Assessing the Complexity of Plan Recognition

Christopher W. Geib

Honeywell Labs
3660 Technology Drive
Minneapolis, Minnesota 55418
christopher.geib@honeywell.com

Abstract

This paper presents a discussion of the theoretical complexity of plan recognition on the basis of an analysis of the number of explanations that any complete plan recognition algorithm must consider given various properties of the plan library. On the basis of these results it points out properties of plan libraries that make them computationally expensive.

Introduction

Plan recognition is a well studied problem in the Artificial Intelligence literature. Following others, we distinguish between plan recognition/task tracking and goal identification. By plan recognition we mean the process of identifying not only the top level goal an agent is pursuing but also the plan that is being followed and the actions that have been done in furtherance of the plan.

The algorithms that have been used to address plan recognition range from, graph covering (Kautz 1986), to Bayes nets (Bui 2003, Horvitz 1998), to Probabilistic State Dependent Grammars (Pynadath 2000). While a significant amount of information is known about the complexity of these algorithms what has previously been lacking in the literature is a discussion of how hard the actual problem for an individual plan library is. Without a discussion of the complexity of the actual problem, we may find ourselves using very powerful algorithms to solve problems that are amenable to simpler algorithms.

The rest of this paper is organized as follows, we will discuss HTN plans as a representation for plan libraries and define an explanation for a given set of observations. Then, making sure to divorce ourselves from any particular plan recognition algorithm, we will discuss the complexity of a plan library in terms of the number of possible explanations licensed by a given set of observations and the features of the domain that control this. This will leave us in a position to make predictions about the difficulty any complete algorithm for plan recognition will have with a particular domain.

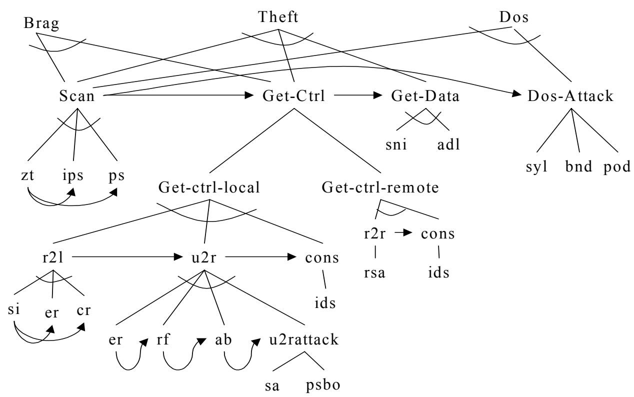

Figure 1: A small HTN plan library for a cyber security

Modeling Plans and Explanations

Much of the past work in plan recognition has at least tacitly been based on simple hierarchical task networks (HTN) (Kutluhan, Hendler, and Nau 1994a, 1994b) as the representation for plans. Figure 1 is an HTN plan library represented as partially ordered and/or trees. In most plan recognition systems this kind of plan library is given to the system to define the set of plans it is expected to recognize. In the figure, interior "and nodes" representing plan decomposition (all the children must be performed for the parent to be achieved) are represented by an undirected arc across the lines connecting the parent node to its children. Interior "or nodes", which represent choice points in the plan (only one of the children must be performed to achieve the parent) do not have this arc. Finally, basic actions that are directly observable by a plan recognition system are shown as leaf nodes of the trees. Directed arcs represent ordering constraints between plan nodes. For example, in Figure 1, action **zt** must be executed before **ips** and **ps**. In this paper, we will be considering the complexity of plan recognition limited to plans that can be represented in this formalism.

We define an *explanation of a set of observations* as a minimal forest of instances of plan trees with expansions chosen for "or" nodes sufficient to allow an assignment of each observation to a specific basic action in the plan. For example, Figure 2 shows one of many possible

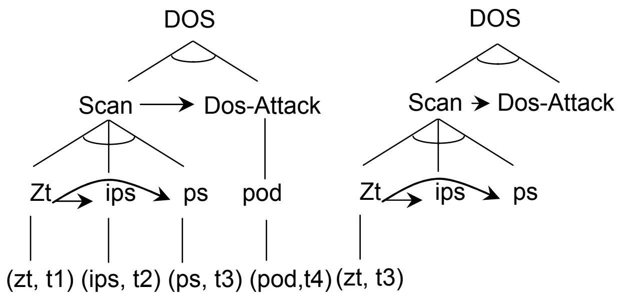

Figure 2: An Explanation

explanations for the set of observations (action, time pairs): {(zt, t1), (ips, t2), (zt, t3), (ps, t4), (pod, t5)} given the plan library shown in Figure 1. Note that the second Dos-Attack is not expanded in this explanation since it is not required to explain the observations. Also note that in this discussion we will be using an integer time notation.

