Practical Goal Recognition for ISS Crew Activities

Yolanda E-Martín[1,2] and **María D. R-Moreno**[1] and **David E. Smith**[3]
[1] Departamento de Automática. Universidad de Alcalá. Ctra Madrid-Barcelona, Km. 33,6 28871 Alcalá de Henares (Madrid), Spain.
{yolanda,mdolores}@aut.uah.es
[2] Universities Space Research Association. 615 National Ave, Suite 220, Mountain View, CA 94043
[3] Intelligent Systems Division. NASA Ames Research Center. Moffett Field, CA 94035-1000
david.smith@nasa.gov

Abstract

In recent years, there has been increasing interest in the development of intelligent robots that can assist with astronaut activities. An example of this is the use of free-flying robots to assist astronauts aboard the International Space Station (ISS). The Smart Synchronized Position Hold, Engage, Reorient Experimental Satellites (SPHERES) free-flying robots currently aboard the ISS are capable of functioning autonomously, conducting zero gravity robotics experiments, carrying mobile sensors, and inspecting items using a built in camera. They can also potentially offer supervision, advice, and support for the astronauts, but goal recognition is essential for this capability. More generally, recognizing an agent's goals from some or all of the agent's observed actions is critical for any application that involves cooperation between people and machines. In this paper, we describe a fast goal recognition technique that can be applied to ISS crew member activities.

Introduction

Recognizing an agent's goals from some or all of the agent's observed actions is an important technological capability for applications that involves cooperation between humans and machines, such as intelligent personal assistants (Weber and Pollack, 2008), smart environments (Wu et al., 2007), monitoring user's needs (Pollack et al., 2003; Kautz et al., 2002), and intelligent tutoring systems (Brown, Burton, and Hausmann, 1977). For human space exploration, there has been increasing interest in the development of intelligent robots that can assist with astronaut activities. An example of this is the use of free-flying robots to assist astronauts aboard the International Space Station (ISS). The Smart Synchronized Position Hold, Engage, Reorient Experimental Satellites (SPHERES) free-flying robots currently aboard the ISS are capable of functioning autonomously, conducting zero gravity robotics experiments, carrying mobile sensors, and inspecting items using a built in camera. They can also potentially offer supervision, advice, and support for the astronauts. However, goal recognition is essential for this capability, because an astronaut often doesn't fully communicate his or her intentions and objectives to a robotic assistant or to others in the vicinity.

During the last few years, AI planning has been applied to solve goal recognition problems. In particular, Jigui and Minghao (2007) developed Incremental Planning Recognition (IPR) that is based on reachability in a plan graph. Ramírez and Geffner (2010) developed a goal recognition theory for estimating the probability of each possible goal, given the observations. The likelihood of a goal given a sequence of observations is computed using the cost difference between achieving the goal complying with the observations, and achieving the goal not complying with the observations. This cost is computed by means of two calls to a planner for each possible goal. A significant drawback to Ramírez's approach is the computational cost of calling a planner twice for each possible goal. This makes the approach impractical for real-time goal recognition in non-trivial domains.

Nevertheless, Ramírez and Geffner's framework provides a compelling domain-independent theory of the likelihood of a goal given a sequence of action observations. We make use of this framework to develop a heuristic goal recognition technique (E-Martín, R-Moreno, and Smith, 2015a). This approach builds a plan graph and propagates cost and interaction information through it. These cost estimates are more accurate than usual because of the use of *interaction* (Bryce and Smith, 2006). Using these cost estimates, we can then make use of Ramírez and Geffner's framework to quickly estimate goal probabilities.

We are applying this approach to an ISS Crew Activities Domain (ISS-CAD) that focuses on maintenance tasks for the Environmental Control and Life Support System (ECLSS) (Bagdigian, 2008).

In the next section we provide an overview of the goal recognition techniques from the perspective of planning. Next we describe our ISS Crew Activities Domain. Finally, we present some empirical results, and discuss future work.

Goal Recognition Background

Different techniques have been used to solve plan recognition problems: hand-coded action taxonomies (Kautz and Allen, 1986), probabilistic belief networks (Huber, Durfee, and Wellman, 1994), consistency graphs (Lesh and Etzioni, 1995), Bayesian inference (Albrecht et al., 1997; Bui, 2003), machine learning (Bauer, 1998), parsing algorithms (Geib and Goldman, 2009), and more recently AI planning. The following subsections describe two goal recognition techniques from the perspective of planning.

Incremental Plan Recognition

Jigui and Minghao (2007) developed a framework for plan recognition that narrows the set of possible goals by incrementally pruning a plan graph as actions are observed. The approach consists of building a plan graph to determine which actions and which propositions are true (1), false (-1), or unknown (0) given the observations. For level zero, since it is assumed that the initial state is true, every proposition has value 1. In addition, when an action is observed at a level it gets value 1. The process incrementally builds a plan graph and updates it level by level. The values of propositions and actions are updated according to the following rules:

1. An action in the plan graph gets value -1 when any of its preconditions or any of its effects is -1.
2. An action in the plan graph gets value 1 when it is the sole producer of an effect that has value 1.
3. A proposition in the plan graph gets value -1 when all of its producers are -1, `noop` included.
4. A proposition in the plan graph gets value 1 when any of its consumers or any of its producers is 1.
5. An action or proposition in the plan graph gets value -1 when it is mutually exclusive with an action or proposition that has value 1.

The process results in a plan graph where each proposition and each action is labeled as 1, -1, or 0. Those propositions and actions identified as -1 can be ignored for plan recognition purposes, meaning that these are pruned from the resulting plan graph. To illustrate this propagation and pruning technique, consider a simple problem with three operators:

$$\begin{aligned} A &: y \rightarrow z \\ B &: y \rightarrow, \neg y, t \\ C &: t \rightarrow k, \neg t \end{aligned} \tag{1}$$

Suppose that the sequence of observed actions is A at level 0, and C at level 2. Figure 1 shows the plan graph for this problem. The numbers above the propositions and actions are the values for each proposition and action computed using the above propagation rules. As a result of the propagation, z must be true (has value 1) at level 1 because action A was observed. As a result, since A and B are mutually exclusive, action B and its effects t and $\neg y$ are false (have value -1) at level 0. As a consequence of t being false, C is also false at level 1 along with its effects k and $\neg t$. At level 2, k and $\neg t$ must be true because action C was observed. (This results in t being true at level 1.) Since A and C, and B and C are mutually exclusive, A, B, $\neg y$, and t are not possible (have value -1) at level 2. Action B is unknown (has value 0) at level 1 since there is not enough information to determine whether it is true or false. The proposition y is true at level 0 since it is assumed that the initial state is true, and is unknown at level 1 because there is not enough information to determine whether it is true or false. However, it is false at level 2 due to the mutual exclusion between C and noop-y. Proposition z is true at every level since there are no operators in the domain that delete it.

