Problems with Intent Recognition for Elder Care

Christopher W. Geib
Honeywell Labs
3660 Technology Drive
Minneapolis, MN 55418 USA
{geib,goldman}@htc.honeywell.com

Abstract

Providing contextually appropriate help for elders requires the ability to identify the activities they are engaged in. However, inferring agent intent in these kinds of very general domains requires answers to a number of problems that existing AI research on intent recognition/task tracking has not addressed. This paper identifies some of these limitations, sketches some of the solutions and points at some promising directions for future work.[1]

Introduction

To provide contextually relevant aid for elders, assistant systems must be able to observe the actions of the elder, infer their goals, and make predictions about their future actions. In the artificial intelligence (AI) literature this process of deducing an agent's goals from observed actions is called plan recognition or task tracking. We argue that plan recognition must be a central component in assistant systems for elders.

However, most existing AI literature on intent recognition makes a number of assumptions preventing its application to the elder care domain. In this paper we will discuss the requirements of this domain. In our previous work (Geib & Goldman 2001b; 2001a; Goldman, Geib, & Miller 1999) we have described an approach to plan recognition that does not make many of the restrictive assumptions of other intent recognition systems. In this paper we briefly discuss its application to the recognition of elders plans in relatively unconstrained environments.

Requirements on Plan Recognition

The domain of plan recognition for elder care requires confronting a number of issues that have not been examined in more theoretical or academic domains. The following is a list of issues, properties or assumptions made by some or all previous AI plan recognition research that are critical in the elder care domain.

Abandoning plans: All people abandon plans at one time or another. They forget what they are doing, the get distracted by other events, or due to changing circumstances they explicitly decide to abandon a goal. In the case of providing care to elders failing memory and mental faculties may cause the elder to unintentionally abandon goals that we would like the system to remind them of. Consider an elder that begins to take their medicine but then gets distracted by a phone call and does not return to the medication. To remind the elder that they need to finish taking their medication, a system will first have to recognize that they have abandoned that goal. We will discuss this in more detail in the second half of the paper.

Hostile agents: Most previous work in plan recognition has assumed cooperative agents, that is agents that do not mind or are even helpful in having their plans recognized. Unfortunately it is unrealistic to assume that all elders will be willing to have their actions observed by an assistant system. Like all adults, elders highly value their independence and may be willing to take extensive measures to prevent an assistant system from observing them.

To correctly handle agents that are hostile to having their goals recognized it is critical that we remove the assumption of a completely observable stream of actions. The assistant system must be able to infer the execution of *unobserved actions* on the basis of other observed actions and observations of unexplained state changes.

Observations of failed actions: Observations of failed actions can tell us at least as much about the intent of the elder as successful actions, however most current work in plan recognition has not considered observations of failed actions in inferring intent.

Partially ordered plans: The plans for the daily activities of an elder are often very flexible in the ordering of their plans steps. Consider making a peanut-butter and jelly sandwich, do we put the jelly on bread first or do we put peanut-butter on bread first? In fact,

[1]This work was performed under the support of the U.S. Department of Commerce, National Institute of Standards and Technology, Advanced Technology Program Cooperative Agreement Number 70NANB0H3020

it doesn't matter but it does result in a large number of acceptable orderings for the plan steps.

Multiple concurrent goals: People often have multiple goals. For example, taking medication while eating lunch, or watching a TV show while doing laundry. Much previous work in plan recognition has looked for the single goal that best explains all the observations. In these domains this is simply unacceptable. It is critical that the system consider when the elder is engaged in multiple tasks at the same time.

Actions used for multiple effects: Often in real life a single action can be used for multiple effects. Consider opening the refrigerator. This can be part of a plan to get a drink as well as making a meal. While this action can be done once for each of the plans it can be *overloaded* for use by both plans if the elder were performing both. Again a critical requirement for our plan recognition system is that it be able to handle these kinds of actions.

Failure to observe: Suppose we observe the front door being opened, and suppose this could be part of a plan to leave the house or to get the mail. If immediately after the open door even we don't see motion in the house we should be more likely to believe that the elder has left the house. It is this kind of reasoning based on the absence of observed actions that is critical to our domain.

Our commitment to considering agents with multiple concurrent goals makes it even more critical that an assistant system be able to engage in this kind of reasoning. It is rare that we will be provided with definitive evidence that an elder is *not* pursing a specific goal. Far more likely is that a lack of evidence for the goal will lower its probability. As a result, reasoning on the basis of the "failure to observe" is critical for these kinds of systems to prefer those explanations where the agent is pursuing a single goal over those where the elder has multiple goals but has not performed any of the actions for one of them.

