Constrained Intelligent Action: Planning Under the Influence of a Master Agent

Eithan Ephrati
Jeffrey S. Rosenschein
Computer Science Department, Hebrew University
Givat Ram, Jerusalem, Israel
tantush@cs.huji.ac.il, jeff@cs.huji.ac.il

Abstract

In this paper we analyze a particular model of control among intelligent agents, that of *non-absolute control*. Non-absolute control involves a "supervisor" agent that issues orders to a "subordinate" agent. An example might be a human agent on Earth directing the activities of a Mars-based semi-autonomous vehicle.

Both agents operate with essentially the same goals. The subordinate agent, however, is assumed to have access to some information that the supervisor does not have. The agent is thus expected to exercise its judgment in following orders (i.e., following the true intent of the supervisor, to the best of its ability). After presenting our model, we discuss the planning problem: how would a subordinate agent choose among alternative plans? Our solutions focus on evaluating the distance between candidate plans.

Introduction

Imagine that you have sent your robot Explorer to analyze the Martian surface. Though your communication with him is limited, you are able, from time to time, to send him general goals, such as "Get a soil sample from Region A." Explorer is an obedient robot, but after failing to hear from him for some time, you ask why there's been a delay in retrieving the soil. Explorer informs you that because there is a large valley between his current location and Region A (a valley of which you, his master, had been unaware), he has begun construction of a small plane to fly over the valley. In truth, getting the soil sample wasn't that important to you. Had you known the cost of Explorer's plan to accomplish your goal, you would have been willing to have him get a sample from Region B. Explorer *knew* that you didn't know about the valley, so he could have reasoned that your plan for the soil collection was much simpler than his. If only he had had the intelligence to inform you of the discrepancy!

On another occasion, you tell Explorer to move to Region C. While you believe he is able to do this using a particular route, he knows of two alternative shortcuts. Each shortcut will have side effects (one route may pollute the only well of water in the region and the second route might cause damage to his photoelectric cells). By what criteria should Explorer plan his travels, given his knowledge of your goals?

Multi-Agent Planning

Research on planning in Distributed Artificial Intelligence (DAI) has focused on two major paradigms: *planning for multiple agents* and *distributed problem solving*. In the first paradigm, a single intelligent agent (the "master") constructs a plan to be carried out by a group of agents (the "slaves"), then hands out pieces of the plan to the relevant individuals [Rosenschein, 1982; Konolige, 1982; Lesser, 1987; Katz and Rosenschein, 1992]. In the second paradigm, a group of intelligent agents jointly construct the final plan, and subsequently cooperate in carrying it out [Genesereth *et al.*, 1986; Durfee and Lesser, 1987].

In the master-slave model, the slave follows the exact detailed orders of the master. It has no interests of its own, and does not attempt to reason about the master's knowledge or plans. In the second model, all agents may share equally in control and decision making. Each agent might have its own goals and control its own actions. To promote its interests, each can reason about other agents' knowledge and goals. In some scenarios, agents have conflicting goals, negotiate, and compete over resources [Kraus *et al.*, 1991; Zlotkin and Rosenschein, 1991].

We are interested in that important family of scenarios in which the control method combines both aspects. In these scenarios, there are *supervising* and *supervised* agents, where the control of a supervising agent over the supervised one is non-absolute. Such a hierarchy is typically exhibited in organizations with a pyramid structure, but also characterizes all other situations in which the supervised agent is expected to show some intelligence. We will refer to the supervising agent as the "supervisor," and to the supervised agent simply as the "agent."

Determining Plan Deviation

We are concerned with the following possible control relationships between the supervisor and the agent:

1. The supervisor generates and sends the agent a complete plan;
2. The supervisor does not (or cannot) generate a complete plan, but generates and sends the agent a partial (abstract) plan;
3. The supervisor has not generated a complete (or partial) plan, or perhaps has generated it but cannot send it because of constraints on communication. Instead, the supervisor sends a high-level goal.

Given a complete plan, the agent should in principle follow it, but this may not be possible or desirable if the supervisor's model of the world is faulty. Thus, even in this situation, or when the agent has been given a partial plan or a goal, the agent is expected to generate a complete plan based on its own, more accurate knowledge. The agent's complete plan should not differ radically from the supervisor's specified plan, or (if no complete plan was specified), what the agent predicts the supervisor *would* have generated as a plan.