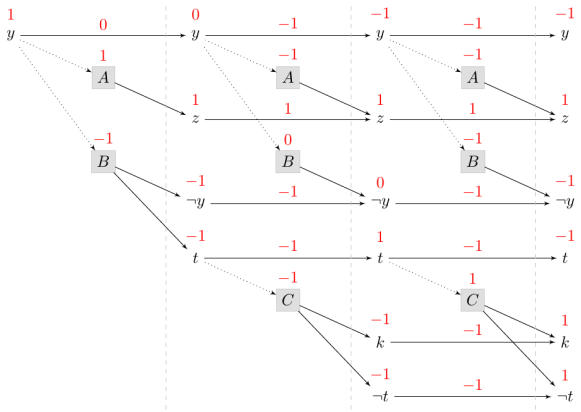

Figure 1: A plan graph with status values of propositions and actions

Plan Recognition as Planning

Ramírez and Geffner (2009) developed an approach that identifies goals where the observed actions are compatible with an optimal plan for one or more of those goals. To identify those plans, they used optimal and satisficing planners, as well as a heuristic estimator (Keyder and Geffner, 2007), which approximates the solution by computing a relaxed plan. The limitation of this work is the assumption that agents act optimally – suboptimal plans compatible with the given observations are not considered. To remove this limitation, they developed an approach for estimating the probability of each possible goal, based on how far the observations deviate from good plans for that goal (2010; 2012). They define a goal recognition problem to be a tuple $T = \langle P, \mathcal{G}, O, Pr \rangle$ where P is a planning domain and initial conditions, $\mathcal{G}$ is a set of possible goals or hypotheses, O is the observed action sequence $O = o_1, ..., o_n$, and Pr is the prior probability distribution over the goals in $\mathcal{G}$. The solution to a plan recognition problem is a probability distribution over the set of goals $G \in \mathcal{G}$ giving the relative likelihood of each goal. These posterior goal probabilities $P(G|O)$ can be characterized using Bayes Rule as:

$$Pr(G|O) = \alpha \, Pr(O|G) \, Pr(G) \tag{2}$$

where α is a normalizing constant, $Pr(G)$ is the prior distribution over $G \in \mathcal{G}$, and $Pr(O|G)$ is the likelihood of observing O when the goal is G. Ramírez goes on to characterize the likelihood $Pr(O|G)$ in terms of cost differences for achieving G under two conditions: complying with the observations O, and not complying with the observations O. More precisely, Ramírez characterizes the likelihood, $Pr(O|G)$, in terms of a Boltzman distribution:

$$Pr(O|G) = \frac{exp\{-\beta \, \Delta(G, O)\}}{1 + exp\{-\beta \, \Delta(G, O)\}} \tag{3}$$

where β is a positive constant and $\Delta(G, O)$ is the cost difference between achieving the goal with and without the observations:

$$\Delta(G, O) = Cost(G|O) - Cost(G|\overline{O}) \tag{4}$$

Putting equations (2) and (3) together yields:

$$Pr(G|O) = \alpha \, \frac{exp\{-\beta \, \Delta(G, O)\}}{1 + exp\{-\beta \, \Delta(G, O)\}} \, Pr(G) \tag{5}$$

By computing $\Delta(G, O)$ for each possible goal, equation 5 can be used to compute a probability distribution over those goals. The two costs necessary to compute Δ can be found by optimally solving the two planning problems $G|O$ and $G|\overline{O}$. Ramírez shows how the constraints O and $\overline{O}$ can be compiled into the goals, conditions and effects of the planning problem so that a standard planner can be used to find plans for $G|O$ and $G|\overline{O}$.

Heuristic Goal Recognition

We developed a heuristic approach for goal recognition based on Ramirez's framework that combines two principal ideas: 1) cost and interaction estimates using a plan graph (E-Martín, R-Moreno, and Smith, 2015b), and 2) the pruning technique of Incremental Plan Recognition. As with Ramírez, it assumes that a goal recognition problem $T = \langle P, \mathcal{G}, O, Pr \rangle$ is given, which includes the problem P (planning domain and initial conditions), the set of possible goals G, a set of observations O, and a prior distribution over the possible goals G. It is also assumed that the sequence of observed actions may be incomplete, but is accurate (not noisy).

Plan Graph Cost Estimation Simple propagation of cost estimates in a plan graph is a technique that has been used in a number of planning systems to do heuristic cost estimation (e.g: (Do and Kambhampati, 2002)). Unfortunately, the cost information computed this way is often inaccurate because it assumes independence between propositions and between actions.

Cost Interaction is a quantity that represents how more or less costly it is that two propositions or actions are established together instead of independently. Formally, the optimal Interaction, I^*, considers n-ary interaction relationships among propositions and among actions (p_0 to p_n) in the plan graph, and it is defined as:

$$I^*(p_0, ..., p_n) = cost^*(p_0 \wedge ... \wedge p_n) - [cost^*(p_0) + ... + cost^*(p_n)] \quad (6)$$

where the term $cost^*(p_0 \wedge ... \wedge p_n)$ is the minimum cost among all the possible plans that achieve all the members in the set. Computing I^* would be computationally prohibitive. As a result, we limit the calculation of these values to pairs of propositions and pairs of actions in each level of a plan graph – in other words, binary interaction:

$$I^*(p, q) = cost^*(p \wedge q) - [cost^*(p) + cost^*(q)] \quad (7)$$

A value $I < 0$ means that two propositions or actions are synergistic – that is, the cost of establishing both is less than the sum of the costs of establishing the two independently. When $I = 0$ the two propositions or actions are independent. When $I > 0$ the two propositions or actions interfere with each other, so it is harder to achieve them both than to achieve them independently. The extreme value $I = \infty$, indicates that the two propositions or actions are mutually exclusive. One can therefore think of interaction as a more nuanced generalization of the notion of mutual exclusion.

