Adjustable Autonomy: From Theory to Implementation

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

1. INTRODUCTION

Recent exciting, ambitious applications in agent technology involve agents acting individually or in teams in support of critical activities of individual humans or entire human organizations. Applications range from intelligent homes [13], to "routine" organizational coordination[16], to electronic commerce[4] to long-term space missions[12, 6]. These new applications have brought forth an increasing interest in agents' *adjustable autonomy* (AA), i.e., in agents' *dynamically adjusting their own level of autonomy based on the situation*[8]. In fact, many of these applications will not be deployed, unless reliable AA reasoning is a central component. At the heart of AA is the question of whether and when agents should make autonomous decisions and when they should transfer decision-making control to other entities (e.g., human users).

Unfortunately, previous work in adjustable autonomy has focused on individual agent-human interactions and the techniques developed fail to scale-up to complex heterogeneous organizations. Indeed, as a first step, we focused on a small-scale, but real-world agent-human organization called *Electric Elves*, where an individual agent and human worked together within a larger multiagent context. Although the application limits the interactions among entities, key weaknesses of previous approaches to adjustable autonomy are readily apparent. In particular, previous approaches to transfer-of-control are seen to be too rigid, employing one-shot transfers-of-control that can result in unacceptable coordination failures. Furthermore, the previous approaches ignore potential costs (e.g., from delays) to an agent's team due to such transfers of control.

To remedy such problems, we propose a novel approach to AA, based on the notion of a *transfer-of-control strategy*. A transfer-of-control strategy consists of a conditional sequence of two types of actions: (i) actions to transfer decision-making control (e.g., from the agent to the user or vice versa) and (ii) actions to change an agent's pre-specified coordination constraints with team members, aimed at minimizing miscoordination costs. The goal is for high quality individual decisions to be made with minimal disruption to the coordination of the team. We operationalize such strategies via Markov decision processes (MDPs) which select the optimal strategy given an uncertain environment and costs to individuals and teams. We have developed a general reward function and state representation for such an MDP, to facilitate application of the approach to different domains. We present results from a careful evaluation of this approach, including via its use in our real-world, deployed Electric Elves system.

2. ADJUSTABLE AUTONOMY – THE PROBLEM

In the following, a formal definition of the AA problem is given so as to clearly define the task of the AA reasoning. A team, which may consist of agents and users, has some joint activity, α, which the entities work cooperatively towards. The primary task of the agent is the success of α which it pursues by performing some role, ρ. Performing ρ requires that one or more non-trivial decisions are made. To make a decision, d, the agent can draw upon entities from a set $E = \{e_1 \ldots e_n\}$. Each entity in E, though not necessarily in the team, is capable of making decision d. Typically, the agent can also make the decision itself. Different entities will have differing abilities to make the decisions due to, e.g., available computational resources or access to relevant information. The decision is made in a context Π, that includes both the environment and any other tasks being performed by related entities. The agent will often not have complete information about Π. Coordination *constraints*, $\asymp$, exist between ρ and the roles of other members of the team, e.g., various roles might need to be executed simultaneously or within some total cost. A critical facet of the successful completion of the joint task is ensuring that coordination between team members is maintained, i.e., $\asymp$ are not violated. Thus, we can describe an AA problem instance with the tuple: $\langle A, \alpha, \rho, \asymp, d, E, \Pi \rangle$.

From an AA perspective an agent can take two types of actions. The first type of AA action is to transfer control to an entity in E. In general, there are no restrictions on when, how often or for how long decision making control can be transferred to a particular entity. In general, we assume that when the agent transfers control it does not have any guarantee on the timeliness or quality of the decision made by the entity to which control is transferred, indeed in many cases that entity will not make the decision at the time required by the coordination constraints. The second type of action that an agent can take is to change the coordination constraints, $\asymp$. A coordination change might involve changing the timing of tasks or changing the role, ρ, or even the team plan. Changing $\asymp$ has some cost, though it may be better to incur that cost than violate coordination constraints. Thus, given a problem instance, $\langle A, \alpha, \rho, \asymp, d, E, \Pi \rangle$, the agent must decide whether to transfer control or act autonomously or change coordination constraints to maximize the overall expected utility of the team.

2.1 The Electric Elves

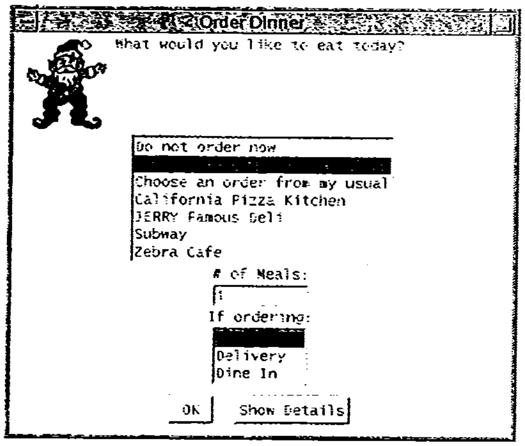

Figure 1: Friday asking the user for input regarding ordering a meal.

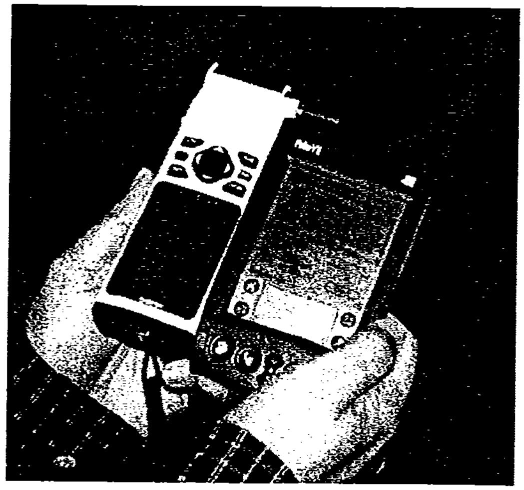

Figure 2: Palm VII for communicating with users and GPS device for detecting their location.