Figure 2 shows two instances of the DOS (Denial of Service) goal being explained by this series of observations. Much research in plan recognition (Geib 2003, 2001a, 2001b, Goldman 1999, Goldman 1993) is committed to recognizing the interleaved execution of multiple goals that are active at the same time. In defining what is an acceptable explanation for a series for observations we will follow this tradition. What is less accepted in the research community is that plan recognition algorithms should consider multiple instances of the same goal as part of an explanation. Allowing multiple instances of the same goal makes the problem of plan recognition significantly harder, therefore we will provide a short justification for why algorithms must address this case.

Multiple Instances of the Same Goal

Consider the cyber security domain from Figure 1. In the real world, it is common for a determined cyber attacker to launch multiple different attacks against a single host, and even multiple instances of the same attack, to achieve a single goal. This is done for a number of reasons: diversity of target susceptibility, attack success likelihood, and to create confusion, among others. Thus in this domain it is very common to see multiple instances of the same goal being pursued by different, very similar, or even identical instances of plans. The explanation presented in Figure 2 has remained agnostic about the specific Dos-Attack that will be launched, however since the **zt** (zone-transfer) observed at time **t3** is consistent with a second DOS goal any complete algorithm for plan recognition must consider the possibility that there are multiple interleaved instances of this goal being pursued by a single agent at the same time.

Having explicitly, stated that we will be looking at multiple interleaved goals and multiple instances of the same goal. It is worth noting some possible aspects of plan recognition that will not be covered in this discussion. Plans with explicit looping constructs, observations that contribute to more than one plan, domains that are only partially observable, questions of the certainty of a given observation, and the possible abandonment of plans, can all increase the number of viable explanations for a given set of observations and hence the complexity of the plan recognition task. A complete treatment of these issues are outside the scope of this paper, however, since they all result in increasing the number of possible explanations for a set of observations, this work can be seen as a lower bound on the complexity of plan recognition in domains that address these issues.

Counting Explanations

We are interested in the worst case complexity of each problem domain. One way to examine the complexity of a domain to examine the maximal number of explanations that a given set of observations can have. Since any complete algorithm for plan recognition will have to examine all of the viable explanations for a given set of observations, this will give us a handle on the complexity of the domain. However, long sets of observations may result in more explanations than short sets. Therefore, we need to find a way to examine the complexity of the domain outside of a specific set of observations. That is, we are interested in identifying those plan properties that result in large sets of possible explanations. We will identify these properties by examining a single explanation, observing how the addition of a single observation changes the number of explanations that are licensed, and then identifying the properties of the plans in the explanation that enabled the changes.

One of the properties of an explanation is that each observation is assigned to a specific basic action. We will say an observation *explains* a basic action if it can be legally bound to the basic action within an explanation. Further, we define an *attachment point* as any basic action that does not have an observation assigned to it that could be explained by the current observation.

As with other highly constrained systems, the addition of a single observation can increase, decrease, or leave unchanged the number of explanations. That is, given an explanation, a new observation is either inconsistent with the explanation, in which case the number of explanations decreases, or it is consistent with the explanation in which case then number of explanation remains the same or increases.

In the following subsections we will examine the addition of an observation to a single explanation under various conditions and the effects it will have on the number of licensed explanations. We will first consider the simple case of observations that do not introduce new root goals to the explanation and then consider the case of adding observations that do.

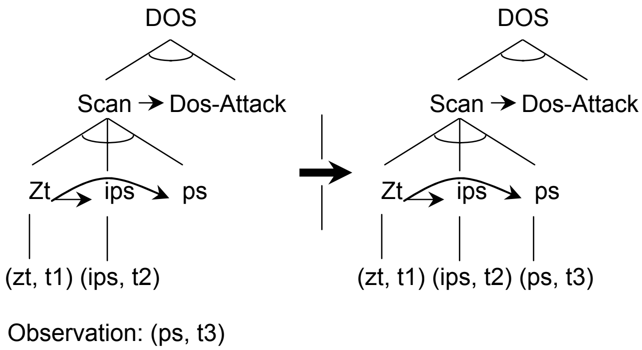

Figure 3: Left: an observation and explanation pair with a single point of attachment. Right: resulting explanation.

