Model-Based Goal Recognition with Unknown Domain Models

Diego Aineto[1] , **Sergio Jiménez**[1] , **Eva Onaindia**[1] and , **Miquel Ramírez**[2]

[1]Departamento de Sistemas Informáticos y Computación. Universitat Politècnica de València. Valencia, Spain

[2]School of Computing and Information Systems. The University of Melbourne. Melbourne, Victoria. Australia

{dieaigar,serjice,onaindia}@dsic.upv.es, miquel.ramirez@unimelb.edu.au

Abstract

The paper shows how to relax one key assumption of the *plan recognition as planning* for *goal recognition* that is knowing the action model of the observed agent. The paper introduces a novel formulation that fits together the *learning of planning action models* with the *plan recognition as planning* approach. The empirical evaluation evidences that our novel formulation allows to solve standard goal recognition benchmarks without having knowing the action model of the observed agent.

1 Introduction

Goal recognition is a particular classification task in which each class represents a different goal and each example is an observation of an agent acting to achieve one of that goals. Despite there is a wide range of diffenrent approaches for *goal recognition*, *plan recognition as planning* [Ramírez and Geffner, 2009; Ramírez, 2012] is one of the most popular since it is at the core of several interesting tasks such as, *goal recognition design* [Keren *et al.*, 2014], *deceptive planning* [Masters and Sardina, 2017], *planning for transparency* [MacNally *et al.*, 2018] or *counterplanning* [Pozanco *et al.*, 2018].

Plan recognition as planning leverages the action model of the observed agent and an off-the-shelf classical planner to compute the most likely goal of the agent. In this paper we show that we can relax the key assumption of *plan recognition as planning* for *Goal recognition* that is knowing the action model of the observed agent. In particular, the paper introduces a novel formulation that fits together the *learning of planning action models* with the *plan recognition as planning* approach. The evaluation of our formulation evidences that it allows to solve goal recognition tasks, even when the action model of the observed is unknown.

2 Background

This section formalizes the *planning model* we follow as well as the kind of *observations* that are given as classification examples for the *goal recognition* task.

2.1 Classical planning with conditional effects

Let F be the set of *fluents* or *state variables* (propositional variables) describing a state. A *literal* l is a valuation of a fluent $f \in F$; i.e. either $l = f$ or $l = \neg f$. A set of literals L represents a partial assignment of values to fluents (without loss of generality, we will assume that L does not contain conflicting values). Given L, let $\neg L = \{\neg l : l \in L\}$ be its complement. We use $\mathcal{L}(F)$ to denote the set of all literal sets on F; i.e. all partial assignments of values to fluents. A *state* s is a full assignment of values to fluents; $|s| = |F|$.

A *classical planning frame* is a tuple $\Phi = \langle F, A\rangle$, where F is a set of fluents and A is a set of *actions*. Each classical planning action $a \in A$ has a precondition $\mathsf{pre}(a) \in \mathcal{L}(F)$, a set of effects $\mathsf{eff}(a) \in \mathcal{L}(F)$, and a positive action cost $cost(a)$. The semantics of actions $a \in A$ is specified with two functions: $\rho(s, a)$ denotes whether action a is *applicable* in a state s and $\theta(s, a)$ denotes the *successor state* that results of applying action a in a state s. Then, $\rho(s, a)$ holds iff $\mathsf{pre}(a) \subseteq s$, i.e. if its precondition holds in s. The result of executing an applicable action $a \in A$ in a state s is a new state $\theta(s, a) = (s \setminus \neg\mathsf{eff}(a)) \cup \mathsf{eff}(a)$. Subtracting the complement of $\mathsf{eff}(a)$ from s ensures that $\theta(s, a)$ remains a well-defined state. The subset of action effects that assign a positive value to a state fluent is called *positive effects* and denoted by $\mathsf{eff}^+(a) \in \mathsf{eff}(a)$ while $\mathsf{eff}^-(a) \in \mathsf{eff}(a)$ denotes the *negative effects* of an action $a \in A$.