The computation of cost and interaction information begins at level zero of a plan graph and proceeds sequentially to higher levels. For level zero we assume 1) the cost for propositions at this level is 0 because the initial state is given, and 2) the interaction between each pair of propositions is 0, that is, the propositions are independent. With these assumptions, we start the propagation by computing the cost of the actions at the first level of the plan graph. In general, for an action *a* at level *l* with a set of preconditions $\mathcal{P}_a$, the cost is approximated as:

$$cost(a) = cost(\mathcal{P}_a) \approx \sum_{x_i \in \mathcal{P}_a} cost(x_i) + \sum_{\substack{(x_i, x_j) \in \mathcal{P}_a \\ j<i}} I(x_i, x_j)$$

The next step is to compute the interaction between two actions. The interaction between two actions *a* and *b* at level *l* is:

$$I(a, b) = \begin{cases} \infty \text{ if } a \text{ and } b \text{ are mutex by inconsistent effects or interference} \\ Cost(a \wedge b) - Cost(a) - Cost(b) \text{ otherwise} \end{cases}$$

where $cost(a \wedge b)$ is defined to be $cost(\mathcal{P}_a \cup \mathcal{P}_b)$, which is approximated as in () by:

$$cost(\mathcal{P}_a \cup \mathcal{P}_b) \approx \sum_{x_i \in \mathcal{P}_a \cup \mathcal{P}_b} cost(x_i) + \sum_{\substack{(x_i, x_j) \in \mathcal{P}_a \cup \mathcal{P}_b \\ j<i}} I(x_i, x_j)$$

If the actions are mutex by inconsistent effects, or interference, then the interaction is ∞. Otherwise, the interaction above simplifies to:

$$I(a, b) \approx \sum_{\substack{x_i \in \mathcal{P}_a - \mathcal{P}_b \\ x_j \in \mathcal{P}_b - \mathcal{P}_a}} I(x_i, x_j) - \left[\sum_{x_i \in \mathcal{P}_a \cap \mathcal{P}_b} cost(x_i) + \sum_{\substack{(x_i, x_j) \in \mathcal{P}_a \cap \mathcal{P}_b \\ j<i}} I(x_i, x_j) \right]$$

For a proposition *x* at level *l*, achieved by the actions $\mathcal{A}_x$ at the preceding level, the cost is calculated in the same way as for traditional plan graph cost estimation:

$$Cost(x) = \min_{a \in \mathcal{A}(x)} [Cost(a) + Cost_a]$$

where $\mathcal{A}_x$ is the set of actions that support x.

Finally, in order to compute the interaction between two propositions at a level *l*, we need to consider all possible ways of achieving those propositions at the previous level. That is, all the actions that achieve the pair of propositions and the interaction between them. Suppose that $\mathcal{A}_x$ and $\mathcal{A}_y$ are the sets of actions that achieve the propositions *x* and *y* at level *l*. The interaction between *x* and *y* is then:

$$\begin{aligned} I(x, y) &= cost(x \wedge y) - cost(x) - cost(y) \\ &= \min_{\substack{a \in \mathcal{A}_x \\ b \in \mathcal{A}_y}} \left\{ cost(a \wedge b) \right\} - cost(x) - cost(y) \\ &\approx \min \left\{ \begin{array}{l} \min_{a \in \mathcal{A}_x \cap \mathcal{A}_y} cost(a) + Cost_a \\ \\ \min_{\substack{a \in \mathcal{A}_x - \mathcal{A}_y \\ b \in \mathcal{A}_y - \mathcal{A}_x}} \left[\begin{array}{l} cost(a) + Cost_a + \\ cost(b) + Cost_b + \\ I(a, b) \end{array} \right] \end{array} \right\} \\ &\quad - cost(x) - cost(y) \end{aligned}$$

Using these equations, a cost-plan graph is built until quiescence. On completion, each possible goal proposition has an estimated cost of achievement, and there is an interaction estimation between each pair of goal propositions. Using this information, one can estimate the cost of achieving a conjunctive goal $G = g_1, ..., g_n$ as:

$$Cost(G) \approx \sum_{i=1}^{n} \left[Cost(g_i) + \sum_{j<i} I(g_i, g_j) \right] \quad (8)$$

With this information, we can estimate the cost of a plan for each possible goal G. While this allows us to estimate $Cost(G)$, what we need for goal recognition is to compute $\Delta(G, O)$, which requires $Cost(G|O)$ and $Cost(G|\overline{O})$. Unless O is a subsequence of every optimal plan for G, $Cost(G|\overline{O}) = Cost(G)$. Even when this is not the case, these two costs tend to be similar because the cost of the cheapest suboptimal plan is usually very close to the cost of the optimal plan. (This is confirmed in our experiments; for most problems there is little or no difference between $Cost(G|\overline{O})$ and $Cost(G)$.) As a result, we approximate $Cost(G|\overline{O})$ by $Cost(G)$, which can be estimated as shown above. To estimate $Cost(G|O)$ we modify the plan graph as described in the next section, and repropagate cost and interaction information.

Relaxing the Time Step Assumption We have modified the IPR pruning technique in order to relax the assumption of knowing the time step of each action in the observed sequence. Like Ramírez and Geffner (2010), we assume that the sequence of actions is sequential. Initially, we assign an *earliest time step* (*ets*) i to each action o in the observed sequence. The *ets* is given by the order of each action in the observed sequence. That is, given $(o_0, o_1, ..., o_i)$, the *ets* for each action is: *ets*(o_0)=0, *ets*(o_1)=1, *ets*(o_2)=2, etc. When the pruning process starts, we establish that an observed action o is possible to be observed at the assigned level i if all its preconditions are true (value 1) and/or unknown (value 0), and they are not mutually exclusive at level $i - 1$. Otherwise, the action cannot be executed at that level, which results in an update of the *ets* of each remaining action in the observed sequence. For instance, considering the initial sequence where *ets*(o_0)=0, *ets*(o_1)=1, *ets*(o_2)=2, and o_0 can be executed at level 0. If o_1 cannot be executed at level 1, then *ets*(o_1)=2 and *ets*(o_2)=3. If necessary, the cost-plan graph will be expanded until an *ets* is assigned to each observed action in the sequence. To illustrate this method, consider the example from Figure 1. Let us suppose the sequence of observed actions is A and C, with initial *ets* 0 and 1 respectively. As a result of the propagation, z must be true (have value 1) at level 1 because action A was observed. As a result, since A and B are mutually exclusive, action B and its effects t and $\neg y$ are false (have value -1) at level 0. Action C is initially assumed to be at level 1, but this cannot be the case because its precondition t is false at level 0. Therefore, the *ets* for C is updated to 2. The result of this updating is that each observed action is assumed to occur at the earliest possible time consistent with both the observation sequence and the constraints found in constructing the plan graph, using the interaction information.

