.

Our research aims at a decision-theoretic approach to adjustable autonomy (henceforth AA) in rich environments with agent teams embedded within large-scale human organizations. In such environments, not only do individual agents interact with individual people, but they also coordinate with each other and act jointly in teams. The required teamwork and coordination give rise to novel AA challenges that previous work does not address. Whereas most existing research focuses on the interaction between an individual agent and an individual person, we focus on two key novel challenges in AA: the *AA coordination challenge* and the *AA team-decision challenge*.

The *AA coordination challenge* arises in transferring decision-making control. In AA, the problem of when an agent should transfer decision-making control to a human (or vice versa) is already a well-known central problem. In a team setting, the novel challenge is that an agent must transfer control while also avoiding miscoordination with its teammates and while ensuring effective team performance. Techniques from previous AA research on transferring decision-making control fail to address this challenge. For instance, in one existing method, an agent avoids an autonomous decision if it has high uncertainty about the correctness of its decision, and it relies on human input instead (Gunderson & Martin 1999). Applying such a technique in a team setting, an agent may transfer control to a

human to reduce decision-making uncertainty, but if the human fails to respond, then the agent may be unable to communicate with its teammates, resulting in costly miscoordination. On the other hand, while an agent's autonomous decision may avoid miscoordination, it may also be erroneous and jeopardize the overall team activity. Thus, we address the *AA coordination challenge* by applying decision-theoretic means to optimally balance possible miscoordination through inaction against possibly erroneous actions.

The second *AA team-decision challenge* arises due to the multiple levels of decision making in teams, i.e., an individual agent's decision may typically lead to negotiations and decisions within a subteam, and inputs from subteams may lead to decisions at the team level. Since individual human users interact with only individual agents, there is a key difficulty in ensuring effective team decisions. Unfortunately, despite responsible decisions by individual agents, the agent team's collective decision may still be highly undesirable. Thus, AA must consider the impact of an individual decision at multiple levels of team decision making.

Our research in AA addresses the above challenges in the context of a real-world, multi-agent system called *Electric Elves* (or E-Elves) (Pynadath *et al.* 2000), supporting everyday human collaboration for several months at USC/ISI. E-Elves assists a group of 12 researchers and one project assistant in their daily activities and provides a unique, exciting opportunity to test our ideas in a deployed team of intelligent agents[1]. To address the *AA coordination challenge*, agents in E-Elves explicitly reason about team coordination. In particular, they follow a novel three-step approach: (i) Before transferring decision-making control, an agent explicitly weighs the cost of waiting for user input and any potential team miscoordination against the cost of erroneous autonomous action; (ii) If transferring control is the best option, then an agent does not rigidly commit to this decision (as is often the case in previous work), but rather it flexibly reevaluates, sometimes reversing its decision to ensure team coordination (e.g., if an agent transfers control to a user, but the user fails to act, then the agent may act autonomously to avoid team miscoordination); (iii) Unfortunately, while an agent may appropriately regain control over a decision to avoid miscoordination, it may face significant uncertainty or cost over that decision, so rather than force a risky decision in such situations, an agent may elect to change its coordination arrangements, postponing or reordering activities, to potentially buy time to lower decision cost/uncertainty. Since each such coordination decision and change incurs various costs, agents can look ahead over the different possible sequences of changes in coordination and select the one that maximizes team benefits.

With respect to the *AA team-decision challenge*, two classes of approaches are possible. One approach is to ensure perfect individual agent decisions, such that the (sub)team reaches the desired decision. A second approach is to simplify the individual agent decision, but to introduce AA at the team level; thus, the agent team may consult a user team if the collective decision involves significant uncertainty, cost, etc. E-Elves relies on the second approach, introducing AA at multiple levels of team decision-making. A key novelty in AA at team-level decision-making is that the transfer-of-control reasoning focuses on collective team features (e.g., majority of team members) rather than specific individual features.

We have implemented the above decision-theoretic approach to AA in E-Elves using Markov Decision Processes (MDPs) (Puterman 1994). MDPs allow explicit representation of individual and team decision costs, as well as explicit representation and reasoning about uncertainty of an agent's state of knowledge or user responses. MDPs also provide a mechanism for decision-theoretic planning that allows agents to choose an optimal policy of actions representing possible coordination changes.

Electric Elves: A Deployed Multi-Agent System

The past few years have seen a revolution in the field of software agents, with agents now proliferating in human organizations, helping individuals in information gathering, activity scheduling, managing email, etc. The Electric Elves (E-Elves) effort at USC/ISI is now taking the next step: dynamic teaming of all such heterogeneous agents, as well as proxy agents for humans, to serve not just individuals, but to facilitate the functioning of entire organizations. The ultimate goal of our work is to build agent teams that assist in all organizational activities, enabling organizations to act coherently, to robustly attain their mission goals, and to react swiftly to crises. For instance, in an external crisis (e.g., a natural disaster) E-Elves agent teams may help an organization to urgently locate relevant personnel, coordinate their movement to the crisis site, coordinate the shipping of their equipment, provide them with the latest information, etc. The results of this work are potentially relevant to all organizations, including the military, disaster rescue organizations, corporations, and research institutions.

As a step towards this goal, we have had a team of 15 agents, including 13 proxies (for 13 people) running 24/7 since June 1, 2000, at USC/ISI. Each proxy is called Friday (from Robinson Crusoe's Friday), and it acts on behalf of its user in the agent team. Thus, if a user is delayed to a meeting, then Friday can reschedule that meeting by informing other Fridays, which, in turn, will inform their users. If there is an open slot to give a research presentation, Friday may volunteer or decline the invitation for that slot on behalf of its user. In addition, Friday can also order its user's meals by automatically faxing orders to local restaurants. Friday also tracks the user's location and posts this information on a web page for others to see. Friday communicates with users using different types of mobile wireless devices, such as personal digital assistants (Palm VIIs) and WAP-enabled mobile phones. A Palm VII connected to a Global Positioning Service (GPS) device allows Friday to track its user's location using wireless transmission. These mobile devices also enable easy communication between users and their Fri-

[1]E-Elves is a joint project among several research groups at USC/ISI. In this paper, we describe only the role of our own group within the overall project

day proxies.