This research was initiated in response to issues that arose in a real application and the resulting approach was extensively tested in the day-to-day running of that application. In the following, the application and an early failed approach to implementing AA reasoning are presented in order to motivate the eventual solution. The Electric Elves (E-Elves) is a project at USC/ISI to deploy an agent organization in support of the daily activities of a human organization[3]. The operation of a human organization requires dozens of everyday tasks to ensure coherence in organizational activities, e.g., monitoring the status of activities, gathering information and keeping everyone informed. Teams of software agents can aid organizations in accomplishing these tasks, facilitating coherent functioning and rapid, flexible response to crises. While a number of underlying AI technologies support E-Elves[16, 3], AA emerges as the central research issue in agent-human interactions.

In E-Elves, each user is supported by an agent proxy, called Friday (after Robinson Crusoes' man-servant Friday) that acts on their behalf in the agent team (see [23] for details of Friday's design). Friday can perform a variety of tasks for its user. If a user is delayed to a meeting, Friday can reschedule the meeting, informing other Fridays, who in turn inform their users. If there is a research presentation slot open, Friday may respond to the invitation to present on behalf of its user. Friday can also order its user's meals (see Figure 1) and track the user's location, posting it on a Web page. Friday communicates with users using wireless devices, such as personal digital assistants (PALM VIIs) (see Figure 2) and WAP-enabled mobile phones, and via user workstations. Each Friday's team behavior is based on a teamwork model, called STEAM[22]. STEAM encodes and enforces the constraints between roles that are required for the success of a joint activity, e.g., meeting attendees should arrive at a meeting simultaneously.

AA is critical to E-Elves since, despite the range of sensing devices, Friday has considerable uncertainty about the user's intentions and location. Thus, it is somewhat risky for Friday to make decisions on behalf of the user; yet, it cannot continually ask the user for input, given that user's time is valuable. There are currently four decisions in E-Elves where AA reasoning is applied: (i) whether the user is willing to perform a task in its team, (ii) if and what to order for lunch, (iii) selecting a presenter for a team meeting, (iv) rescheduling meetings (which we focus on here). In the meeting context, the AA problem can be described as follows: the meeting is α, while Friday's role, ρ, is to ensure that the user arrives at the meeting at the same time as other users. Friday may reschedule the meeting (i.e., changing coordination) as needed. Friday can transfer control to a human user (the set E = {user, Friday}) to seek user input about the meeting, thus, creating a problem instance, $\langle A, \alpha, \rho, \asymp, d, E, \Pi \rangle$. The challenge for AA here is as follows: If Friday acts autonomously despite the uncertainty and takes an incorrect action on behalf of the user (e.g., saying the user will not attend the meeting), the other attendees may unnecessarily cancel the meeting. If Friday transfers control to the user and waits for her input, and if she is unable to provide timely input (e.g., she is stuck in traffic), there may be significant miscoordination, as other meeting attendees may unnecessarily wait at the meeting location. Thus, the AA challenge for Friday is to avoid making errors, while also avoiding miscoordination due to transfers of control – this last part about miscoordination is a novel challenge for AA in team settings, such as E-Elves.

Everyday coordination at university research groups, commercial businesses and governmental organizations is not the only coordination that can benefit from such agent technology. Unfortunate natural and man-made disasters require coordination of many people and organizations, cooperating on many joint tasks[5]. Efficient, coordinated leveraging of both physical and decision making resources will lead to the most effective response to the disaster. Facilitating the details of this coordination can be undertaken by teams of intelligent agents with an AA capability. For example, a team of robots might be assigned the task of searching an area of the city for survivors. A robot may transfer the responsibility for planning a route to the area to a satellite with a better view of the city. Some coordination with other robots in the team may be required to ensure that the robot most in need of the satellites resources has access to them first. If it cannot get a response from the satellite, the robot may ask a human user at a central control center or it may need to plan its route autonomously. In some cases the robot might reason that the team is better off it if exchanges roles with another robot, since it has a good known route available to the other robot's search location. In such a scenario the robot performs several transfers of control and perhaps a coordination change to maximize the performance of the overall team.

2.2 Decision-tree approach

One logical avenue of attack on the AA problem for E-Elves was to apply an approach used in a previously reported, successful meeting scheduling system, in particular CAP[14]. Like CAP, Friday learned user preferences using C4.5 decision-tree learning [17]. Friday recorded values of a dozen carefully selected attributes and the user's preferred action (identified by asking the user) whenever it had to make a decision. Friday used the data to learn a decision tree that encoded its autonomous decision making. For AA, Friday also asked if the user wanted such decisions taken autonomously in the future. From these responses, Friday used C4.5 to learn a second decision tree which encoded its rules for transferring control. Initial tests with the approach were promising [23], but a key problem soon became apparent. When Friday encountered a decision for which it had learned to transfer control to the user, it would wait indefinitely for the user to make the decision, even though this inaction could lead to miscoordination with teammates if the user did not respond or attend the meeting. To address this problem, if a user did not respond within a fixed time limit, Friday took an autonomous action.

Although performance improved, when the resulting system was deployed 24/7, it led to some dramatic failures. One such failure occurred when one user's Friday incorrectly cancelled the group's weekly research meeting when a time-out forced the choice of an risky autonomous action (and the action turned out to be wrong). On another occasion, a Friday delayed a meeting almost 50 times, each time by 5 minutes. It was correctly applying a learned rule but ignoring the nuisance to the rest of the meeting participants. It turns out that AA in a team context requires more careful reasoning about the costs and benefits of acting autonomously and transferring control. In particular, an agent needs to be more flexible in its AA reasoning, not restricting itself to a single transfer of control and a fixed timeout. Moreover, it needs to plan ahead to find sequences of actions that handle various contingencies that might arise and take into account costs to the team. (In theory, using C4.5, Friday might have eventually been able to handle the complexity of AA in a multiagent environment, but a very large amount of training data would be required, even for this relatively simple decision.)