No New Root Goals

We will break the case of observations that merely extend the root goals that are already in an explanation into two sub-cases. First we consider the case where there is only a single attachment point in the explanation for the observation. In this case, the observation is added, the explanation remains consistent and no new explanations need to be introduced. An example of this is shown in Figure 3. Note that if there were no basic actions the observation could explain then the explanation could be discarded and the number of explanations would decrease.

Second, we consider the more interesting case where there are multiple attachment points for the observation in the explanation. In this case, the number of consistent explanations is multiplied by the number of attachment points. A complete algorithm must consider the possible binding of the observation to each of the candidate attachment points resulting in multiple possible explanations.

Informal empirical evaluation of one complete, incremental plan recognition algorithm (Geib and Goldman 2003, 2001a, 2001b) over three different domains suggest that the most common cases that do not introduce new root goals are those where there is only a single possible attachment point. While there are many cases in these domains where multiple instances of the same basic action could have potentially resulted in multiple attachment points, the incremental nature of the algorithm and the restrictions imposed by ordering constraints within the plan libraries prevented it. Thus both theoretical and empirical analysis suggest that these are not the computationally expensive cases for plan recognition.

New Root Goals the Simple Case

To discuss observations that introduce new goals, it will be helpful to define a *leader* as any basic action such that there exists a root goal in the plan library, such that, neither the basic action, nor any of its ancestors, is ordered after some other action within at least one possible plan for the root goal. Intuitively, a leader could be the first observed action within a plan for the designated goal. As in the case where no new goal is introduced it will be helpful to divide our discussion into two cases: the introduction of root goals and plans for them that have a single leader action and those that have multiple leader actions.

The introduction of single leader plans is very similar to the cases that do not introduce a new root goal. The number of explanations increases linearly with the number of root goals for which the action is a leader. Thus, it is actually only necessary to introduce a new explanation for each of the possible root goals that could be introduced by the action. In effect, each of the root goals for which the action is a leader can be seen as continually active attachment point. Figure 4 presents an example of this case.

New Root Goals with Multiple Leaders

For the rest of this discussion, we will make a small alteration to the plan library that we have been using in our examples. For the rest of this discussion we will assume that the ordering constraints from **zt** to **ips** and **ps** have been deleted. That is **zt** is no longer the sole leader for all three of the root goals. Instead all three goals share the common set of unordered leaders {**zt, ips, ps**}.

The first observed leader for a root goal with multiple leaders is very similar to the previous case of a new goal with a single leader. The new observation only requires that the set of explanations be increased linearly in the number of root goals for which the observed action is a leader. In our example plan library, this means that observing any one of **zt**, **ips**, or **ps** will generate three explanations, one where the observed action contributes to a new instance of each of **DOS**, **Theft**, or **Brag.** This looks just like Figure 4 from the previous case. However, a very different thing happens when a second leader for the same root goal is seen.

Since the second observed leader for the root goal is unordered with respect to the first, we are no longer allowed to assume that it must be part of the same root goal. That is, we cannot be certain if the observation is contributing to an existing plan or if it is introducing a new goal. Thus if we are to consider all the possible explanations, we must consider both the possibility that the observation is contributing to the existing goal and that it is introducing a new root goal. An example of this is shown in Figure 5.

Now consider what happens when the third leader for this example is seen. We must consider three cases: 1) all three actions are contributing to the same goal, 2) that any two of them may be contributing to a single goal and the other to a second goal, and 3) that all three contribute to distinct root goal instances. On top of this all three of these cases must be enumerated for all goals that share the common leaders.

If we extend this case to have m unordered leaders that are shared by n root goals then, when we see all m leaders we will be forced to consider the set of all subsets of the up to m instances of the n root goals and all the possible distributions of the m leaders between them. Clearly this will grow quite fast, but having identified this as the case that will result in a large increase in the number of explanations, the question remains, how fast does this grow? Given a plan library that has a set of m unordered leaders shared by n plans. Then in the worst case where one of each of the unordered leaders is observed in a row, a single explanation will expand to exactly:

$$\sum_{i=i}^{m} \left\{ {m \atop i} \right\} n^{i}$$

explanations. The summation captures the fact that the number of possible root goals varies from one to m. The first term in the summation represents the distribution of observed leaders to root goal instances, captured by Sterling's numbers of the second kind (Bogart 1990). This number is exactly the number of ways that the m actions can be distributed to the i root goals so that each goal has at least one action. The second term in the summation reflects the number of possible permutations of the root goals. To provide some intuition about the rate of growth of this function, consider the following. This function is bounded below by n^m which is seen by considering the last term of the summation.