The intuition is that if the world does not differ greatly from what the supervisor believes, and if the agent's model of the supervisor's beliefs is accurate, the plans they generate will also not differ greatly. If there is sufficient deviation between their plans, then someone's model is wrong, and some action should probably be taken (e.g., communicate). This is true when the the agent's plan is much "better" than the supervisor's plan, and when it is much "worse;" in both cases, some realignment may be necessary. Thus, we are concerned with measuring the distance between plans.

Note that we consider plan deviation among correct plans; even though they all achieve the goal, some are more suitable than others. We are concerned with more than just having a plan to achieve a goal—we are concerned with the structure of the resulting correct plan.

The main research issues of concern in these situations are reasoning about knowledge, nonmonotonic reasoning, shared knowledge and knowledge revision, communication, and planning. In this paper, we limit ourselves to issues of planning.

A Blocks World Example

Consider a scenario in the slotted blocks world as described in Figure 1. This example will be used several times in the paper. The domain is described by the following predicates: **Blocks**(n) — The total number of blocks in the world; **Stacked**($block1$, $block2$, $slot$) — $block1$ is on top of $block2$, and located at $slot$ (if $block1$ is on the table, $block2$ and $slot$ are identical); **At**(s) — The agent is at slot s. The domain operators are: **Go**(s_i, s_j) — The agent goes from s_i to s_j; **Carry**(b, s_i, s_j) — Agent carries block b from s_i to s_j; **Paint**($b, color$) — Agent paints block b a particular color; **Destroy**(b) — Agent destroys block b; **Create**(b) — Agent creates a new block b.

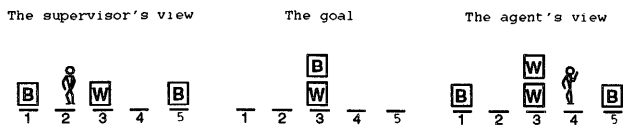

Figure 1: A scenario in the slotted blocks world

In the initial state the agent believes the supervisor's knowledge of the world to be described by:
$\{Blocks(3), Stacked(B, s_1, s_1),$[1]
$At(s_2), Stacked(W, s_3, s_3), Stacked(B, s_5, s_5)\}$.
The agent's own knowledge of the world is:
$\{Blocks(4), Stacked(B, s_1, s_1), At(s_4),$
$Stacked(W, s_3, s_3), Stacked(W, W, s_3),$
$Stacked(B, s_5, s_5)\}$.
Given only the supervisor's goal, $\{Stacked(W, s_3, s_3), Stacked(B, W, s_3)\}$, the agent must decide which of the many alternative plans for achieving the goal he should carry out.

Assumptions and Definitions

- The agent keeps a model of the supervisor's knowledge base (KB) concerning the domain that the agent operates in, and updates it during the interaction. This model may in fact differ from the actual view of the domain held by the supervisor.
- The *goal* G is a set of predicates that might be given by the supervisor to the agent. S_G are the states of the world in which G holds; we use s_G to stand for any state in S_G.
- An agent a's plan $P(a, s_0^a, s_G^a)$ (denoted by P^a) is a set of consecutive operations: $[op_1^a, op_2^a, \ldots, op_n^a]$, that brings about s_G^a starting from the initial state s_0^a. The set of all such alternative plans is $\mathcal{P}^a$.
- Each operation of a plan P transforms the world from one state into another $[s_0, s_1, \ldots, s_{n+1}]$ (such that $s_{n+1} = s_G$).
- Given a *cost function* ($C: OP \rightarrow \mathbb{R}$) over the domain's operators [Finger, 1986], we define the cost of a plan $C(P)$ to be $\sum_{k=1}^{n} C(op_k)$.

Based on a comparison with the supervisor's complete plan (either transmitted, or generated by the agent from a goal or partial plan using his model of the supervisor's knowledge base), the agent has to choose one plan from $\mathcal{P}^a$. In fact, the generation of the plans in $\mathcal{P}^a$ can themselves be guided by the supervisor's plan P^s, and the entire set $\mathcal{P}^a$ may not need to be explicitly generated.

Assume the following cost function over the operators of our example: $C(Go(s_i, s_j)) = \mid i - j \mid$, $C(Carry(b, s_i, s_j)) = 2* \mid i - j \mid$, $C(Paint(b, color)) =$

[1] B refers to any Black block, W to any White one.

6 if color is Black and 8 if it is White, $C(Destroy(b))$ = 3 and $C(Create(b))$ = 18.

