Goal Recognition in Incomplete STRIPS Domain Models

Ramon Fraga Pereira and **Felipe Meneguzzi**
Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil
Postgraduate Programme in Computer Science, School of Informatics (FACIN)
ramon.pereira@acad.pucrs.br
felipe.meneguzzi@pucrs.br

Abstract

Recent approaches to goal recognition have progressively relaxed the assumptions about the amount and correctness of domain knowledge and available observations, yielding accurate and efficient algorithms. These approaches, however, assume completeness and correctness of the domain theory against which their algorithms match observations: this is too strong for most real-world domains. In this paper, we develop a goal recognition technique capable of recognizing goals using *incomplete* (and possibly incorrect) domain theories as well as noisy observations. Such recognition needs to cope with a much larger space of plan hypotheses consistent with observations. We show the efficiency and accuracy of our approach empirically against a large dataset of goal recognition problems with incomplete domains.

1 Introduction

Goal recognition is the problem of identifying the correct goal intended by an observed agent, given a sequence of observations as evidence of its behavior in an environment and a domain model describing how the observed agent generates such behavior. Approaches to solve this problem vary on the amount of domain knowledge used in the behavior, or plan generation, model employed by the observed agent (Sukthankar et al. 2014), as well as the level of observability and noise in the observations used as evidence (Sohrabi, Riabov, and Udrea 2016). Recent work has progressively relaxed the assumptions about the accuracy and amount of information available in observations required to recognize goals (E.-Martín, R.-Moreno, and Smith 2015; Sohrabi, Riabov, and Udrea 2016; Pereira and Meneguzzi 2016; Pereira, Oren, and Meneguzzi 2017a). However, regardless of the type of domain model formalism describing the observed agent's behavior, all recent approaches assume that the planning domain models are correct and complete, restricting its application to realistic scenarios in which the domain modeler either has an incomplete or incorrect model of the behavior under observation.

Specifically, real world domains have two potential sources of uncertainty:

- ambiguity in how actions performed by agents are realized; and
- ambiguity from how imperfect sensor data reports features of the world.

The former stems from an incomplete understanding of the action being modeled and requires a domain modeler to specify a number of alternate versions of the same action to cover the possibilities. For example, an action to turn on the gas burner in a cooker may or may not require the observed agent to press a spark button. The latter stems from imperfections in the way actions themselves may be interpreted from real-world noisy data, *e.g.*, if one uses machine learning algorithms to classify objects to be used as features (*e.g.*, logical facts) of the observations (Granada et al. 2017), certain features may not be recognizable reliably, so it is useful to model a domain with such feature as optional.

In this paper, we develop a goal recognition approach that can cope with incomplete planning domain models (Nguyen, Sreedharan, and Kambhampati 2017). This paper has four main contributions. First, we formalize goal recognition in incomplete domains (Section 2.3) by combining the standard formalization of Ramírez and Geffner (2009; 2010) for plan recognition and that of Nguyen, Sreedharan, and Kambhampati (2017). Second, we develop an algorithm, adapted from (Hoffmann, Porteous, and Sebastia 2004), that extracts *possible* landmarks in incomplete domain models (Section 3). Third, we develop a notion of *overlooked* landmarks that we can extract online as we process (*on the fly*) observations that we can use to match candidate goals to the multitude of models induced by incomplete domains. Fourth, we develop an algorithm to recognize goals very efficiently using a heuristic that accounts for the various types of landmark as evidence in the observations (Section 4).

We evaluate our approach using a new dataset constructed by modifying an existing one (Pereira, Oren, and Meneguzzi 2017a; Pereira and Meneguzzi 2017) of planning-based goal and plan recognition problems (Section 5). We have built this new dataset by removing just information from the complete domain model and annotating it with possible preconditions and effects, creating and adding an incomplete domain model. Using these modified datasets, we show that our approach is fast and accurate for recognizing goals in large and non-trivial incomplete domain models at most percentages of domain incompleteness.

2 Problem Formulation

2.1 STRIPS Domain Models

We assume that the agents being observed reason using planning domains described using the STRIPS (Fikes and Nilsson 1971) domain model $\mathcal{D} = \langle \mathcal{R}, \mathcal{O} \rangle$, where: $\mathcal{R}$ is a set of predicates with typed variables. Grounded predicates represent logical values according to some interpretation as facts, which are divided into two types: positive and negated facts, as well as constants for truth ($\top$) and falsehood ($\bot$); $\mathcal{O}$ is a set of operators $op = \langle pre(op), \mathit{eff}(op) \rangle$, where $\mathit{eff}(op)$ can be divided into positive effects $\mathit{eff}^+(op)$ (the add list) and negative effects $\mathit{eff}^-(op)$ (the delete list). An operator op with all variables bound is called an action and allows state change. An action a instantiated from an operator op is applicable to a state S iff $S \models pre(a)$ and results in a new state S' such that $S' := (S \cup \mathit{eff}^+(a))/\mathit{eff}^-(a)$.

A planning problem within $\mathcal{D}$ and a set of typed objects Z is defined as $\mathcal{P} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, G \rangle$, where: $\mathcal{F}$ is a set of facts (instantiated predicates from $\mathcal{R}$ and Z); $\mathcal{A}$ is a set of instantiated actions from $\mathcal{O}$ and Z; $\mathcal{I}$ is the initial state ($\mathcal{I} \subseteq \mathcal{F}$); and G is a partially specified goal state, which represents a desired state to be achieved. A plan π for a planning problem $\mathcal{P}$ is a sequence of actions $\langle a_1, a_2, ..., a_n \rangle$ that modifies the initial state $\mathcal{I}$ into a state $S \models G$ in which the goal state G holds by the successive execution of actions in a plan π.