Each Friday is based on a teamwork model, called STEAM (Tambe 1997), that helps it communicate and coordinate with other Fridays. For instance, Friday models each meeting as a team's joint intention (Cohen & Levesque 1991). Then, by the rules of STEAM, Friday agents keep each other informed of the status of this joint intention (e.g., a meeting is delayed, cancelled, etc). Furthermore, Friday uses STEAM's role relationships (e.g., *AND*, *OR*, and *role-dependency*) to model the relationships among team members attending a meeting. For instance, the role of the presenter is critical since the roles of the other attendees depend on its successful fulfillment. Thus, if the presenter cannot attend, the team recognizes a critical role failure, and requires a role substitution, i.e., the team must recruit a new presenter so that the meeting can succeed.

The design of the Friday proxy is discussed in detail elsewhere (Pynadath *et al.* 2000; Tambe *et al.* 2000; Tambe, Pynadath, & Chauvat 2000) (where they are sometimes referred to as TEAMCORE proxies). While the basic proxy design presented there still pertains, we have made several significant advances since those initial reports. For instance, we extended STEAM's method of allocating roles to team members. Previously, if there was an important role that was vacant, an agent with the relevant capability would obtain it on a first-come-first-served basis (Tambe 1997). Now, the team instead auctions off the role, allowing it to consider a more complex combination of factors in choosing a replacement. Figure 1 shows the auction tool that allows human users to view auctions in progress and to possibly intervene in such auctions. As an example, some meetings have a presenter role. Given a topic of presentation, Friday bids on behalf of its user, indicating whether its user is capable and willing to give a presentation on that topic at that time. In the auction shown in progress in Figure 1, the Friday representing Jay Modi has bid that Jay is capable of giving the presentation, but is unwilling to do so. Here, a Friday looks up its user's capability (with respect to the presentation topic) in a capability knowledge base and bids 1 or 0 on the capability as appropriate. While Friday makes this capability decision fully autonomously, it does not do so for the willingness decision (as we describe in a following section). In this specific auction, Paul Scerri's agent has submitted the highest bid, and the team indeed correctly chose Paul to be the presenter for the meeting.

AA is of critical importance to Friday agents. Clearly, the more decisions that Friday makes autonomously, the more time it save its user. Yet, given the high uncertainty in Friday's beliefs about its user's state, it could potentially make very costly mistakes while acting autonomously. For example, it may order an expensive dinner when the user is not hungry, or volunteer a busy user to give a presentation. Thus, each Friday must make intelligent decisions about when to consult its user and when to act autonomously.

The E-Elves domain itself adds to the challenge of AA. There are significant costs associated with incorrect autonomous decisions, as already discussed. Furthermore, a Friday agent faces significant uncertainty in its autonomous decision (e.g., does a user's absence from the meeting location at meeting time indicate a delay or does s/he not plan to attend at all?). In addition to uncertainty and cost, the E-Elves domain raises the two novel AA challenges mentioned in the opening section. First, consider the *AA coordination challenge*. Suppose that, when faced with uncertainty, a Friday agent consults its user (e.g., to check whether the user plans to attend a meeting), but the user is unable to respond for a long time (e.g., s/he is caught in traffic). By waiting for a response from its user, this Friday may produce miscoordination with its teammates (i.e., the other Friday agents), since its inaction means that it does not inform them whether the user will attend or not. This, in turn, means that the other meeting attendees (humans) may end up wasting their time by waiting. However, if Friday avoids the miscoordination by taking an autonomous action and this action is erroneous (e.g., erroneously informing teammates that the user will not attend), then the results are potentially disastrous for the team as well.

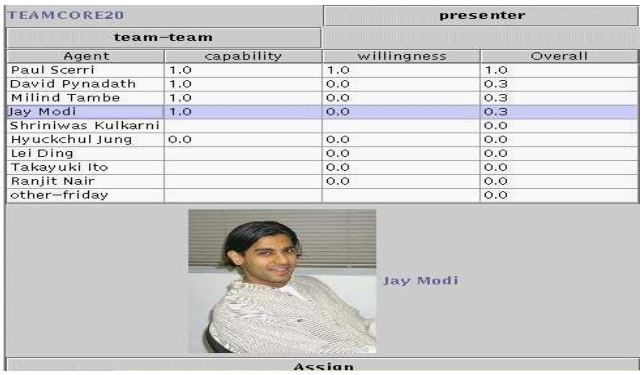

Figure 1: E-Elves auction tool.

The second *AA team-decision challenge* is to ensure effective team-level decisions. In particular, AA in individual Friday agents may ensure effective individual decisions, but the resulting team decision may still be highly undesirable. As a simple example, suppose all individual agents act responsibly, and, as a result, a team receives a reasonably high bid on an unallocated role. If not all team members have submitted bids yet, the team itself faces a difficult choice: should the team immediately assign the role to the current highest bidder, or should it wait for the possible submission of a higher bid? Waiting too long may mean that a user may not have sufficient time to prepare for their presentation, but immediate assignment may mean a suboptimal choice of a presenter.

A Decision-Tree Approach to AA

Our initial attempt at AA in E-Elves was inspired by CAP (Mitchell *et al.* 1994), the well-known agent system for advising a human user on scheduling meetings. As with CAP, Friday learned user preferences using C4.5 decision-tree learning (Quinlan 1993), although Friday's focus was on *re*scheduling meetings. In its training mode, Friday recorded values of a dozen carefully selected attributes, as well as the user's preferred action (identified by a query using a dialog

box as shown in Figure 2). Friday used the recorded data and user responses to learn a decision tree (e.g., *if* the user has a meeting with his/her advisor, but the user is not at ISI at the meeting time, *then* delay the meeting 15 minutes). In acquiring its data, Friday also queried the user if s/he wanted Friday to make this decision autonomously or not. Friday again used C4.5 to learn a second decision tree to indicate whether it should ask for user input or not. The key idea was to resolve the transfer-of-control decision by learning from user input.