In the example of Figure 1, the assumed supervisor's plan is $\langle Go(s_2, s_1), Carry(B, s_1, s_3)\rangle$ $\{cost5\}$. The agent is faced with choosing one of the following alternative plans (for simplicity, we ignore some other possibilities):

$P_1^a = \langle Go(s_4, s_3),\ Paint(W, Black)\rangle\{cost7\}$
$P_2^a = \langle Go(s_4, s_3),\ Destroy(W),\ Go(s_3, s_5),$
$Carry(B, s_5, s_3)\rangle\{cost10\}$
$P_3^a = \langle Go(s_4, s_3),\ Destroy(W),\ Go(s_3, s_1),$
$Carry(B, s_1, s_3)\rangle\{cost10\}$
$P_4^a = \langle Go(s_4, s_3),\ Carry(W, s_3, s_4),\ Go(s_4, s_5),$
$Carry(B, s_5, s_3)\rangle\{cost8\}$
$P_5^a = \langle Go(s_4, s_3),\ Carry(W, s_3, s_2),\ Go(s_2, s_1),$
$Carry(B, s_1, s_3)\rangle\{cost8\}$.

What follows are some plausible criteria for measuring distances between plans. Although we refer here to the entire plan, it may be possible to take into consideration *abstract* plans (i.e., to measure plan distances at higher levels of abstraction, incompletely specified [Chapman, 1987]), so as to reduce the computational complexity.

Comparing Outcomes and Costs

One way of evaluating the distance between plans is to take into consideration only the final outcome of a plan or its overall cost. The following are three alternative ways of doing so:

(1) Comparing the cost of plans generated by the agent to the cost of the supervisor's plan: One method of comparison would be to consider only the cost of each plan, regardless of its actual steps or side effects. Such a consideration suggests the following cost metric:

$$D_c(P_1, P_2) = \mid C(P_1) - C(P_2) \mid .$$

Under certain circumstances, it might be satisfactory to choose the plan that minimizes the cost difference from the supervisor's (assumed) plan, or any plan such that $D_c(P, P^s)$ does not exceed a freedom threshold parameter T_c. Using the cost function given above, we get the following distance values: $D_c(P_1^a, P^s) = 2$, $D_c(P_2^a, P^s) = 5$, $D_c(P_3^a, P^s) = 5$, $D_c(P_4^a, P^s) = 3$, $D_c(P_5^a, P^s) = 3$. With the criterion of cost difference minimization, P_1^a would be chosen.

(2) Comparing the deviation of the KBs: The freedom the agent has in generating plans may lead to some unpredicted side effects, and there is the potential that these side effects are undesirable from the supervisor's point of view. One symptom of such side effects is deviation between the supervisor and agent KBs. An intuitive solution is, therefore, to have the agent choose actions so as to minimize KB deviation. One method of evaluating the distance between KBs is to consider the cost of the plan that would transform one into the other. Using this criterion, the agent might be expected to minimize (or bound to T_o) the following "outcome metric":

$$D_o(P^a, P^s) = \max(C(P(a, s_G^a, s_G^s)), C(P(s, s_G^s, s_G^a)))$$

Applying the outcome metric to the example we get the following values: $D_o(P_1^a, P^s) = 22$, $D_o(P_2^a, P^s) = 0$, $D_o(P_3^a, P^s) = 12$, $D_o(P_4^a, P^s) = 32$, $D_o(P_5^a, P^s) = 20$. P_2^a minimizes KB deviation. For example, the goal state that is created by P_1^a differs from the one that is generated by P^s by $\{Blocks(4), Stacked(B, s_1, s_1)\}$. The plan that transforms the world from the agent's actual goal state to the supervisor's presumed goal state is $\langle Go(s_3, s_1),\ Destroy(B),\ Go(s_1, s_3)\rangle\{cost7\}$. The plan that transforms the world from the supervisor's presumed goal state to the agent's actual goal state is $\langle Go(s_3, s_1),\ Create(B),\ Go(s_1, s_3)\rangle\{cost22\}$. Thus, the cost of the more expensive plan is 22.

A more sophisticated outcome metric would distinguish between differences caused by the plan, and differences that preceded plan execution. In the example above, there was already a difference in the number of blocks in the world prior to the plan's execution, and the outcome metric should perhaps not consider the variation in number of blocks at plan's end.