2.2 Incomplete STRIPS Domain Models

The agent reasoning about the observations and trying to infer a goal has information described using the formalism of incomplete domain models from Nguyen, Sreedharan, and Kambhampati (2017), defined as $\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$. Here, $\widetilde{\mathcal{O}}$ contains the definition of incomplete operators comprised of a six-tuple $\widetilde{op} = \langle pre(\widetilde{op}), \widetilde{pre}(\widetilde{op}), \mathit{eff}^+(\widetilde{op}), \mathit{eff}^-(\widetilde{op}), \widetilde{\mathit{eff}}^+(\widetilde{op}), \widetilde{\mathit{eff}}^-(\widetilde{op}) \rangle$, where: $pre(\widetilde{op})$ and $\mathit{eff}(\widetilde{op})$ have the same semantics as in the STRIPS domain models; and *possible* preconditions $\widetilde{pre}(\widetilde{op}) \subseteq \mathcal{R}$ that *might* be required as preconditions, as well as $\widetilde{\mathit{eff}}^+(\widetilde{op}) \subseteq \mathcal{R}$ and $\widetilde{\mathit{eff}}^-(\widetilde{op}) \subseteq \mathcal{R}$ that *might* be generated as *possible* effects either as add or delete effects. An incomplete domain $\widetilde{\mathcal{D}}$ has a *completion set* $\langle\langle \widetilde{\mathcal{D}} \rangle\rangle$ comprising all possible domain models derivable from the incomplete one. There are 2^K possible such models where $K = \sum_{\widetilde{op} \in \widetilde{\mathcal{O}}} (|\widetilde{pre}(\widetilde{op})| + |\widetilde{\mathit{eff}}^+(\widetilde{op})| + |\widetilde{\mathit{eff}}^-(\widetilde{op})|)$, and a single (unknown) ground-truth model $\mathcal{D}^*$ that actually drives the observed state. An incomplete planning problem derived from an incomplete domain $\widetilde{\mathcal{D}}$ and a set of typed objects Z is defined as $\widetilde{\mathcal{P}} = \langle \mathcal{F}, \widetilde{\mathcal{A}}, \mathcal{I}, G \rangle$, where: $\mathcal{F}$ is the set of facts (instantiated predicates from Z), $\widetilde{\mathcal{A}}$ is the set of incomplete instantiated actions from $\widetilde{\mathcal{O}}$ with objects from Z, $\mathcal{I} \subseteq \mathcal{F}$ is the initial state, and $G \subseteq \mathcal{F}$ is the goal state.

Like most planning approaches in incomplete domains (Weber and Bryce 2011; Nguyen and Kambhampati 2014; Nguyen, Sreedharan, and Kambhampati 2017), we reason about possible plans with incomplete actions by assuming that they succeed under the *most optimistic* conditions, namely that: possible preconditions do not need to be satisfied in a state; possible add effects are always assumed to occur in the resulting state; and delete effects are ignored in the resulting state. Formally, an incomplete action $\widetilde{a}$ instantiated from an incomplete operator $\widetilde{op}$ is applicable to a state S iff $S \models pre(\widetilde{a})$ and results in a new state S' such that $S' := (S \cup \mathit{eff}^+(a) \cup \widetilde{\mathit{eff}}^+(\widetilde{a}))/\mathit{eff}^-(a)$. Thus, a valid plan π that achieves a goal G from $\mathcal{I}$ in an incomplete planning problem $\widetilde{\mathcal{P}}$ is a sequence of actions that corresponds to an *optimistic* sequence of states. Example 1 from Weber and Bryce (2011) illustrates an abstract incomplete domain and a valid plan for it.

Example 1 *Consider the following incomplete planning problem $\widetilde{\mathcal{P}}$, where:*

- $\mathcal{F} = \{p, q, r, g\}$*;*
- $\widetilde{\mathcal{A}} = \{\widetilde{a}, \widetilde{b}, \widetilde{c}\}$*, where:*
  - $pre(\widetilde{a}) = \{p, q\}, \widetilde{pre}(\widetilde{a}) = \{r\}, \widetilde{\mathit{eff}}^+(\widetilde{a}) = \{r\}, \widetilde{\mathit{eff}}^-(\widetilde{a}) = \{p\}$
  - $pre(\widetilde{b}) = \{p\}, \mathit{eff}^+(\widetilde{b}) = \{r\}, \mathit{eff}^-(\widetilde{b}) = \{p\}, \widetilde{\mathit{eff}}^-(\widetilde{b}) = \{q\}$
  - $pre(\widetilde{c}) = \{r\}, \widetilde{pre}(\widetilde{c}) = \{q\}, \mathit{eff}^+(\widetilde{c}) = \{g\}$
- $\mathcal{I} = \{p, q\}$*; and*
- $G = \{g\}$.

The $[\widetilde{a}, \widetilde{b}, \widetilde{c}]$ sequence of actions is a valid plan to achieve goal state $\{g\}$ from the initial state $\{p, q\}$. It corresponds to the optimistic state sequence: $s_0 = \{p, q\}, s_1 = \{p, q, r\}, s_2 = \{q, r\}, s_3 = \{q, r, g\}$. The number of completions for this example is $|\langle\langle \widetilde{\mathcal{D}} \rangle\rangle| = 2^5$ (2 possible preconditions and 3 possible effects, i.e., 1 possible add effect and 2 possible delete effects).

2.3 Goal Recognition in Incomplete Domains

Goal recognition is the task of recognizing and anticipating agents' goals by observing their interactions in an environment. Whereas most planning-based goal recognition approaches assume that a complete domain is available (Ramírez and Geffner 2009; 2010; Keren, Gal, and Karpas 2014; E.-Martín, R.-Moreno, and Smith 2015; Sohrabi, Riabov, and Udrea 2016; Pereira, Oren, and Meneguzzi 2017a), we assume that the observer has an incomplete domain model while the observed agent is planning and acting with a complete domain model. To account for such uncertainty, the model available to the observer contains possible preconditions and effects, much like the incomplete domain models from previous planning approaches (Weber and Bryce 2011; Nguyen, Sreedharan, and Kambhampati 2017). We formalize goal recognition over incomplete domain models in Definition 1.

Definition 1 (Goal Recognition Problem) *A goal recognition problem with an incomplete domain model is a quintuple $\widetilde{\mathcal{T}} = \langle \widetilde{\mathcal{D}}, Z, \mathcal{I}, \mathcal{G}, Obs \rangle$, where:*

- $\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$ *is an incomplete domain model (with possible preconditions and effects). Z is the set of typed objects in the environment, in which $\mathcal{F}$ is the set instantiated predicates (facts) from Z, and $\widetilde{\mathcal{A}}$ is the set of incomplete instantiated actions from $\widetilde{\mathcal{O}}$ with objects from Z;*

- $\mathcal{I} \in \mathcal{F}$ *an initial state;*
- $\mathcal{G}$ *is the set of possible goals, which include a correct hidden goal G (i.e.,* $G \in \mathcal{G}$*); and*
- $Obs = \langle o_1, o_2, ..., o_n \rangle$ *is an observation sequence of executed actions, with each observation* $o_i \in \widetilde{\mathcal{A}}$*. Obs corresponds to the sequence of actions (i.e., a plan) to solve a problem in a complete domain in* $\langle\langle\widetilde{\mathcal{D}}\rangle\rangle$.

A solution for a goal recognition problem in incomplete domain models $\widetilde{\mathcal{T}}$ is the correct hidden goal $G \in \mathcal{G}$ that the observation sequence Obs of a plan execution achieves. As most goal recognition approaches, observations consist of the action signatures of the underlying plan[1], more specifically, we observe incomplete actions with possible precondition and effects, in which some of the preconditions might be required and some effects might change the environment. While a full (or complete) observation sequence contains all of the action signatures of the plan executed by the observed agent, an incomplete observation sequence contains only a sub-sequence of actions of a plan and thus misses some of the actions actually executed in the environment.