Impact of world state on adopted plans: World state can have significant impact on the goals that are adopted by an elder. An elder may be more likely to bath in the evening than in the morning, thus we should be more likely to suspect that activity in the bathroom at night is part of a bathing goal than if we saw the same activity in the morning.

Ranking multiple possible hypotheses: Providing a single explanation for the observed actions in general is not going to be as helpful as ranking the possibilities. Rather than giving just one of possibly many explanations for a set of observations it is much more helpful to report the relative likelihood of each of the possibilities.

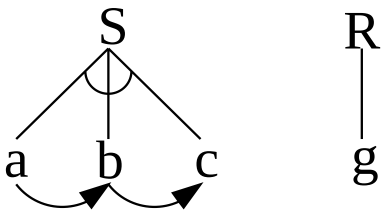

Figure 1: A very simple example plan library.

Background

To met the previous set of requirements, Geib and Goldman have developed the plan/intent recognition framework PHATT (Probabilistic Hostile Agent Task Tracker) (Geib & Goldman 2001b; 2001a) based on a model of the execution of simple hierarchical task network (HTN) plans (Erol, Hendler, & Nau 1994). Figure 1 shows a very simple examples of the kind of HTN plans that we assume are initially given to the system. The plans in the library may be represented as and/or trees. In both figures, "and nodes" are represented by an undirected arc across the lines connecting the parent node to its children; "or nodes" do not have this arc. Ordering constraints in the plans are represented by directed arcs between the actions that are ordered. For example in Figure 1 action a must be executed before b before c.

The central realization of the PHATT approach is that plans are executed dynamically and that at any given moment the agent is able to execute any one of the actions in its plans that have been enabled by its previous actions. To formalize this, initially the executing agent has a set of goals and chooses a set of plans to execute to achieve these goals. The set of plans chosen determines a set of pending primitive actions. The agent executes one of the pending actions, generating a new set of pending actions from which the next action will be chosen, and so forth.

For each possible agent state (explanatory hypothesis), then, there is a unique corresponding pending set. Each pending set is generated from the previous set by removing the action just executed and adding newly enabled actions. Actions become enabled when their required predecessors are completed. This process is illustrated in Figure 2. To provide some intuition for the probabilistically-inclined, the sequence of pending sets can be seen as a Markov chain, and the addition of the action executions with unobserved actions makes it a hidden Markov model.

This view of plan execution provides a simple conceptual model for the generation of execution traces. To use this model to perform probabilistic plan recognition, we use the observations of the agent's actions as an execution trace. By stepping forward through the trace, and hypothesizing goals the agent may have, we

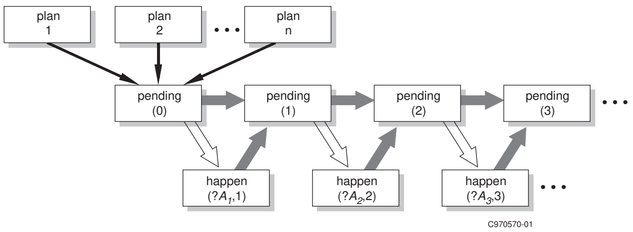

Figure 2: A simple model of plan execution.

can generate the agent's resulting pending sets. Once we have reached the end of the execution trace we will have the complete set of pending sets that are consistent with the observed actions and the sets of hypothesized goals that go with each of these sets. Once we have this set, we establish a probability distribution over it and can determine which of the possible goals the agent is most likely pursuing.

PHATT builds this probability distribution over the set of explanations by considering three kinds of probabilistic features for each of the goals in each explanation. First, the prior probability of each of the root goals being adopted by the actor. In PHATT, this is assumed to be a given with the plan library. Second, for each goal (or node) for which there are multiple methods (children), PHATT must have a probability of method choice given goal. Typically, we have assumed that each method is equally likely given its parent goal, but this simplifying assumption is not required by the framework. Third and most relevant to this work, for each pending set, PHATT computes the probability that the observed action is the one chosen next for execution. We will discuss this third computation in greater detail in the next section.

On the basis of these three probabilities, PHATT computes the prior probability of a given explanation by multiplying together the priors for each goal, the probability of the expansion choices, and the probability of the observed actions being chosen. The set of explanations is, by construction, an exclusive and exhaustive covering of the set of observations, so the priors can be normalized to yield conditional probabilities given the observations. For large problems, some pruning of the hypothesis set and approximation of posteriors will obviously be necessary.

By looking at plan recognition in terms of plan execution, PHATT is able to handle a number of problems that are critical to application to real world domains. In our previous work we have discussed PHATT's ability to handle the issues mentioned in earlier. (Goldman, Geib, & Miller 1999) discusses the problems of, multiple interleaved goals, partially ordered plans, the effects of context on the goals adopted, the effect of negative evidence or failure to observe ("the dog didn't bark"). (Geib & Goldman 2001b; 2001a) discusses PHATT's ability to handle hostile agent's in the form of missing observations, and observations of "failed actions." We refer interested readers to these papers for a more complete discussion of the system, its operation, and these issues. In the following we will sketch a solution to the goal abandonment problem.