Computing Goal Probabilities With the plan graph cost estimation technique and the observation pruning technique described in the previous sections, we can now put these pieces together to allow fast estimation of cost differences $\Delta(G, O)$, giving us probability estimates for the possible goals $G \in \mathcal{G}$. The steps are:

1. Build a plan graph for the problem P (domain plus initial conditions) and propagate cost and interaction information through this plan graph according to the technique described in Section .
2. For each (possibly conjunctive) goal $G \in \mathcal{G}$ estimate the $Cost(G)$ from the plan graph using equation (8).
3. Prune the plan graph, based on the observed actions O, using the technique described in the previous section.
4. Compute new cost and interaction estimates for this pruned plan graph, considering only those propositions and actions labeled 0, or 1.
5. For each (possibly conjunctive) goal $G \in \mathcal{G}$ estimate the $Cost(G|O)$ from the cost and interaction estimates in the pruned plan graph, again using equation (8). The pruned plan graph may discard propositions and/or actions in the cost-plan graph necessary to reach the goal. This constraint provides a way to discriminate possible goals. However, it may imply that 1) the real goal is discarded, 2) the calculated costs are less accurate. Therefore, computation of $Cost(G|O)$ has been developed under two strategies:
   (a) $Cost(G|O)$ is computed using the pruned plan graph.
   (b) $Cost(G|O)$ is computed after the pruned plan graph is expanded to quiescence again. This will reintroduce any pruned goals that are still possible given the observations.
6. For each goal $G \in \mathcal{G}$, compute $\Delta(G, O)$, and using equation (5) compute the probability $Pr(G|O)$ for the goal given the observations.

To illustrate this computation, consider again the actions A, B, and C from equation (1), and the plan graph shown in Figure 1. Suppose that A, B, and C have costs 2, 1, and 3 respectively, and that the possible goals are $g_1 = \{z, k\}$ and $g_2 = \{k, t\}$. Propagating cost and interaction information through the plan graph, we get $Cost(t) = 1$, $Cost(z) = 2$, $Cost(k) = 4$, and interaction values $I(k, t) = \infty$ and $I(k, z) = 0$ at level 3. Now consider the hypothesis $g_1 = \{z, k\}$; in order to compute $Cost(k \wedge z)$, we use the cost and interaction information propagated through the plan graph. In order to compute $Cost(k \wedge z|O)$, the cost and interaction information is propagated again only in those actions with status 1 and 0. In our example, these costs are:

$$Cost(k \wedge z) \approx Cost(z) + Cost(k) + I(k, z) = 2 + 4 + 0 = 6$$
$$Cost(k \wedge z|O) \approx Cost(k) + Cost(z) + I(k, z) = 4 + 2 - 1 = 5$$

Thus, the cost difference is:

$$\Delta(g_1, O) = Cost(g_1|O) - Cost(g_1) = 5 - 6 = -1$$

As a result:

$$Pr(O|g_1) = \frac{exp\{-(-1)\}}{1 + exp\{-(-1)\}} = 0.73$$

For the hypothesis $g_2 = \{k, t\}$, the plan graph dismisses this hypothesis as a solution because once the plan graph is pruned, propositions *t* is labeled as -1. Therefore:

$$Cost(k \wedge t|O) \approx Cost(k) + Cost(t) + I(k,t) = \infty$$

So:

$$Pr(O|g_2) = \frac{exp\{-\infty\}}{1 + exp\{-\infty\}} = \frac{0}{1} = 0$$

If we expand the pruned cost-plan graph until quiescence again, the solution is still the same because A and B are permanently mutually exclusive.

Assuming uniform priors, $Pr(G)$, after normalizing the probabilities, we get that $Pr(g_1|O) = 1$ and $Pr(g_2|O) = 0$, so the goal g_1 is certain in this simple example, given the observations of actions A and C.

The ISS Crew Activities Domain

The Environmental Control and Life Support System (ECLSS) is a critical subsystem aboard the ISS that provides oxygen and potable water, removes carbon dioxide, distributes cabin air between modules, maintains cabin temperature, humidity, and pressure levels, monitors and controls nitrogen, oxygen, carbon dioxide, methane, hydrogen, and water vapor levels, and filters particulates and microorganisms from the air. Maintaining the ECLSS equipment involves regular inspection, repair, and replacement of the components, which requires retrieving and using various tools, measurement instruments, and replacement parts from stowage in different modules aboard the space station.

The ISS is divided into modules where the ECLSS subsystems are located. An ECLSS subsystem, such as the Air Revitalization System (ARS), or Water Recovery System (WRS), may be available in more than one module. The ISS-CAD domain is concerned with the maintenance tasks that the astronaut must conduct for the ECLSS subsystems, and the astronaut's health. We developed our model of activities based on descriptions in (Bagdigian, 2008). The operators described in the domain involve moving around the modules, taking, replacing, repairing, or inspecting components, measuring air and humidity levels using different instruments, cleaning modules, exercising, eating, monitoring brain activity, body radiation, blood pressure, etc. Propositions in the domain model the connection among modules, the location of astronauts, subsystems, tools, and components, the availability of instruments and components, the state (on, off, enabled, disabled) of the electrical equipment, and the results of repair, replace, measure, etc, operators.

