Equivalence-Based Abstractions for Learning General Policies

Dominik Drexler[1], **Simon Ståhlberg, Blai Bonet**[2], **Hector Geffner**[3,1]
[1]Linköping University, Sweden
[2]Universitat Pompeu Fabra, Spain
[3]RWTH Aachen University, Germany
dominik.drexler@liu.se, simon.stahlberg@gmail.com, bonetblai@gmail.com, hector.geffner@ml.rwth-aachen.de

Abstract

Identifying state symmetries plays a crucial role in minimizing the number of states explored during search, yet identifying precisely all symmetries is computationally hard. In the context of learning general policies that solve instances of arbitrary size from small instances, however, this computational bottleneck is not a problem. In this paper, we address the task of identifying all state symmetries through the lens of the graph isomorphism problem. To accomplish this, we represent states as undirected, labeled graphs that reflect the relational structure of states and the goal. We then use off-the-shelf graph isomorphism algorithms to determine whether two states are isomorphic with respect to the goal. The isomorphism relationship forms equivalent classes that result in an abstract state space that can be used instead of the original one to learn general policies more efficiently. While this abstract state space can be used for many different learning tasks, we focus on learning symbolic general policies where we show that the proposed approach can lead to significant speedups.

Introduction

In generalized planning, the task involves obtaining a general strategy for solving a class of instances $\mathcal{Q}$ drawn from the same planning domain. A classical planning domain ensures that all instances in $\mathcal{Q}$ share a structure given by a common set of action schemas and predicates (Srivastava, Immerman, and Zilberstein 2011; Jiménez, Segovia-Aguas, and Jonsson 2019; Toyer et al. 2020; Yang et al. 2022; Srivastava 2022).

These general strategies, also called general plans or policies, are predominantly learned (Srivastava, Immerman, and Zilberstein 2008; Bonet, Palacios, and Geffner 2009; Hu and Giacomo 2011; Belle and Levesque 2016; Celorrio, Segovia-Aguas, and Jonsson 2019; Illanes and McIlraith 2019) rather than synthesized, and this is a done from a diverse set of small instances from $\mathcal{Q}$ (Ståhlberg, Bonet, and Geffner 2022a,b). For this, these approaches generate the complete (reachable) state space for each problem within the training set, and find a policy that works across all such states. The learned policy is then validated on slightly larger problems from $\mathcal{Q}$, referred to as the validation set. In the symbolic setting, where the learning problem is formulated as a combinatorial optimization problem, the resulting general policies can be understood and be shown to be correct over *all* problems in the target class $\mathcal{Q}$, even if the number of such problems is infinite (Bonet, Francès, and Geffner 2019; Francès, Bonet, and Geffner 2021). Similar methods have been used to learn sketches as well with similar formal properties (Drexler, Seipp, and Geffner 2022).

A computational bottleneck of the symbolic approach to generalized planning is that even for small instances in $\mathcal{Q}$, the complete state space can be huge; for example, in Gripper, the task is to move balls from one room to another, and with n balls and 2 rooms, there are at least 2^n reachable states. The size of the instances in the training set limits the scope of the combinatorial approach for learning general policies that cannot deal optimally with training instances of large sizes. It turns out, however, that many pairs of states in the training set are *equivalent*, meaning that a solution for one state implies a solution for the other. This suggests that the number of states for the training set can be significantly reduced by considering just one representative of each equivalent class of states.

In this work, we take the first steps toward formalizing this intuition and developing methods that are computationally effective. For this, we define a class of problem abstractions that reduce the size of the state space while preserving solutions as captured by general policies expressed in terms of features defined from the domain predicates in first-order logic. We show then that these abstractions can be obtained by finding the states that are structurally equivalent, a notion that is formalized in terms of isomorphisms over relational structures, and which is computed by reducing relational structures (states) to plain colored graphs and using state-of-the-art algorithms for testing graph isomorphism (GI). While GI is not known to be in P, there are very efficient codes for testing GI over large graphs, much larger than those resulting from the states in the training instances.

The paper is organized as follows. After discussing related work, we review planning, generalized planning, relational structures, and graphs. Then we introduce faithful and uniform abstractions, look at the notion of isomorphic relational structures (states), and the computation of such abstractions via graph isomorphisms. We conclude with the experiments and summary.

Related Work

Identifying Symmetries. The concept of identifying symmetries is closely related to equivalence relations, where symmetric states are considered equivalent and thus mapped to the same abstract state. Identifying such states helps to prune the search space (Shleyfman et al. 2015), generate abstractions to either solve the problem directly or to construct heuristic functions (Edelkamp 2001; Haslum et al. 2007; Helmert et al. 2014; Nissim, Hoffmann, and Helmert 2011), and transform the underlying representation of the instance (Riddle et al. 2016). However, a common thread among existing approaches, to the best of the authors' knowledge, is that actions are explicitly considered in identifying symmetries. Contrary to these approaches, our approach considers them implicitly. We show that as long as action schemas have a specific structure, explicit consideration is not necessary. For example, the method proposed by Pochter, Zohar, and Rosenschein (2011) uses off-the-shelf algorithms to compute automorphisms on a *Problem Description Graph* (PDG). Notably, this graph contains state variables and ground actions. The PDG has been used in subsequent research (Shleyfman et al. 2015), including lifted versions that encode action schemas rather than ground actions (Sievers et al. 2019). Other approaches use abstract representations to identify symmetries (Sievers et al. 2017), where this representation includes actions. It is worth noting that some symmetry breaking methods consider both the initial state and the goal; however, we are only interested in whether two states are equivalent with respect to the goal.

General policies from logic. The problem of learning general policies has a long history (Khardon 1999; Martín and Geffner 2004; Fern, Yoon, and Givan 2006; Jiménez, Segovia-Aguas, and Jonsson 2019), general policies have been formulated in terms of logic (Srivastava, Immerman, and Zilberstein 2011; Illanes and McIlraith 2019), regression (Boutilier, Reiter, and Price 2001; Wang, Joshi, and Khardon 2008; Sanner and Boutilier 2009), and policy rules (Francès, Bonet, and Geffner 2021; Drexler, Seipp, and Geffner 2022; Yang et al. 2022; Srivastava 2023; Silver et al. 2024).