(3) Avoiding irreversible consequences: The solution presented above might not always be compatible with the kind of intelligence we would like the agent to exhibit. Consider a user (the supervisor) ordering "**rm ~/pub/***" to an intelligent UNIX operating system agent (the agent) [Finger, 1986]. If the agent is aware that the user does not know about the existence of some files in his **/pub** directory, we would like it to warn the user, although the deletion of all files in this case would actually bring the two KBs closer.

Situations can thus occur where the KBs get closer, but there is a loss of resources known only to the agent. Therefore, we want to prevent the agent from changing important parts of the world that are not known to the supervisor ($s_0^a - s_0^s$). This could be done by bounding the cost of the plan needed to recreate the resources:

$$C_r(P^a, P^s) = C(P(a, s_G^a, s_0^a - s_0^s)).$$

Notice that this constraint can be integrated into the cost metric if the cost of an operation reflects the cost of reversing it. This is not, however, always desirable since (for instance) we would not want the **rm** operation to be very expensive under all circumstances.

In our example, the difference between the two knowledge bases is:
$\{Blocks(4),\ At(4),\ Stacked(W, W, s_3)\}$. Therefore $C_r(P_1^a, P^s) = 9$, $C_r(P_2^a, P^s) = 22$, $C_r(P_3^a, P^s) = 22$, $C_r(P_4^a, P^s) = 7$, $C_r(P_5^a, P^s) = 7$. For instance, to restore his exclusive knowledge from the final state of P_1^a, the agent would perform $\langle Paint(B, White),$ $Go(s_3, s_4)\rangle$. Starting at the final state of P_2^a, the agent would perform $\langle Carry(B, s_3, s_2),\ Go(s_2, s_3),$ $Create(W), Go(s_3, s_4)\rangle$. P_4^a and P_5^a minimize the cost needed to recreate resources.

Distance Between Sets of States

Above, we treated each plan as a single, indivisible unit. Alternatively, we could take into consideration in our evaluation *each step* of a plan, and the corresponding state to which it leads. These states can be used to measure distance between the plans. We associate with two plans (P_1, P_2) their corresponding sets of states (S^{P_1}, S^{P_2}). Figure 2 shows the corresponding sets of states of P_1^a, P_3^a, P_5^a and P^s. Assuming the existence of a distance metric between two states $d(s_1, s_2)$ (see the following section), the agent could make geometric measurements of distance between *sets* of points adequate for measuring the difference between two plans. Here are three plausible metrics between sets of points.

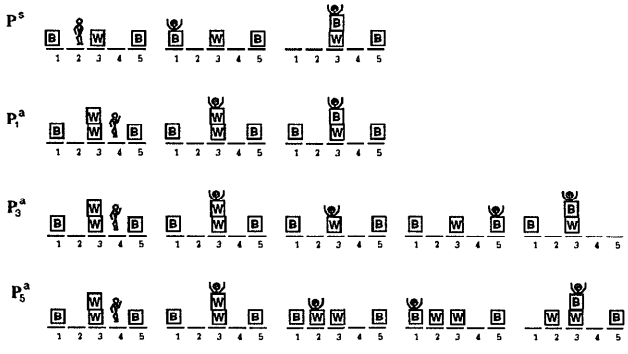

Figure 2: Sets of states corresponding to plans

(1) The Hausdorff metric: One way of measuring inter-plan distance is to choose one "representative" state from each plan, and consider their distance. Above, for example, we considered the plans' *final* states to be representative. Other alternatives might be to take the plans' maximally distant states (or minimally distant) as representative, or the two "centers of gravity" of both plans.

The Hausdorff metric is also based on the distance between a representative state from each plan. This metric is used in the field of image processing to measure the distance between two shapes [Atallah, 1983; Arkin *et al.*, 1990]. Taking the state in the first set that is closest to any state in the second set, one measures the distance between this state in the first set and the *farthest* state in the second set. Doing this in both directions (both plans as the first set), one takes the maximal distance: $H_{max}(S^s, S^a) = \max(h(S^s, S^a), h(S^a, S^s))$, where $h(A, B) = \max_{a \in A} \min_{b \in B} d(a, b)$. It may be preferable at times to use the sum, H_{sum}, of the h metrics.