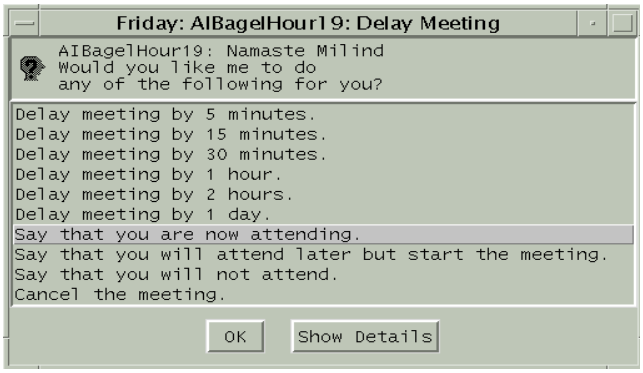

Figure 2: Dialog for delaying meetings in Electric Elves.

Initial tests based on the above setup were successful (Tambe *et al.* 2000). However, soon thereafter, one key problem became apparent: a user would suggest Friday to not take some specific decision autonomously, but then s/he would not be available to provide any input. Thus, a Friday would end up waiting for user input and miscoordinate with its team. To address this problem, timeouts were introduced so that if a user did not respond within a fixed time limit, Friday acted autonomously according to the rules of its learned decision tree. In our initial tests, the results still looked promising. Unfortunately, when we deployed the system 24/7 in our research group, it led to some dramatic failures. A few illustrative examples include the following:

1. User Tambe's Friday incorrectly cancelled a meeting with his division's director. C4.5 had overgeneralized, incorrectly taking an autonomous action from the initial set of training examples.
2. User Pynadath's Friday incorrectly cancelled the group's weekly research meeting. The time-out forced an incorrect autonomous action when Pynadath was unavailable to respond in time.
3. One of the Fridays delayed a meeting almost 50 times, each in 5 minute increments. The agent was applying its learned rule to cause a small delay each time, but ignoring the nuisance to the rest of the meeting participants.
4. Tambe's proxy automatically volunteered him for a presentation, even though he was not willing. Again, C4.5 had overgeneralized from a few examples and, because of the timeout, taken an undesirable autonomous action.

From the growing list of failures, it became increasingly clear that our original approach faced some fundamental problems. The first problem was clearly the *AA coordination challenge*, which required the agent to balance the possibility of team miscoordination against effective team action. The fixed timeouts failed to address this challenge, since the agent sometimes had to take autonomous actions when it was ill-prepared to do so, causing the problems illustrated by examples 2 and 4. Second, C4.5 was not considering the cost to the team due to erroneous autonomous actions, as seen in example 1 and 2. Third, decision-tree learning lacked the ability to look ahead and plan actions that work better over the longer term. For instance, in example 3, each 5-minute delay is appropriate for its corresponding state *in isolation*, but the C4.5 rules did not take into account the consequences of one action on future actions. Such planning could have preferred a one-hour delay instead of several 5-minute delays. Such planning and consideration of cost could also enable an agent to choose a low-cost action of delaying a meeting while it double-checks with its user regarding the higher-cost cancellation action (example 1). As a final note, this work did not address the *team-decision challenge* at all, given that it first failed to overcome the *AA coordination challenge*.

One possible solution to these problems would be to obtain more data, to find a better set of attributes, and to exploit C4.5's confidence factors to support some reasoning about uncertainty. Unfortunately, the E-Elves domain is very rich, so it would be difficult to gather the amount of training data required by C4.5, and the agents must still be able to perform reasonably in the meantime. The overall point is not that learning is inapplicable in AA, but rather that the AA problem has significant complexities in team settings that complicate domain-independent learning approaches.

Addressing Challenges in Adjustable Autonomy

To more directly address the *AA coordination challenge*, we need a mechanism that would allow us to explicitly represent and reason with the different types of costs as well as uncertainty (e.g., costs of miscoordination vs. costs of taking an erroneous action). Second, it should enable lookahead, allowing the agent to plan a systematic transfer of decision-making control and provide an optimal action policy over the longer term. Third, it should allow us to encode significant amount of initial domain knowledge in terms of costs and uncertainty, so that the agent does not have to learn everything from scratch (as C4.5 required).

MDPs fit the above requirements, and the rest of this paper describes how we use them in addressing the AA challenges. Indeed, Friday invokes the MDP-based AA reasoning for each and every decision that it makes, from rescheduling meetings, to volunteering a user to give a presentation, to ordering meals. To lessen the computational complexity of the reasoning, we rely on partitioning the MDPs. Thus, each individual agent reasons about each meeting or presentation separately, using separate MDPs for each possible decision. Overall, there are four types of MDPs used. The first subsection discusses the basic representation of these MDPs, using the *delay MDP* as an illustrative example (two other MDPs, the *bid-for-role MDP* and the *order-meal MDP*, are similar). The next subsection illustrates how these MDPs address the *AA coordination challenge*. The third subsection discusses the *assign-role MDP*,

which focuses on the *AA team-decision challenge*.

MDP Representation

This section describes our MDP representation using the delay MDP for illustration. The delay MDP represents a class of MDPs covering all types of meetings for which the agent may take rescheduling actions. For each meeting, an agent can perform any of 12 actions. In particular, it can autonomously perform any of the 10 actions shown in the dialog of Figure 2. For example, it could ask for a meeting delay of 15 minutes so that its user is in attendance on time, or it could announce the user will not attend or report that the user is in attendance etc. It can also wait and sit idly without doing anything. Finally, an agent can autonomously reduce its own autonomy and ask a user (the user can also directly issue any of the 10 commands from Figure 2).