A *classical planning problem* is a tuple $P = \langle F, A, I, G\rangle$, where I is the initial state and $G \in \mathcal{L}(F)$ is the set of goal conditions over the state variables. A *plan* π is an action sequence $\pi = \langle a_1, \ldots, a_n\rangle$, with $|\pi| = n$ denoting its *plan length* and $cost(\pi) = \sum_{a\in\pi} cost(a)$ its *plan cost*. The execution of π on the initial state I of P induces a *trajectory* $\tau(\pi, s_0) = \langle s_0, a_1, s_1, \ldots, a_n, s_n\rangle$ such that $s_0 = I$ and, for each $1 \leq i \leq n$, it holds $\rho(s_{i-1}, a_i)$ and $s_i = \theta(s_{i-1}, a_i)$. A plan π solves P iff the induced *trajectory* $\tau(\pi, s_0)$ reaches a final state $G \subseteq s_n$, where all goal conditions are met. A solution plan is *optimal* iff it minimizes the sum of action costs.

An *action with conditional effects* $a_c \in A$ is defined as a set of preconditions $\mathsf{pre}(a_c) \in \mathcal{L}(F)$ and a set of *conditional effects* $\mathsf{cond}(a_c)$. Each conditional effect $C \rhd E \in \mathsf{cond}(a_c)$ is composed of two sets of literals: $C \in \mathcal{L}(F)$, the *condition*, and $E \in \mathcal{L}(F)$, the *effect*. An action a_c is applicable in a state s if $\rho(s, a_c)$ is true, and the result of applying action a_c in state s is $\theta(s, a_c) = \{s \setminus \neg\mathsf{eff}_c(s, a) \cup \mathsf{eff}_c(s, a)\}$ where

$\mathsf{eff}_c(s,a)$ are the *triggered effects* resulting from the action application (conditional effects whose conditions hold in s):

$$\mathsf{eff}_c(s,a) = \bigcup_{C \rhd E \in \mathsf{cond}(a_c), C \subseteq s} E,$$

2.2 The observation model

Given a planning problem $P = \langle F, A, I, G\rangle$, a plan π and a trajectory $\tau(\pi, P)$, we define the *observation of the trajectory* as an interleaved combination of actions and states that represents the observation from the execution of π in P. Formally, $\mathcal{O}(\tau) = \langle s_0^o, a_1^o, s_1^o \ldots, a_l^o, s_m^o\rangle$, $s_0^o = I$, and:

- The **observed actions** are consistent with π, which means that $\langle a_1^o, \ldots, a_l^o\rangle$ is a sub-sequence of π. Specifically, the number of observed actions, l, can range from 0 (fully unobservable action sequence) to $|\pi|$ (fully observable action sequence).
- The **observed states** $\langle s_0^o, s_1^o, \ldots, s_m^o\rangle$ is a sequence of possibly *partially observable states*, except for the initial state s_0^o, which is fully observable. A partially observable state s_i^o is one in which $|s_i^o| < |F|$; i.e., a state in which at least a fluent of F is not observable. Note that this definition also comprises the case $|s_i^o| = 0$, when the state is fully unobservable. Whatever the sequence of observed states of $\mathcal{O}(\tau)$ is, it must be consistent with the sequence of states of $\tau(\pi, P)$, meaning that $\forall i, s_i^o \subseteq s_i$. In practice, the number of observed states, m, range from 1 (the initial state, at least), to $|\pi|+1$, and the observed intermediate states will comprise a number of fluents between $[1, |F|]$.

We assume a bijective monotone mapping between actions/states of trajectories and observations [Ramírez and Geffner, 2009], thus also granting the inverse consistency relationship (the trajectory is a superset of the observation). Therefore, transiting between two consecutive observed states in $\mathcal{O}(\tau)$ may require the execution of more than a single action ($\theta(s_i^o, \langle a_1, \ldots, a_k\rangle) = s_{i+1}^o$, where $k \geq 1$ is unknown but finite. In other words, having $\mathcal{O}(\tau)$ does not imply knowing the actual length of π.