Assume that the inter-state metric returns the following distance values between each of the states in the supervisor's plan, and each of the states in the agents' plans (as described in Figure 2). For simplicity, we look at only three agent plans:

$(s_0^s$ —) ($\langle 5,4,4\rangle$, $\langle 5,4,1,3,5\rangle$, $\langle 5,4,3,4,8\rangle$)),[2]
$(s_1^s$ —) ($\langle 6,5,5\rangle$, $\langle 6,5,2,4,6\rangle$, $\langle 6,5,4,3,9\rangle$)),
$(s_2^s$ —) ($\langle 8,7,3\rangle$, $\langle 8,7,4,6,8\rangle$, $\langle 8,7,8,9,3\rangle$)).

These values yield $H_{sum}(S^{P^s}, S^{P_1^a}) = 10$, $H_{sum}(S^{P^s}, S^{P_3^a}) = 9$, $H_{sum}(S^{P^s}, S^{P_5^a}) = 8$. P_5^a minimizes the Hausdorff metric.

(2) The "Shifting" distance: This measurement considers the overall differences between two plans. The measurement is done by summing the distance of each state in one plan from the set of states in the other. Defining the distance between one state and a set of states to be $D(s', S) = \min_{s \in S} d(s', s)$, we get the distance between one set of states and another to be the total shifts needed for merging this set with the other:

$$Sh(S^a, S^s) = \sum_{s \in S^a} D(s, S^s).$$

The distance between the sets can be defined by the *min*, the *max*, or, more informatively, the summation of $(Sh(S^a, S^s), Sh(S^s, S^a))$. As with the use of an "outcome metric," this measurement suffers from the fact that as the states converge the distance gets smaller, even though important data may be lost.

Following the values given in the example above, we get: $Sh_{sum}(S^{P^s}, S^{P_1^a}) = 24$, $Sh_{sum}(S^{P^s}, S^{P_3^a}) = 25$, $Sh_{sum}(S^{P^s}, S^{P_5^a}) = 27$. P_1^a minimizes the Shifting Distance metric.

(3) The "Dynamic deviation" distance: This metric considers the sequential order in which the states of each plan were generated. Let $S^s = \{s_0^s, s_1^s, \ldots, s_m^s\}$ and $S^a = \{s_0^a, s_1^a, \ldots, s_n^a\}$. Assuming (without loss of generality) that $m = \min(m, n)$, we define the distance between the plans to be the summation of the sequential relative deviations produced by operators of each plan:

$$D_s(S^a, S^s) = \sum_{i=1}^{m} d(s_i^a, s_i^s) + \sum_{i=m}^{n} d(s_i^a, s_m^s).$$

Referring to the example, this metric yields: $D_s(S^{P^s}, S^{P_1^a}) = 13$, $D_s(S^{P^s}, S^{P_3^a}) = 28$, $D_s(S^{P^s}, S^{P_5^a}) = 30$.

The deviation metric captures, in a sense, the "direction" of the deviation (i.e., whether it is growing or shrinking). The deviation metric's advantage with respect to the other two metrics is its relatively easy computation and the fact that it can dynamically guide search. Using the Hausdorff metric, on the other hand, is preferable when we are concerned only in preventing the agent from radical exceptions to the supervisor's intended plan, while not being bothered by many small variations. A computational advantage of the Hausdorff metric is that it can better constrain the generation of a plan dynamically, since it considers only one member of S^a while the Shifting measurement

[2]The notation compares the distance between the first state of the supervisor's plan, s_0, and the 3 states in P_1^a, the 5 states in P_3^a, and the 5 states in P_5^a.

takes all of them into account. The Shifting metric, on the other hand, is suitable when the overall distance from the intended plan (whether caused by one drastic change or many small ones) is of importance. All these metrics (or their variations) may be taken into account in accordance with different freedom parameters. For example, using the Hausdorff metric, if our agent plans "forward" and follows the "best first" strategy, after generating the first two states of the three plans it would choose P_5^a and will not find it necessary to further explore the two other alternatives.

Metric Between States

Our techniques above depended on our being able to provide a distance metric between states. There are several sensible ways to define such a distance metric, and the following are some basic ideas. Any of the following state metrics can be used as part of any of the plan metrics mentioned above.