General policies from neural nets. Deep learning (DL) and deep reinforcement learning (DRL) (Sutton and Barto 1998; Bertsekas 1995; François-Lavet et al. 2018) have been used to learn general policies (Kirk et al. 2023). In some cases, the planning representation of the domain is used (Toyer et al. 2020; Bajpai, Garg et al. 2018; Rivlin, Hazan, and Karpas 2020); in most cases, it is not (Groshev et al. 2018; Chevalier-Boisvert et al. 2019), and in practically all cases, the neural networks are GNNs or variants.

Background

We review basic notions of planning, generalized planning, relational structures, and graphs.

Classical Planning

A **planning problem** is a pair $P = \langle D, I\rangle$ where D is a general first-order *domain* containing a set of predicates (or relations) R, each with given arity, and a set of action schemas of the form $\langle \mathit{pre}, \mathit{eff}\rangle$ where pre is an arbitrary first-order formula and eff is an arbitrary effect, and I is specific instance information that contains the set of objects O, two sets of *ground atoms, Init* and *Goal*, that describe the initial and goal situations, respectively. The problem P defines the state model $S_P^\circ = \langle S, s_I, G, \mathit{Act}, A, f\rangle$ where the states in S are the truth valuations over the ground atoms, where each such valuation is represented by the set of atoms true in the valuation. The function A maps states s into the set $A(s)$ of ground actions from Act that are applicable in s, and the state transition function f maps states s and $a \in A(s)$ into the resulting state $s' = f(s, a)$.

The *unlabeled state model* for the problem P is the tuple $S_P = \langle S, s_I, G, \text{Succ}\rangle$ where the actions are compiled away, and states have a set of possible successor states instead. In this unlabeled model, the first three components are those for S_P°, while $\text{Succ} = \{(s, f(s, a)) \mid a \in A(s)\}$ is the (unlabeled) successor relation.

A trajectory seeded at state s_0 in P is a state sequence $s_0, s_1, \ldots, s_n$ such that (s_i, s_{i+1}) is in Succ, $0 \leq i < n$. A state s is reachable in P if there is a trajectory seeded at the initial state s_I that ends in s. For a reachable state s, a plan from s is a trajectory seeded at s that ends in a goal state.

Generalized Planning

A **generalized planning problem** (Bonet and Geffner 2018) is defined by a class $\mathcal{Q}$ of problems P over a common domain D. A state feature ϕ for $\mathcal{Q}$ is a function that maps the reachable states for the problems in $\mathcal{Q}$ into values. The feature ϕ is said to be Boolean if the values are Boolean, and numerical if the values are non-negative integers. The value of a feature ϕ at state s is denoted by $\phi(s)$. If Φ is a set of features, $\Phi(s)$ denotes the vector of values $\phi(s)$ for $\phi \in \Phi$.

A general policy π for $\mathcal{Q}$ is a relation on state pairs. A trajectory $s_0, s_1, \ldots, s_n$ is a π-trajectory seeded at s_0 if (s_i, s_{i+1}) is a state transition that is in both P and in π, $0 \leq i < n$. The relation π solves s if each maximal π-trajectory seeded at s reaches a goal state, it solves P if it solves the initial state of P, and it solves $\mathcal{Q}$ if it solves each problem P in $\mathcal{Q}$.

In generalized planning, *goals are encoded as part of the state* as follows. For each atom $p(\bar{o})$ that appears in the goal condition G, a new relational symbol p_g of the same arity of p is created. Then, the initial situation I is extended with the atoms $\{p_g(\bar{o}) \mid p(\bar{o}) \in G\}$ which are static and thus remain in every reachable state (Martín and Geffner 2004). Adding these "goal atoms" in the state allows general policies/sketches to take the specific goal of the instance into account. In this way, the resulting policies generalize not just to instances with different numbers of objects and different initial states but also to instances with different goals.

States, Relational Structures, and Graphs

A (planning) state defines a **relational structure** $\mathfrak{A}^s$ with universe $U^s = O$ for the set of objects O in s, and interpretations $R^s \subseteq (U^s)^k$ for each predicate R of arity k in the planning domain D, where $\langle o_1, o_2, \ldots, o_k\rangle \in R^s$ iff $R(o_1, o_2, \ldots, o_k)$ is true in s. The *signature* of a relational structure $\mathfrak{A}$ is the set of relational symbols in $\mathfrak{A}$. We assume fully relational structures that contain no functions nor constants (nullary functions). This type of structures are adequate for planning problems described in PDDL.

While a planning state defines a relational structure, relational structures can be encoded by graphs, a mapping that we will use to test state equivalence. Recall that a directed graph, or graph, is a pair $G = (V, E)$ where V is the set of vertices and $E \subseteq V^2$ is the set of edges. An undirected graph is a directed graph G where E is symmetric; i.e., $(v, w) \in E$ iff $(w, v) \in E$. Two graphs $G = (V, E)$ and $G' = (V', E')$ are **isomorphic**, denoted by $G \simeq_g G'$, if there is a bijection $f : V \to V'$ such that $(u, v) \in E$ iff $(f(u), f(v)) \in E'$.

A *vertex-colored graph* is a tuple $G = (V, E, \lambda)$ where (V, E) is a graph, and $\lambda : V \to \mathcal{C}$ maps vertices to the colors in $\mathcal{C}$. Two vertex-colored graphs $G = (V, E, \lambda)$ and $G' = (V', E', \lambda')$ are **isomorphic**, denoted as $G \simeq_g G'$, iff there is a *color preserving isomorphism* f from G to G', i.e., $\lambda(v) = \lambda'(f(v))$ for $v \in V$. If the graphs G and G' are isomorphic via the bijection f, we write $f : G \to G'$.