3. MODELING TRANSFER OF CONTROL STRATEGIES

To avoid rigid one-shot transfers of control and allow team costs to be considered we introduce the notion of a *transfer-of-control strategy*. A transfer-of-control strategy consists of a conditional sequence of two types of actions: (i) actions to transfer decision-making control (e.g., from the agent to the user or vice versa) and (ii) actions to change an agent's pre-specified coordination constraints with team members, aimed at minimizing miscoordination costs. An agent executes such a strategy by performing the actions in sequence, transferring control to the specified entity and changing coordination as required, until some point in time when the entity currently in control exercises that control and makes the decision. Given a problem instance, $\langle A, \alpha, \rho, \asymp, d, E, \Pi \rangle$, agent A can transfer decision-making control for d to any entity $e_i \in E$, and we denote such a transfer-of-control action with the symbol e_i. When the agent transfers decision-making control to an entity, it may stipulate a time limit for a response from that entity. To capture this additional stipulation, we denote transfer-of-control actions with a time limit as an action $e_i(t)$, i.e., e_i has decision-making control for a maximum time of t.[1] Such an action has two possible outcomes: either e_i responds before time t and makes the decision, or it does not respond and decision d remains unmade at time t. In addition, the agent has some action through which it can change coordination constraints, which we denote $\mathcal{D}$.

Since the outcome of a transfer-of-control action is uncertain and some potential outcomes are undesirable, an agent needs to carefully consider the potential consequences of its actions and plan for the various contingencies that might arise. Moreover, the agent needs to consider sequences of transfer-of-control actions to properly deal with a single decision. Considering multi-step strategies allows an agent to exploit decision making sources considered too risky to exploit without the possibility of retaking control. For example, control could be transferred to a very capable but not always available decision maker then taken back if the decision was not made before serious miscoordination occurred. More complex strategies, possibly including several changes in coordination constraints, can provide even more opportunity for obtaining high quality input. For instance, the strategy $H(5)A$ would specify that the agent first give up control and ask entity H. If the H responds with a decision within 5 minutes, then the task is complete. If not, then the agent proceeds to the next transfer-of-control action in the sequence, in this case transferring control to A (denoting itself). We can define the space of all possible strategies as follows:

$$\mathrm{S} = (E \times \mathcal{R}) \times \bigcup_{n=0}^{\infty} ((E \times \mathcal{R}) \cup \{\mathcal{D}\})^n \tag{1}$$

To select between strategies we compare the expected utility (EU) of the candidate strategies. The calculation of a strategy's EU takes into account the benefits, i.e., likely relative quality of different entities' decisions and the probability of getting a response from an entity at a particular time, and the costs, i.e., the cost of delaying a decision and the costs of changing coordination constraints. The first element of the EU calculation is the expected quality of an entity's decision. In general, we capture the quality of an entity's decision at time t with the functions $\mathbf{EQ} = EQ_e^d(t) : \mathcal{R} \rightarrow \mathcal{R}$. The quality of a decision reflects both the likelihood that the entity will make an "appropriate" decision and the costs

[1]For readability, we will frequently omit the time specifications from the transfer-of-control actions and instead write just the order in which the agent transfers control among the entities and executes $\mathcal{D}$s (e.g., e_1e_2 instead of $e_1(5)e_2$).

incurred if the decision if the decision is wrong. We assume the agent has a model of $EQ^d_e(t)$. The second element of the EU calculation is a representation of the probability an entity will respond if control is transferred to it. The functions, $\mathbf{P} = \{P^e_\top(t) : \mathcal{R} \rightarrow [0,1]\}$, represent continuous probability distributions over the time that the entity e will respond, i.e., the probability that e_i will respond at time t_0 is $P^{e_i}_\top(t_0)$. The final element of the EU calculation is a representation of the cost of inappropriate timing of a decision. In general, not making a decision until a particular point in time occurs some cost that is a function of both the time and the coordination constraints, $\asymp$, between team members. We focus on cases of constraint violations due to delays in making decisions. Thus, the cost is due to the violation of the constraints caused by not making a decision until that point in time. We can write down a *wait-cost function* function: $\mathbf{W} = f(\asymp, t)$ which returns the cost of not making a decision until t given coordination constraints, $\asymp$. We assume that there is some point in time, $\lhd$, after which no more costs accrue, i.e., if $t \geq \lhd$ $f(\asymp, t) = f(\asymp, \lhd)$.[2] Finally, we assume that, in general, until $\lhd$ the wait cost function is non-decreasing, reflecting the idea that bigger violations of constraints lead to higher wait costs. The coordination change action, $\mathcal{D}$, reduces the wait costs that are incurred from the time the action is taken, onwards. For example, a $\mathcal{D}$ action might be to change the order in which two roles are performed, thereby changing the time at which decisions in the roles need to be made, but at the cost of reorganizing the team. We represent the effect of the $\mathcal{D}$ by letting $\mathcal{W}$ be a function of $t - \mathcal{D}_{value}$ (i.e., $\mathcal{D}_{value}$ is the value of the $\mathcal{D}$ action) after the $\mathcal{D}$ action, although other models might also be used. We represent the $\mathcal{D}$ as having a fixed cost, $\mathcal{D}_{cost}$, incurred immediately upon its execution, although, again, more complex models might be used.