3 Extracting Landmarks in Incomplete STRIPS Domain Models

In planning, landmarks are facts (or actions) that must be achieved (or executed) at some point along all valid plans to achieve a goal from an initial state (Hoffmann, Porteous, and Sebastia 2004). Landmarks are often used to build heuristics (Richter, Helmert, and Westphal 2008) for planning algorithms (Richter and Westphal 2010). However, in the planning literature, landmark-based heuristics extract landmarks from complete and correct domain models. In this paper, we extend the landmark extraction algorithm proposed by Hoffmann *et al.* in (2004) to extract *definite* and *possible* landmarks in incomplete STRIPS domain models.

The landmark extraction algorithm proposed by Hoffmann *et al.* in (2004) uses a Relaxed Planning Graph (RPG), which is a leveled graph that ignores the delete-list effects of all actions, thus containing no mutex relations (Hoffmann and Nebel 2001). Once the RPG is built, this algorithm extracts a set of *landmark candidates* by back-chaining from the RPG level in which all facts of the goal state G are possible, and, for each fact g in G, it checks which facts must be true until the first level of the RPG. For example, if fact B is a landmark and all actions that achieve B share A as precondition, then A is a landmark candidate. To confirm that a landmark candidate is indeed a necessary condition, and thus a landmark, the algorithm builds a new RPG removing actions that achieve the landmark candidate and checks the solvability over this modified problem. If the modified problem is unsolvable, then the landmark candidate is a necessary landmark. This means that the actions that achieve the landmark candidate are necessary to solve the original planning problem. Deciding the solvability of a relaxed planning problem using an RPG structure can be done in polynomial time (Blum and Furst 1997).

[1]Our approach is not limited to using just actions as observations and can also deal with logical facts as observations.

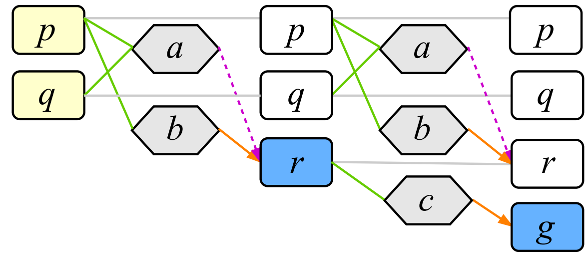

Figure 1: ORPG for Example 1. Green arrows represent preconditions, Orange arrows represent add effects, and Purple dashed arrows represent possible add effects. Light-Blue boxes represent the set of *definite* landmarks and Light yellow boxes represent the set of *possible* landmarks. Hexagons represent actions.

We adapt the extraction algorithm from (Hoffmann, Porteous, and Sebastia 2004) to extract landmarks from incomplete domain models by building an Optimistic Relaxed Planning Graph (ORPG) instead of the original RPG. An ORPG is leveled graph that deals with incomplete domain models by assuming the most *optimistic* conditions. Thus, besides ignoring the delete-effects of all actions, this graph also ignores possible preconditions and possible delete-effects, whereas we use all possible add effects. Replacing an RPG for an ORPG allows us to extract *definite* and *possible* landmarks, formalized in Definitions 2 and 3, respectively.

Definition 2 (Definite Landmark) *A definite landmark* $L_{Definite}$ *is a fact landmark extracted from a known add effect (*$eff^+(a)$*) of an achiever*[2] a *(action) in the ORPG.*

Definition 3 (Possible Landmark) *A possible landmark* $L_{Possible}$ *is a fact landmark extracted from a possible add effect (*$\widetilde{eff}^+(a)$*) of an achiever* a *(action) in the ORPG.*

Figure 1 shows an ORPG for Example 1. For this example, the set of *definite* and *possible* landmarks is $\{p, q, r, g\}$. The set of *definite* landmarks is $\{r, g\}$ (Light-Blue in Figure 1), and the set of *possible* landmarks is $\{p, q\}$ (Light-Yellow in Figure 1). The classical landmark extraction algorithm from Hoffmann, Porteous, and Sebastia (without the most *optimistic* conditions), returns $\{p, r, g\}$ as landmarks. The classical landmark extraction algorithm does not extract q as a fact landmark because it does not assume the most *optimistic* condition that possible add effects always occur, therefore, the a action was not considered as a possible achiever (action). Thus, by using an ORPG instead an RPG we can extract not only *definite* landmarks but also *possible* landmarks. Although we use this modification (ORPG instead RPG) to recognize goals in incomplete domain models, these landmarks can easily be used to build heuristics for planning in incomplete domains.

[2]An achiever is an action at the level before a candidate landmark in the ORPG (or RPG) that can be used to achieve this candidate landmark.

4 Heuristic Goal Recognition For Incomplete Domain Models

Key to our goal recognition approach is observing the evidence of achieved landmarks during observations to recognize which goal is more consistent with the observations. To do so, our approach combines the concepts of *definite* and *possible* with that of *overlooked* landmarks. An overlooked landmark is an actual landmark, *i.e.*, a necessary fact for all valid plans towards a goal, that was not detected by approximate landmark extraction algorithms. Since we are dealing with incomplete domain models, and it is possible that they have few (or no) *definite* and/or *possible* landmarks, we extract *overlooked* landmarks from the evidence in the observations as we process them in order to enhance the set of landmarks useable by our heuristic. This *on the fly* landmark extraction checks if the facts in the known preconditions and known and possible add effects are not *definite* and *possible* landmarks, and if they are not, we check if these facts are *overlooked* landmarks. To do so, we use the ISLANDMARK function that builds a new ORPG removing actions that achieve a fact (i.e. a potentially *overlooked* landmark) and checks the solvability of this modified problem. If the modified problem is indeed unsolvable, then this fact is an *overlooked* landmark. We check every candidate goal using this function to extract additional landmarks.

Combining the concept of *definite*, *possible*, and *overlooked* landmarks, we develop a goal recognition heuristic for recognizing goals in incomplete domain models. Our heuristic estimates the correct goal in the set of candidate goals by calculating the ratio between achieved *definite* ($\mathcal{AL}_G$), *possible* ($\widetilde{\mathcal{AL}}_G$), and *overlooked* ($\mathcal{ANL}_G$) landmarks and the amount of *definite* ($\mathcal{L}_G$), *possible* ($\widetilde{\mathcal{L}}_G$), and *overlooked* ($\mathcal{NL}_G$) landmarks. This estimate, computed using Equation 1, represents the percentage of achieved landmarks for a candidate goal from observations.

$$h_{\widetilde{GR}}(G) = \left(\frac{\mathcal{AL}_G + \widetilde{\mathcal{AL}}_G + \mathcal{ANL}_G}{\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G} \right) \qquad (1)$$