Abstractions

We aim at formalizing the notion of equivalence relations $\sim$ on the reachable states for the problems in $\mathcal{Q}$ with the objective of reducing the size of the training sets when learning policies and sketches. If $\mathcal{Q}_\mathcal{T} \subseteq \mathcal{Q}$ is a training set, $\mathcal{Q}_\mathcal{T}/\!\sim$ consists of the *abstract unlabeled state models* induced by $\sim$ for the problems in $\mathcal{Q}_\mathcal{T}$. For a problem P in $\mathcal{Q}_\mathcal{T}$ with unlabeled state model $S_P = \langle S, s_I, G, \text{Succ}\rangle$, the abstract unlabeled state model is the tuple $\tilde{S}_P = \langle \tilde{S}, [s_I], \tilde{G}, \widetilde{\text{Succ}}\rangle$ where

1. $\tilde{S} \doteq \{[s] \mid s \in S\}$ is the set of equivalence classes for P,
2. $[s_I]$ is the equivalence class for initial state s_I of P,
3. $\tilde{G} \doteq \{[s] \mid s \in G\}$ is the set of goal classes, and
4. $\widetilde{\text{Succ}} \doteq \{([s], [s']) \mid (s, s') \in \text{Succ}\}$.

The successor relation in $\tilde{S}_P$ is the *existential quantification* of the successor relation in S_P. Hence, generalized plans that solve $\tilde{S}_P$ do not necessarily solve S_P. For this, we need further conditions:

Definition 1 (Faithful Relations)**.** *Let $\mathcal{Q}$ be a class of planning problems, and let $\sim$ be an equivalence relation over the reachable states in $\mathcal{Q}$. The relation $\sim$ is **faithful** iff*

1. *for any P in $\mathcal{Q}$, any reachable transition (s, s') in P, and any reachable state t in P with $t \sim s$, there is a state t' such that $t' \sim s'$, and (t, t') is a transition in P, and*
2. *if $s \sim t$ for reachable states s and t in P, then s is a goal state iff t is a goal state.*

We write $(s, s') \sim (t, t')$, for two pairs (s, s') and (t, t'), to denote that $s \sim t$ and $s' \sim t'$. A faithful relation is nothing else than a bisimulation among the states in problem P under unlabeled transitions (Sangiorgi 2012).

Theorem 2 (Faithful Abstractions)**.** *Let $\sim$ be a **faithful** relation on the class $\mathcal{Q}$, and let P be a problem in $\mathcal{Q}$. Then, 1) if $s_0, s_1, \ldots, s_n$ is a trajectory in S_P, then $[s_0], [s_1], \ldots, [s_n]$ is a trajectory in $\tilde{S}_P$, and 2) if $[s_0], [s_1], \ldots, [s_n]$ is a trajectory in $\tilde{S}_P$, for each s'_0 in $[s_0]$, there is trajectory $s'_0, s'_1, \ldots, s'_n$ in S_P with $s'_i \sim s_i$ for $0 \leq i \leq n$.*

Proof. The first claim is direct by the definition of $\tilde{S}_P$. For the second, notice that $([s_i], [s_{i+1}])$ in $\widetilde{\text{Succ}}$ implies there is a transition (s''_i, s''_{i+1}) with $(s_i, s_{i+1}) \sim (s''_i, s''_{i+1})$, for $0 \leq i < n$. We construct the required trajectory in S_P inductively. By faithfulness, there is s'_1 such that (s'_0, s'_1) is in Succ and $s'_1 \sim s''_1$. Hence, $s'_1 \sim s_1$. After constructing $s'_0, s'_1, \ldots, s'_k$, we have $s'_k \sim s_k$. By faithfulness, there is transition (s'_k, s'_{k+1}) with $s'_{k+1} \sim s''_{k+1}$. Thus, $s'_{k+1} \sim s_{k+1}$, and the trajectory can be extended with s'_{k+1}. ☐

Corollary 3. *Let $\sim$ be a **faithful** relation on the class $\mathcal{Q}$, and let P be a problem in $\mathcal{Q}$. If s and t are reachable states in P such that $s \sim t$, then $V^*(s) = V^*(t)$.*

Faithfulness allows us to work with the abstraction, but it does not take into account the form of the policy π. Namely, it can be the case that a transition (s, s') in P belongs to π but not a transition (t, t') with $(t, t') \sim (s, s')$. This will not happen, however, for the large class of *uniform policies*:

Definition 4 (Uniform Policies)**.** *Let $\mathcal{Q}$ be a class of planning problems, let $\sim$ be an equivalence relation over the reachable states in $\mathcal{Q}$, and let Π be a class of policies for $\mathcal{Q}$. A policy π in Π is **uniform for** $\sim$ iff for any problem P in $\mathcal{Q}$, and any reachable transition (s, s') in P, if (t, t') is a reachable transition in P and $(s, s') \sim (t, t')$, then (s, s') is in π iff (t, t') is in π. When the relation $\sim$ is clear from context, we just say that π is **uniform.** The class Π of policies is uniform if each policy π in Π is uniform.*

Uniform classes of policies allow us to reduce the training set with the equivalence relation:

Theorem 5 (Solvability)**.** *Let $\mathcal{Q}$ be a class of problems, let $\sim$ be an equivalence relation for the reachable states in $\mathcal{Q}$, and let Π be a class of policies for $\mathcal{Q}$. If $\sim$ is **faithful** and Π is **uniform**, then for any policy π in Π: π solves $\mathcal{Q}$ iff π solves the abstraction $\mathcal{Q}/\!\sim$.*

Proof. Let us assume that π solves $\mathcal{Q}$, and suppose that it does not solve $\mathcal{Q}/\!\sim$. Then, there is a **maximal** trajectory $[s_0], [s_1], \ldots, [s_n]$ seeded at the initial class $[s_0]$ of $\mathcal{Q}/\!\sim$ that is not goal reaching. By Theorem 2, there is a trajectory $s'_0, s'_1, \ldots, s'_n$ in P such that $s'_i \sim s_i$, for $0 \leq i \leq n$. By faithfulness and uniformity, such a trajectory is a maximal π-trajectory. On the other hand, the state s'_n cannot be a goal state since $[s_n]$ is not a goal state. Hence, π cannot solve $\mathcal{Q}$, which contradicts the assumption. The other direction is shown similarly. ☐

A standard way to obtain policies that are uniform for the equivalence relation is to define them in terms of state functions. We say that a policy π for a class $\mathcal{Q}$ is **function-** or **feature-based** if there is a function f that maps reachable states in $\mathcal{Q}$ into a domain Dom_f such that π chooses which transition to take at a state s in problem P in $\mathcal{Q}$ by just looking at the value pairs $\langle f(s), f(s')\rangle$ for the possible state transitions (s, s') from s. In general, if f is invariant under $\sim$, π is automatically uniform. The **rule-based policies** used in symbolic approaches for generalized planning, e.g., (Bonet, Francès, and Geffner 2019), are function-based policies, as they select transitions by just looking at the value pairs $(\Phi(s), \Phi(s'))$ for a fixed set Φ of state features; and also the **GNN-based policies** (Ståhlberg, Bonet, and Geffner 2022a) that greedily select transitions (s, s') by minimizing the state values $V(s')$ of successor states s'. Theorem 5 implies that in these cases, one can partition the states in the training set into equivalence classes, while just keeping one state per partition, without loosing any information. Policies that work for the selected states are guaranteed to work for the pruned states as well. This is elaborated below.