Using the three elements, i.e., expected decision quality, probability of response and wait costs, we can compute the EU of an arbitrary strategy, s. The total utility is the quality of decision being made by the entity in control minus the costs incurred from waiting, i.e., $EU^d_{e_c}t = EQ^d_e(t) - \mathcal{W}(t)$. If a coordination change action has been taken it will also have an effect on utility. If a single $\mathcal{D}$ action has been taken at $t = \Delta$ the second component of the EU calculation is: $\mathcal{W}(t|\mathcal{D}) = \mathcal{W}(\Delta) - \mathcal{W}(\Delta - \mathcal{D}_{value}) + \mathcal{W}(t - \mathcal{D}_{value}) + \mathcal{D}_{cost}$. To calculate the EU of an arbitrary strategy, we multiply the probability of response at each instant of time by the EU of receiving a response at that instant, and then integrate over the products. Hence, the EU for a strategy s is: $EU_s = \int_0^\infty P_\top(t') EU^d_{e_c}(t')\, .dt'$. To correctly calculate the EU of a strategy, we need to ensure that the probability of response function and the wait-cost calculation reflect the control situation at that point in the strategy. To achieve this, the integral can be broken into terms, one term for each action in the strategy. For example, for a strategy $e(T)A$ there would be one term for when e has control and another for when A has control, i.e.:

$$EU^d_{eA}t = \int_0^T P_\top(t') \times (EQ^d_e(t') - \mathcal{W}(t')).dt' + \int_T^\infty P_\top(t') \times (EQ^d_e(t') - \mathcal{W}(t')).dt' \quad (2)$$

More complex strategies have more complex EU equations, e.g., for $e(\Delta)\mathcal{D}e(T)A$ we have:

$$\begin{aligned} EU^d_{e\mathcal{D}eA} = {} & \textstyle\int_0^\Delta P_\top(t')(EQ^d_e(t') - \mathcal{W}(t')).dt' + \\ & \textstyle\int_\Delta^T P_\top(t')(EQ^d_e(t') - \mathcal{W}(\Delta) + \mathcal{W}(\Delta - \mathcal{D}_{value}) - \\ & \mathcal{W}(t' - \mathcal{D}_{value}) - \mathcal{D}_{cost}).dt' + \\ & \textstyle\int_T^\infty P_\top(t')(EQ^d_A(t') - \mathcal{W}(\Delta) + \mathcal{W}(\Delta - \mathcal{D}_{value}) - \\ & \mathcal{W}(T - \mathcal{D}_{value}) - \mathcal{D}_{cost}).dt' \end{aligned} \quad (3)$$

In the above EU equations the functions $EQ^d_e(t)$, $P_\top(t)$ and $\mathcal{W}(t)$ are all general functions, i.e., there is no limitation on the nature of the functions. By instantiating these functions with concrete ones we can calculate an exact number for the EU of a strategy. For example, if we use a Markovian response probability function for the user ($P^t_\top(U) = \epsilon \exp^{-\epsilon t}$), an exponential wait cost function ($\mathcal{W}(t) = \omega \exp^{\omega t}$) and constant decision making ability ($EQ^d_A(t) = \alpha$ and for $EQ^d_U(t) = \beta$), Equation 4 becomes:

$$\begin{aligned} EU^d_{e\mathcal{D}eA}t = {} & \tfrac{\epsilon\omega}{\delta}(exp^{-\Delta\delta} - 1) + \beta(1 - \exp^{-\Delta\epsilon}) + \\ & \tfrac{\epsilon\omega \exp^{-\omega\mathcal{D}_{value}}}{\delta}(\exp^{-T\delta} - \exp^{-\Delta\delta}) + \\ & (\mathcal{D}_{cost} - \beta)(\exp^{-\epsilon T} - \exp^{-\epsilon\Delta}) + \\ & \omega \exp^{\Delta\omega}(\exp^{-\omega\mathcal{D}_{value}} - 1)(\exp^{-\epsilon\Delta} - \exp^{-\epsilon T}) - \\ & \exp^{-\epsilon T}(\mathcal{D}_{cost} - \alpha + \omega(\exp^{\omega\Delta} - \\ & \exp^{\omega(\Delta - \mathcal{D}_{value})} + \exp^{\omega(T - \mathcal{D}_{value})})) \end{aligned} \quad (4)$$

Given the ability to calculate the EU of transfer-of-control strategies, the AA problem for the agent reduces to finding and following the transfer-of-control strategy that maximizes its EU. Formally, the agent's problem is:

Definition 3.1 *For a problem $\langle A, \alpha, \rho, \asymp, d, E, \Pi \rangle$, the agent must select $s \in \mathbf{S}$ such that*
$\forall s' \in \mathbf{S}, s' \neq s, EU_s^{\langle A,\alpha,\rho,\asymp,d,E,\Pi \rangle} \geq EU_{s'}^{\langle A,\alpha,\rho,\asymp,d,E,\Pi \rangle}$

Ideally, we wish to select the right strategy for a domain such as E-Elves and use the strategy throughout. Unfortunately, we can show that even with fixed functions, the best strategy in one situation may not be best in another. In fact, the strategy with the highest EU may vary greatly from situation to situation. Below we give three Lemmas which show when various types of strategies are useful. The Lemmas lead to the conclusion that complex strategies are not necessarily superior to single-shot strategies, even in a multi-agent context, and that in fact no particular strategy dominates all other strategies.

