Utility-Based Plan Recognition: An Extended Abstract

Dorit Avrahami-Zilberbrand Gal A. Kaminka*

The MAVERICK Group
Computer Science Department
Bar Ilan University, Israel
{avrahad1,galk,}@cs.biu.ac.il

ABSTRACT

Plan recognition is the process of inferring other agents' plans and goals based on their observable actions. Essentially all previous work in plan recognition has focused on the recognition process itself, with no regard to the use of the information in the recognizing agent. As a result, low-likelihood recognition hypotheses that may imply significant meaning to the observer, are ignored in existing work. In this paper, we present novel efficient algorithms that allows the observer to incorporate her own biases and preferences—in the form of a utility function—into the plan recognition process. This allows choosing recognition hypotheses based on their expected utility to the observer. We call this Utility-based Plan Recognition (UPR). We briefly discuss a hybrid symbolic decision-theoretic plan recognizer, and demonstrate the efficacy of this approach in an example.

1. INTRODUCTION

Keyhole plan recognition [2, 3] focuses on mechanisms for recognizing the unobservable state of an agent, given observations of its interaction with its environment. Most approaches to plan recognition utilize a plan library, which encodes the behavioral repertoire of observed agents. Observations are matched against this plan library in sequence.

Essentially all plan recognition techniques ignore the decision processes of the recognizing agent. Existing work focuses on probabilistic or heuristic ranking of recognition hypotheses. As a result, low-likelihood recognition hypotheses that may carry significant gains or costs to the observer, might be ignored.

For instance, suppose we observe a sequence of Unix commands that can be explained by for some intention I or for a more common intention L. Most plan recognition systems will prefer the most likely hypothesis L, and ignore I. Yet, if the expected cost (risk) of I for the observer is high (e.g., if I is a plan to take down the computer system), then that hypothesis should be preferred when trying to recognize suspicious behavior.

We advocate a novel plan recognition approach, *utility-based plan recognition* (UPR), in which the observer folds its biases and preferences—in the form of a utility function—into the plan recognition process itself. Using UPR, the recognition process ranks recognition hypotheses based on their *expected utility to the observer*. This allows the observer, for instance, to select hypotheses based on their expected costs (e.g., in the case of a risk-averse observer), or expected gains. Unfortunately, while in principle UPR can be carried out via influence diagrams or other means, such reasoning about interactions with others is intractable in the general case [5, 7].

We present an efficient UPR recognizer, able to carry out plan recognition in worst-case complexity of $O(NDT)$, where N is the size of a hierarchical plan library, D is the depth of the library, and T is the number of observations. This complexity is achieved by using an hybrid approach that combines an efficient symbolic plan recognizer [1], with a decision-theoretic inference mechanism. We restrict these algorithms to the case of *keyhole recognition*, where the observed agent does not modify its behavior based on the knowledge that it is being observed.

2. RELATED WORK AND MOTIVATION

There has been considerable research exploring plan recognition algorithms. Almost all of it ignores the use of utilities; we leave those aside for lack of space.

[6] address utilities of the other agent's actions to itself, but in contrast to our work they consider the impact of recognition hypotheses on the observed agent, not to the observer.

More closely-related work examined reasoning about the utility of recognition hypotheses *for the observer*. RESC [8] takes a heuristic approach that prefers hypotheses that imply significant costs to the observer (e.g., potential destruction). The relative likelihood of such hypotheses is ignored. While we are inspired by this work, we take a principled, decision-theoretic, approach. In the algorithms we present, the likelihood of hypotheses is combined with their utilities, to calculate the expected impact on the observer.

In general, a UPR recognizer could be implemented by extending the use of plan-recognition Bayesian networks [2] to influence diagrams [5]. However, the run-time complexity of inference in such representations is inhibitory for real-world cases.

3. A HYBRID UPR TECHNIQUE

This section presents an efficient hybrid UPR technique. Here, a highly efficient symbolic plan recognizer [1] is used to filter through hypotheses, maintaining only those that are consistent with the observations. We then use a decision-theoretic layer on top of the symbolic recognizer for ranking the hypotheses.

3.1 Efficient Symbolic Plan Recognition

We exploit SBR, a highly-efficient symbolic plan recognizer, briefly described below. The reader is referred to [1] for details.

*This research was supported by ISF Grant #1211/04.