Algorithm 1 implements our approach to recognize goals in incomplete domain models, using the $h_{\widetilde{GR}}$ heuristic and the extended landmark extraction process from Section 3. This algorithm takes as input a goal recognition problem $\widetilde{\mathcal{T}}$, which contains an incomplete domain model $\widetilde{\mathcal{D}}$, a set of typed objects Z, an initial state $\mathcal{I}$, a set of candidate goals $\mathcal{G}$, and a sequence of observations Obs with incomplete actions. For every candidate goal $G \in \mathcal{G}$ this algorithm extracts *definite* and *possible* landmarks for G from $\mathcal{I}$ in Line 5. The algorithm then iterates over the observations Obs by checking observed facts from preconditions ($pre(o)$) and effects ($\mathit{eff}^+(o)$ and $\widetilde{\mathit{eff}}^+(o)$) to store in OF_o, and verifies which *definite* and *possible* landmarks have been achieved from the set of observed facts OF_o (Lines 9 and 10). In Line 12, our algorithm uses the ISLANDMARK to extract possibly overlooked landmarks from the observations storing the set of *overlooked* landmarks in variables $\mathcal{NL}_G$ and $\mathcal{ANL}_G$ in Lines 13 and 14. Note that since the overlooked landmarks are extracted from the observations, they are all also assumed to be achieved. Finally, the algorithm uses heuristic $h_{\widetilde{GR}}$ to compute a score (estimated value, a percentage of achieved landmarks) for G in Line 18, returning the candidate goals with the highest percentage of achieved landmarks in Line 20.

Algorithm 1 Recognize Goals in Incomplete Domain Models.

Input: $\widetilde{\mathcal{T}} = \langle \widetilde{\mathcal{D}}, Z, \mathcal{I}, \mathcal{G}, Obs \rangle$, *in which* $\widetilde{\mathcal{D}}$ *is an incomplete domain model*, Z *is a set of typed objects*, $\mathcal{I}$ *is the initial state*, $\mathcal{G}$ *is the set of candidate goals, and* Obs *represents observations with incomplete action sequences.*
Output: *Recognized goal(s).*

1: **function** RECOGNIZE($\widetilde{\mathcal{D}}, Z, \mathcal{I}, \mathcal{G}, O$)
2: $\quad \mathcal{H}_\mathcal{G} := \langle\rangle$ ▷ *Map of goals to heuristic values.*
3: $\quad$ **for each** goal G in $\mathcal{G}$ **do**
4: $\quad\quad \mathcal{NL}_G := \langle\rangle$ ▷ *Set of overlooked landmarks for G.*
5: $\quad\quad \mathcal{L}_G, \widetilde{\mathcal{L}}_G :=$ EXTRACTLANDMARKS($\widetilde{\mathcal{D}}, Z, \mathcal{I}, G$)
6: $\quad\quad \mathcal{AL}_G, \widetilde{\mathcal{AL}}_G, \mathcal{ANL}_G := \langle\rangle$ ▷ *Achieved landmarks.*
7: $\quad\quad$ **for each** observed action o in Obs **do**
8: $\quad\quad\quad OF_o :=$ all facts in $pre(o) \cup \mathit{eff}^+(o) \cup \widetilde{\mathit{eff}}^+(o)$ ▷ *Observed facts in the incomplete model of action o.*
9: $\quad\quad\quad \mathcal{AL}_G :=$ all landmarks l in $\mathcal{L}_G$ s.t $l \in OF_o$
10: $\quad\quad\quad \widetilde{\mathcal{AL}}_G :=$ all landmarks $\widetilde{l}$ in $\widetilde{\mathcal{L}}_G$ s.t $\widetilde{l} \in OF_o$
11: $\quad\quad\quad$ **for each** fact f in OF_o s.t $f \notin (\mathcal{L}_G \cup \widetilde{\mathcal{L}}_G)$ **do**
12: $\quad\quad\quad\quad$ **if** ISLANDMARK($f, \mathcal{I}, G$) **then**
13: $\quad\quad\quad\quad\quad \mathcal{NL}_G := \mathcal{NL}_G \cup f$
14: $\quad\quad\quad\quad\quad \mathcal{ANL}_G := \mathcal{ANL}_G \cup f$
15: $\quad\quad\quad\quad$ **end if**
16: $\quad\quad\quad$ **end for**
17: $\quad\quad$ **end for**
18: $\quad\quad \mathcal{H}_\mathcal{G} := \mathcal{H}_\mathcal{G} \cup \langle G, \left(\frac{\mathcal{AL}_G + \widetilde{\mathcal{AL}}_G + \mathcal{ANL}_G}{\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G} \right) \rangle$ ▷ *Use* $h_{\widetilde{GR}}$ *heuristic for calculating the estimated value for G.*
19: $\quad$ **end for**
20: $\quad$ **return** all G s.t $\langle G, v \rangle \in \mathcal{H}_\mathcal{G}$ and $v \geq (\max_{v_i} \langle G', v_i \rangle \in \mathcal{H}_\mathcal{G})$
21: **end function**

5 Experiments and Evaluation

We now describe the experiments carried out to evaluate our goal recognition approach in incomplete domain models, describing how we have built and modified a dataset from literature, as well as describing the metrics we used for evaluation. We conclude this section by discussing our results over this modified dataset.

5.1 Dataset and Setup

For experiments, we used openly available goal and plan recognition datasets (Pereira and Meneguzzi 2017)[3], which contain thousands of recognition problems. These datasets contain large and non-trivial planning problems (with optimal and sub-optimal plans as observations) for 15 planning domains, including domains and problems from datasets that were developed by Ramírez and Geffner (2009; 2010)[4]. The planning domains used in these datasets

[3] https://doi.org/10.5281/zenodo.825878

[4] https://sites.google.com/site/prasplanning

are: BLOCKS-WORLD, CAMPUS, DEPOTS, DRIVER-LOG, DOCK-WORKER-ROBOTS (DWR), IPC-GRID, FERRY, INTRUSION-DETECTION (INTRUSION), KITCHEN, LOGISTICS, MICONIC, ROVERS, SATELLITE, SOKOBAN, and ZENO-TRAVEL (ZENO). All planning domains in these datasets are encoded using the STRIPS fragment of PDDL (McDermott et al. 1998). Each goal/plan recognition problem in these datasets contains a (complete) domain definition, an initial state, a set of candidate goals, a correct hidden goal in the set of candidate goals, and an observation sequence. An observation sequence contains actions that represent an optimal plan or sub-optimal plan that achieves a correct hidden goal, and this observation sequence can be full or partial. A full observation sequence represents the whole plan that achieves the hidden goal, *i.e.*, 100% of the actions having been observed. A partial observation sequence represents a plan for the hidden goal, varying in 10%, 30%, 50%, or 70% of its actions having been observed. To evaluate our goal recognition approach in incomplete domain models, we modify the (complete) domain models of these datasets by adding annotated possible preconditions and effects (add and delete lists). Thus, the only modification to the original datasets is the generation of new, incomplete, domain models for each recognition problem, varying the percentage of incompleteness (possible preconditions and effects) in these domains.