Isomorphic Relational Structures (States)

As we only consider classes $\mathcal{Q}$ over STRIPS domains D, the relation that deems two states s and t equivalent when their relational structures are *isomorphic* is a faithful equivalence relation over states.

Definition 6 (Isomorphic Structures)**.** *Two relational structures $\mathfrak{A}$ and $\mathfrak{B}$, over a common universe U and signature (without constants), are **isomorphic**, written as $\mathfrak{A} \simeq \mathfrak{B}$, iff there is a permutation σ on U such that for each relation R of arity k, $R^{\mathfrak{B}} = \{\sigma(\bar{u}) \mid \bar{u} \in R^{\mathfrak{A}}\}$, where $\sigma(\bar{u})$ for the tuple $\bar{u} = \langle u_1, u_2, \ldots, u_k\rangle$ is the tuple $\langle \sigma(u_1), \sigma(u_2), \ldots, \sigma(u_k)\rangle$. We say that σ maps $\mathfrak{A}$ into $\mathfrak{B}$, and write $\sigma : \mathfrak{A} \to \mathfrak{B}$.*

Isomorphic structures satisfy the same set of sentences and the same set of formulas under suitable permutations. The following is a standard result.

Lemma 7. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two relational structures, and let $\varphi(\bar{x})$ be a first-order formula whose free variables are among the ones in $\bar{x}$. If $\sigma : \mathfrak{A} \to \mathfrak{B}$, then for any tuple $\bar{u}$ of objects of the same length as $\bar{x}$, $\mathfrak{A} \vDash \varphi(\bar{u})$ iff $\mathfrak{B} \vDash \varphi(\sigma(\bar{u}))$. In particular, if φ is a sentence (i.e., it has no free variables), $\mathfrak{A} \vDash \varphi$ iff $\mathfrak{B} \vDash \varphi$.*

We now establish the main result about faithful relations, and the uniformity of feature-based and GNN-based classes of policies.

Theorem 8 (Main)**.** *Let $\mathcal{Q}$ be a class of problems. If $\sim_{iso}$ is the binary relation on the reachable states in $\mathcal{Q}$ given by $s \sim_{iso} t$ iff $\mathfrak{A}^s \simeq \mathfrak{A}^t$, then $\sim_{iso}$ is a **faithful** relation.*

Proof (sketch). That $\sim_{iso}$ is an equivalence relation follows from $\simeq$ being an equivalence relation on structures.

Let P be a problem in $\mathcal{Q}$, let (s, s') be a reachable transition in P, and let t be a reachable state in P with $t \sim_{iso} s$. We need to show that there is a transition (t, t') in P with $t' \sim_{iso} s'$. By assumption, $\sigma : \mathfrak{A}^s \to \mathfrak{A}^t$ for some permutation σ, and there is a ground action $a(\bar{o})$ with $s' = f(s, a(\bar{o}))$. In particular, $\mathfrak{A}^s \vDash pre(\bar{o})$ and thus, by Lemma 7, $\mathfrak{A}^t \vDash pre(\sigma(\bar{o}))$ (i.e. the ground action $a(\sigma(\bar{o}))$ is applicable in t). It is not hard to show that $t' \sim_{iso} s'$ for $t' = f(t, a(\sigma(\bar{o})))$.

For the second condition, let P be a problem in $\mathcal{Q}$. As the states are assumed to contain goal atoms that determine the goal in P, the **sentence** $\varphi_g = \bigwedge_p \forall\bar{x}\big[p_g(\bar{x}) \to p(\bar{x})\big]$, where the conjunction is over all predicates p in D, p_g is the goal predicate for p, and the size of $\bar{x}$ is the arity of p, determines whether a state s in P is a goal state; i.e., s is a goal state iff $\mathfrak{A}^s \vDash \varphi_g$. Hence, if s and t are reachable states in P such that $s \sim_{iso} t$, then $\mathfrak{A}^s \vDash \varphi_g$ iff $\mathfrak{A}^t \vDash \varphi_g$; i.e., s is a goal state iff t is a goal state. □

A state feature ϕ for the class $\mathcal{Q}$ is **first-order definable** (on the set of predicates in the underlying domain D) iff for each value v in Dom_ϕ, there is a sentence $\varphi_{\phi,v}$ such that for any reachable state s in $\mathcal{Q}$, $\phi(s) = v$ iff $\mathfrak{A}^s \vDash \varphi_{\phi,v}$. Therefore, if Π is a class of rule-based policies, where each π in Π is defined over a set $\Phi = \Phi_\pi$ of first-order definable features, then Π is uniform for $\sim_{iso}$, since $s \sim_{iso} t$ for two reachables states s and t implies $\Phi(s) = \Phi(t)$. This is actually the case for the symbolic approaches mentioned above as they work with pools of first-order definable features. Likewise, if the graphs $G(s)$ or $G(s, s')$ induced by states s or transitions (s, s') are preserved under $\sim_{iso}$, the class Π of such policies is uniform for $\sim_{iso}$. This holds for the graphs used in the GNN-based approaches mentioned before.

Computing Abstractions

Checking $\sim_{iso}$ on two reachable states can be reduced to a graph-isomorphism test on vertex-colored graphs that we call *object graphs* and that encode relational structures as vertex-colored undirected graphs.

On the theoretical side, the exact complexity of graph isomorphism is still unknown, but it can be tested in quasi-polynomial time (Babai 2016). However, in practice, the test can be performed efficiently (McKay and Piperno 2014); see discussion in Babai (2016, page 83). Indeed, we use the tool called `nauty` by McKay and Piperno (2014) that computes a **canonical representation** of an input graph which is preserved under graph isomorphism. `nauty` is an state-of-the-art tool that applies Color Refinement, recursively, using a technique called vertex individualization.