3 Model-Based Goal Recognition with Unknown Domain Models

Goal recognition is a particular classification task in which each class represents a different goal and each example is an observation of an agent acting to achieve one of that goals. Following the *naive Bayes classifier*, the *solution* to the *goal recognition* task is the subset of goals in $G[\cdot]$ that maximizes this expression.

$$argmax_{g \in G[\cdot]} P(\mathcal{O}|g)P(g). \tag{1}$$

The *Plan recognition as planning* approach shows how to compute estimates of the $P(\mathcal{O}|g)$ likelyhood leveraging the action model of the observed agent and an off-the-shelf classical planner. Recent works show that faster, but less accurate estimates, of this $P(\mathcal{O}|g)$ likelyhood can also be computed using relaxations of the classical planning tasks [Pereira *et al.*, 2017].

3.1 Well-defined STRIPS action schemata

STRIPS action schemata provide a compact representation for specifying classical planning models.*A* STRIPS *action schema* ξ is defined by four lists: A list of *parameters* $pars(\xi)$, and three list of predicates (namely $pre(\xi)$, $del(\xi)$ and $add(\xi)$) that shape the kind of fluents that can appear in the *preconditions*, *negative effects* and *positive effects* of the actions induced from that schema.

Let be Ψ the set of *predicates* that shape the propositional state variables F, and a list of *parameters* $pars(\xi)$. The set of elements that can appear in $pre(\xi)$, $del(\xi)$ and $add(\xi)$ of the STRIPS action schema ξ is given by FOL interpretations of Ψ over the parameters $pars(\xi)$ and is denoted as $\mathcal{I}_{\Psi,\xi}$. For instance, in the *blocksworld* the $\mathcal{I}_{\Psi,\xi}$ set contains five elements for a `pickup`(v_1) schemata, $\mathcal{I}_{\Psi,pickup}$={`handempty`, `holding`($v_1$), `clear`($v_1$), `ontable`($v_1$), `on`($v_1, v_1$)} while it contains eleven elements for a `stack`(v_1, v_2) schemata, $\mathcal{I}_{\Psi,stack}$={`handempty`, `holding`(v_1), `holding`(v_2), `clear`(v_1), `clear`(v_2), `ontable`(v_1), `ontable`(v_2), `on`(v_1, v_1), `on`(v_1, v_2), `on`(v_2, v_1), `on`(v_2, v_2)}.

Despite any element of $\mathcal{I}_{\Psi,\xi}$ can *a priori* appear in the $pre(\xi)$, $del(\xi)$ and $add(\xi)$ of schema ξ, the space of possible STRIPS schemata is bounded by a set of constraints $\mathcal{C}$ of three kinds:

1. *Syntactic constraints*. STRIPS constraints require $del(\xi) \subseteq pre(\xi)$, $del(\xi) \cap add(\xi) = \emptyset$ and $pre(\xi) \cap add(\xi) = \emptyset$. Considering exclusively these syntactic constraints, the size of the space of possible STRIPS schemata is given by $2^{2\times|\mathcal{I}_{\Psi,\xi}|}$. *Typing constraints* are also of this kind [McDermott *et al.*, 1998].
2. *Domain-specific constraints*. One can introduce domain-specific knowledge to constrain further the space of possible schemata. For instance, in the *blocksworld* one can argue that `on`(v_1, v_1) and `on`(v_2, v_2) will not appear in the $pre(\xi)$, $del(\xi)$ and $add(\xi)$ lists of an action schema ξ because, in this specific domain, a block cannot be on top of itself. *State invariants* are also constraints of this kind [Fox and Long, 1998].
3. *Observation constraints*. An observations $\mathcal{O}(\tau)$ depicts *semantic knowledge* that constraints further the space of possible action schemata.

