Recognizing Agent Plan/Goal Abandonment
A significant extension to work on abductive probabilistic plan recognition

Problem
The ability to recognize when an agent abandons a plan or goal is important to successful implementation of plan recognition in real systems.

Historical Approaches
Existing plan recognition systems have adopted a number of methods to work around the problem:

• Only worry about the agent’s current goal [Horvitz et al., 1996]
• Assume the agent only has one goal [Conati et al., 1997; Darrell et al., 1998]
• Assume the agent will always come back [Katz and Allen, 1986]
• Rely on a cooperative agent for disambiguation

The Exact Solution
Background
Goldman et al. [2002b]; Gab and Goldman [2001b; 2001c] provide an algorithm for plan recognition based on a model of the execution of hierarchical task network (HTN) plans. As the agent executes one of the set of pending actions, a new set of pending actions is generated from which the next action will be chosen, and so forth.

To handle more than one time step and to handle partially ordered plans, this formula must be generalized slightly.

• The existing search space is multiplied by 2^n (the number of possible sets of abandoned goals)
• The need for a probability distribution over the set of goals the agent could abandon
• Assume the agent only has one goal [Conati et al., 1997; Gertner et al., 1998]

Thus, computing the exact probability of an explanation considering goal abandonment presents two problems:

• Accuracy
  – The existing search space is multiplied by 2^n (the number of possible sets of abandoned goals)
  – The need for a probability distribution over the set of goals the agent could abandon

APPROXIMATION BY MODEL REVISION
Lackey [1991]; Jensen [Jensen et al., 1992]; Halpern [1976] have suggested that unexpectedly small values for the statistic of (Observations/model) indicate a model mismatch. In our case this mismatch can be interpreted as the abandonment of a goal.

In this light, consider computing the probability that none of the observed actions in a subsequence (say a to t) contribute to one of the goals (call it G). (we denote this as: P(notContrib(G,a,t)))

Computing “not Contrib”
Given this very simple plan library and assuming the following sequence of observations: happen(a,0), happen(b,1), happen(g,2), happen(g,3), the probability of seeing a at time 2 is given by P(abandoned(a)|c) where c is the number of elements in the pending set that have c as the next action. The probability that we do not see c is then:

\[ 1 - \frac{1}{\text{PS}_G} \]

To handle more than one time step and to handle partially ordered plans, this formula must be generalized slightly:

\[ \prod_{i=1}^{t} \left( \frac{1}{\text{PS}_G} \right) \]

Honeywell

Runtime
Runtime tests were done using the test library:

The chart below shows a scatter of runs with the PAT set at 0.75. The system believed 1138 of the runs had at least one abandoned goal, but still completed the majority of the runs in less than one second.