Definition 9 (Object Graphs)**.** *Let $\mathfrak{A}$ be a relational structure with universe U, and relational symbols R_i, each of arity k_i, $0 \leq i < n$. The **object graph** for $\mathfrak{A}$ is the **vertex-colored undirected** graph $G(\mathfrak{A}) = (V, E, \lambda)$ where the set V of vertices consists of*

1. *vertices $v = \langle u\rangle$ with color $\lambda(v) = \bot$ for $u \in U$, and*
2. *vertices $v = \langle R_i, j, \bar{u}\rangle$ with color $\lambda(v) = \langle R_i, j\rangle$ for each relation R_i, $1 \leq j \leq k_i$, and tuple $\bar{u} \in (R_i)^{\mathfrak{A}}$.*

The set of edges E consists of

1. *edges connecting the vertices $\langle u_j\rangle$ and $\langle R_i, j, \bar{u}\rangle$ if $\bar{u} = \langle u_1, u_2, \ldots, u_{k_i}\rangle$, and*
2. *edges connecting the vertices $\langle R_i, j, \bar{u}\rangle$ and $\langle R_i, j{+}1, \bar{u}\rangle$ for $1 \leq j < k_i$.*

The vertices of the form $\langle u\rangle$ are called *object vertices*, and vertices of the form $\langle R, j, \bar{u}\rangle$ are called *positional-argument vertices*. The first type of edge connects object vertices to corresponding positional-argument vertices, while the second type of edge connects the j-th with the $(j + 1)$-st positional-argument vertex.

Example. *Consider the Gripper domain, where the goal is to move all balls from room* A *to room* B *with a robot. The robot has two grippers, it can move between the rooms, and it can pick and drop balls with any of the grippers. Figure 1 shows the object graph $G(\mathfrak{A}^s)$ for a state s where there is a*

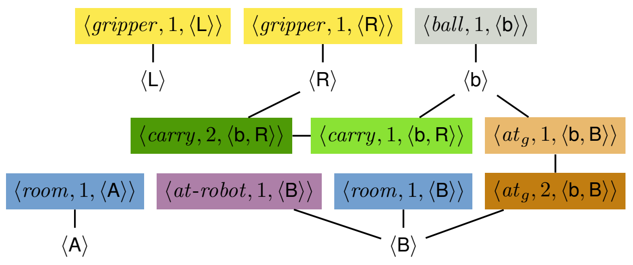

Figure 1: Object graph $G(\mathfrak{A}^s)$ for a state s in Gripper instance containing two grippers L and R, one ball b, and two rooms A and B. In the state s, the robot is at B, the ball is at gripper R, and the goal is for the ball to be room B. The goal is specified in the state using the goal predicate at_g. This graph is isomorphic to the graph $G(\mathfrak{A}^t)$ for a state t that is like s except that the ball is at gripper L.

single ball, the robot is at room B*, and the ball is being held. This graph is isomorphic to $G(\mathfrak{A}^t)$ where the state t is like s, except that the ball is held by the other gripper.* □

Theorem 10 (Reductions)**.** *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two relational structures over a common universe U and signature (with no constant symbols). Then, $\mathfrak{A} \simeq \mathfrak{B}$ iff $G(\mathfrak{A}) \simeq_g G(\mathfrak{B})$.*

Proof (sketch). First assume $\mathfrak{A} \simeq \mathfrak{B}$ with $\sigma : \mathfrak{A} \to \mathfrak{B}$. We construct a color-preserving isomorphism f from $G(\mathfrak{A})$ to $G(\mathfrak{B})$: for object vertices, $f(\langle u \rangle) \doteq \langle \sigma(u) \rangle$, while for positional-argument vertices, $f(\langle R, j, \bar{u} \rangle) \doteq \langle R, j, \sigma(\bar{u}) \rangle$. It can be seen that f is an edge-preserving bijection between the vertices of both graphs. Additionally, $\lambda(\langle u \rangle) = \bot = \lambda(\langle \sigma(u) \rangle)$, and $\lambda(\langle R, j, \bar{u} \rangle) = \langle R, j \rangle = \lambda(\langle R, j, \sigma(\bar{u}) \rangle)$. Hence, f is a color-preserving isomorphism.

For the converse, let us assume that f is a color-preserving isomorphism from $G(\mathfrak{A})$ to $G(\mathfrak{B})$. Consider the function $\sigma : U \to U$ defined by $\sigma(u) = u'$ iff $f(\langle u \rangle) = \langle u' \rangle$. As no object vertex has the color of a positional-argument vertex, σ is a U-permutation. We need to show $\sigma : \mathfrak{A} \to \mathfrak{B}$; i.e., for each relation R,

$$R^{\mathfrak{B}} = \{\sigma(\bar{u}) \mid \bar{u} \in R^{\mathfrak{A}}\}. \qquad (1)$$

The set of vertices related to a tuple $\bar{u}$ in $R^{\mathfrak{A}}$ is $V(\mathfrak{A}, \bar{u}) = \{\langle u_i \rangle \mid u_i \in \bar{u}\} \cup \{\langle R, j, \bar{u} \rangle \mid 1 \leq j \leq k\}$. Such a set induces a subgraph $G(\mathfrak{A}, \bar{u})$ of $G(\mathfrak{A})$. It is not hard to see that (1) holds iff the subgraphs $G(\mathfrak{A}, \bar{u})$ and $G(\mathfrak{B}, \sigma(\bar{u}))$, for all tuples $\bar{u} \in R^{\mathfrak{A}}$, are isomorphic through the (restriction of) f. As this is the case, (1) holds, and $\mathfrak{A} \simeq \mathfrak{B}$. □

By Theorem 10, we can use `nauty` to identify equivalent states. Other state encodings have been proposed that are not aimed at testing structural equivalence but at using standard GNN libraries (Ståhlberg, Bonet, and Geffner 2022a; Chen, Trevizan, and Thiébaux 2023). While the theoretical relationship between GNNs and first-order logics with counting quantifiers $\mathfrak{C}_k$ is known (Grohe 2021), the theoretical capabilities on preserving this relationship for different state encodings (e.g., object graphs) is not clear.