Definition 1 (Well-defined STRIPS action schemata) *Given a set of* predicates Ψ*, a list of action* parameters $pars(\xi)$*, and set of FOL constraints* $\mathcal{C}$*,* ξ *is a* **well-defined STRIPS action schema** *iff its three lists* $pre(\xi) \subseteq \mathcal{I}_{\Psi,\xi}$*,* $del(\xi) \subseteq \mathcal{I}_{\Psi,\xi}$ *and* $add(\xi) \subseteq \mathcal{I}_{\Psi,\xi}$ *only contain elements in* $\mathcal{I}_{\Psi,\xi}$ *and they satisfy all the constraints in* $\mathcal{C}$*.*

We say a planning model $\mathcal{M}$ is *well-defined* if all its STRIPS action schemata are *well-defined*.

3.2 Edit distances for STRIPS planning models

First, we define the two edit *operations* on a schema ξ that belongs to a STRIPS model $\mathcal{M} \in M$:

- *Deletion*. Given $\xi \in \mathcal{M}$, an element from any of the lists $pre(\xi)$/$del(\xi)$/$add(\xi)$ is removed such that the result is a *well-defined* STRIPS action schema.
- *Insertion*. Given $\xi \in \mathcal{M}$, an element in $\mathcal{I}_{\Psi,\xi}$ is added to any of the lists $pre(\xi)$/$del(\xi)$/$add(\xi)$ such that the result is a *well-defined* action schema.

Second, let us define when to action models are comparable. For instance, we claim that the `stack(?v1,?v2)` and `unstack(?v1,?v2)` actions schemata from a four operator *blocksworld* [Slaney and Thiébaux, 2001] are comparable while, the `stack(?v1,?v2)` and `pick-up(?v1)` schemata are not. Last but not least, we say that two STRIPS models $\mathcal{M}$ and $\mathcal{M}'$ are *comparable* iff there exists a bijective function $\mathcal{M} \mapsto \mathcal{M}^*$ that maps every action schema $\xi \in \mathcal{M}$ to a comparable schemata $\xi' \in \mathcal{M}'$ and vice versa.

Definition 2 (Comparable STRIPS action schemata) *Two* STRIPS *schemata ξ and ξ' are* **comparable** *iff $pars(\xi) = pars(\xi')$, i.e, both share the same list of parameters.*[1]

We are now ready to formalize an *edit distance* that quantifies how similar two given STRIPS models are. The distance is symmetric and meets the *metric axioms* provided that the two edit operations, *deletion* and *insertion*, have the same positive cost.

Definition 3 (Edit distance) *Let $\mathcal{M}$ and $\mathcal{M}'$ be two* comparable *and* well-defined STRIPS *planning models within the same set of predicates Ψ. The* **edit distance** $\delta(\mathcal{M}, \mathcal{M}')$ *is the minimum number of* edit operations *that is required to transform $\mathcal{M}$ into $\mathcal{M}'$.*

Since $\mathcal{I}_{\Psi,\xi}$ is a bounded set, the maximum number of edits that can be introduced to an action schema is bounded as well. The **maximum edit distance** of a STRIPS model $\mathcal{M}$ built with predicates Ψ is $\delta(\mathcal{M}, *) = \sum_{\xi \in \mathcal{M}} 3 \times |\mathcal{I}_{\Psi,\xi}|$ (note that if we consider the set of syntactic constraints then $\delta(\mathcal{M}, *) = \sum_{\xi \in \mathcal{M}} 2 \times |\mathcal{I}_{\Psi,\xi}|$).

An observation of the execution of a plan generated with $\mathcal{M}$ further constraints the space of possible action schemata of $\mathcal{M}$. The *semantic knowledge* included in the observations introduce a third type of constraints, that we will call *observation constraints*, and that can be added to the set $\mathcal{C}$. In addition, *observation constraints* allow us to define an edit distance to elicit the value of $P(\mathcal{O}|\mathcal{M})$. It can be argued that the shorter this distance the better the given model explains the given observation.