The ISS-CAD domain has a total of 47 operators, and 96 goals. The length of a plan solution depends on the number of tasks to be performed, and the number of astronauts involved. Figure 2 shows a simplified description of an ISS-CAD problem with three actions: *move*, *take a replacement*, and *replace a component*, and an astronaut (fe1). The goal is to replace a sensor in the ARS subsystem located in Destiny module. A valid plan for the problem in Figure 2 is the following sequence of actions:

```
(define (domain ISS-CAD)
 (:requirements :strips :typing :action-costs)
 (:types crew module system component tool)
 (:predicates (connected ?m1 ?m2 - module) (at ?c - crew ?m - module)
              (in ?c - component ?s - system ?m - module)
              (replacement-in ?t - component ?m - module)
              (taken-replacement ?t - component ?c - crew)
              (replaced ?cp - component ?s - system ?m - module ?c - crew))

 (:functions (total-cost))

 (:action move
  :parameters (?c - crew ?m1 ?m2 - module)
  :precondition (and (at ?c ?m1) (connected ?m1 ?m2))
  :effect (and (not (at ?c ?m1)) (at ?c ?m2) (increase (total-cost) 1)))

 (:action take-replacement
  :parameters (?t - component ?c - crew ?m - module)
  :precondition (and (at ?c ?m) (replacement-in ?t ?m))
  :effect (and (taken-replacement ?t ?c) (increase (total-cost) 20)))

 (:action replace-component
  :parameters (?o - component ?s - system ?m - module ?c - crew)
  :precondition (and (at ?c ?m) (in ?o ?s ?m) (taken-replacement ?o ?c))
  :effect (and (replaced ?o ?s ?m ?c) (not (taken-replacement ?o ?c))
               (increase (total-cost) 20)))
```

```
(define (problem ISS-CAD-1)
 (:domain ISS-CAD)
 (:objects fe1 - crew Harmony Destiny Unity - module
           ars - system sensor - component
 (:init (connected Harmony Destiny) (connected Destiny Harmony)
        (connected Destiny Unity) (connected Unity Destiny)
        (replacement-in sensor Unity) (in sensor ars Destiny)
        (at fe1 Harmony) (= (total-cost) 0))
 (:goal (replaced sensor ars Destiny fe1))
 (:metric minimize (total-cost)))
```

Figure 2: A fragment of a PDDL domain and problem description on the ISS Crew Activities Domain

$$\pi = \left\{ \begin{array}{l} \texttt{(move fe1 Harmony Destiny)}, \\ \texttt{(move fe1 Destiny Unity)}, \\ \texttt{(take-replacement sensor fe1 Unity)}, \\ \texttt{(move fe1 Unity Destiny)} \\ \texttt{(replace-component sensor ars Destiny fe1)} \end{array} \right\}$$

Experimental Results

This section shows an empirical evaluation firstly on classical planning domains used by Ramírez and Geffner, and secondly on the ISS Crew Activities Domain described above.

Results on Classical Planning Domains

We have conducted an experimental evaluation on planning domains used by Ramírez and Geffner: BlocksWord, Intrusion, Kitchen, and Logistics. Each domain has 15 problems. The hypotheses set and actual goal for each problem were chosen at random with the priors on the goal sets assumed to be uniform. For each problem in each of the domains, we ran the LAMA planner (Richter, Helmert, and Westphal, 2008) to solve the problem for the actual goal. The set of observed actions for each recognition problem was taken to be a subset of this plan solution, ranging from 100% of the actions, down to 10% of the actions. The experiments were conducted on an Intel Xeon CPU E5-1650 processor running at 3.20GHz with 32 GB of RAM.

Ramírez evaluates his technique using an optimal planner HSP^*_f (Haslum, 2008), and LAMA, a satisficing planner that is used in two modes: as a greedy planner that stops when it finds the first plan ($LAMA_G$), and as a planner that returns the best plan found in a given time limit (LAMA). For purposes of this test, Ramírez technique is also evaluated using