We vary the percentage of incompleteness of a domain from 20 to 80 percent. For example, consider that a complete domain has, for all its actions, a total of 10 preconditions, 10 add effects, and 10 delete effects. A derived model with 20% of incompleteness needs to have 2 possible preconditions (8 known preconditions), 2 possible add effects (8 known add effects), and 2 possible delete effects (8 known delete effects), and so on for other percentages of incompleteness. Like (Nguyen and Kambhampati 2014; Nguyen, Sreedharan, and Kambhampati 2017), we used the following conditions to generate incomplete domain models with possible preconditions, possible add effects, and possible delete effects: (1) we randomly move a percentage of known preconditions and effects into possible lists of preconditions and effects; (2) we randomly add possible preconditions from delete effects that are not preconditions of a corresponding operator; and (3) we randomly add into possible lists (of preconditions, add effects, or delete effects) predicates whose parameters fit into the operator signatures and are not precondition or effects of the operator. By following all these three conditions, we generate three different incomplete domain models from a complete domain model, since the lists of preconditions and effects are generated randomly. Thus, each percentage of domain incompleteness has three domain models with different possible lists of preconditions and effects.

Using these modified recognition problems, we ran all experiments using a single core of a 12 core Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz with 16GB of RAM. The JavaVM ran experiments with a 2GB memory limit and a 2-minute time limit.

5.2 Evaluation Metrics

We evaluated our approach using the following metrics: the recognition time in seconds (*Time*); *Accuracy*[5], representing the fraction of time steps in which the correct goal was among the goals found to be most likely, *i.e.*, how good our approach is for recognizing the correct goal G in $\mathcal{G}$ over time; and *Spread in* $\mathcal{G}$ represents the average number of returned goals.

As each percentage of domain incompleteness has three different incomplete domain models, the percentage columns (20%, 40%, 60%, and 80%) in Table 1 report averages for recognition time (*Time*), *Accuracy*, and *Spread in* $\mathcal{G}$ by taking into account the results of the three incomplete domain models. We also show in Table 1 the total number of goal recognition problems for each domain name (first column). Each row in the table expresses averages for the number of candidate goals $|\mathcal{G}|$; the percentage of the plan that was actually observed (% Obs); the average number of observations per problem $|O|$; and $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$, representing the number of possible complete domain models (completion set) for each percentage of domain incompleteness.

We adapt the *Receiver Operating Characteristic* (ROC) curve metric to highlight the trade-off between true positive and false positive results. A ROC curve is often used to compare not only true positive predictions, but also to compare the false positive predictions of the experimented approaches. Here, each prediction result of our goal recognition approach represents one point in the space, and thus, instead of a curve, our graphs show the spread of our results over ROC space. In the ROC space, the diagonal line represents a random guess to recognize a goal from observations. This diagonal line divides the ROC space in such a way that points above the diagonal represent good classification results (better than random guess), whereas points below the line represent poor results (worse than random guess). The best possible (perfect) prediction for recognizing goals are points in the upper left corner (*i.e.*, coordinate x = 0 and y = 100) in ROC space.

5.3 Results

Table 1 shows the experimental results of our goal recognition approach in incomplete domain models. Apart from IPC-GRID and SOKOBAN that took substantial recognition time, for most planning domains our approach yields high accuracy at low recognition time. SOKOBAN exceeds the time limit of 2 minutes for most goal recognition problems because this dataset contains large problems with a huge number of objects, leading to an even larger number of instantiated predicates and actions. For example, as domain incompleteness increases (i.e. the ratio of possible to definite preconditions and effects), the number of possible actions (moving between cells and pushing boxes) in a grid with 9x9 cells and 5 boxes increases substantially because as there are very few definite preconditions for several possible

[5]This metric is analogous to the *Quality* metric (also denoted as Q), used for most planning-based goal recognition approaches (Ramírez and Geffner 2009; 2010; E.-Martín, R.-Moreno, and Smith 2015; Sohrabi, Riabov, and Udrea 2016).