Definition 4 (Observation edit distance) *Given a planning problem P, an observation $\mathcal{O}(\tau)$ of the execution of a plan that solves P and a* STRIPS *planning model $\mathcal{M}$ (all defined within the same set of predicates Ψ). The* **observation edit distance**, *$\delta^o(\mathcal{M}, \mathcal{O})$, is the minimal edit distance from $\mathcal{M}$ to any* comparable *and well-defined model $\mathcal{M}'$ s.t. $\mathcal{M}'$ produces a trajectory $\tau(\pi, P)$ that reaches the goals in P and is* consistent *with $\mathcal{O}(\tau)$;*

$$\delta^o(\mathcal{M}, \mathcal{O}) = \min_{\forall \mathcal{M}' \rightarrow \mathcal{O}} \delta(\mathcal{M}, \mathcal{M}')$$

$\delta^o(\mathcal{M}, \mathcal{O})$ can also be defined through the edition that the observation $\mathcal{O}(\tau)$ requires to fit $\mathcal{M}$. This implies defining *edit operations* that modify the observation $\mathcal{O}(\tau)$ instead of the model $\mathcal{M}$ [Yang *et al.*, 2007; Sohrabi *et al.*, 2016]. Our definition of *observation edit distance* is more practical since the size of $\mathcal{I}_{\Psi,\xi}$ is usually much smaller than F (the number of variables in the action schemata should normally be lower than the number of objects in a planning problem).

Definition 5 (*Closest consistent models*) *Given a model $\mathcal{M}$, the set M^* of the* **closest consistent models** *is the set of models $\mathcal{M}'$ that: (1) produce a trajectory $\tau(\pi, P)$ that reaches the goals in P and is* consistent *with $\mathcal{O}(\tau)$ and (2) their* edit distance *to $\mathcal{M}$ is minimal;*

$$\arg\min_{\forall \mathcal{M}' \rightarrow \mathcal{O}} \delta(\mathcal{M}, \mathcal{M}')$$

4 Evaluation

5 Conclusions

References

[Fox and Long, 1998] Maria Fox and Derek Long. The automatic inference of state invariants in tim. *Journal of Artificial Intelligence Research*, 9:367–421, 1998.

[Keren *et al.*, 2014] Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design. In *International Conference on Automated Planning and Scheduling, (ICAPS-14)*, pages 154–162, 2014.

[MacNally *et al.*, 2018] Aleck M MacNally, Nir Lipovetzky, Miquel Ramirez, and Adrian R Pearce. Action selection for transparent planning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1327–1335. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[Masters and Sardina, 2017] Peta Masters and Sebastian Sardina. Deceptive path-planning. In *IJCAI 2017*, pages 4368–4375. AAAI Press, 2017.

[McDermott *et al.*, 1998] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL – The Planning Domain Definition Language, 1998.

[Pereira *et al.*, 2017] Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. Landmark-based heuristics for goal recognition. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. AAAI Press, 2017.

[Pozanco *et al.*, 2018] Alberto Pozanco, Yolanda E.-Martín, Susana Fernández, and Daniel Borrajo. Counterplanning using goal recognition and landmarks. In *International Joint Conference on Artificial Intelligence, (IJCAI-18)*, pages 4808–4814, 2018.

[Ramírez and Geffner, 2009] Miquel Ramírez and Hector Geffner. Plan recognition as planning. In *International Joint conference on Artifical Intelligence, (IJCAI-09)*, pages 1778–1783. AAAI Press, 2009.

[1] In STRIPS models, $pars(\xi) = pars(\xi')$ implies the number of parameters must be the same. For other planning models that allow object typing, the equality implies that parameters share the same type

[Ramírez, 2012] Miquel Ramírez. *Plan recognition as planning*. PhD thesis, Universitat Pompeu Fabra, 2012.

[Slaney and Thiébaux, 2001] John Slaney and Sylvie Thiébaux. Blocks world revisited. *Artificial Intelligence*, 125(1-2):119–153, 2001.

[Sohrabi *et al.*, 2016] Shirin Sohrabi, Anton V. Riabov, and Octavian Udrea. Plan recognition as planning revisited. In *International Joint Conference on Artificial Intelligence, (IJCAI-16)*, pages 3258–3264, 2016.

[Yang *et al.*, 2007] Qiang Yang, Kangheng Wu, and Yunfei Jiang. Learning action models from plan examples using weighted MAX-SAT. *Artificial Intelligence*, 171(2-3):107–143, 2007.