The agent may choose to perform any of these 12 actions in various states of the world. The most salient features of the delay MDP's state space are the user's current location, which helps determine if and possibly when the user will attend the meeting, and the current time. Figure 3 shows a portion of this state space, showing just these two features, as well as the transitions between states possible for some of the available actions (a transition labeled "delay n" corresponds to the action "delay by n minutes"). Each state also contains a feature representing the number of times the meeting has previously been delayed and a feature representing what the agent has told the other attendees about the user's attendance (either nothing, that the user is currently attending, that the user will be late, or that the user will not be attending). There are a total of 768 possible states (per each individual meeting).

As shown in the figure, the delay MDP's reward function has its maximum value in state **S2** where the user is at the location of the meeting when the meeting starts. We denote the component of the reward function that focuses on a user's being at meeting time at the meeting location as r_{user}. This component gives the agent an incentive to delay meetings when the individual user is unlikely to make it to the meeting location by the originally scheduled time. However, in isolation, r_{user} could drive the agent to choose arbitrarily large delays, giving the user as much time as possible to arrive before the start of the meeting. However, there is clearly a high team cost incurred by forcing all of the attendees to rearrange their schedules. Fortunately, we can easily incorporate this team cost into the delay MDP's reward function by adding a negative reward, r_{repair} for delay actions, with the magnitude of the cost increasing with the magnitude of the delay. The magnitude is also an increasing function in the number of attendees (e.g., rescheduling a meeting of a large group is more costly than rescheduling a one-on-one meeting).

On the other hand, we do not want the rescheduling cost to scare the agent into *never* delaying meetings. In fact, in addition to the potential benefit to the individual user, explicitly delaying a meeting can produce an additional benefit to the team itself, since, without a delay, the other attendees may end up wasting their time waiting for the agent's user to arrive. Therefore, the delay MDP's reward function includes a component, r_{time}, that is negative in states after the start of the meeting, but before the user's arrival. The magnitude of this reward, like that of the team rescheduling cost, also increases in magnitude with the number of attendees. The reward function also includes an additional component r_{role}, which, like r_{user}, is positive when the user is in attendance and zero everywhere else. However, the magnitude of r_{role} increases with the importance of the user's role (e.g., speaker vs. passive participant) to the success of the meeting, thus representing the value of the user's attendance to the *team*. Finally, the reward function includes a component, $r_{meeting}$, which is positive once the meeting starts and zero everywhere else (to deter meeting cancellation). The overall reward function for a state *s* is a weighted sum of these components:

$$r(s) = \lambda_{user} r_{user}(s) + \lambda_{repair} r_{repair}(s) + \lambda_{time} r_{time}(s) + \lambda_{role} r_{role}(s) + \lambda_{meeting} r_{meeting}(s)$$

Thus, the agent reasons about different tradeoffs in team costs. This reasoning follows a fundamental tenet of teamwork in our system, that the individual team members act responsibly towards the team. Nonetheless, Friday's decisions are on behalf of only its individual user; the team may or may not concur with them. Thus, even if Friday follows its reasoning about individual and team costs in requesting a delay, the team may deliberate on it and may or may not accept it, as discussed in a following subsection.

For the agent to reason effectively, it must consider the likelihood over the possible times of the user's arrival. The delay MDP's state transition probabilities represent the likelihoods over possible user movement (e.g., from office to meeting location) in a given interval of time (as illustrated in Figure 3 by the multiple transitions possible through a "wait" action). Without any user input, the agent uses probabilities learned over observations of the user's movements in the past. In most cases, from state **S1**, the transition to **S2** (where the user arrives on time) is much more likely than the one to **S3** (where the user arrives late). The thickness of the arrows in Figure 3 represents the relative likelihood of the transitions.

The "ask" action, by which the agent gives up autonomy and queries the user, has two possible outcomes. First, the user may not respond at all, in which case, the agent is performing the equivalent of a "wait" action. We use a communication model (Tambe *et al.* 2000) to provide the probability of receiving a user's response in the amount of time left before the meeting. We also derive the cost of the "ask" action from the cost of interrupting the user, again taken from the communication model. These probabilities and costs vary with the mode of interaction. For example, a dialog box on the user's workstation is cheaper than sending a page to the user's cellular phone. A second possible outcome of the "ask" action is that the user responds with any one of the 10 responses from Figure 2. Thus, we compute an expected value of receiving user input by summing over the value of each possible response, weighted by its likelihood. The value of each user response is computed using the reward function mentioned above, but with the assumption that user response is accurate. For instance, if the user provides an

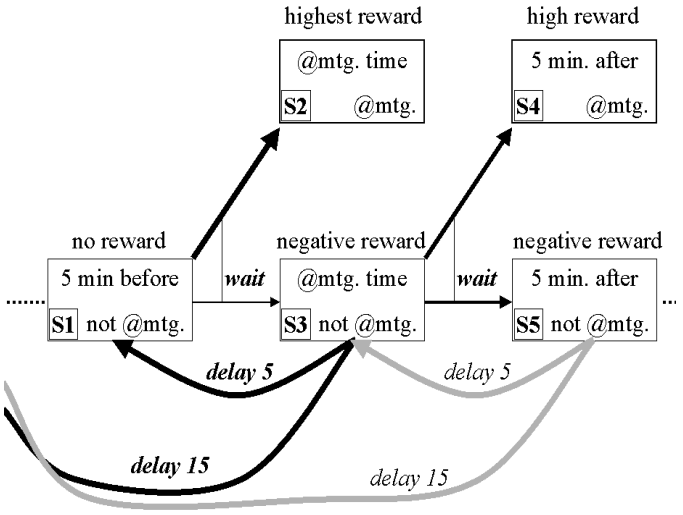

Figure 3: A small portion of simplified version of the delay MDP, with no response from the user.

input suggesting a 5 minute delay, then the agent knows that it will incur the cost of the 5-minute delay but will then receive the highest possible reward when the user arrives at the (rescheduled) meeting on time. We compute the desired likelihoods based on using the delay MDP as a (possibly erroneous) model for the user's own decision-making process.