One requirement of a strategy is to have the agent strike the right balance between not waiting indefinitely for a user response and not taking a risky autonomous action. The agent should reason that it will eventually make a decision (either after giving control to the user or immediately) if the expected costs of continued waiting exceeds the difference between the user's decision quality and its own. In fact, we can state:

LEMMA 1: *If $s \in \mathbf{S}$ is a strategy ending with $e \in E$, and s' is sA, then $EU^d_{s'} > EU^d_s$ iff $\forall e \in E, \exists t < \lhd$ such that $\int_t^\lhd P_\top(t')\mathcal{W}(t').dt' - \mathcal{W}(t) > EQ^d_e(t) - EQ^d_A(t)$*

Lemma 1 says that if at any point in time the expected costs of indefinitely leaving control in the hands of the user

[2]This implies that the total cost of delaying a decision is finite provided $f(\asymp, t)$ is finite.

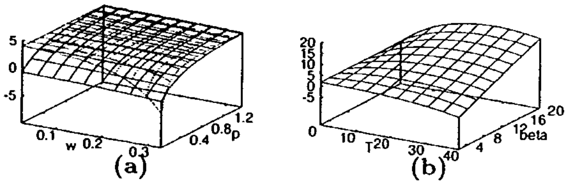

Figure 3: (a) EU of strategies $e\mathcal{D}eA$ and e (dotted line is e). (b) Strategy eA plotted against T (transfer time) and *beta* (the user's decision quality).

exceed the difference between the quality of the agent and user's decision then strategies which ultimately give the agent control dominate those which do not. An interesting consequence of the Lemma is that when the *expected* wait costs do not change over time the agent could leave control in the hands of a user even though very high total costs are eventually incurred, i.e., it may choose a simple one shot strategy. Using similar types of arguments we can prove two further Lemmas:

LEMMA 2: *if $s \in \mathbf{S}$ has no $\mathcal{D}$ and s' is s with a $\mathcal{D}$ added then* $EU^d_{s'} > EU^d_s$ **iff**
$\int P_\top(t')\mathcal{W}(t).dt' - \int P_\top(t')\mathcal{W}(t|\mathcal{D}).dt' > \mathcal{D}_{cost}$

LEMMA 3: $\forall K \in \mathbf{N}, \exists \mathcal{W}(t) \in \mathbf{W}, \exists \mathrm{P}_\top(t) \in \mathbf{P}, \exists \mathrm{EQ}^d_e(t) \in \mathbf{EQ}$ *such that K is the optimal number of $\mathcal{D}$s*

Lemmas 1-3 show that no particular transfer-of-control strategy dominates. Moreover, very different strategies, from single-shot strategies to arbitrarily complex strategies, are appropriate for different situations. The utility gained by taking different transfer-of-control actions is not always obvious and the range of situations where an action provides benefit can be quite narrow. Hence, careful attention must be paid to the specific situation when choosing a strategy to follow.

By plotting the EU equations against various parameters, we can see that the EU of a strategy depends on a variety of factors and that for different parameters different strategies have higher EU. Using the instantiation of functions described above, Figure 3(a) plots the EU of the strategies $e\mathcal{D}eA$ and e as the rate of wait cost accrual (w) and the probability of response parameter (p) are varied. Notice that for some values of the parameters, strategy e has higher EU, while for others strategy $e\mathcal{D}eA$ has higher EU. In Figure 3(b) the EU of the strategy eA is plotted varying the time at which control is transferred from the user to the agent (T) and the quality of the user's decision making (beta). The figure shows that the EU of the strategy is sensitive to these parameters (as it is to other parameters.)

4. FROM THEORY TO IMPLEMENTATION

While the mathematical model of strategies for AA presented above clearly illustrates that different strategies dominate in different situations, it does not directly provide a means for operationalizing AA reasoning. Some mechanism is required to choose an appropriate strategy given the current situation. The mechanism should also execute the strategy and dealing with contingencies the strategy does not handle, e.g., unexpected changes in expected response time due to a user changing location. MDPs were chosen as the mechanism for operationalizing such reasoning. We can interpret the policy produced by using standard algorithms on the MDP[15] as many transfer-of-control strategies, with the strategy to follow chosen depending on the current state. The overall state, within the MDP representation for an AA problem, $\langle A, \alpha, \rho, \asymp, d, E, \Pi\rangle$, has the following features (recall that ρ is the agent's role with the team's joint activity, α):

- *controlling-entity* is the entity that currently has decision making control.
- *team-orig-expect-ρ* is what the team originally expected of the fulfilling of ρ.
- *team-expect-ρ* is the team's current expectations of what fulfilling the role ρ implies.
- *agent-expect-ρ* is the agent's (probabilistic) estimation for how ρ will be fulfilled.
- *e_i-response* is any response the user has made to the agent's requests for input.
- "other α attributes" encapsulates other aspects of the joint activity that are impacted by the decision.

The set of actions for this MDP representation is $\Gamma = E \cup \{\mathcal{D}, \text{wait}\}$. The E action, i.e., transfer control, and the $\mathcal{D}$ action, i.e., perform a coordination change, have been discussed above. The "wait" action puts off transferring control and making any autonomous decision, without changing coordination with the team. The transition probabilities of the MDP represent the effects of the actions as a distribution over their effects, e.g., one set of transition probabilities captures the probability of getting a response from an entity. When the agent chooses an action that transfers decision-making control to an entity other than the agent itself, there are two possible outcomes: either the entity makes a decision (producing a terminal state), or the decision remains unmade (the result being as if the agent had simply waited). We compute the relative likelihood of these two possible transitions by using the response times modeled in $\mathbf{P}$. The $\mathcal{D}$ action has a deterministic effect, in that it changes the coordination of α (affecting the expectations on the user's role through the state feature, *team-expect-ρ*).