Example. *Let us consider an instance P of Gripper with n balls, 2 grippers, and 2 rooms. It turns out that two states*

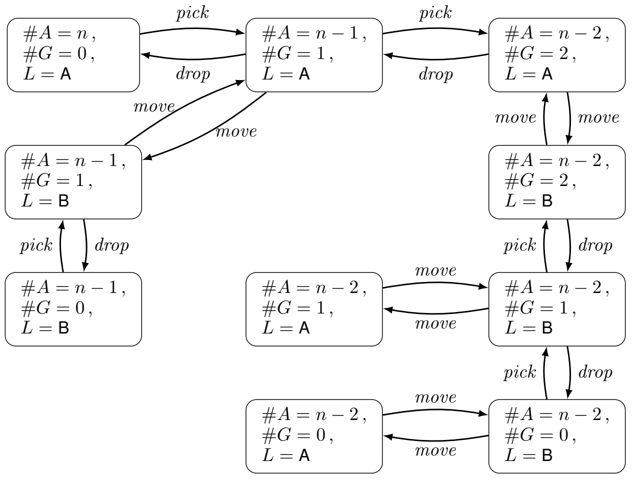
#A = n, #G = 0, L = A
#A = n − 1, #G = 1, L = A
#A = n − 2, #G = 2, L = A
#A = n − 1, #G = 1, L = B
#A = n − 2, #G = 2, L = B
#A = n − 1, #G = 0, L = B
#A = n − 2, #G = 1, L = A
#A = n − 2, #G = 1, L = B
#A = n − 2, #G = 0, L = A
#A = n − 2, #G = 0, L = B
pick
drop
move

Figure 2: Fragment of the state model $\tilde{S}_P$ for a Gripper instance with n balls, 2 grippers, and 2 rooms. Each equivalence class is identified by the number of balls at room A ($\#A$), the number of balls being held ($\#G$), and the position of the robot (L). For better understanding, we label transition with the action schemas that induce them. The abstraction contains $6n$ abstract states (see text).

s and t are equivalent if both have the same number of balls in each room, and the robot is in the same room in each state. Thus, the number of non-isomorphic states is $6n = 2[(n+1) + n + (n-1)]$: for each of the two possible positions of the robot, there are $n+1$ states with no ball being held, n states with 1 ball being held, and $n-1$ states with two balls being held. On the other hand, the (plain) state space contains an exponential number of states: when no ball is being held, for example, each ball and the robot can be in either room, for a total of 2^{n+1} states. Thus, the reduction translates into abstractions for Gripper that are exponentially smaller. Figure 2 shows a fragment of the state model $\tilde{S}_P$ for the abstraction of such an instance P, where each "abstract state" is represented with the features $\Phi = \{\#A, \#G, L\}$ where $\#A$ counts the number of balls in room A*, $\#G$ counts the number of balls being held, and L is the position of the robot, either* A *or* B*. The number of balls in room* B *is determined by the features $\#A$ and $\#G$.* □

Experiments

The testing for the equivalence relation $\sim_{iso}$ is implemented in Python using the graph-isomorphism tool `nauty` (McKay and Piperno 2014), and the planning library Mimir (Ståhlberg 2023). The relation $\sim_{iso}$ is used to reduce the training sets for learning sketches and general policies (i.e., sketches of width zero) within the sketch learning framework of Drexler, Seipp, and Geffner (2022) which is implemented in Clingo (Gebser et al. 2012). The benchmark set consists of different tractable classical planning domains from the International Planning Competition (IPC).

The learning is done on two Intel Xeon Gold 6130 CPUs with 32 cores, 96 GiB of memory, and a time budget of 24

	with equivalence-based reduction				without equivalence-based reduction			
Domain	M	T_{pre}	T_{learn}	$\mathcal{Q}_\mathcal{T}/{\sim_{iso}}$	M	T_{pre}	T_{learn}	$\mathcal{Q}_\mathcal{T}$
Blocks3ops	9	**537**	**6,876**	**4,901**	9	992	71,299	145,680
Blocks4ops-clear	1	**2**	**3**	**86**	1	4	71	30,540
Blocks4ops-on	**2**	**60**	**185**	**249**	3	192	301	30,540
Delivery	**1**	**166**	**290**	**3,346**	3	820	15,355	411,720
Ferry	1	**14**	70	**265**	1	20	**41**	8,430
Gripper	1	4	**2**	**90**	1	**3**	7	1,084
Miconic	1	40	**37**	**17,661**	1	**10**	77	32,400
Reward	1	23	**26**	**7,026**	1	**5**	93	13,394
Spanner	1	3	**3**	**525**	1	3	5	9,291
Visitall	2	1,761	**14,163**	**446,005**	2	**46**	15,487	476,766

Table 1: Learning general policies with and without equivalence-based reductions. The table shows the memory in GiB (M), the wall-clock times in seconds for preprocessing (T_{pre}), grounding, solving the ASPs and validation (T_{learn}), and the total number of states in the training set ($\mathcal{Q}_\mathcal{T}$), and the reduced training set ($\mathcal{Q}_\mathcal{T}/{\sim_{iso}}$). We use boldface to denote the winner in the pairwise comparison, i.e., the one with strictly fewer resources needed.

hours. Since the reductions are often much smaller, we use training instances with up to 10,000 states instead of the 2,000 states previously used (Drexler, Seipp, and Geffner 2022). We tested the generalization of all learned general policies on much larger instances.

Table 1 shows a summary of the times required for preprocessing (that include the tests for $\sim_{iso}$) and learning of the general policies. The sizes of the training and reduced training sets, $\mathcal{Q}_\mathcal{T}$ and $\mathcal{Q}_\mathcal{T}/{\sim_{iso}}$ respectively, are also shown. We remove problems from $\mathcal{Q}_\mathcal{T}/{\sim_{iso}}$ if the initial state is isomorphic to a state in another instance. As it can be seen, the total overhead incurred by testing $\sim_{iso}$ (i.e., the difference between the two figures for T_{pre}) is relatively small, except for Visitall where the preprocessing time increases from 46 to 1,761 seconds because of the additional graph isomorphism test. On the other hand, the learning time often decreases except for Blocks3ops, where it decreases 10-fold from 71,299 to 6,876 seconds, which is caused by the non-determinism in the parallelized Clingo solver and not by symmetry pruning. The method for learning general policies iteratively selects small training instances, with a total number of states often less than 50 to find a general policy that solves all the training instances. The reduction in training data used in the Clingo encoding in such cases is often small. The largest improvement can be seen in Delivery where a significant reduction in training data from 411,720 states (relational structures) to 3,346 abstract states results in a learning speedup from 15,043 to 290 seconds primarily caused by a much faster validation on the abstract states.