The function is also bounded above by $(mn)^m$. This can be seen by creating an alphabet of letters to assign to each observed unordered leader to tell us what goal it contributes to. Since we know that in the worst case there will be m goals, each possible shared goal α is given m letters $\alpha_1, \alpha_2, \ldots \alpha_m$ each one representing the possible goal instances of α. This creates an alphabet of mn letters. If each leader is given one letter to indicate which instance of which goal it contributes to. Since we are interested in the case of m observations there are $(mn)^m$ such combinations. This represents an upper bound because, many of these combinations are not legal assignments of leaders to root goal instances. For example, none of the leaders could be assigned to contribute to the second instance of a particular goal unless there were already a first instance of the same goal, however, nothing in the construction of our alphabet or the assignment of letters to leaders prevents this.

n^m represents significant growth in the number of explanations and will be a dominating source of complexity for the domain. Note that multiple instances of any one of the leaders will not have this kind of exponential growth. Consider the case of the following observations {(**zt**, 1), (**zt**, 2), (**zt**,3)}. Since any given instance of the root goals can only have one instance of **zt** we know that each of these observations must introduce a new root goal. Since they can't contribute to each other the set of explanations remains constant in size (introducing a new goal with each action.)

It is important to realize that while this problem is significantly amplified by having multiple root goals that share the unordered leaders, this problem occurs even when the leaders are not shared. Note that the exponent of this problem is the length of the unordered leader, not the number of goals sharing the leader. Thus the number of

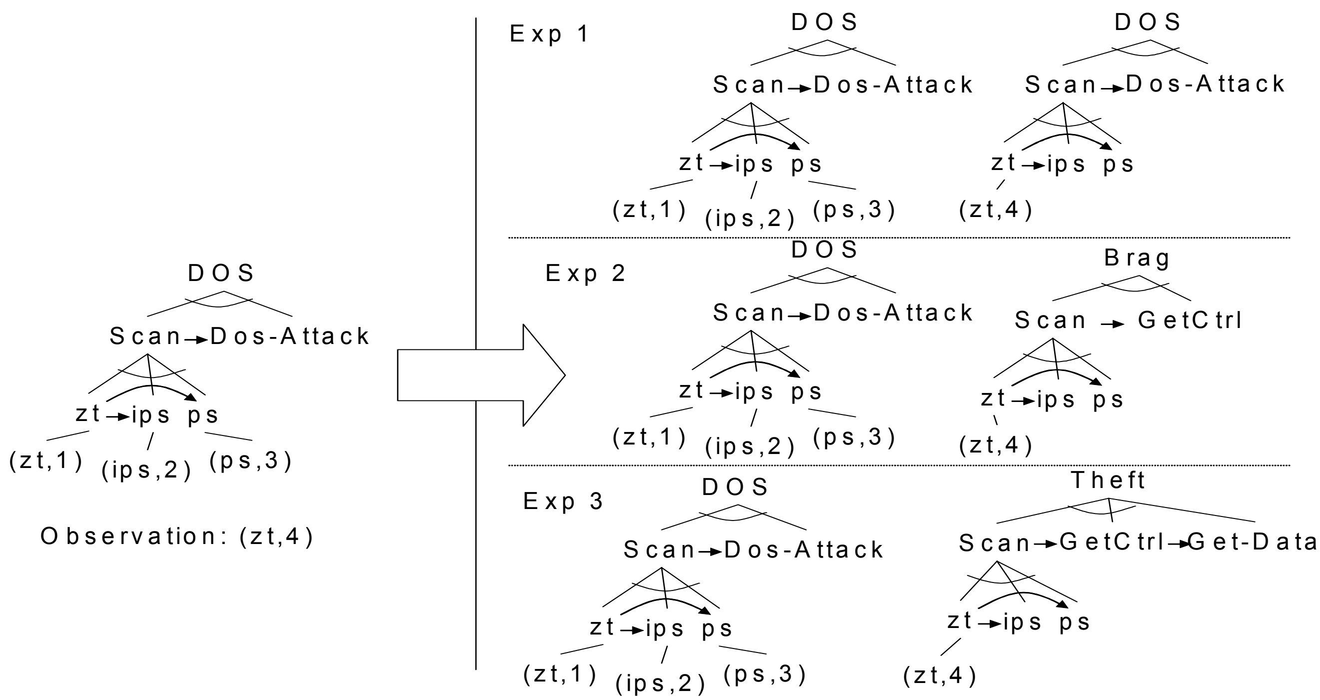

Figure 4: The Explanations that result from observing the single leader for three root goals

explanations will grow exponentially even if the leaders are not shared. That is, since, any complete algorithm will have to consider multiple instances of the same root goal, this problem will occur even for a single plan with unordered leaders, albeit to a lesser degree.