Incompleteness of $\widetilde{\mathcal{D}}$ (%)				20%				40%				60%				80%			
#	$\|\mathcal{G}\|$	*Obs* (%)	$\|O\|$	$\|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle\|$	*Time*	*Accuracy*	*Spread in* $\mathcal{G}$	$\|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle\|$	*Time*	*Accuracy*	*Spread in* $\mathcal{G}$	$\|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle\|$	*Time*	*Accuracy*	*Spread in* $\mathcal{G}$	$\|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle\|$	*Time*	*Accuracy*	*Spread in* $\mathcal{G}$
		10	1.82		0.045	33.7%	1.45		0.047	29.3%	1.74		0.048	41.1%	2.46		0.052	45.0%	2.89
		30	4.93		0.047	44.9%	1.33		0.048	50.9%	1.67		0.050	60.8%	2.04		0.054	56.4%	2.11
BLOCKS (12192)	20.3	50	7.66	42.22	0.051	52.6%	1.33	1782.89	0.052	62.1%	1.54	75281.09	0.050	69.8%	1.69	3178688.02	0.058	65.0%	1.71
		70	11.1		0.053	72.4%	1.32		0.052	76.6%	1.44		0.055	85.8%	1.67		0.059	82.2%	1.76
		100	14.5		0.057	94.6%	1.31		0.060	96.4%	1.38		0.061	99.6%	1.61		0.063	99.3%	1.66
		10	1.0		0.004	75.6%	1.22		0.003	84.4%	1.24		0.005	57.8%	1.29		0.005	80.0%	1.49
		30	2.0		0.006	77.8%	1.11		0.005	80.0%	1.22		0.007	62.2%	1.24		0.007	84.4%	1.47
CAMPUS (900)	2.0	50	3.0	7131.55	0.007	91.1%	1.09	50859008.46	0.008	91.1%	1.13	3.63E+11	0.009	64.4%	1.16	2.59E+15	0.008	77.8%	1.36
		70	4.47		0.009	97.8%	1.02		0.009	95.6%	1.09		0.010	77.8%	1.11		0.011	82.2%	1.29
		100	5.4		0.008	100.0%	1.02		0.007	97.8%	1.09		0.011	86.7%	1.09		0.012	86.7%	1.24
		10	3.13		0.079	30.2%	1.38		0.071	37.3%	1.87		0.075	39.3%	2.01		0.067	52.8%	2.68
		30	8.61		0.079	40.5%	1.33		0.067	57.5%	1.62		0.083	48.0%	1.47		0.075	67.1%	2.37
DEPOTS (4368)	8.9	50	14.04	168.89	0.079	61.5%	1.32	28526.20	0.075	75.8%	1.38	4817990.10	0.087	63.9%	1.35	813744135.40	0.095	86.5%	1.85
		70	19.71		0.079	78.6%	1.28		0.079	90.5%	1.28		0.099	79.8%	1.33		0.107	92.1%	1.71
		100	27.43		0.083	89.3%	1.08		0.083	100.0%	1.11		0.107	90.5%	1.17		0.119	98.8%	1.46
		10	2.61		0.282	35.3%	1.45		0.310	36.1%	1.90		0.346	36.1%	2.01		0.377	53.6%	3.13
		30	6.96		0.307	45.2%	1.37		0.325	47.2%	1.86		0.363	43.7%	1.92		0.406	60.7%	2.01
DRIVER (4368)	7.1	50	11.18	48.50	0.329	56.0%	1.32	2352.53	0.344	63.5%	1.73	114104.80	0.392	61.1%	1.84	5534417.30	0.424	79.0%	1.95
		70	15.64		0.331	69.0%	1.26		0.339	75.4%	1.62		0.409	72.2%	1.76		0.452	90.1%	1.89
		100	21.71		0.338	78.6%	1.21		0.341	82.1%	1.46		0.428	79.8%	1.65		0.464	95.2%	1.82
		10	5.71		0.221	19.0%	1.31		0.365	44.0%	2.42		0.413	48.8%	3.65		0.468	63.9%	4.15
		30	16.0		0.237	37.3%	1.26		0.392	66.3%	1.88		0.449	72.6%	2.97		0.603	77.0%	3.34
DWR (4368)	7.3	50	26.21	512.00	0.274	50.0%	1.27	262144.00	0.348	75.0%	1.69	134217728.00	0.480	80.2%	2.15	6.88E+10	0.718	84.1%	2.58
		70	36.86		0.278	67.1%	1.11		0.484	91.7%	1.36		0.561	91.7%	1.91		0.802	92.9%	2.25
		100	51.89		0.286	85.7%	1.01		0.507	98.8%	1.23		0.624	100.0%	1.74		0.917	100.0%	2.12
		10	2.93		0.071	73.8%	1.59		0.083	62.7%	1.71		0.091	67.9%	2.02		0.098	80.6%	3.10
		30	7.68		0.075	94.0%	1.43		0.102	87.7%	1.62		0.110	85.7%	1.78		0.113	88.9%	1.97
FERRY (4368)	7.6	50	12.36	8.00	0.087	98.8%	1.32	64.00	0.103	96.0%	1.44	512.00	0.126	95.6%	1.63	4096.00	0.125	92.5%	1.85
		70	17.36		0.095	100.0%	1.24		0.117	100.0%	1.37		0.138	100.0%	1.58		0.144	97.6%	1.69
		100	24.21		0.112	100.0%	1.0		0.129	100.0%	1.0		0.141	100.0%	1.0		0.153	100.0%	1.11
		10	1.92		0.022	46.7%	2.07		0.034	34.6%	1.92		0.043	25.7%	2.39		0.049	16.5%	1.82
		30	4.48		0.031	86.3%	1.31		0.039	65.7%	1.45		0.046	34.9%	2.68		0.050	23.8%	1.97
INTRUSION (5580)	16.7	50	6.7	16.00	0.037	93.7%	1.15	256.00	0.041	79.4%	1.24	4096.00	0.048	49.2%	2.81	65536.00	0.053	31.4%	2.18
		70	9.55		0.048	95.2%	1.12		0.052	85.7%	1.17		0.059	61.0%	2.79		0.066	44.1%	2.27
		100	13.07		0.053	100.0%	1.03		0.060	87.4%	1.13		0.062	87.4%	3.27		0.075	58.5%	2.21
		10	2.8		0.492	28.8%	1.08		0.924	30.3%	2.02		7.083	45.1%	3.24		13.532	99.1%	7.58
		30	7.49		0.495	45.1%	1.19		0.946	44.1%	1.65		8.499	46.8%	2.76		14.294	99.3%	6.01
IPC-GRID (8076)	8.7	50	12.02	10.55	0.497	66.7%	1.11	111.43	0.967	66.7%	1.51	1176.26	9.547	58.6%	2.06	12416.75	15.063	99.8%	3.99
		70	17.16		0.503	73.4%	1.11		1.203	75.4%	1.35		10.259	67.8%	1.82		15.588	99.3%	3.21
		100	21.84		0.432	82.5%	1.05		1.317	85.7%	1.29		10.573	81.4%	1.46		16.208	100.0%	2.25
		10	1.33		0.002	51.1%	1.47		0.003	15.6%	1.33		0.003	26.7%	1.07		0.005	57.8%	1.33
		30	3.33		0.004	40.0%	1.18		0.006	26.7%	1.29		0.005	33.3%	1.07		0.007	48.9%	1.33
KITCHEN (900)	3.0	50	4.0	2767208.65	0.006	53.3%	1.07	7.66E+12	0.008	48.9%	1.36	2.11+19	0.008	37.8%	1.04	5.86E+25	0.009	48.9%	1.33
		70	5.0		0.008	71.1%	1.21		0.007	53.3%	1.33		0.008	46.7%	1.09		0.009	44.4%	1.33
		100	7.47		0.008	73.3%	1.21		0.007	48.9%	1.33		0.010	42.2%	1.09		0.010	48.9%	1.33
		10	2.82		0.205	42.5%	2.01		0.233	52.3%	2.72		0.271	59.0%	4.07		0.312	80.8%	5.51
		30	8.01		0.227	66.0%	1.53		0.258	72.5%	1.68		0.314	79.3%	2.85		0.436	88.9%	3.08
LOGISTICS (8076)	10.5	50	13.07	27.85	0.246	81.5%	1.31	776.04	0.319	81.0%	1.53	21618.81	0.355	85.6%	2.21	602248.76	0.504	92.6%	2.23
		70	18.33		0.253	91.7%	1.14		0.362	93.2%	1.19		0.421	93.5%	1.45		0.515	97.6%	1.44
		100	24.41		0.324	97.3%	1.04		0.412	99.5%	1.08		0.466	97.3%	1.09		0.564	100.0%	1.02
		10	3.96		0.282	56.3%	1.58		0.291	84.5%	2.21		0.294	64.7%	2.93		0.302	68.7%	3.12
		30	11.14		0.286	81.3%	1.29		0.299	95.2%	1.67		0.302	81.7%	1.82		0.333	83.3%	2.07
MICONIC (4368)	6.0	50	18.07	9.18	0.294	94.4%	1.19	84.44	0.303	97.2%	1.31	776.05	0.311	88.9%	1.46	7131.55	0.354	87.7%	1.79
		70	25.32		0.311	98.4%	1.07		0.329	99.6%	1.13		0.345	92.5%	1.51		0.369	94.8%	1.94
		100	35.57		0.298	100.0%	1.0		0.340	100.0%	1.02		0.374	94.0%	1.07		0.381	100.0%	1.26
		10	3.0		0.579	51.6%	1.81		0.659	41.7%	1.88		0.738	41.3%	1.94		0.675	68.7%	3.19
		30	7.93		0.583	64.7%	1.49		0.676	57.9%	1.49		0.765	61.5%	1.78		0.721	81.0%	2.45
ROVERS (4368)	6.0	50	12.75	9410.14	0.591	76.2%	1.19	88550676.93	0.687	79.4%	1.28	8.34E+11	0.790	78.2%	1.49	7.84E+15	0.762	92.5%	1.96
		70	17.96		0.595	88.9%	1.13		0.702	86.9%	1.14		0.806	90.5%	1.26		0.794	99.6%	1.74
		100	24.93		0.602	94.0%	1.07		0.714	92.9%	1.04		0.821	97.6%	1.12		0.846	100.0%	1.28
		10	2.07		0.105	50.0%	1.88		0.108	50.4%	2.06		0.112	58.3%	2.02		0.124	65.1%	2.46
		30	5.43		0.119	59.9%	1.52		0.124	63.1%	1.63		0.128	71.8%	1.66		0.131	75.8%	1.77
SATELLITE (4368)	6.4	50	8.71	27.85	0.123	65.1%	1.25	776.04	0.141	75.0%	1.37	21618.81	0.152	81.0%	1.46	602248.76	0.145	85.3%	1.52
		70	12.29		0.143	72.6%	1.14		0.155	85.7%	1.29		0.164	86.1%	1.19		0.173	90.9%	1.26
		100	16.89		0.158	82.1%	1.06		0.169	95.2%	1.08		0.173	94.0%	1.08		0.197	100.0%	1.10
		10	3.18		4.799	26.6%	1.33		*Timeout*	-	-		*Timeout*	-	-		*Timeout*	-	-
		30	8.82		5.591	29.0%	1.23		*Timeout*	-	-		*Timeout*	-	-		*Timeout*	-	-
SOKOBAN (4368)	7.1	50	14.07	9.18	6.347	32.5%	1.22	84.44	*Timeout*	-	-	776.04	*Timeout*	-	-	7131.55	*Timeout*	-	-
		70	19.86		6.803	35.3%	1.19		*Timeout*	-	-		*Timeout*	-	-		*Timeout*	-	-
		100	27.71		7.595	45.2%	1.14		*Timeout*	-	-		*Timeout*	-	-		*Timeout*	-	-
		10	2.61		0.895	37.3%	1.38		1.352	38.1%	1.43		1.441	56.7%	2.12		1.697	79.4%	3.02
		30	6.75		0.909	50.8%	1.43		1.376	52.0%	1.49		1.465	71.0%	1.73		2.184	88.1%	2.12
ZENO (4368)	6.9	50	10.82	48.50	0.937	63.1%	1.29	2352.53	1.412	66.3%	1.33	114104.80	1.501	83.3%	1.43	5534417.30	2.273	93.7%	1.41
		70	15.21		0.948	79.0%	1.19		1.430	77.4%	1.22		1.548	91.7%	1.29		2.452	96.8%	1.33
		100	21.14		0.954	91.7%	1.07		1.571	92.9%	1.08		1.513	96.4%	1.04		2.758	100.0%	1.04