Table 1: Goal recognition with random observations

Domain		Blocks					Intrusion					Kitchen					Logistics				
Approach	%O	100	70	50	30	10	100	70	50	30	10	100	70	50	30	10	100	70	50	30	10
HSP^*_fu	T	558.08	419.45	379.81	357.94	357.94	447.41	281.12	151.37	3.58	3.55	480.51	171.08	49.62	37.93	37.92	36.26	32.46	14.08	7.04	7.04
	Q	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.93	0.66	0.8	0.8
	S	1.06	1.13	4.06	11.46	11.46	1	1	1.06	4.46	4.6	1	1	1.33	1.4	1.4	1	1.13	2.26	3.6	3.6
LAMA	T	1603.24	1522.96	1260.6	1077.62	1082.15	92.85	89.01	64.97	17.57	17.48	26.33	26.2	20.45	20.3	20.3	45.17	41.71	26.53	11.38	11.39
	Q	0.33	0.8	0.8	0.93	1	0.93	1	1	1	1	0.73	1	1	1	1	1	0.93	0.66	0.8	0.8
	S	1	1.13	3.86	10.4	10.93	0.93	1	1.06	4.46	4.6	0.73	1	1.33	1.4	1.4	1	1.13	2.26	3.6	3.6
	Q_{20}	1	1	1	1	1	0.93	1	1	1	1	0.73	1	1	1	1	1	1	0.93	1	1
	Q_{50}	1	1	1	1	1	0.93	1	1	1	1	0.73	1	1	1	1	1	1	1	1	1
	d	0.24	0.316	0.192	0.048	0.068	1.739	1.12	0.095	7×10^{-6}	7×10^{-6}	0.358	3×10^{-3}	0.016	0.012	0.012	0.019	0.673	1.051	0.956	0.956
$LAMA_G$	T	849.08	840.76	814.95	803.04	809.01	3.32	2.63	2.21	2.08	2.08	0.42	0.36	0.33	0.32	0.32	5.47	4.95	4.5	4.33	4.36
	Q	0.93	0.8	0.73	0.66	0.46	0	0.4	1	1	1	0.73	1	1	1	1	1	0.8	0.4	0.46	0.46
	S	1	1.2	3	6.2	4.2	0	0.4	1.13	4.46	4.6	0.73	1	1.33	1.4	1.4	1	1.2	1.8	2.93	2.93
	Q_{20}	0.93	1	1	1	1	0	0.4	1	1	1	0.73	1	1	1	1	1	1	0.66	0.6	0.6
	Q_{50}	0.93	1	1	1	1	0	0.4	1	1	1	0.73	1	1	1	1	1	1	0.93	0.86	0.86
	d	0.068	0.34	0.404	0.322	0.341	1.86	1.12	0.108	7×10^{-6}	7×10^{-6}	0.358	3×10^{-3}	0.016	0.012	0.012	0.019	0.675	1.179	1.063	1.063
h^s_a	T	1.04	0.89	0.81	0.81	0.8	0.7	0.46	0.39	0.35	0.36	0.066	0.049	0.044	0.045	0.046	0.61	0.55	0.51	0.51	0.51
	Q	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.13	0.06	0.13	0.06	0.06
	S	20.26	20.26	20.26	20.26	20.26	16.66	16.66	16.66	16.66	16.66	3	3	3	3	3	2.8	1.86	1.93	1	1
	Q_{20}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.93	0.93	0.86	0.86	0.86
	Q_{50}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.93	0.93	0.86	0.86	0.86
	d	0.092	0.087	0.062	0.035	0.035	0.123	0.124	0.119	0.075	0.073	0.443	0.436	0.284	0.226	0.226	0.123	0.117	0.099	0.074	0.074
FGR_I	T	1.007	1.029	1.265	1.452	1.452	0.885	0.493	0.212	0.209	0.21	0.261	0.195	0.140	0.135	0.135	0.886	1.015	1.19	1.266	1.271
	Q	1	0.66	0.4	0.13	0.13	1	1	0.93	0.93	0.93	1	1	1	1	1	1	0.86	0.53	0.6	0.6
	S	1.06	0.8	1.06	1.73	1.73	1	1	1	4.4	4.53	1	1	1.2	1.26	1.26	1	1.26	1.6	2.46	2.46
	Q_{20}	1	0.66	0.53	0.46	0.46	1	1	1	0.93	0.93	1	1	1	1	1	1	0.93	0.66	0.73	0.73
	Q_{50}	1	0.73	0.73	0.8	0.8	1	1	1	1	1	1	1	1	1	1	1	0.93	0.8	0.86	0.86
	d	0.149	0.962	0.751	0.336	0.336	3.2×10^{-4}	0.055	0.872	0.241	0.241	3.98×10^{-4}	0.036	0.107	0.192	0.192	0.264	0.786	1.163	0.718	0.718
FGR_IE	T	9.936	8.211	4.542	3.696	3.687	1.293	1.191	0.996	0.743	0.738	0.287	0.266	0.230	0.193	0.193	7.535	4.24	3.016	2.842	2.834
	Q	0.46	0.53	0.46	0.13	0.13	1	1	0.93	0.93	0.93	1	1	1	1	1	0.86	0.66	0.66	0.6	0.6
	S	1.26	1.13	1.86	1.73	1.73	1	1	1	4.4	4.53	1	1	1.2	1.26	1.26	1.13	1.4	1.8	2.8	2.8
	Q_{20}	0.46	0.73	0.66	0.4	0.4	1	1	1	0.93	0.93	1	1	1	1	1	0.86	0.8	0.8	0.73	0.73
	Q_{50}	0.46	0.8	0.86	0.8	0.8	1	1	1	1	1	1	1	1	1	1	0.93	1	0.86	0.86	0.86
	d	1.094	1.025	0.742	0.358	0.358	3.2×10^{-4}	0.055	0.872	0.241	0.241	3.98×10^{-4}	0.036	0.107	0.192	0.192	0.387	0.943	1.107	0.771	0.771
FGR^+	T	0.761	0.609	0.643	0.782	0.783	0.886	0.491	0.196	0.192	0.193	0.258	0.191	0.136	0.128	0.129	0.806	0.405	0.448	0.508	0.51
	Q	1	0.46	0.46	0.46	0.46	1	1	1	1	0.93	1	1	1	1	1	1	0.4	0.46	0.6	0.6
	S	1	0.53	1.2	2.06	2.06	1	1.13	1.13	4.06	3.93	1	1	1.33	1.4	1.4	1	1	2.13	2.93	2.93
	Q_{20}	1	0.46	0.6	0.6	0.6	1	1	1	1	0.93	1	1	1	1	1	1	0.46	0.6	0.66	0.66
	Q_{50}	1	0.46	0.6	0.73	0.73	1	1	1	1	1	1	1	1	1	1	1	0.53	0.73	0.8	0.8
	d	0.088	1.084	1.036	1.004	1.004	0.151	0.764	1.25	0.311	0.285	2×10^{-6}	0.038	5×10^{-6}	0.022	0.022	0.011	1.196	1.29	0.899	0.899

the heuristic estimator h^s_a, which was used in (Ramírez and Geffner, 2009). Like our technique, this requires no search since the cost is given by a heuristic estimator. We compare our goal recognition technique, FGR, against Ramírez's technique for those three planners and h^s_a, on the aforementioned domains. We present two variations of our technique, with and without extension of the plan graph after pruning:

- FGR_I: the propagation of cost information through the plan graph considers interaction information.
- FGR^+: the propagation of cost information through the plan graph does not consider interaction information.

Table 1 show the results. For each domain we show a table where for each planner, each column shows average performance over the 15 problems in each domain. The first row in the table represents the optimal solution where gHSP$^*_f$u computes $\Delta(G, O)$ using $Cost(G)$ instead of $Cost(G|\overline{O})$ (Equation (4)). This approach was allowed to run for an unlimited amount of time. The other rows represent different measures of quality and performance:

- T shows the average time in seconds taken for solving the problems.
- Q shows the fraction of times the actual goal was among the goals found to be the most likely.
- S shows the *spread*, that is, the average number of goals in $\mathcal{G}$ that were found to be the most likely.
- Q_{20} and Q_{50} show the fraction of times the actual goal is in the top 20% and top 50% of the ranked goals. Although Q might be less than 1 for some problem, Q_{20} or Q_{50} might be 1, indicating that the actual goal was *close* to the top.
- d is the mean distance between the probability scores produced for all the goal candidates, and the probability scores produced by gHSP$^*_f$u. More precisely, if the set of possible goals is $\{G_1, ..., G_n\}$, a method produces probabilities $\{e_1, ..., e_n\}$ for those goals, and gHSP$^*_f$u produces $\{p_1, ..., p_n\}$, d is defined as:

$$d = 1/n \sum_{i=1}^{n} |e_i - p_i| \quad (9)$$

The use of an optimal planner like HSP$^*_f$u is generally impractical for real-time goal recognition on any non-trivial domain. Surprisingly, LAMA does not perform any better on the harder domains, and the solution quality is uneven. $LAMA_G$ is much faster, but still no faster than HSP$^*_f$u in the blocks world domain. The h^s_a heuristic for approximating costs is quite fast, but the cost estimates are not very accurate, leading to poor quality results for goal recognition. The FGR_I heuristic is also quite fast, but yields much better results. On the harder domains, it is two orders of magnitude faster than HSP$^*_f$u, LAMA, or $LAMA_G$, and yields results of comparable quality to both LAMA and $LAMA_G$ for the higher observation percentages.