Given the state space, action space, transition probabilities, and reward function of our MDP, we can use standard techniques like value iteration to compute the expected value of taking a particular action in a particular state. We can then use these expected values to generate a *policy* specifying the proper action for the agent to take in each and every possible state. We have designed and implemented an algorithm that computes such a policy as the basis for the agent's decision-making on when to give up its autonomy and what action to take when acting autonomously. One possible policy, generated for a subclass of possible meetings, specifies "ask" and then "wait" in state **S1** of Figure 3, which prompts the agent to give up its autonomy. If the agent then reaches state **S3**, the policy again specifies "wait", so the agent continues acting without autonomy. However, if the agent then reaches state **S5**, the policy chooses "delay 15", which the agent then executes autonomously.

Addressing the AA coordination challenge

The MDP from previous section enables Friday to address the *AA coordination challenge* using the three-step approach discussed in the opening section: (i) weighing costs of waiting for user input and team miscoordination against cost of possible erroneous actions; (ii) flexibly transferring autonomy rather than rigidly committing to an initial decision; (iii) electing to change the coordination rather than taking risky actions in uncertain states. The previous section's description of the delay MDP illustrates how the agent follows the first step in the team-related components of its reward function and its decision-theoretic approach to trading off individual preferences and input against the team's needs. This section describes how such MDPs support the other two steps of our AA coordination approach as well.

The second step of our approach requires agents to avoid rigidly committing to an initial decision about transferring decision making control. In particular, in typical approaches to AA, the agent never reconsiders its decision on transferring decision-making control. In a team context, such rigidity can cause costly miscoordination. That is, if the agent decides to give up its autonomy for a decision, it cannot wait indefinitely for the user's response as that could jeopardize the overall team activity. Instead, the agent must continuously reassess the developing situation, possibly changing its previous autonomy decision.

The MDP representation supports this second step in addressing the AA coordination challenge by providing the necessary flexibility and responsiveness in its autonomy decisions. The policies generated through value iteration specify what action is optimal in each and every possible state. Therefore, the agent can immediately respond to any state changes by following the policy's specified action for the new state. In this respect, the agent's AA decision-making is an ongoing process rather than a single decision, as the agent acts according to its MDP policy throughout the entire sequence of states it finds itself in.

The third step of our approach arises because an agent may need to regain control over decision in order to avoid miscoordination, yet it may face significant uncertainty and risk in acting autonomously. In such cases, an agent could carefully plan a change in coordination, looking ahead using different costs of team miscoordination vs. erroneous actions. In the meeting scenario, changes in coordination are essentially delaying actions. Such changes in coordination could, among other things, buy time to reduce the uncertainty or cost. The delay MDP is especially suitable for producing such a plan because it generates policies while looking ahead at all of the different outcomes.

For instance, the delay MDP supports reasoning about the

fact that the 15-minute delay gives the user more time to respond to a query by the agent and reduce its uncertainty. Since the MDP explicitly represents the possibility that the user may not respond in a limited amount of time, the computed policy is sensitive to the possibility that a "delay" followed by an "ask" may be a more effective strategy than an "ask" by itself. Of course, the policy is also sensitive to the cost of introducing the "delay" action, so it makes the optimal tradeoff in choosing between such alternate strategies in each possible state.

Furthermore, the lookahead in the value iteration algorithm enables a more effective long-term solution. For instance, the delay MDP considers a 5-minute delay in terms of its effect on the future team states, since its reward function explicitly represents the costs of rescheduling. In addition, as already mentioned, the MDP state space includes a feature representing the number of rescheduling actions already taken, and the cost of rescheduling, r_{repair}, increases as more and more such actions occur. This provides a compact summary sufficient for supporting some history dependency in the cost of future states. Thus, even if the user is very likely to arrive at the meeting in the next 5 minutes, the uncertainty associated with that particular state transition is sufficient, when coupled with the cost of subsequent delays if the user does not arrive, for the delay MDP policy to specify an initial 15-minute delay (rather than risk three 5-minute delays). Thus, the agent reasons about the likelihood of possible subsequent delays.

Addressing the AA Team-Decision Challenge

Once individual team members provide their input to the team (e.g., suggestions to delay meetings or bids for roles), the team must make a collective decision based on the input. As discussed in the opening section, the *AA team-decision challenge* requires ensuring effectiveness of such team decisions. In particular, even if individual Fridays act responsibly, the team may still make a highly undesirable collective decision. One approach, suggested in previous work, is to improve the decision making at the individual level so as to avoid such team-level problems (Hartrum & Deloach 1999). Unfortunately, this would greatly complicate the individual reasoning, because each agent would need to model and predict the actions of its teammates. We instead propose a novel approach, by introducing AA at the team-decision level, enabling the agent team to directly tap into the human team's expertise and knowledge in difficult decisions. Thus, our approach introduces AA at all levels of decision-making in a team hierarchy—at individual, subteam, and team levels. At each level, we face the same problems already discussed for the individual AA case (e.g., while consulting the human team, the agent team must not overburden the users).

In a team context the AA problem is for the team to decide whether to rely on autonomous teamwork reasoning or to relinquish the decision to the human team. As mentioned earlier, our agents rely on the STEAM teamwork model for their teamwork decisions. The integration of AA in STEAM's decisions is beneficial for the following reasons. Clearly, in some cases, STEAM's teamwork rules may have inadequate input, and any decision it reaches may have a reasonable likelihood of an error (e.g., in choosing a presenter in the absence of all bids). Thus, consulting the human team may be appropriate in such situations. Indeed, in general, we can regard any team decision that STEAM reaches as having some (low) likelihood of error. For instance, STEAM's decision based on given role-relationships (*AND*, *OR*, and *role-dependency*) may be completely accurate, but there may be potential errors in the original modeling of the role relationships. Furthermore, each such error has different cost.