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

SBR's plan library is a single-root directed graph, where vertices denote *plan steps*, and edges can be of two types: Decomposition edges decompose plan steps into sub-steps, and sequential edges specify the temporal order of execution. The graph is acyclic along decomposition transitions.

Each plan has an associated set of conditions on observable features of the agent and its actions. When these conditions hold, the observations are said to match the plan. At any given time, the observed agent is assumed to be executing a *plan decomposition path*, root-to-leaf through decomposition edges. An observed agent is assumed to change its internal state in two ways. First, it may follow a sequential edge to the next plan step. Second, it may reactively interrupt plan execution at any time, and select a new (first) plan. Figure 1 shows an example portion of a plan library.

The recognizer operates as follow: First, it matches observations to specific plan steps in the library according to the plan step's conditions. Then, after matching plan steps are found, they are tagged by the time-stamp of the observation. These tags are then propagated up the plan library, so that complete plan-paths (root to leaf) are tagged to indicate they constitute hypotheses as to the internal state of the observed agent when the observations were made. The propagation process tags paths along decomposition edges. However, the propagation process is not a simple matter of following from child to parent. A plan may match the current observation, yet be *temporally inconsistent*, when a history of observations is considered. SBR is able to quickly determine the temporal consistency of a hypothesized recognized plan [1].

At the end of the SBR process we are left with a set of *current-state hypotheses*, i.e., a set of paths through the hierarchy, that the observed agent may have executed at the time of the last observation. The overall worst-case run-time complexity of this process is $O(LD)$ [1]. Here, L is the number of plan-steps that directly match the observations; D is depth of a degenerate plan-library (i.e., a linked list).

3.2 The Expected Utility of an Hypothesis

After getting all *current state hypotheses* from the symbolic recognizer, the next step is to compute the expected utility of each hypothesis. This is done by multiplying the posterior probability of a hypothesis, by its utility to the observer.

We follow in the footsteps of *Hierarchical Hidden Markov Model* (HHMM) [4] in representing probabilistic information in the plan library. We denote plan-steps in the plan library by q_i^d, where i is the plan-step index and d is its hierarchy depth, $1 \leq d \leq D$. For each plan step, there are three probabilities maintained:

Sequential transition. For each internal state q_i^d, there is a state transition probability matrix denoted by $A^{q^d} = (a_{i,j}^{q^d})$, where $a_{i,j}^{q^d} = P(q_j^d|q_i^d)$ is theprobability of making a sequential transition from the i^{th} plan-step to the j^{th} plan-step. Note that self-cycle transitions are also included in A^{q^d}.

Interruption. We denote by $a_{i,end}^{q^d}$ a transition to a special plan step end^d which signifies an interruption of the sequence of current plan step q_i^d, and immediate return of control to its parent, q^{d-1}.

Decomposition transition. When the observed agent first selects a decomposable plan step q_i^d, it must select between its (first) children for execution. The decomposition transition probability is denoted $\Pi^{q^d} = \pi^{q^d}(q^{d+1}) = P(q_k^{d+1}|q_i^d)$, the probability that plan-step q_i^d will initially activate the plan-step q_k^{d+1}.

Observation Probabilities. each leaf has output probability vector denoted by $B^{q^d} = (b^{q^d}(k))$, the probability that state q^d will output symbol k.

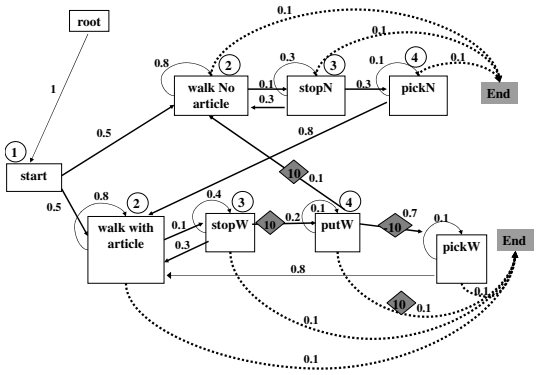

Figure 1: An example plan library. Recognition time-stamps in the text appear in circles. Costs appear in diamonds on edges.

In addition to transition and interruption probabilities, we add utility information on the edges in the plan library. The utilities on the edges represent the cost or gains to the observer, given that the observed agent selects the edge. For the remainder of the paper, we use the term cost to refer to a positive value associated with an edge or node. As in the probabilistic reasoning process, for each node we have three kinds of utilities: (a) E^{q^d} is the sequential transition utility (cost) to the observer, conditioned on the observed agent transitioning to the next plan-step, paralleling A^{q^d}; (b) $e_{i,end}^{q^d}$ is the interruption utility; and (c) Ψ^{q^d} is the decomposition utility to the observer, paralleling Π^{q^d}.