We have conducted a second test where the set of observed actions for each recognition problem was considered to be the prefix of the plan solution, ranging from 100% of the actions, down to 10% of the actions. (Priors on the goal sets are also assumed to be uniform.) The reason for this test is to model incremental goal recognition as actions

are observed. This produces essentially the same results as the previous test for all the different techniques. We have also tested the approaches in Table 1 on test problems where the observation sequences were generated using an optimal planner (rather than LAMA). This also makes no difference in the results.

Preliminary Results on the ISS domain

This section shows some preliminary results on the ISS Crew Activities Domain, which consists of 15 problems. The hypotheses set includes 96 goals that a single astronaut could potentially be doing. The actual goal for each problem was chosen at random with the priors on the goal sets assumed to be uniform. As in the previous section, for each problem we ran the LAMA planner (Richter, Helmert, and Westphal, 2008) to solve the problem for the actual goal. We have conducted a test where the observed sequence was considered to be the prefix of the plan solution, since we assume that the observed action sequence is provided by a free-flying robot monitoring the astronaut's activities. In this case, the observed sequence ranges from 100% of the actions, down to 50% of the actions. We did not include a test where the observed sequence is lower than 50% because the quality of the solutions were poor for all the approaches. The reason for this is that the first steps of the plan solution only correspond to actions where the astronaut moves among the different modules collecting tools and equipment, which is usually not enough information to discriminate among the possible goals.

For purposes of this test, Ramírez technique is evaluated using the heuristic estimator h^s_a and LAMA$_G$.

Table 2 shows the results when the observed actions sequence is a prefix of the plan solution. HSP$^*_f$u finds the actual goal with the highest probability ($Q = 1$), and the spread increases as the percentages of observed actions drops. However, the computation time is high enough that it makes the use of this planner impractical in this domain. LAMA$_G$ solves all the problems within 300 seconds and produces high quality solutions, which means that the actual goal is among the most likely goals for most of the problems. However, the spread increases considerably for lower percentages, which means that it does not discriminate well for lower numbers of observed actions. The h^s_a heuristic solves all the problems within 55 seconds, and finds the actual goal with the highest probability. However, it does not discriminated among the possible goals very well since the spread is very high for all the percentages. FGR$_I$ and FGR$^+$ solve all the problems within 113 and 46 seconds respectively, and find the actual goal with the highest probability. On top of that, the spread of the solution is very close to the one computed by the optimal solution, being slightly better for FGR$_I$. This means that these heuristic approaches discriminate among the possible goals quite well. FGR$^+$ is faster than FGR$_I$, and the quality of the solutions is only a bit lower. Interaction information helps to compute more accurate estimates of cost when subgoals interfere with each other. It appears that the reason FGR$^+$ works so well for these problems is that the degree of interference among subgoals is low.

Table 2: ISS-CAD Evaluation with Sequential Observation

Approach	%O	100	90	80	70	60	50
gHSP$^*_f$u	T	1134.33	1069.81	1164.44	1164.23	1165.05	1164.7
	Q	1	1	1	1	1	1
	S	1	7.06	16.33	16.33	16.33	16.33
LAMA$_G$	T	302.98	304.37	300.46	298.49	297.49	296.32
	Q	1	0.93	0.86	0.73	0.86	0.86
	S	1.13	5.13	14.4	32.86	66.93	71.66
	Q_{20}	1	1	1	0.73	0.86	0.86
	Q_{50}	1	1	1	0.93	0.86	0.86
	d	0.019444	0.018206	0.015936	0.012142	0.007288	0.006890
h^s_a	T	55.51	55.42	55.32	55.33	55.32	55.34
	Q	1	1	1	1	1	1
	S	85	85	85	85	85	85
	Q_{20}	1	1	1	1	1	1
	Q_{50}	1	1	1	1	1	1
	d	0.019230	0.014479	0.006027	0.006027	0.006027	0.006027
FGR$_I$	T	113.28	113.29	113.38	113.45	113.7	113.72
	Q	1	1	1	1	1	1
	S	1	7.53	17.46	17.46	17.46	17.46
	Q_{20}	1	1	1	1	1	1
	Q_{50}	1	1	1	1	1	1
	d	0	0.001544	0.004283	0.004283	0.004283	0.004283
FGR$^+$	T	46.3	46.33	46.31	46.39	46.25	46.3
	Q	1	1	1	1	1	1
	S	1	8.4	19.26	19.26	19.26	19.26
	Q_{20}	1	1	1	1	1	1
	Q_{50}	1	1	1	1	1	1
	d	0	0.001560	0.004134	0.004134	0.004134	0.004134

We have performed another test where the observed sequence was randomly taken to be a subset of this plan solution, ranging from 100% of the actions, down to 30% of the actions, even though it does not exactly fit the ISS-CAD problem where the free-flying robot is progressively monitoring the astronaut. Table 3 shows the results of this test, which produces essentially the same results as the previous test for all the different techniques. HSP$^*_f$u finds the actual goal with the highest probability ($Q = 1$), but the computation time is high. LAMA$_G$ solves all the problems within 300 seconds. It provides high quality solutions that degrade as the percentage of observed actions decreases. The h^s_a heuristic solves all the problems within 54–58 seconds, and finds high quality solutions. However, it does not discriminates among the possible goals very well. FGR$_I$ solves all the problems within 88–105 seconds. The quality of the solutions decreases as the percentage of observed actions drops, and, in general, is lower than the solutions provided by LAMA$_G$ (although the spread is lower for FGR$_I$). FGR$^+$ solves all the problems within 30-40 seconds. It find a high quality solutions, but with relatively high spread for low percentages of observed actions. In general, when the sequence of observed actions is randomly chosen, solutions are lower quality than when the observed sequence is sequential.