Model Revision

Rather than explicitly computing the probability that a given goal is abandoned we have approached this problem as one of model fitting or model revision. If we are using a model of plan execution that does not consider plan abandonment to recognize observation streams in which the agent is abandoning plans, we expect that the computed probabilities for the observation streams will be quite low. Laskey (1991), Jensen (Jensen *et al.* 1990), Habbema (1976) and others (Rubin 1984) have suggested that cases of an unexpectedly small $P(observations|Model)$ should be used as evidence of a model mismatch.

In this case where we are interested in recognizing a specific kind of model mismatch/failure (namely that the agent is no longer executing a plan that it has begun.) As a result, the statistic of $P(observations|model)$ is not sufficient. While this statistic will drop rather rapidly as we fail to see evidence of the agent carrying out the steps of their plan, it does not provide us with sufficient information to determine which of the agent's goals has been abandoned, the critical information that we need in order to repair the model. Instead of the general $P(observations|model)$ statistic we propose the use of a more specific statistic. Our intuitions about how we recognize plans will help us to define this statistic.

Suppose I am observing someone that I initially believe has two high level goals, call them α and β. At the outset they mostly alternate the steps of the plans, but at about half way through the agent stops working on β and instead only executes actions that are part of α. As the string of actions that contribute only to α gets longer, and we see no more actions that contribute to β we should start to suspect that the agent has abandoned β.

We can formalize this idea as the probability that none of the observed actions in a subsequence (from say s to t) contribute to one of the goals (call it G), and we denote it $P(noneContrib(G,s,t)|model, observations)$. If this probability gets unexpectedly small we consider this as evidence of a mismatch between the model and the real world. Namely the model predicts that the agent is still working on the goal, while the agent may have abandoned it. The following section will detail how to compute this probability.

For a Single Explanation

Consider the plan library shown in Figure 1. The first plan is a very simple plan for achieving S by executing a, b, and c and the second plan for R has only the single step of g. Given this plan library, assume the following sequence of observations:

$$happen(a, 0), happen(b, 1), happen(g, 2), happen(g, 3).$$

In this case we know that at time 0 and 1 that the agent has as a goal achieving S. Further we know that the probability of seeing c at time 2 is given by: $(m/|PS_2|)$ where m is the number of elements in the pending set that have c as the next action. The probability that we don't see c (that is the probability that any other element of the pending set is chosen at time 2) is just:

$$1 - (m/|PS_2|)$$

or more generally the probability that we have seen b at time $(s-1)$ and not seen c by time t:

$$\prod_{i=s}^{t} 1 - (m/|PS_i|)$$

To handle partially ordered plans, this formula must be generalized slightly. With partially ordered plans it is possible for more than a single next action to contribute to the specified root goal. Thus, if $m_{q,i}$ represents the number of elements (with any next action) in the pending set at time i that contribute to goal q, *(s-1)* is the last time we saw an action contribute to q and t is the current time, $P(noneContrib(q, s, t)|model, obs) =$

$$\prod_{i=s}^{t} 1 - (m_{q,i}/|PS_i|)$$

Thus, under the assumptions that we have made we can compute the probability of the subsequence of actions *not* contributing to a given plan or goal.

Given the large number of possible sets of abandoned goals, simply computing this probability is not enough. We will need a way to prune the search space of these possible alternatives, that is, trigger model revision and add the assumption that the plan has been abandoned. To do this we require the system designer to specify a threshold value we will call the *Probability of Abandonment Threshold* (PAT). The PAT represents how confident the system must be that no actions have contributed to a given goal, before it can assume that the goal actually has been abandoned.

By computing the probability of the action sequence not contributing to the goal and comparing it to the user set PAT, we can consider any explanation in which this probability drops below the threshold as sufficient evidence of a model mismatch and revise the model to reflect the goal's abandonment. This requires removing all the elements from the current pending set that contribute to the abandoned goal. Modeling of the execution of the rest of the goals can continue as before.

Estimating $P(abandon(g)|Obs)$

If we keep a rolling computation of $P(noneContrib(g, s, t)|model, obs)$ for each g and threshold our models as described in the previous section, the G&G algorithm will now provide us with explanations that include abandoned goals and probabilities for each of them. On the basis of this information we can estimate the probability of a goal's abandonment by:

$$P(abandoned(g)|Obs) \approx \frac{\sum_{e}^{Exp_{A(g)}} P(e|Obs)}{\sum_{e}^{Exp} P(e|Obs)}$$

where Exp represents the set of all explanations for the observations, and $Exp_{A(g)}$ represents the set of explanations in which goal g is marked as abandoned.