Implications

To summarize, shared unordered leaders in a plan library have the potential to exponentially increase the number of explanations that have to be considered by any complete plan recognition algorithm. All other cases have, at worst, a increase in the number of explanations that is linear in the number of attachment points. Thus, plan libraries that have exceptionally large numbers of repeated actions or long shared unordered plan leaders will likely produce very large runtimes. Note that this shows that partial ordering of the plans can in fact have a more significant impact on the runtime of any plan recognition algorithm than even the sharing of actions across plans.

What is the most disheartening about this result is that this growth in the number of explanations will happen even when there is only a single instance of the root goal that is intended and a single set of the shared unordered leaders is executed. Encouragingly, in the case where there is only a single instance of one of the root goals being executed by the agent, once the leaders have all been executed a significant reduction of the number of explanations is possible. Given the ordering constraints that by definition end the unordered leaders, we can begin to eliminate explanations that involve multiple goals since they will be inconsistent. For example, consider the following set of observations: {(**ips**, 1), (**ps**, 2), (**zt**, 3), (**pod**, 4) again taken from the plan library of Figure 1 (with the initial ordering constraints removed between **zt**, **ips** and **ps**.). If we had just the unordered shared leaders the set of explanations would be quite large, however for the presence of the **pod** action to be legal within this plan library all three of the shared leaders must contribute to a single instance of the **scan** action. This allows us to eliminate all but one of the possible explanations.

Given these results some might be tempted to artificially introduce ordering into the plans in the library in order to reduce the number of possible explanations. One could enumerate all of the possible orderings for the shared unordered leaders, producing a plan library that had only shared ordered leaders. Of course in the end all this will do is push the implicit complexity of the ordering constraints into explicit complexity of the plan library itself. Further since all of the orderings must be enumerated and none of the complexity associated with multiple goals is addressed, this will not succeed in reducing the complexity of the problem.

To Enumerate or Not

Beyond the conclusions about the properties of plan libraries that make plan recognition complex, these results shed light on the question of when to use which plan recognition algorithm. Some algorithms call for an explicit enumeration of the explanation space before seeing any of the observations (Bui 2003, Kautz 1986). Since the size of the explanation space depends on the number of observations that will be processed these approaches have often limited themselves to a single instance of each root goal. Further, as these results have shown, enumeration of the explanation space will require an exponential number