Thus, analogous to the AA at the individual level, the team-based AA needs to reason about uncertainty, cost, and potential future developments in the world. Hence, we again chose an MDP as the underlying AA reasoning mechanism. The states of the team-level MDP are based on abstract team attributes rather than individual attributes (e.g., *few*, *half*, *many*, or *most* bids have been submitted for a role). This MDP once again compares the expected value of consulting with the human team against the expected value of making an autonomous decision. Here, the team-level decision to delay a meeting is of sufficiently low uncertainty and low cost that it is not worth consulting with the human team; instead, the team makes this decision. In contrast, the team-level decision to assign a presenter with only *half* the number of potential bids has sufficiently high uncertainty and cost that the agent team consults the human team. However, if the time draws sufficiently close to the the meeting time, the agent team takes an autonomous action.

Evaluation

We have used the Electric Elves system within our research group at ISI for the last six months and have used it continuously (24 hours a day, 7 days a week) since June 1, 2000 (occasionally interrupted for bug fixes and enhancements). There are twelve agent proxies belonging to twelve users, one agent proxy for a project assistant, one capability matcher (with proxy), and an interest matcher (with proxy). As described in an earlier section, the key functionality provided by this system is: (i) to monitor and reschedule meetings; (ii) to auction off research presentations; (iii) to order meals; and (iv) to track user locations. Figure 4 plots the number of daily messages exchanged by the proxies for seven months (6/1/2000-12/31/2000). The size of the daily counts demonstrates the large amount of coordination actions necessary in managing all of the activities such as meeting rescheduling, while the high variability illustrates the dynamic nature of the domain. Weekends usually have very little activity, with the notable exception of the weekends of June 3, the start date of the Agents 2000 conference.

The fact that E-Elves users were (and still are) willing to use the system over such a long period and in a capacity so critical to their daily lives is a testament to the effectiveness of the Fridays' decision making. We can also make several more specific observations. First, over the past several months, few emails have been exchanged among our group members indicating to each other that they may get delayed to meetings. Instead, Friday agents automatically address such delays. Thus, the overhead of waiting for delayed members in meeting rooms has also gone down. Sec-

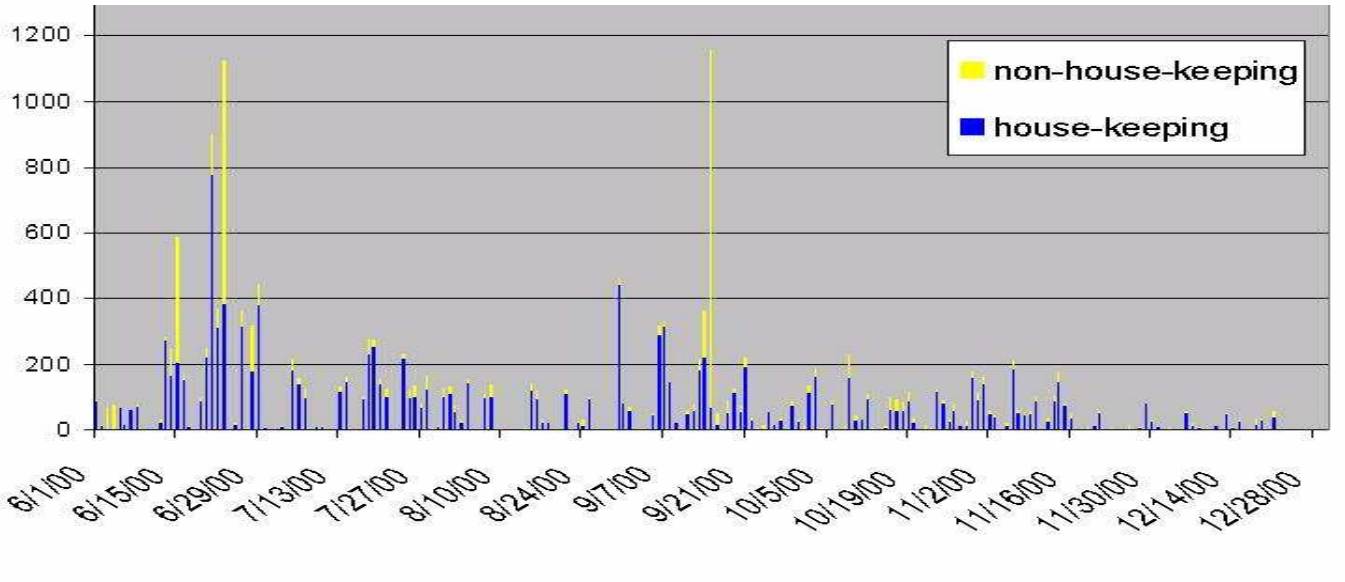

Figure 4: Number of daily coordination messages exchanged by proxies over three-month period.

ond, whereas, in the past, one of our research group members would need to circulate email trying to recruit a presenter for research meetings and making announcements, this overhead has been almost completely eliminated — weekly auctions automatically select the presenters at our research meetings. Third, a web page, where different Friday agents post their user's location, enables us to track our group members quickly, again, avoiding the overhead of trying to track them down manually. Fourth, mobile devices keep us informed remotely of changes in our activities, while also enabling us to remotely delay meeting, volunteer for presentations, order meals, etc. Finally, we have begun relying on our agents to order lunch for us.

The rest of this section presents more details regarding these general observations. Over the seven-month period, the proxies monitored a total of 1128 meetings for the 12 users. Overall, 285 of these meetings were rescheduled. Of these reschedulings, 230 occurred autonomously, so the agents are acting autonomously in a large number of instances. Equally importantly, humans are intervening in the significant remainder of the meetings, indicating the critical importance of adjustable autonomy in Friday agents.