Implementation

We have implemented this algorithm for estimating model mismatch and plan abandonment in the PHATT system. We are in the process of evaluating the algorithms accuracy in estimating plan abandonment. However, we have done some simple work on the algorithms runtime. Figure 3 shows the runtimes for seventeen hundred randomly generated two goal observation streams. The plans were taken from an extensive plan library that we have for a different domain. The x-axis graphs the length of the observation stream (the number of actions observed) and the y-axis is a log scale of the runtime in milliseconds. For these runs the PAT was set at the relatively low level of 0.24 to encourage a reasonable number of goals to be abandoned by the system. As a result, 1138 of the runs had at least one abandoned goal, and the majority of the examples still had a runtime under one second.

It is worth noting that, in general, running the same set of examples with a higher PAT can result in higher runtimes. Since hypothesized goals and plans are kept active longer the system takes more time to determine the correct set of explanations and probabilities for the observations. Likewise, lowering the PAT and abandoning more goals often shortens the runtime for the system to generate each explanation. This results from the reduced size of the pending set and the resulting

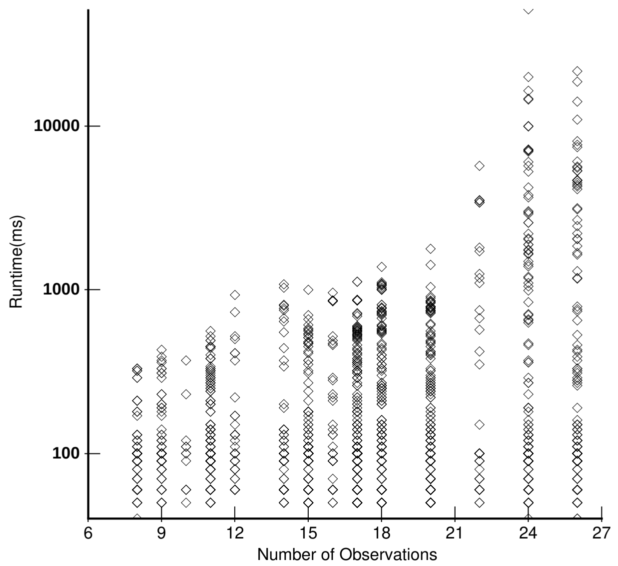

Figure 3: 1700 runs with 2 goals each.

reduction in the number of possible plans for the goals considered.

Preliminary analysis of this data shows that the worst cases for the algorithm result when there is a long common prefix for more than one goal in the plan library. If two plans with this common prefix are interleaved, alternating one action from each plan, the number of possible explanations is quite large. Further, since the actions are alternating, neither of the plans is believed to be abandoned. In these cases, the full algorithm is being run with the added overhead of computing $P(noneContrib(g, s, t)|model, obs)$ and attempting to find abandoned plans.

Conclusions

In this paper we have argued for using a probabilistic model of plan recognition in assistant systems for elder care. We have identified many requirements that are placed on the process of plan recognition by this domain and have shown how our model of plan recognition based on plan execution meets these requirements. These extensions remove a major assumption of previous research in plan recognition and significantly broadens the domains where plan recognition can be applied.

References

Erol, K.; Hendler, J.; and Nau, D. S. 1994. UMCP: A sound and complete procedure for hierarchical task network planning. In Hammond, K. J., ed., *Artificial Intelligence Planning Systems: Proceedings of the Second International Conference*, 249–254. Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Geib, C. W., and Goldman, R. P. 2001a. Plan recognition in intrusion detection systems. In *Proceedings of DISCEX II, 2001.*

Geib, C. W., and Goldman, R. P. 2001b. Probabilistic plan recognition for hostile agents. In *Proceedings of the FLAIRS 2001 Conference.*

Goldman, R. P.; Geib, C. W.; and Miller, C. A. 1999. A new model of plan recognition. In *Proceedings of the 1999 Conference on Uncertainty in Artificial Intelligence.*

Habbema, J. 1976. Models for diagnosis and detections of combinations of diseases. In *Decision Making and Medical Care.* North Holland.

Jensen, F. V.; Chamberlain, B.; Nordahl, T.; and Jensen, F. 1990. Analysis in hugin of data conflict. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence.*

Laskey, K. B. 1991. Conflict and surprise: Heuristics for model revision. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence.*

Rubin, D. 1984. Bayesianly justifiable and relevent frequency calculations for the applied statistician. In *The Annals of Statistics*, volume 12(4), 1151–1172.