The final part of the MDP representation is the reward function. In general, our AA MDP framework uses a general reward function:

$$
\begin{aligned}
R(s,a) = \quad & f(\textit{team-orig-expect-}\rho(s), \textit{team-expect-}\rho(s), && (5)\\
& \quad \textit{agent-expect-}\rho(s), \alpha\textit{-status}(s), a) \\
& = \textstyle\sum_{e\in E} EQ^d_e \cdot \textit{e-response} - && (6)\\
& \lambda_1 f_1(\|\ \textit{team-orig-expect-}\rho(s) - \textit{team-expect-}\rho(s)\ \|) \\
& -\lambda_2 f_2(\|\ \textit{team-expect-}\rho(s) - \textit{agent-expect-}\rho(s)\ \|) \\
& +\lambda_3 f_3(\alpha\textit{-status}(s)) + \lambda_4 f_4(a)
\end{aligned}
$$

The first component of the reward function captures the value of getting a response from a decision-making entity (notice, that only one entity will actually respond, therefore only one e-response will be non-zero). This corresponds to the $EQ^e_d(t)$ function used in the mathematical model. The f_1 function reflects the inherent value of performing a role as the team originally expected, hence deterring the agent from coordination changes (corresponds to $\mathcal{D}_{cost}$ from the mathematical model). The f_2 function reflects the value of keeping the agent's expectation of their performance of the role in agreement with the team's understanding of how the role will be performed. The agent receives most reward when the role is performed exactly as the team expects, thus encouraging it to keep other team members informed of the role's status. This component largely corresponds to the wait cost function, $\mathcal{W}(t)$, from the mathematical model. The fourth component of the reward function, f_3, influences overall reward based on the successful completion of the joint activity, which encourages the agent to take actions that maximize the likelihood the joint activity succeeds. The desire to have the joint task succeed is implicit in the mathematical model but must be explicitly represented in the MDP. The fourth component, f_4, factors in the specific, immediate costs of an action and varies with the type of action, discouraging the agent from taking costly actions (like coordination changes) unless it can gain some indirect value from doing so. For example, there is some cost to asking the user for input via a mobile device. Notice, that these detailed, domain specific costs do not directly appear in the mathematical model.

Given the MDP's state space, actions, transition probabilities, and reward function, an agent can use *value iteration* to generate a policy $P{:}S{\rightarrow}\Gamma$ that specifies the optimal action in each state [15]. Effectively, the value iteration process efficiently calculates the EU of the different strategies and compares them. The agent then executes the policy by taking the action that the policy dictates in each and every state in which it finds itself. A policy may include several transfers of control and deadline-delaying actions, as well as a final autonomous action. The particular series of actions depends on the activities of the user.

An example of an AA MDP is the generic *delay MDP*, for doing AA reasoning about whether to delay meetings based on the user not being able to attend on time. The MDP can be instantiated for any meeting for which Friday may act on behalf of its user. In this case the joint activity, α, is for the meeting attendees to attend the meeting simultaneously. Friday's role, ρ, is to ensure that its user arrives at the currently scheduled meeting time. The constraints between Friday's role and other agent's roles is that they occur simultaneously, i.e., the users must attend at the currently scheduled time. Changing $\asymp$ corresponds to delaying the meeting, i.e., a $\mathcal{D}$ is a meeting delay. Friday has a variety of $\mathcal{D}$ actions of various lengths at its disposal, as well the ability to cancel a meeting entirely. The user can also request a $\mathcal{D}$, e.g., via a dialog box, to buy more time to make it to the meeting. If the user decides a $\mathcal{D}$ is required, Friday is the conduit through which other Fridays (and hence their users) are informed.

In the delay MDP's state representation, *team-orig-expect-ρ* is *originally-scheduled-meeting-time*, since attendance at the originally scheduled meeting time is what the team originally expects of the user and is the best possible outcome. *team-expect-ρ* is *time-relative-to-meeting*, which may increase if the meeting is delayed. *α-status* becomes *status-of-meeting*. *agent-expect-ρ* is not represented explicitly; instead, *user-location* is used as an observable heuristic of when the user is likely to attend the meeting. For example, a user who is away from the department shortly before a meeting should begin is unlikely to be attending on time, if at all. The general reward function is mapped to the *delay MDP* reward function in the following way. $f_1 = \{g(N, \alpha)$, where N is the number of times the meeting is rescheduled. The exact size of the reward, i.e., the function g, depends on factors like the number of meeting attendees and role of the user in the meeting. $f_2 = \{h(late, \alpha)$ if $late > 0, 0$ otherwise$\}$, where *late* is the difference between the scheduled meeting time and the time the user arrives at the meeting room. *late* is probabilistically calculated by the MDP based on the user's current location and a model of the user' behavior. $f_3 = \{r_{user}$ if user attends, 0 otherwise$\}$, where r_{user} models the user's value to α. f_4 depends on the medium being used, e.g., there is higher cost to communicating via a WAP phone than via a workstation dialog box and the length of the meeting delay (longer delays are more costly).

Expected decision quality for the user and agent is implicitly calculated by the MDP. When the user is asked for input, it is assumed that if they respond their response will be "correct", i.e., if the user says to delay the meeting by 15 minutes we assume the user will arrive on time for the re-scheduled meeting. The expected quality of the agent's decision is calculated by considering the agent's proposed decision and the possible outcomes of that decision, i.e., the benefits if the decision is correct and the costs if it is wrong. The delay MDP also represents probabilities that a change in user location (e.g., from office to meeting location) will occur in a given time interval. The designer encodes the initial probabilities, which a learning algorithm may then tailor to individual users. Evaluation of the delay MDP is given in the next section.