(1) Distance between predicates: Since we consider each state to be described by a set of predicates, a straightforward metric can be generated by the summation of the differences between conflicting predicates of the two states in question.[3] As a trivial example, consider two states that differ only by the location of one object (obj). The definition $Diff(At(obj, x_1, y_1), At(obj, x_2, y_2)) = \sqrt{|\, x_1 - x_2 \,| + |\, y_1 - y_2 \,|}$ is sufficient to serve as a distance metric between these states. This method can be refined by giving different weights to such metrics in accordance with some special considerations. For example, if there is a dangerous region in the agent's operating domain the above metric should be given a higher weight if $(x_2, y_2) \in DangerSet$. In that case the location might be used to change the weight of other differences such as $Diff(Armed(obj), UnArmed(obj))$.

The values that were used as the metric between states in the previous examples followed the following rough definitions (based on the cheapest operation that can eliminate the difference): $Diff(Blocks(n_1), Blocks(n_2)) = 3* |\, n_1 - n_2 \,|$, $Diff(At(s_i), At(s_j)) = |\, i - j \,|$, $Diff(Stacked(b, loc, s_i), Stacked(b, loc, s_j)) = 2* |\, i - j \,|$.

(2) Considering ultimate goals: A more meaningful metric might consider either the agent's or the supervisor's global goals (or both). For example, if one of the supervisor's global goals is $Above(obj_1, obj_2)$ and both states satisfy $Stacked(obj_3, obj_2)$, we would like $Diff(Stacked(obj_1, obj_3), Stacked(obj_1, obj_4))$ to return a higher value than, for instance $Diff(Stacked(obj_1, obj_5), Stacked(obj_1, obj_4))$. Global goals can be given at a higher level of abstraction, such as "$\min_{obj \in U.S.Navy} In(obj, Iraq)$." The fact that such ultimate goals can be from different sources (i.e., from the supervisor and agent) allows the consideration of subjective points of view (with respect to these sources) and thus enriches the overall control mechanism. If the supervisor of the example had as a global goal the maximization of the number of blocks in the state, we would not like P_2^a or P_3^a to be considered. This may be done by giving a very high value to $Diff(Blocks(n_1), Blocks(n_2))$.

(3) Using the domain operators: Another possibility is to measure the deviation of two states with regard to the operators that generated them. When comparing two states, the metric will be based on how the current state was reached, with particular operators considered to cause greater variations than others. For example, we might define $Diff(Go(s_i, s_j), Go(s_k, s_l)) = |\, s_j - s_l \,|$ to reflect the deviation or convergence caused by following these two operations. This measure is most appropriate for use with the sequential deviation metric (D_s). In the example, defining $Diff(Carry(b, s_i, s_j), Paint(b, c)) > 20$ would make P_1^a inferior to all other plans according to all the suggested metrics.

(4) Considering the plan that unifies the two states: According to this approach, the metric between any two states is defined by the plan needed to bring about the differing predicates of one state starting from the other:
$Diff(s_1, s_2) = \max(P^*(s_1, s_2), P^*(s_2, s_1))$, where $P^*(s_1, s_2) = \min_P P(a, s_1, s_2)$ (the minimization criterion may be the cost of the plan or just the number of operations). A further consideration might also be to take into account the perspective of the supervisor: $Diff(s_1, s_2) = \max(P_a^*(s_1, s_2), P_a^*(s_2, s_1), P_s^*(s_1, s_2), P_s^*(s_2, s_1))$. Notice that the use of such a metric makes the outcome metric (i.e., bounding KB deviation) redundant.

All of the distance computations above may be costly to perform directly, but the calculation of distance between sets need not use a single distance function uniformly. For example, one might use a (relatively cheap) approximate distance function to map states into an n-dimensional geometric space. Only then would one use the states' approximate "locations" to decide on which ones to apply the more expensive and accurate distance function.

Related Work

The subject of non-absolute control (i.e., how an agent can coherently integrate local goals with outside goals) has received some attention in DAI. Durfee [Durfee, 1988] has looked at the problem in the domain of distributed vehicle sensing. His method of "partial global plans" (PGP) allows agents to coordinate activity around common tasks. The agents may exist at the same level of an authority hierarchy, or one may be above another in that hierarchy. Agents do not, however, reconcile their activity by measuring distance be-

[3] This is related to the measures (such as cardinality of sets) proposed by Ginsberg for evaluating distance between possible worlds in his research on counterfactual planning [Ginsberg, 1986].

tween candidate partial global plans. In addition, all local plans that fully satisfy the constraints imposed by a PGP are satisfactory; in our approach, we derive useful information from plan deviation, even though all plans are technically correct (i.e., achieve the goals).