As already mentioned, auctions decided who would give research group presentations. The system automatically opened an auction when it received notification of a meeting requiring a presentation. A summary of the results appears in Table 1 below. Column 1 shows the dates of the research presentations (over the summer, we cancelled several weekly meetings due to conference travel and vacation). Column 2 shows the total number of bids received before a decision. The key here is that the system can make auction decisions without all bids being in. In fact, in one case, the system received only 4 bids. The rest of the group simply did not bid until the winner was announced. Column 3 shows the winning bid. A winner typically bid $\langle 1, 1 \rangle$, indicating that the user is both capable and willing to the presentation. The auction MDP considers a person being both capable and willing for the task as a high-quality bid. When there was only one such bid, the MDP could confidently choose the winner; otherwise, it would wait for user input. Interestingly, the winner of July 27 had a bid of $\langle 0, 1 \rangle$, indicating that the user was not capable, but was willing. Thus, the team was able to settle on a winner despite the bid not being the highest possible, illustrating its flexibility. Finally, column 4 indicates whether the assignment was made autonomously or through human intervention. In six of the eight times, the system automatically selected the winner. There were two exceptions. First, for the meeting on July 6, human users intervened with the agent team and assigned Scerri, since they wanted a quicker resolution of the bids. Second, for the meeting on September 19, a visitor gave a talk in our research group. Agents were unable to easily accommodate a visitor proxy in the bidding, so we made his assignment manually. Automatic handling of visitors remains an issue for future work.

Auction Date	# of bids	Winner $\langle$bid$\rangle$	Autonomous?
July 6	7	Scerri$\langle 1, 1 \rangle$	No
July 20	9	Scerri$\langle 1, 1 \rangle$	Yes
July 27	7	Kulkarni$\langle 0, 1 \rangle$	Yes
August 3	8	Nair$\langle 1, 1 \rangle$	Yes
August 31	4	Tambe$\langle 1, 1 \rangle$	Yes
Sept 19	6	Visitor$\langle -, - \rangle$	No
Oct 31	7	Tambe$\langle 1, 1 \rangle$	Yes
Nov 21	7	Nair$\langle 1, 1 \rangle$	Yes

Table 1: Table showing results for auctioning research presentation slot.

We also performed a number of experiments to illustrate

the MDPs' abilities to react to changes in the environment. As expected, as one increases the *cost* of asking the user, the number of situations in which the agent would relinquish autonomy decreases (Figure 5). Furthermore, as the *likelihood* of the user replying to an agent decreases, so does the number of states in which the agent would ask (Figure 6). This demonstrates that the agent is able to trade off intelligently between the wasted effort if the user does not respond and the value of the information the user could provide.

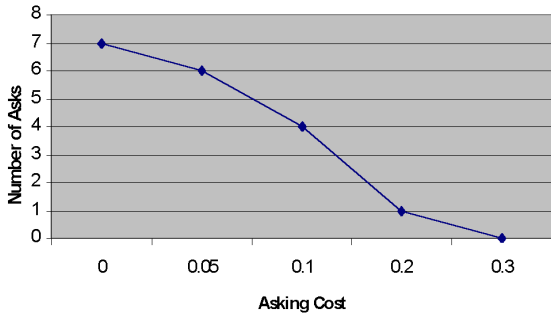

Figure 5: Changes in the number of states where the agent will ask the user for input when the cost of asking is changed.

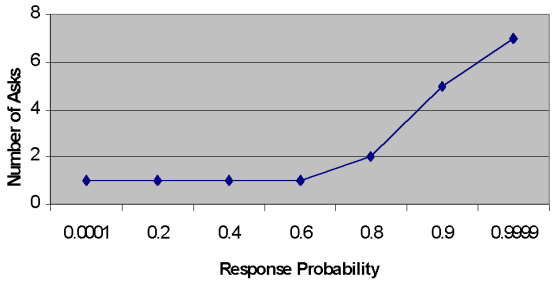

Figure 6: Changes in the number of states where the agent will ask the user when the probability that the user responds changes.

Related Work

This section discusses some of the related work that has influenced our research. However, as discussed earlier, most of this work has focused on an interaction between an individual agent and an individual human, while our research has focused on adjustable autonomy in the team context, raising new challenges such as the AA coordination challenge and the AA team-decision challenge.

Mitchell's CAP system (Mitchell *et al.* 1994), an intelligent interface to an online calendar, provided the initial inspiration for our AA work. CAP learns a decision-tree to capture "general regularities" in the way the user schedules meetings then uses the decision tree to make suggestions when new meetings are to be scheduled. However, although successful for the calendar application, as discussed above, the decision-tree approach did not scale to the challenges of AA in a team context.

Mixed-Initiative (MI) systems share the responsibility for a tasks between a human and agent. TRAINS-95 (Ferguson, Allen, & Miller 1996), for example, allows collaboration between an agent and a user in the performance of a fairly complex planning task. The planning problem is split up so that the overall system leverages the abilities of the respective parties. For example, it gives problems having hard-to-define objective functions to the user and problems involving consideration of many small details to the machine. In such systems there is generally only a single user, assumed to be continually available, and only a single agent; hence, the team-decision and coordination challenges are avoided.

One aspect of Friday's decision to relinquish autonomy is an assessment of the costs and benefits to the individual user. The PRIORITIES system also uses decision theory to reason about the costs, benefits and uncertainty associated with alerting a user to the arrival of new email (Horvitz, Jacobs, & Hovel 1999). The reasoning does not, however, consider the team-level issues that are considered in E-Elves. One of the focuses of the PRIORITIES research is to use Bayesian networks to build a probabilistic model of the user's intentions, so as to better assess the costs involved in interrupting them. That research complements the E-Elves work, which currently makes very simple assumptions about the intentions of the users.

Some of the major research in AA has been motivated by the requirements of NASA space missions (Dorais *et al.* 1998; Kortenkamp *et al.* 1999). An important thread of that research has been the development of an interface layer to the 3T architecture which provides mechanisms that allow users to take back control of an agent at whatever level of control is most appropriate to the situation (Schreckenhost 1999). This interface layer approach is fundamentally different to the E-Elves approach, as the agent does not explicitly reason about reducing own autonomy. Similarly, the AA coordination or team-decision challenges have not yet needed to be addressed in this domain.