Table 1: Experimental results of our approach for recognizing goals in incomplete STRIPS domain models.

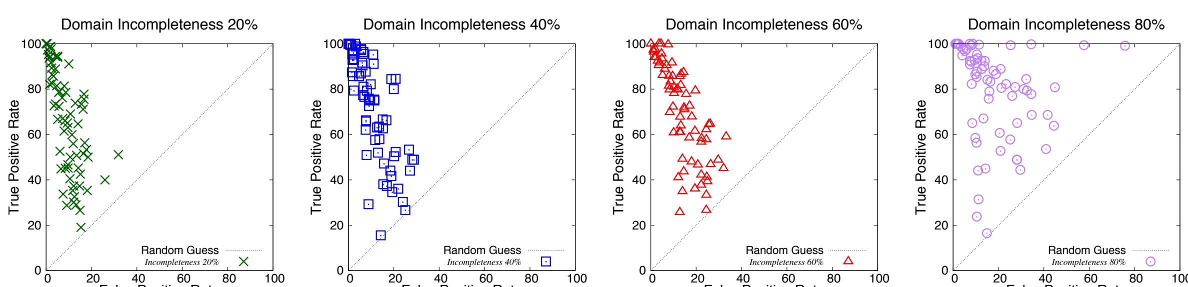

Figure 2: ROC space for all four percentage of domain incompleteness.

preconditions. The average number of possible complete domain models $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$ is huge for several domains (CAMPUS, DWR, KITCHEN, and ROVERS), showing that the task of goal recognition in incomplete domains models is quite difficult and complex.

Figure 2 shows four ROC space graphs corresponding to recognition performance over the four percentages of domain incompleteness we used in our experiments. We aggregate multiple recognition problems for all domains and plot these results in ROC space varying the percentage of domain incompleteness. Although the true positive rate is high for most recognition problems at most percentages of domain incompleteness, as the percentage of domain incompleteness increases, the false positive rate also increases, leading to several problems being recognized with a performance close to the random guess line. We argue that this happens because the number of extracted landmarks decreases significantly the as the number of definite preconditions and effects diminishes, and consequently, all candidate goals have just a few (if any) landmarks. For example, in several cases in which the domain incompleteness is 60% and 80%, the set of landmarks is quite similar, leading our approach to return more than one candidate goal (*Spread* in $\mathcal{G}$) as the correct one (*i.e.*, there is more uncertainty in the result of the recognition process).

6 Related Work

To the best of our knowledge, the only earlier work that deals directly with plan or goal recognition with incomplete domains are that of Lee and McCartney (1998) and Kerkez and Cox (2002). In the first one, Lee and McCartney (1998) developed plan recognition approach that uses machine learning and stochastic techniques (Hidden Markov Models) to learn actions and properties based on an incomplete agent behavior model from partial observation, which are stored as a history of interactions in a dataset. They use an incomplete plan library to represent the agent behavior model. The approach of Kerkez and Cox (2002) takes as input an incomplete plan library and deals with incomplete domain information by using a planner. More specifically, the approach of Kerkez and Cox uses a planner to fill and complete an incomplete plan library from the observations, and then recognize the observed agent's goal.