5. EVALUATION

The strategy approach to AA reasoning in a multiagent context has been carefully evaluated via its use in the E-Elves. The E-Elves was heavily used by five to ten users between June and December 2000 and by a smaller group of users since then. The agents ran continuously, around the clock, seven days a week. The most heavily used AA reasoning was for delaying meetings. We make three key observations about the use of AA reasoning. Over the course of six months (June to December, 2000) nearly 700 meetings where monitored (Figure 4(a)). Most users had about 50% of their meetings delayed. Figure 4(b) shows that usually 50% or more of delayed meetings were autonomously delayed. The graphs show that the agents are acting autonomously in a large number of instances, but equally importantly, users are also often intervening, indicating the critical importance of AA in Friday. Figure 4(c) shows a frequency distribution of the number of actions taken per meeting. The number of actions taken for a meeting corresponds to the length of the strategy followed. The figure shows both that the MDP followed complex strategies in the real world and that it followed *different* strategies at different times. The most emphatic evidence for the utility of the MDP approach was that it never repeated the catastrophic mistakes of the C4.5 implementation. Although mistakes did occur they were generally small errors such as asking the user earlier than required.[3]

To further determine the suitability of MDPs to the AA reasoning task we performed a series of experiments where various parameters of the MDP's reward function were varied and the resulting policies observed. The experiments aimed to investigate some properties of MDPs for AA, in particular whether policies changed in expected ways when parameters were varied and whether small changes in the parameters would lead to large changes in the policy. We

[3]The inherent subjectivity of the application makes an objective evaluation of the system's success difficult.

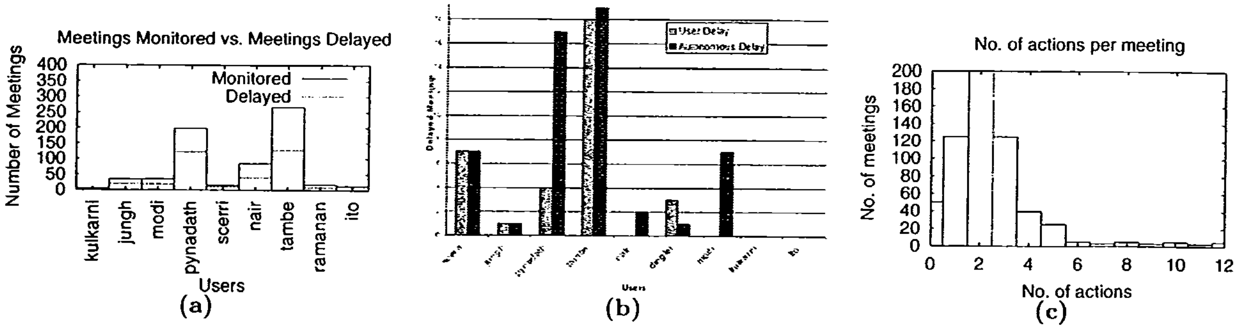

Figure 4: (a) Monitored vs. delayed meetings per user. (b) Meetings delayed autonomously (darker bar) vs. by hand. (c) Frequency distribution of the number of actions taken for a particular meeting.

describe the results of varying one parameter below. In this experiment, the varied parameter is *team wait cost*, which determines the cost of having other team members waiting in the meeting room for the user (corresponds to λ_2 in Equation 7). Each graph in Figure 5 shows the how the frequency of a certain type of action in the resulting MDP policy varied as the *team wait cost* was varied. Notice in Figure 5(a) the phenomena of the number of asks increasing then decreasing as the team wait cost is increased. Friday transfers control whenever the potential costs of asking are lower than the potential costs of errors it makes – as the cost of time waiting for a user decision increases, the balance tips towards acting. When waiting costs are very low Friday acts since the cost of its errors are very low, while when they are very high it acts because it cannot afford to wait for user input. Figure 5(b) shows that as the cost of teammates time increases Friday acts autonomously more often. The number of times Friday will say attending changes rapidly for low values of the parameter, hence considerable care would need to be taken to set this parameter appropriately.

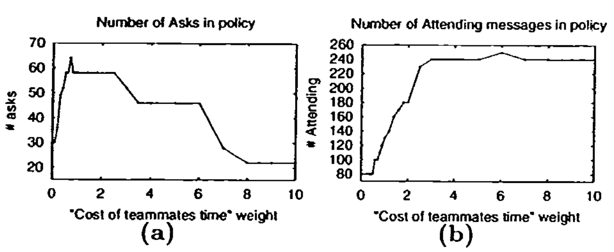

Figure 5: Properties of the MDP policy as team mate time cost is varied.

To determine whether the need for complex strategies was an unusual feature of the E-Elves domain, a simple experiment was run with randomly generated configurations of entities. In each configuration, factors like the rate of wait cost accrual and number of entities was randomly varied. Figure 6(a) shows a frequency distribution of the number of transfer of control actions of the optimal strategies found for 25,000 configurations. Strategies of two actions are optimal in over fifty percent of situations but strategies of up to eight actions were sometimes optimal. Notice that the model on which the experiment is based excludes many complicating factors like the dynamic environment and interacting goals, yet often complex strategies are still required.

Importantly, our theoretical model helps to explain and predict the behavior of other AA systems, not only our

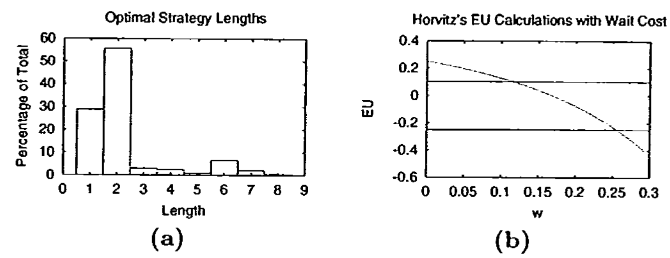

Figure 6: (a) Graph showing the relative percentages of optimal strategy lengths for randomly generated configurations of entities and decisions. (b) EU of different agent options in Horvitz's work. The solid line shows the EU of acting, the dotted line shows the EU of not acting and the dashed line shows the EU of dialog. Each is plotted against increasing wait cost accrual rate.

own. For example, Horvitz has used decision theory to develop general, theoretical models for AA reasoning[10]. A critical difference between his work and this work is that Horvitz pays no attention to the possibility of not receiving a (timely) response and hence, complex strategies are not required. Figure 6 shows that when Horvitz's work is modeled using our transfer-of-control strategy model we advocate the same choice of transfer-of-control action as he does when there are no wait costs (w = 0) but that we might choose differently if there were significant wait costs. The fact that the optimal strategy varies with wait cost suggests that Horvitz's strategy would not immediately transfer to a domain where wait costs were non-negligible.