Discussion

We have presented a general and principled method for reducing the training sets for learning policies in generalized planning while preserving both information and solutions. This is important because in symbolic approaches for learning provably correct general policies (and also sketches), the scope of the methods is limited by the scalability of the combinatorial solvers. Large instances are needed to obtain policies that generalize, but the solvers cannot deal with large instances optimally. The proposed method is based on the notion of state abstractions that satisfy two key properties: faithfulness and uniformity. It is then shown that state (relational structure) isomorphism delivers abstractions that are faithful and uniform, and that state isomorphism can be computed efficiently by reducing states to vertex-colored graphs and using state-of-the-art codes for testing graph isomorphism. Interestingly, slightly different mapping of planning states into graphs have been used recently for learning planning heuristics using graph neural networks (Chen, Trevizan, and Thiébaux 2023; Chen, Thiébaux, and Trevizan 2023), while (Ståhlberg, Bonet, and Geffner 2022a,b, 2023) learn general policies using *relational* GNNs. The reduction methods proposed in this work can also be used with GNNs. Moreover it can be used with architectures that extend the expressive power of GNNs, like k-GNNs (Ståhlberg, Bonet, and Geffner 2024), as it is well known all these architectures cannot distinguish graphs (states) that are structurally isomorphic (Grohe 2021).

Finally, the abstract state graphs that result from the equivalence reduction and which compile the object names away, represent meaningful structures that can be used to learn general policies more effectively. For example, the Boolean and numerical features introduced in Figure 2 above to identify the states in the resulting abstraction suffice to define a general policy for the domain. This idea can be generalized and may lead to an alternative way of learning the features that support general policies for a given domain; an idea that we would like to explore in the future.

Conclusions

This paper presented a formal method for reducing the training data needed to learn general policies using symmetry reduction based on graph isomorphism. Our general method is applicable to different learning tasks in planning and reinforcement learning. The overall reduction in training data can be exponential, and the resulting abstractions are meaningful on their own. Our experiments show a significant reduction in training data and resulting improvements in learning general policies.

Acknowledgments

The research of H. Geffner has been supported by the Alexander von Humboldt Foundation with funds from the Federal Ministry for Education and Research. The research has also received funding from the European Research Council (ERC), Grant agreement No. No 885107, and Project TAILOR, Grant agreement No. 952215, under EU Horizon 2020 research and innovation programme, the Excellence Strategy of the Federal Government and the NRW Länder, and the Knut and Alice Wallenberg (KAW) Foundation under the WASP program. Resources were also provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) and the Swedish National Infrastructure for Computing (SNIC) at the National Supercomputer Centre at Linköping University partially funded by the Swedish Research Council through grant agreements no. 2022-06725 and no. 2018-05973.

References

Babai, L. 2016. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, 684–697. Association for Computing Machinery.

Bajpai, A. N.; Garg, S.; et al. 2018. Transfer of deep reactive policies for MDP planning. In *Proc. NeurIPS 2018*, 10965–10975.

Belle, V.; and Levesque, H. J. 2016. Foundations for Generalized Planning in Unbounded Stochastic Domains. In *Proc. KR 2016*, 380–389.

Bertsekas, D. P. 1995. *Dynamic Programming and Optimal Control*. Athena Scientific.

Bonet, B.; Francès, G.; and Geffner, H. 2019. Learning Features and Abstract Actions for Computing Generalized Plans. In *Proc. AAAI 2019*, 2703–2710.

Bonet, B.; and Geffner, H. 2018. Features, Projections, and Representation Change for Generalized Planning. In *Proc. IJCAI 2018*, 4667–4673.

Bonet, B.; Palacios, H.; and Geffner, H. 2009. Automatic Derivation of Memoryless Policies and Finite-state Controllers Using Classical Planners. In *Proc. ICAPS 2009*, 34–41.

Boutilier, C.; Reiter, R.; and Price, B. 2001. Symbolic Dynamic Programming for First-Order MDPs. In *Proc. IJCAI 2001*, 690–700.

Celorrio, S. J.; Segovia-Aguas, J.; and Jonsson, A. 2019. A review of generalized planning. *Knowl. Eng. Rev.*, 34: e5.

Chen, D. Z.; Thiébaux, S.; and Trevizan, F. 2023. GOOSE: Learning Domain-Independent Heuristics. In *NeurIPS 2023 Workshop on Generalization in Planning*.

Chen, D. Z.; Trevizan, F.; and Thiébaux, S. 2023. Graph Neural Networks and Graph Kernels For Learning Heuristics: Is there a difference? In *NeurIPS 2023 Workshop on Generalization in Planning*.

Chevalier-Boisvert, M.; Bahdanau, D.; Lahlou, S.; Willems, L.; Saharia, C.; Nguyen, T. H.; and Bengio, Y. 2019. BabyAI: A Platform to Study the Sample Efficiency of Grounded Language Learning. In *Proc. ICLR 2019*.

Drexler, D.; Seipp, J.; and Geffner, H. 2022. Learning Sketches for Decomposing Planning Problems into Subproblems of Bounded Width. In *Proc. ICAPS 2022*, 62–70.

Edelkamp, S. 2001. Planning with Pattern Databases. In *Proc. ECP 2001*, 84–90.

Fern, A.; Yoon, S.; and Givan, R. 2006. Approximate policy iteration with a policy language bias: Solving relational Markov decision processes. *Journal of Artificial Intelligence Research*, 25: 75–118.

Francès, G.; Bonet, B.; and Geffner, H. 2021. Learning General Planning Policies from Small Examples Without Supervision. In *Proc. AAAI 2021*, 11801–11808.

François-Lavet, V.; Henderson, P.; Islam, R.; Bellemare, M. G.; and Pineau, J. 2018. An Introduction to Deep Reinforcement Learning. *Foundations and Trends in Machine Learning*.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool Publishers.