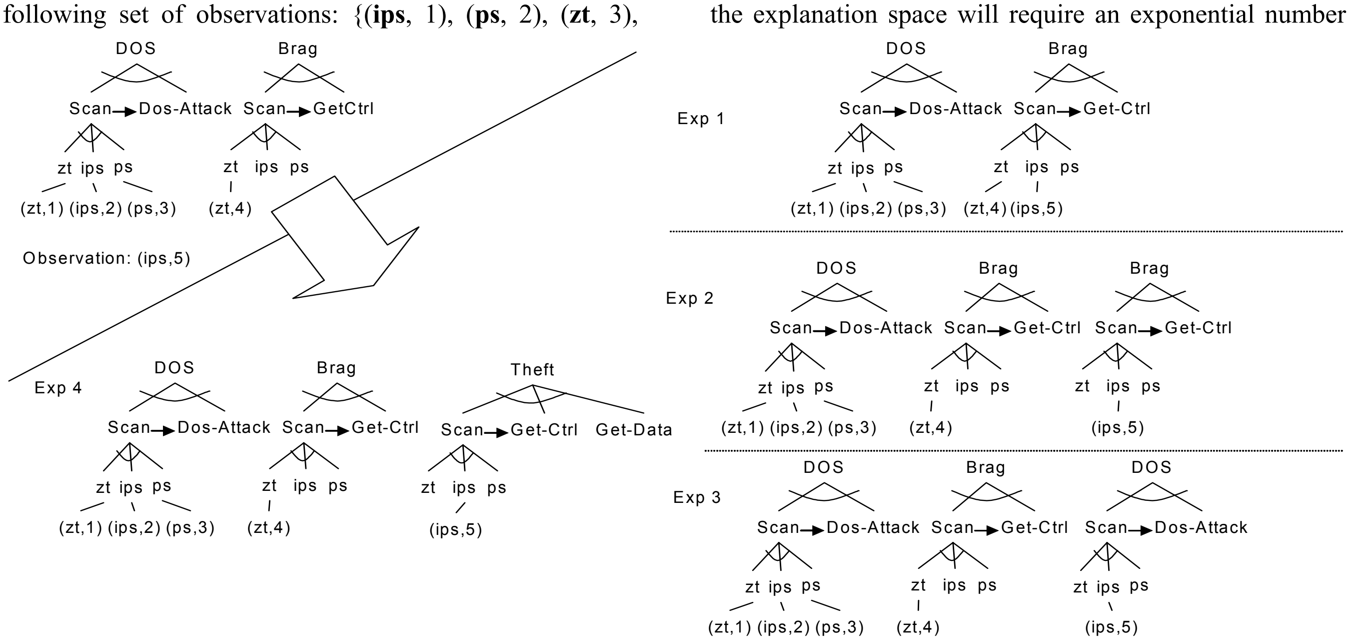

Figure 5:Upper Left: One of the explanations that explains an observation of the first unordered leaders. Lower Right: The four explanations required to explain the observation of the second unordered leader for the three root goals DOS, Brag, and Theft

of cases for some plan libraries. Thus these results should caution against such approaches to plan recognition in domains with shared unordered leaders. Instead, approaches that are incremental and only build those portions of the explanation space that are needed by the observations will be more effective.

Conclusions

In this paper we have examined the way in which the set of possible explanations for a given set of observations grows in an effort to understand the complexity of complete plan recognition algorithms. We have demonstrated that the number of such explanations can grow exponentially in the number of root goals that share a common unordered prefix of actions.

These results have given us a concrete handle on those aspects of a plan library that may make it computationally difficult. Shared leaders are a result of having domains where many different plans for different goals have a "common prefix" of actions. Where the steps of such a prefix are unordered plan recognition will involve consideration of a large number of explanations.

This strongly argues that when given a choice, if a plan library must contain multiple plans for root goals with a common prefix that this prefix should be kept as short as possible and that ordering constraints should be introduced whenever possible within such a prefix.

Acknowledgements

This material is based upon work supported by the Advanced Research and Development Activity (ARDA) under contract No. NBCHC030078.

References

Bogart, K., 1990, *Introductory Combinatorics*, (2nd Ed.), Orlando, Fl., Harcourt Brace Jovanovich.

Bui, H., 2003, "A General Model for Online Plan Recognition," *Proceedings of IJCAI-2003.*

Geib, C., and Goldman, R., 2003, "Recognizing Plan/Goal Abandonment," *Proceedings of IJCAI-2003*.

Geib, C., and Goldman, R., 2001a, "Probabilistic Plan Recognition for Hostile Agents," *Proceedings of the FLAIRS01 Conference*, Key West.

Geib, C. and Goldman, R., 2001b, "Plan Recognition in Intrusion Detection Systems," *Proceedings of the DISCEX-II Conference*.

Goldman, R., Geib, C., and Miller, C., 1999, "A New Model of Plan Recognition," In *Proceedings of the 1999 Conference on Uncertainty in Artificial Intelligence*.

Goldman, R. and Charniak, E., 1993, "A Bayesian Model of Plan Recognition", *Artificial Intelligence*, vol 64(1)

Horvitz, E., Breese, J., Heckerman, D., Hovel, D., and Koos, R., 98, "The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users", *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI-98)*.

Kautz, H., and Allen, J., 1986, "Generalized Plan Recognition", *Proceeding of the National Conference on Aritficial Intelligence (AAAI-86).*

Kutluhan, E., Hendler, J. and Nau, D., 1994, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," *Artificial Intelligence Planning Systems: Proceedings of the Second International Conference (AIPS-94)*, Chicago.

Kutluhan, E., Hendler, J. and Nau, D., 1994, "HTN Planning: Complexity and Expressivity," *Proceedings of the National Conference on Artificial Intelligence (AAAI-94)*.

Poole, D., 1997, "The Independent Choice Logic for Modeling Multiple Agents Under Uncertainty," *Artificial Intelligence*, 94(1-2):7-56.

Pynadath, D., and Wellman, M., 2000, "Probabilistic State-Dependent Grammars for Plan Recognition", *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI-2000).*