Malone has also been interested in issues relating to the organization of agents (both human and artificial), and has published various analyses of different human organizations and their relationship to machine organization [Malone, 1986]. In the field of telerobotics, there has also been discussion of how to effectively manipulate robots at a distance, though robots there are generally less intelligent than those in which we are interested (see, for example, [Conway *et al.*, 1990]).

Conclusions

We have considered how a subordinate agent might choose among alternate plans for achieving his supervisor's goals. The techniques exploit the knowledge that the subordinate has about the supervisor, and allow the agent to choose its actions appropriately. The discussion focused on metrics for evaluating the distance between plans, considering several techniques (such as the potential cost of moving from one final state to the other, and a cost on difficult to reverse consequences). One key point is that a plan can be considered a set of states, and the distance between plans can be modeled using a distance metric between sets of states (e.g., the Hausdorff metric). To use any distance metric between sets of states, it is also necessary to establish a suitable distance metric between individual states.

Techniques for making plan comparisons can serve in other scenarios where plans are to be compared (for example, maintaining consistency among several subordinate agents, choosing among agents to carry out a plan, predicting the likelihood that a plan will be carried out). Issues such as these will be of importance in building flexible multi-agent systems.

Acknowledgments

We want to thank Barbara Grosz, Michael Werman, and an anonymous referee for their insightful comments on previous versions of this paper. This research was partially supported by the Israel National Council for Research and Development (Grant 032-8284).

References

Arkin, E.; Chew, L. P.; Huttenlocher, D. P.; Kedem, K.; and Mitchell, J. S. B. 1990. An efficiently computable metric for comparing polygonal shapes. In *Proceedings of the First ACM-SIAM Symposium on Discrete Algorithms.*

Atallah, M. J. 1983. A linear time algorithm for the Hausdorff distance between convex polygons. *Information Processing Letters* 17:207–209.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence* 32(3):333–377.

Conway, Lynn; Volz, Richard A.; and Walker, Michael W. 1990. Teleautonomous systems: Projecting and coordinating intelligent action at a distance. *IEEE Transactions on Robotics and Automation* 6(2):146–158.

Durfee, E. H. and Lesser, V. R. 1987. Using partial global plans to coordinate distributed problem solvers. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan. 875–883.

Durfee, Edmund H. 1988. *Coordination of Distributed Problem Solvers.* Kluwer Academic Publishers, Boston.

Finger, J. J. 1986. *Exploiting Constraints in Design Synthesis.* Ph.D. Dissertation, Stanford University, Stanford, CA.

Genesereth, Michael R.; Ginsberg, Matthew L.; and Rosenschein, Jeffrey S. 1986. Cooperation without communication. In *Proceedings of The National Conference on Artificial Intelligence*, Philadelphia, Pennsylvania. 51–57.

Ginsberg, Matthew L. 1986. Possible worlds planning. In Georgeff, M. and Lansky, A., editors 1986, *Reasoning about Actions and Plans.* Morgan Kaufmann Publishers, Los Altos, California. 213–243.

Katz, Matthew J. and Rosenschein, Jeffrey S. 1992. Verifying plans for multiple agents. *Journal of Experimental and Theoretical Artificial Intelligence.* To appear.

Konolige, Kurt 1982. A first-order formalization of knowledge and action for a multi-agent planning system. *Machine Intelligence* 10.

Kraus, Sarit; Ephrati, Eithan; and Lehmann, Daniel 1991. Negotiation in a non-cooperative environment. *Journal of Experimental and Theoretical Artificial Intelligence* 3(4):255–282.

Lesser, Victor R. 1987. Distributed problem solving. In Shapiro, Stuart C., editor 1987, *Encyclopedia of Artificial Intelligence.* John Wiley and Sons, New York. 245–251.

Malone, Thomas W. 1986. Organizing information processing systems: Parallels between human organizations and computer systems. In Zacharai, W.; Robertson, S.; and Black, J., editors 1986, *Cognition, Computation, and Cooperation.* Ablex Publishing Corp., Norwood, NJ.

Rosenschein, Jeffrey S. 1982. Synchronization of multi-agent plans. In *Proceedings of The National Conference on Artificial Intelligence*, Pittsburgh, Pennsylvania. 115–119.

Zlotkin, Gilad and Rosenschein, Jeffrey S. 1991. Incomplete information and deception in multi-agent negotiation. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Sydney, Australia. 225–231.