Grohe, M. 2021. The logic of graph neural networks. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 1–17.

Groshev, E.; Goldstein, M.; Tamar, A.; Srivastava, S.; and Abbeel, P. 2018. Learning Generalized Reactive Policies Using Deep Neural Networks. In *Proc. ICAPS 2018*, 408–416.

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-Independent Construction of Pattern Database Heuristics for Cost-Optimal Planning. In *Proc. AAAI 2007*, 1007–1012.

Helmert, M.; Haslum, P.; Hoffmann, J.; and Nissim, R. 2014. Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces. *Journal of the ACM*, 61(3): 16:1–63.

Hu, Y.; and Giacomo, G. D. 2011. Generalized Planning: Synthesizing Plans that Work for Multiple Environments. In *Proc. IJCAI 2011*, 918–923.

Illanes, L.; and McIlraith, S. A. 2019. Generalized Planning via Abstraction: Arbitrary Numbers of Objects. In *Proc. AAAI 2019*, 7610–7618.

Jiménez, S.; Segovia-Aguas, J.; and Jonsson, A. 2019. A Review of Generalized Planning. *The Knowledge Engineering Review*, 34: e5.

Khardon, R. 1999. Learning action strategies for planning domains. *Artificial Intelligence*, 113: 125–148.

Kirk, R.; Zhang, A.; Grefenstette, E.; and Rocktäschel, T. 2023. A Survey of Zero-shot Generalisation in Deep Reinforcement Learning. *Journal of Artificial Intelligence Research*, 76: 201–264.

Martín, M.; and Geffner, H. 2004. Learning Generalized Policies from Planning Examples Using Concept Languages. *Applied Intelligence*, 20(1): 9–19.

McKay, B. D.; and Piperno, A. 2014. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60: 94–112.

Nissim, R.; Hoffmann, J.; and Helmert, M. 2011. Computing Perfect Heuristics in Polynomial Time: On Bisimulation and Merge-and-Shrink Abstraction in Optimal Planning. In *Proc. IJCAI 2011*, 1983–1990.

Pochter, N.; Zohar, A.; and Rosenschein, J. S. 2011. Exploiting Problem Symmetries in State-Based Planners. In *Proc. AAAI 2011*, 1004–1009.

Riddle, P.; Douglas, J.; Barley, M.; and Franco, S. 2016. Improving Performance by Reformulating PDDL into a Bagged Representation. In *ICAPS 2016 Workshop on Heuristics and Search for Domain-independent Planning*, 28–36.

Rivlin, O.; Hazan, T.; and Karpas, E. 2020. Generalized Planning With Deep Reinforcement Learning. In *ICAPS Workshop on Bridging the Gap Between AI Planning and Reinforcement Learning (PRL)*, 16–24.

Sangiorgi, D. 2012. *Introduction to Bisimulation and Coinduction*. Cambridge University Press.

Sanner, S.; and Boutilier, C. 2009. Practical Solution Techniques for First-Order MDPs. *Artificial Intelligence*, 173(5-6): 748–788.

Shleyfman, A.; Katz, M.; Helmert, M.; Sievers, S.; and Wehrle, M. 2015. Heuristics and Symmetries in Classical Planning. In *Proc. AAAI 2015*, 3371–3377.

Sievers, S.; Röger, G.; Wehrle, M.; and Katz, M. 2017. Structural Symmetries of the Lifted Representation of Classical Planning Tasks. In *ICAPS 2017 Workshop on Heuristics and Search for Domain-independent Planning*, 67–74.

Sievers, S.; Röger, G.; Wehrle, M.; and Katz, M. 2019. Theoretical Foundations for Structural Symmetries of Lifted PDDL Tasks. In *Proc. ICAPS 2019*, 446–454.

Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L. P.; and Katz, M. 2024. Generalized Planning in PDDL Domains with Pretrained Large Language Models. In *Proc. AAAI 2024*, 20256–20264.

Srivastava, S. 2022. Hierarchical Decompositions and Termination Analysis for Generalized Planning. *jair*, 77: 1203–1236.

Srivastava, S. 2023. Hierarchical Decompositions and Termination Analysis for Generalized Planning. *Journal of Artificial Intelligence Research*, 77: 1203–1236.

Srivastava, S.; Immerman, N.; and Zilberstein, S. 2008. Learning Generalized Plans Using Abstract Counting. In *Proc. AAAI 2008*, 991–997.

Srivastava, S.; Immerman, N.; and Zilberstein, S. 2011. A new representation and associated algorithms for generalized planning. *Artificial Intelligence*, 175(2): 393–401.

Ståhlberg, S. 2023. Lifted Successor Generation by Maximum Clique Enumeration. In *Proc. ECAI 2023*, 2194–2201.

Ståhlberg, S.; Bonet, B.; and Geffner, H. 2022a. Learning General Optimal Policies with Graph Neural Networks: Expressive Power, Transparency, and Limits. In *Proc. ICAPS 2022*, 629–637.

Ståhlberg, S.; Bonet, B.; and Geffner, H. 2022b. Learning Generalized Policies without Supervision Using GNNs. In *Proc. KR 2022*, 474–483.

Ståhlberg, S.; Bonet, B.; and Geffner, H. 2023. Learning General Policies with Policy Gradient Methods. In *Proc. KR 2023*.

Ståhlberg, S.; Bonet, B.; and Geffner, H. 2024. Learning General Policies for Classical Planning Domains: Getting Beyond C2. arXiv:2403.11734 [cs.AI].

Sutton, R. S.; and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press.

Toyer, S.; Thiébaux, S.; Trevizan, F.; and Xie, L. 2020. ASNets: Deep Learning for Generalised Planning. *Journal of Artificial Intelligence Research*, 68: 1–68.

Wang, C.; Joshi, S.; and Khardon, R. 2008. First Order Decision Diagrams for Relational MDPs. *Journal of Artificial Intelligence Research*, 31: 431–472.

Yang, R.; Silver, T.; Curtis, A.; Lozano-Pérez, T.; and Kaelbling, L. P. 2022. PG3: Policy-Guided Planning for Generalized Policy Generation. In *Proc. IJCAI 2022*, 4686–4692.