Conclusions and Future Work

This paper presented a heuristic technique for goal recognition. It computes cost estimates using a plan graph, and uses this information to infer probability estimates for the possible goals. This technique is fast enough to be used in real-time goal recognition problems. We also introduced a goal recognition domain for ISS Crew Activities, and showed that the presented approach provides high quality results when the sequence of observed actions is sequential. The assumption behind our model is that the observation sequence may be incomplete, but the observations are certain. It appears to be possible to extend our approach to deal with uncertain

Table 3: ISS-CAD Evaluation with Random Observations

Approach	$\%O$	100	70	50	30
$g\text{HSP}^*_f\text{u}$	T	1096.10	1174.01	1176.15	1238.74
	Q	1	1	1	1
	S	1	3.46	9.13	7.2
LAMA_G	T	325.19	299.09	297.18	297.01
	Q	1	0.93	0.8	0.8
	S	2.6	5.26	15.6	44.86
	Q_{20}	1	1	0.93	0.86
	Q_{50}	1	1	0.93	0.86
	d	0.019384	0.018876	0.018361	0.017186
h^s_a	T	57.97	56.04	54.33	54.02
	Q	0.86	0.86	0.86	0.86
	S	84.86	84.86	84.86	84.86
	Q_{20}	0.86	0.86	0.86	0.86
	Q_{50}	0.86	0.86	0.86	0.86
	d	0.019229	0.016722	0.012803	0.013572
FGR_I	T	87.05	92.55	100.60	105.89
	Q	1	0.86	0.66	0.53
	S	1.13	3.2	10.73	25.06
	Q_{20}	1	0.86	0.66	0.53
	Q_{50}	1	0.93	0.93	0.73
	d	0.018069	0.018233	0.017504	0.017006
FGR^+	T	29.54	32.29	36.95	41.73
	Q	1	0.93	1	0.8
	S	2.2	9.4	27.4	41.66
	Q_{20}	1	0.93	1	0.8
	Q_{50}	1	0.93	1	0.8
	d	0.018056	0.018139	0.017841	0.017516

observations, but this is the subject of future work.

For the ISS-CAD domain, we intend to test problems where each goal in the hypothesis set involves more than one astronaut, and the astronauts tasks interfere with each other. For these problems we expect that FGR_I will prove more accurate than FGR^+.

Acknowledgments

This work was funded by the JCCM project PEII-2014-015A, USRA, and NASA Ames Research Center.

References

Albrecht, D.; Zukerman, I.; Nicholson, A.; and Bud, A. 1997. Towards a bayesian model for keyhole plan recognition in large domains. *In Proceedings of UM*. Chia Laguna, Sardinia.

Bagdigian, B. 2008. Environmental control and life support system.

Bauer, M. 1998. Acquisition of abstract plan descriptions for plan recognition. *In Proceedings of AAAI*. Madison, WI, USA.

Brown, J. S.; Burton, R.; and Hausmann, C. 1977. Representing and using procedural bugs for educational purposes. *In Proceedings of ACM*. Amsterdam, The Netherlands.

Bryce, D., and Smith, D. E. 2006. Using interaction to compute better probability estimates in plan graphs. *In ICAPS-06 Workshop on Planning Under Uncertainty and Execution Control for Autonomous Systems*. Lake District, United Kingdom.

Bui, H. 2003. A general model for online probabilistic plan recognition. *In Proceedings of IJCAI*. Acapulco, Mexico.

Do, M., and Kambhampati, R. 2002. Planning graph-based heuristics for cost-sensitive temporal planning. *In Proceedings of AIPS*. Toulouse, France.

E-Martín, Y.; R-Moreno, M. D.; and Smith, D. E. 2015a. A fast goal recognition technique based on interaction estimates. *In Proceedings of IJCAI*. Buenos Aires, Argentina.

E-Martín, Y.; R-Moreno, M. D.; and Smith, D. E. 2015b. A heuristic estimator based on cost interaction. *In ICAPS-15 Workshop on Heuristics and Search for Domain-independent Planning*. Jerusalem, Israel.

Geib, C. W., and Goldman, R. P. 2009. A probabilistic plan recognition algorithm based on plan tree grammar. *In Artificial Intelligence* 84(1-2):57—112.

Haslum, P. 2008. Additive and reversed relaxed reachability heuristics revisited. *In Proceedings of the 2008 IPC*. Sydney, Australia.

Huber, M. J.; Durfee, E. H.; and Wellman, M. P. 1994. The automated mapping of plans for plan recognition. *In Proceedings of UAI*. Seattle, WA, USA.

Jigui, S., and Minghao, Y. 2007. Recognizing the agent's goals incrementally: planning graph as a basis. *In Frontiers of Computer Science in China* 1(1):26—36.

Kautz, H., and Allen, J. F. 1986. Generalized plan recognition. *In Proceedings of AAAI*. Philadelphia, PA, USA.

Kautz, H.; Arnstein, L.; Borriello, G.; Etzioni, O.; and Fox, D. 2002. An overview of the assisted cognition project. *In Proceedings of AAAI Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care*. Edmonton, Alberta, Canada.

Keyder, E., and Geffner, H. 2007. Heuristics for planning with action costs. In *Proceedings of the Spanish Artificial Intelligence Conference*.

Lesh, N., and Etzioni, O. 1995. A sound and fast recognizer. *In Proceedings of IJCAI*. Montreal, Canada.

Pollack, M. E.; Brownb, L.; Colbryc, D.; McCarthy, C. E.; Orosz, C.; Peintner, B.; Ramakrishnan, S.; and Tsamardinos, I. 2003. Autominder: an intelligent cognitive orthotic system for people with memory impairment. *In Robotics and Autonomous Systems* 44(3-4):273—282.

Ramírez, M., and Geffner, H. 2009. Plan recognition as planning. *In Proceedings of IJCAI*. Pasadena, CA, USA.

Ramírez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. *In Proceedings of AAAI*. Atlanta, GA, USA.

Ramírez, M. 2012. *Plan Recognition as Planning*. Ph.D. Dissertation, Universitat Pompeu Fabra. Barcelona, Spain.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks revisited. *In Proceedings of AAAI*. Chicago, IL, USA.

Weber, J. S., and Pollack, M. E. 2008. Evaluating user preferences for adaptive reminding. *In Proceedings of CHI EA*. Florence, Italy.

Wu, J.; Osuntogun, A.; Choudhury, T.; Philipose, M.; and Rehg, J. M. 2007. A scalable approach to activity recognition based on object use. *In Proceedings of ICCV*. Rio de Janeiro, Brazil.