Unlike our approach, most recent planning-based recognition approaches (Ramírez and Geffner 2009; 2010; Sohrabi, Riabov, and Udrea 2016) use a planner to recognize goals and plans from observations, running a planner at least $2 \times \mathcal{G}$ times. Conversely, E.-Martín, R.-Moreno, and Smith (2015) and Pereira, Oren, and Meneguzzi (2017a) are similar to our approach and avoid using any planning system during the goal and plan recognition recognition process and use planning-graphs and landmarks, respectively. Keren, Gal, and Karpas (2014) developed an approach that assumes planning domain models are not fixed, and it changes (redesigns) the domain definition to facilitate the task of goal recognition in planning domain models. These approaches differ from ours because they only deal with complete (even if modified) domain models, and most of them transform/compile the goal/plan recognition problem into a planning problem to be solved by a planner. Such transformation or compilation process may not necessarily work with incomplete STRIPS domain models, given the very large number of potential models. The approaches of E.-Martín, R.-Moreno, and Smith and Pereira, Oren, and Meneguzzi could work in incomplete domain models with some adaptations (*e.g.*, by ignoring all *possible* preconditions and effects), though these approaches would likely be less accurate than our approach because these approaches do not deal intentionally with possible preconditions and effects. We leave a comparison with these two approaches to future work.

Finally, the motivation for our use of incomplete domain models comes from work on planning algorithms using such models, of which there are several recent approaches in the literature (Garland and Lesh 2002; Robertson and Bryce 2009; Nguyen, Kambhampati, and Do 2010; Weber and Bryce 2011; Nguyen and Kambhampati 2014; Nguyen, Sreedharan, and Kambhampati 2017).

7 Conclusions and Future Work

In this paper, we have developed a novel goal recognition approach that deals with incomplete domain models that have *possible*, rather than *known*, preconditions and effects. The main contributions of this paper are: first, a landmark extraction algorithm that deals with incomplete domains models using ORPG; second, a goal recognition heuristic for incomplete domain models that relies on landmarks; third, a dataset with thousands of problems that use incomplete domain models (annotated with possible preconditions and effects). Experiments over thousands of goal recognition problems in 15 planning domain models show that our approach is fast and accurate when dealing with incomplete domains at all variations of observability and domain incompleteness.

As future work, we aim to explore multiple refinements of our goal recognition approach. First, we aim to explore the set of possible preconditions for extracting landmarks, since our approach only explores the set of possible add effects to build an ORPG. Second, we intend to use a propagated RPG to reason about impossible incomplete domain models, much like in (Weber and Bryce 2011), to build a planning heuristic. Third, we aim to use a bayesian framework to compute probabilistic estimations of which possible complete domain is most consistent with the observations. Forth, we intend to combine the concept of landmarks and planning heuristics to enhance the process of goal recognition (Pereira, Oren, and Meneguzzi 2017b). Finally, we aim to explore recent work that could be used as part of a complete methodology to develop domains includes an approach to acquire and infer information from domains with incomplete information based plan traces. More specifically, Zhuo, Nguyen, and Kambhampati (2013) developed an approach to refine incomplete domain models based on plan traces.

References

Blum, A. L., and Furst, M. L. 1997. Fast Planning Through Planning Graph Analysis. *Journal of Artificial Intelligence Research (JAIR)* 90(1-2):281–300.

E.-Martín, Y.; R.-Moreno, M. D.; and Smith, D. E. 2015. A Fast Goal Recognition Technique Based on Interaction Estimates. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI 2015)*.

Fikes, R. E., and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Journal of Artificial Intelligence Research (JAIR)* 2(3):189–208.

Garland, A., and Lesh, N. 2002. Plan Evaluation with Incomplete Action Descriptions. In *Eighteenth National Conference on Artificial Intelligence*, 461–467. Menlo Park, CA, USA: American Association for Artificial Intelligence.

Granada, R.; Pereira, R. F.; Monteiro, J.; Barros, R.; Ruiz, D.; and Meneguzzi, F. 2017. Hybrid activity and plan recognition for video streams. In *The AAAI 2017 Workshop on Plan, Activity, and Intent Recognition*.

Hoffmann, J., and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research (JAIR)* 14(1):253–302.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *Journal of Artificial Intelligence Research (JAIR)* 22(1):215–278.

Keren, S.; Gal, A.; and Karpas, E. 2014. Goal Recognition Design. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS 2014)*.

Kerkez, B., and Cox, M. T. 2002. Case-Based Plan Recognition with Incomplete Plan Libraries. In *Proceedings of AAAI Fall 2002 Symposium on Intent Inference*, 52–54.

Lee, J.-J., and McCartney, R. 1998. Partial Plan Recognition with Incomplete Information. In *Proceedings of International Conference on Multi Agent Systems*, 445–446. IEEE.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language.

Nguyen, T. A., and Kambhampati, S. 2014. A Heuristic Approach to Planning with Incomplete STRIPS Action Models. In *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2014, Portsmouth, New Hampshire, USA, June 21-26, 2014*.

Nguyen, T. A.; Kambhampati, S.; and Do, M. B. 2010. Assessing and Generating Robust Plans with Partial Domain Models. In *ICAPS Workshop on Planning under Uncertainty*.

Nguyen, T.; Sreedharan, S.; and Kambhampati, S. 2017. Robust Planning with Incomplete Domain Models. *Artificial Intelligence* 245:134 – 161.

Pereira, R. F., and Meneguzzi, F. 2016. Landmark-Based Plan Recognition. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI 2016)*.

Pereira, R. F., and Meneguzzi, F. 2017. Goal and Plan Recognition Datasets using Classical Planning Domains. Zenodo.

Pereira, R. F.; Oren, N.; and Meneguzzi, F. 2017a. Landmark-Based Heuristics for Goal Recognition. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI 2017)*.

Pereira, R. F.; Oren, N.; and Meneguzzi, F. 2017b. Monitoring plan optimality using landmarks and domain-independent heuristics. In *The AAAI 2017 Workshop on Plan, Activity, and Intent Recognition (PAIR)*.

Ramírez, M., and Geffner, H. 2009. Plan Recognition as Planning. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI 2009)*.

Ramírez, M., and Geffner, H. 2010. Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners. In *Proceedings of the Conference of the American Association of Artificial Intelligence (AAAI 2010)*.

Richter, S., and Westphal, M. 2010. The LAMA Planner: Guiding Cost-based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research (JAIR)* 39(1):127–177.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks Revisited. In *Proceedings of the Twenty-Third National Conference on Artificial Intelligence - Volume 2*, AAAI'08, 975–982. AAAI Press.

Robertson, J., and Bryce, D. 2009. Reachability Heuristics for Planning in Incomplete Domains. In *ICAPS Workshop on Heuristics for Domain Independent Planning*.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan Recognition as Planning Revisited. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*.

Sukthankar, G.; Goldman, R. P.; Geib, C.; Pynadath, D. V.; and Bui, H. H. 2014. *Plan, Activity, and Intent Recognition: Theory and Practice*. Elsevier.

Weber, C., and Bryce, D. 2011. Planning and Acting in Incomplete Domains. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling, ICAPS 2011, Freiburg, Germany June 11-16, 2011*.

Zhuo, H. H.; Nguyen, T. A.; and Kambhampati, S. 2013. Refining Incomplete Planning Domain Models Through Plan Traces. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*.