Figure 1 shows portion of the plan library of an agent walking with or without a suitcase in the airport, occasionally putting it up and picking it up again, an example discussed below. Note the *end* plan step at each level, and the transition from each plan-step to this end plan step. This edge represent the probability to interrupt. The utilities are shown in diamonds (we omitted zero utilities, for clarity). The transitions allowing an agent to leave a suitcase without picking it up are associated with large positive costs, since they signify danger to the observer.

We use these probabilities and utilities to rank the hypotheses selected by the SBR. First, we determine all paths from each hypothesized leaf in time-stamp $t-1$, to the leaf of each of the current state hypotheses in time stamp t. Then, we traverse these paths multiplying the transition probabilities on edges by the transition utilities, and accumulating the utilities along the paths. If there is more then one way to get from the leaf of the previous hypothesis to the leaf of the current hypothesis, then it should be accounted for in the accumulation. Finally, we can determine the *most costly* current plan-step (the current-state hypothesis with maximum expected cost). Identically, we can also find the *most likely* current plan-step, for comparison.

Formally, let us denote hypotheses at time $t-1$ (each a path from root to leaf) as $W = \{W_1, W_2, ..., W_r\}$, and the hypotheses at time t as $X = \{X_1, X_2, ..., X_l\}$. To calculate the maximum expected-utility (most costly) hypothesis, we need to calculate for each current hypothesis X_i its expected cost to the observer, $U(X_i|O)$, where O is the sequence of observations thus far. Due to the use of SBR to filter hypotheses, we know that the first $t-1$ observations in O have resulted in hypotheses W, and that observation t results in new hypotheses X. Therefore, under assumption of Markovian plan-step selection, $U(X_i|O) = U(X_i|W)$.

The most costly hypothesis is computed in Equation 1. We use $P(W_k)$, calculated in the previous time-stamp, and multiply it by the probability and the cost to the observer of taking this step from

W_k to X_i. This is done for all i, k.

$$\hat{X}_i = \operatorname*{argmax}_{X_i} \sum_{W_k \in W} P(W_k) \cdot P(X_i|W_k) \cdot U(X_i|W_k) \quad (1)$$

To calculate the expected utility $E(X_i|W_k) = P(X_i|W_k) \cdot U(X_i|W_k)$, let X_i be composed of plan steps $\{x_i^1, ..., x_i^m\}$ and W_k be composed of $\{w_k^1, ..., w_k^n\}$ (the upper index denotes depth). Given $w \in W_k$ and $x \in X_i$, there are two cases. In the first case, x and w have a common parent, and x is a direct decomposition of this common parent. Here, the expected utility is accumulated by climbing up vertices in w (by taking *interrupt* edges) until we hit a parent common to x and w, and then climbing down (by taking first child decomposition edges) to x. In the second case, x is reached by following a sequential edge from a vertex in w to a vertex in x.

A naive algorithm for computing the expected costs of hypotheses at time t can be expensive to run. It would go over all leaves of the paths in $t-1$ and for each of these, traverse the plan library until getting to all leaves of paths we got in time-stamp t. The worst-case complexity of this process is $O(N^2T)$, where N is the plan library size, and T is the number of observations.

3.3 Efficient UPR Algorithms

We developed a set of algorithms that calculates the expected utilities of hypotheses (Equation 1) in worst-case runtime complexity $O(NDT)$, where D is the depth of the plan library (N,T as above). The algorithms are based on the observation that the structural constraints on the plan library are such, that all the paths from any path (hypothesis) true at time $t-1$, to a given hypothesis X_i, true at time t, must necessarily go through a single node S that is a part of X_i. Moreover, S is necessarily a common node to X_i and one or more paths at time $t-1$. If we can propagate the utilities and probabilities up to this node S, then we could propagate down from it to all paths X in which it is a part, that are true at time t.

This translates into the following procedure. We begin with the leaves of all $t-1$ hypotheses W_k ($1 \leq k \leq n$). We sum the utilities and probabilities while climbing up from the leaves along the hierarchy, all the way to the root, storing intermediate sums in the internal nodes w^j (plan-steps) of the hierarchy. Then, any of those internal nodes (i) that has a child marked at time t (i.e., w^j is a common parent, Eq(w^j, x^j) is true); or (ii) that has a sequential transition to an internal node marked at time t (i.e., $a^j_{w,x} > 0$). In either of these cases, we have found a node (marked time t) through which one or more time t hypotheses X_i pass, i.e., a node S as above. We then propagate down the calculated probability and utility downward.