6. SUMMARY AND RELATED WORK

AA is fundamental to the successful deployment of multiagent systems in human organizations. In this paper, we have presented a theory of ajdustable autonomy, based on transfer-of-control strategies. We then mapped these strategies to a general MDP for AA in a team context. Results, from the Electric Elves domain, showed the technique to be an effective one in a complex multiagent context. Future work will focus on extending the theory and implementation to domains where team plans interact and, hence, the AA decisions of agents interact.

We have already discussed some related work in Section 1, and discussed key weaknesses of prior work that arise from its focus on domains involving single-agent single-user interactions. Indeed, these weaknesses are not only seen in the more recent AA work [1, 9, 11], but in earlier related work in mixed-initiative planning[7], robot teleoperation[19], human-machine function allocation[2, 20].

As we have moved towards more complex environments and introduced the notion of strategies at least three other research areas become relevant: (i) meta-reasoning[18]; (ii) multiprocessor scheduling[21]; (iii) anytime algorithms[24]. Each of these areas makes fundamentally different assumptions than AA. For instance, in meta-reasoning, the output is a sequence of computations to execute in sequence. While AA reasoning also involves reasoning about which computations to execute, i.e., which entities to transfer control to, the AA reasoning focuses on contingencies if entities fail to respond while meta-reasoning assumes the computation will succeed if executed. Furthermore, meta-reasoning looks for a sequence of computations that uses a set amount of time optimally while AA reasoning is dealing with decisions requiring little computation and the available time is something the reasoning decides for itself.

7. REFERENCES

[1] K. Barber, A. Goel, and C. Martin. Dynamic adaptive autonomy in multi-agent systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 12(2):129–148, 2000.

[2] A. Bye, E. Hollnagel, and T. S. Brendeford. Human-machine function allocation: a functional modelling approach. *Reliability engineering and system safety*, 64:291–300, 1999.

[3] H. Chalupsky, Y. Gil, C. Knoblock, K. Lerman, J. Oh, D. Pynadath, T. Russ, and M. Tambe. Electric Elves: Applying agent technology to support human organizations. In *International Conference on Innovative Applications of AI*, pages 51–58, 2001.

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

[5] L. Comfort. *Shared Risk: Complex systems in seismic response.* Oxford, Pergamon Press, 1999.

[6] G. Dorais, R. Bonasso, D. Kortenkamp, B. Pell, and D. Schreckenghost. Adjustable autonomy for human-centered autonomous systems on mars. In *Proceedings of the First International Conference of the Mars Society*, pages 397–420, August 1998.

[7] George Ferguson and James Allen. TRIPS : An intelligent integrated problem-solving assistant. In *Proceedings of Fifteenth National Conference on Artificial Intelligence(AAAI-98)*, pages 567–573, Madison, WI, USA, July 1998.

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

[9] H. Hexmoor. Case studies of autonomy. In *Proceedings of FLAIRS 2000*, pages 246-249, 2000.

[10] Eric Horvitz, Andy Jacobs, and David Hovel. Attention-sensitive alerting. In *Proceedings of Conference on Uncertainty and Artificial Intelligence (UAI'99)*, pages 305–313, Stockholm, Sweden, 1999.

[11] D. Kortenkamp, D. Keirn-Schreckenghost, and R. P. Bonasso. Adjustable control autonomy for manned space flight. In *IEEE Aerospace Conference*, 2000.

[12] David Kortenkamp, Robert Burridge, R. Peter Bonasso, Debra Schreckenghost, and Mary Beth Hudson. Adjustable autonomy issues for control of robots. In *Adjustable Autonomy Workshop, IJCAI'99*, 1999.

[13] V. Lesser, M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, P. Xuan, and S. Zhang. The UMASS intelligent home project. In *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 291–298, Seattle, USA, 1999.

[14] Tom Mitchell, Rich Caruana, Dayne Freitag, John McDermott, and David Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7):81–91, July 1994.

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

[16] David V. Pynadath, Milind Tambe, Hans Chalupsky, Yigal Arens, et al. Electric elves: Immersing an agent organization in a human organization. In *Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents*, 2000.

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

[18] Stuart J. Russell and Eric Wefald. Principles of metareasoning. In Ronald J. Brachman, Hector J. Levesque, and Raymond Reiter, editors, *KR'89: Principles of Knowledge Representation and Reasoning*, pages 400–411. Morgan Kaufmann, San Mateo, California, 1989.

[19] T. Sheridan. *Telerobotics, automation and Human Supervisory Control.* MIT Press, Cambridge, Massachusetts, 1992.

[20] B. Shneiderman. *Designing the User Interface.* Addison Wesley, 1998.

[21] J. Stankovic, K. Ramamritham, and S. Cheng. Evaluation of a flexible task scheduling algorithm for distributed hard real-time system. *IEEE Transactions on Computers*, 34(12):1130–1143, December 1985.

[22] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research (JAIR)*, 7:83–124, 1997.

[23] Milind Tambe, David V. Pynadath, Nicolas Chauvat, Abhimanyu Das, and Gal A. Kaminka. Adaptive agent integration architectures for heterogeneous team members. In *Proceedings of the International Conference on MultiAgent Systems*, pages 301–308, 2000.

[24] S. Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996.