Another area of application for AA mechanisms is interactive theater (Hayes-Roth, Brownston, & van Gent 1997). Here, avatars controlled by agents can be directed interactively to carry out some scenario. The actions the agents actually take are influenced by the director, the environment and built-in personality traits. However, interactive theater agents do not reason about uncertainty, costs and benefits or the preferences of the user for AA, nor are team-coordination or learning issues important.

Summary

Gaining a fundamental understanding of adjustable autonomy (AA) is critical if we are to deploy multi-agent systems in support of critical human activities. Indeed, living and working with the E-Elves has convinced us that AA is a critical part of any human-collaboration software. No matter how carefully we design our agents, we cannot allow them to operate with complete autonomy in their decisions on behalf of a user. Hence, our research has focused on applying decision-theoretic mechanisms to optimize AA in rich

environments such as E-Elves, where agent teams are embedded within a human organization. Agent teamwork and coordination in such environments introduce critical novel challenges in AA that previous work has not addressed. We focused on two key challenges: (i) the *AA coordination challenge* requires an agent to avoid miscoordination with teammates, while simultaneously ensuring effective team action; and (ii) the *AA team-decision challenge* arises due to multiple levels of decision making in teams and focuses on ensuring effective team decisions.

We proposed several key ideas to resolve these challenges. In particular, for resolving the *AA coordination challenge* agents explicitly reason about costs of team miscoordination, they flexibly transfer autonomy rather than rigidly committing to initial decisions, and they may change the coordination rather than taking risky actions in uncertain states. For addressing the *AA team-decision challenge*, we introduced AA at multiple levels of the team decision-making hierarchy. We have implemented this reasoning through MDPs.

Our research seeks to fundamentally improve our understanding of AA in complex multi-agent environments. The work presented in this paper provides new techniques to address the coordination and team-decision challenges in AA. In our future work, we plan to generalize our solutions to arrive at an overall framework that we can apply to new AA problems as they arise in the E-Elves domain. We also plan to investigate two techniques to improve agent autonomy: (i) learning an improved user model, and (ii) explaining agent decisions to users. Each of these problems will raise further novel challenges that we look forward to addressing in our future work in this domain.

Acknowledgments This research was supported by DARPA awards No. F30602-98-2-0108 under the Control of Agent-Based Systems program. The effort is being managed by ARFL/Rome Research Site.

References

Cohen, P. R., and Levesque, H. J. 1991. Teamwork. *Nous* 35.

Collins, J.; Bilot, C.; Gini, M.; and Mobasher, B. 2000. Mixed-initiative decision-support in agent-based automated contracting. In *Proceedings of the International Conference on Autonomous Agents.*

Dorais, G. A.; Bonasso, R. P.; Kortenkamp, D.; Pell, B.; and Schreckenghost, D. 1998. Adjustable autonomy for human-centered autonomous systems on mars. In *Proceedings of the International Conference of the Mars Society*.

Ferguson, G.; Allen, J.; and Miller, B. 1996. TRAINS-95: towards a mixed initiative planning assistant. In *Proceedings of the Conference on Artificial Intelligence Planning Systems*, 70–77.

Call for Papers, 1999. AAAI spring symposium on adjustable autonomy. www.aaai.org.

Gunderson, J. P., and Martin, W. N. 1999. Effects of uncertainty on variable autonomy in maintenence robots. In *Proceedings of the International Conference on Autonomous Agents, Workshop on Autonomy Control Software*.

Hartrum, T., and Deloach, S. 1999. Design issues for mixed-initiative agent systems. In *Proceedings of the AAAI Workshop on Mixed-Initiative Intelligence*.

Hayes-Roth, B.; Brownston, L.; and van Gent, R. 1997. *Readings in Agents*. Morgan Kaufmann. Multiagent collaboration in directed improvization, 141–147.

Horvitz, E.; Jacobs, A.; and Hovel, D. 1999. Attention-sensitive alerting. In *Proceedings of the Conference on Uncertainty and Artificial Intelligence*, 305–313.

Kortenkamp, D.; Burridge, R.; Bonasso, R. P.; Schreckenghost, D.; and Hudson, M. B. 1999. Adjustable autonomy issues for control of robots. In *Adjustable Autonomy Workshop, IJCAI'99*.

Lesser, V.; Atighetchi, M.; Benyo, B.; Horling, B.; Raja, A.; Vincent, R.; Wagner, T.; Xuan, P.; and Zhang, S. X. 1999. A multi-agent system for intelligent environment control. In *Proceedings of the International Conference on Autonomous Agents.*

Mitchell, T.; Caruana, R.; Freitag, D.; McDermott, J.; and Zabowski, D. 1994. Experience with a learning personal assistant. *Communications of the ACM* 37(7):81–91.

Puterman, M. L. 1994. *Markov Decision Processes*. John Wiley & Sons.

Pynadath, D. V.; Tambe, M.; Arens, Y.; Chalupsky, H.; Gil, Y.; Knoblock, C.; Lee, H.; Lerman, K.; Oh, J.; Ramachandran, S.; Rosenbloom, P. S.; and Russ, T. 2000. Electric Elves: Immersing an agent organization in a human organization. In *AAAI Fall Symposium on Socially Intelligent Agents: The Human in the Loop*, 150–154.

Quinlan, J. R. 1993. *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Schreckenhost, D. 1999. Human interaction with control software supporting adjustable autonomy. In Musliner, D., and Pell, B., eds., *Agents with adjustable autonomy*, AAAI 1999 spring symposium series.

Tambe, M.; Pynadath, D. V.; Chauvat, N.; Das, A.; and Kaminka, G. A. 2000. Adaptive agent integration architectures for heterogeneous team members. In *Proceedings of the International Conference on MultiAgent Systems*, 301–308.

Tambe, M.; Pynadath, D.; and Chauvat, N. 2000. Building dynamic agent organizations in cyberspace. *IEEE Internet Computing* 4(2):65–73.

Tambe, M. 1997. Towards flexible teamwork. *Journal of Artificial Intelligence Research* 7:83–124.