Complexity Analysis: the run-time complexity of the algorithm is $O(NDT)$: We first propagate the $t-1$ expected utilities up the hierarchy, not visiting plans that already been visited, in worst-case time $O(N)$. Then, calculating β for different depths, for paths tagged with t, is $O(ND)$. We do this for every observation, of which there are T, thus overall, we get $O(NDT)$.

Note the reliance on the underlying SBR: Since the symbolic recognizer provides the possible paths at times $t-1, t$, we do not need to consider all possible paths, and can begin the propagation process directly at the leaves of paths. Hopefully, many paths are disqualified by the symbolic algorithm, due to temporal coherence; in that case, we expect performance in practice to improve significantly over the worst case complexity.

3.4 Leaving unattended articles

It is important to track a person that leave her articles unattended in the airport. It is difficult, if not impossible, to catch this behavior using only probabilistic information. We demonstrate the process using the plan library in Figure 1. This Plan library is used to track simulated passengers in an airport that walk about carrying articles, which they may put down and pick up again. The recognizer's task is to recognize passengers that put something down, and then continue to walk without it. Note that the task is difficult because the plan-steps are hidden (e.g., we see a passenger bending, but cannot decide whether it pick something up, put something down, or neither; we cannot decide whether a person has an article).

Suppose that in time $t = 2$, the SBR had returned that the two plan-steps marked *walk* match the observations (*walkN* means walking with no article, *walkW* signifies walking with an article); in time $t = 3$ the two *stop* plan steps match ($stopN$ and $stopW$), and in time $t = 4$ the plan step $pickN$ and plan step $putW$, match (e.g., we saw that the observed agent was bending). The probability in $t = 4$ will be $P(putW|stopW) = 0.5 \times 0.2 = 0.1$ (the probability of $stopW$ in previous time-stamp is 0.5, then following sequential link to $putW$), and in the same way $P(pickN|stopN) = 0.5 \times 0.3 = 0.15$. Normalizing the probabilities for the current time $t = 4$, $P(putW|stopW) = 0.4$ and $P(pickN|stopN) = 0.6$. The expected utility in time $t = 4$ is $U(putW|stopW) = P(putW|stopW) \times E(putW|stopW) = 0.4 \times 10 = 4$. The expected utility of $pickN$ is still zero. If we had not paid attention to the utilities and picked just the observation with the maximum probability, we could miss important information, such as a passenger putting down an article and not picking it up.

4. SUMMARY AND FUTURE WORK

This paper presents a utility-based plan recognition (UPR) approach, for incorporating biases and preferences of the observer into keyhole plan recognition. This allows choosing recognition hypotheses based on their expected utility to the observer. While reasoning about such expected utilities is intractable in the general case, we present a hybrid symbolic decision-theoretic plan recognizer, whose complexity is $O(NDT)$, where N is the plan library size, D is the depth of the library and T is the number of observations. We plan to further explore the use of UPR algorithms in additional queries and cases such as *intended recognition*.

5. REFERENCES

[1] D. Avrahami-Zilberbrand and G. A. Kaminka. Fast and complete symbolic plan recognition. In *IJCAI-05*, 2005.

[2] E. Charniak and R. P. Goldman. A Bayesian model of plan recognition. *AIJ*, 64(1):53–79, Nov. 1993.

[3] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *CVPR (1)*, pages 838–845, 2005.

[4] S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

[5] R. A. Howard and J. E. Matheson. Influence diagrams. In R. A. Howard and J. E. Matheson, editors, *Readings on the Principles and Applications of Decision Analysis*. Strategic Decisions Group, 1984.

[6] W. Mao and J. Gratch. A utility-based approach to intention recognition. In *AAMAS 2004 Workshop on Agent Tracking: Modeling Other Agents from Observations*, NY City, NY, USA, July 2004.

[7] S. Noh and P. Gmytrasiewicz. Flexible multi-agent decision-making under time pressure. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 35(5):697–707, 2005.

[8] M. Tambe and P. S. Rosenbloom. RESC: An approach to agent tracking in a real-time, dynamic environment. In *IJCAI-95*, August 1995.