Symmetry Detection and Breaking in Cost-Optimal Numeric Planning

Alexander Shleyfman¹, Ryo Kuroiwa², J. Christopher Beck²

¹The Faculty of Industrial Engineering and Management, Technion, Haifa, Israel
²Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada

shleyfman.alexander@gmail.com, ryo.kuroiwa@mail.utoronto.ca, jcb@mie.utoronto.ca

Abstract

One of the main difficulties affecting the use of domain-independent numeric planning is the complexity of the search problem. The exploitation of structural symmetries in a search problem can constitute an effective method of pruning search branches that may lead to exponential improvements in performance. Over a decade now, symmetry breaking techniques have been successfully used within both optimal and satisficing classical planning. In this work, we show that symmetry detection methods applied in classical planning with some effort can be modified to detect symmetries in linear numeric planning. The detected symmetry group, thereafter, can be used almost directly in the A∗-based symmetry breaking algorithms such as DKS and Orbit Space Search. We empirically validate that symmetry pruning can yield a substantial reduction in the search effort, even if algorithms are equipped with a strong heuristic, such as LM-cut.

Introduction

Deterministic planning is the problem of finding a sequence of actions that brings the actor from a given to some desired state. While the formalisms to describe this paradigm may vary, it seems reasonable to assume that richer models can capture finer aspects, and thus, represent the problem with higher fidelity. For example, in classical planning, the variables of the problem are restricted to finite domains, whereas the numeric variant of planning encompasses both continuous and finite variable ranges. Satisficing planners that can manage numeric fluents were designed at the beginning of the century (Hoffmann 2003), yet, it seems that the progress was slowed due to the theoretical undecidability of even simplest, yet still meaningful, numeric formalisms (Helmert 2002). In recent years, however, there seems to have been a surge of interest in planning with numeric fluents, resulting in the development of multiple heuristics for both optimal and satisficing settings (Aldinger and Nebel 2017; Scala et al. 2016; Scala, Haslum, and Thiébaux 2016; Scala et al. 2017; Piacentini et al. 2018; Scala et al. 2020; Kuroiwa et al. 2021). Unfortunately, having a good heuristic is not enough to assemble an efficient planner, e.g., A∗ can expand an exponential number of states even when equipped with an almost perfect heuristic (Helmert and Röger 2008).

Partially to account for this deficit, pruning methods were developed for classical planning (Fox and Long 2002; Coles and Coles 2010; Nissim, Apsel, and Brafman 2012; Wehrle and Helmert 2012; Holte and Burch 2014), and in the past decade, the use of symmetry-based pruning methods has shown its potential within the context of forward search (Pochter, Zohar, and Rosenschein 2011; Domshlak, Katz, and Shleyfman 2013; Wehrle et al. 2015; Gnad et al. 2017). In particular, symmetry reduction methods such as DKS and Orbit Space Search (OSS) were effectively applied in a wide range of classical planning domains, often substantially reducing the expanded state-space size, with a significant increase in planning performance (Domshlak, Katz, and Shleyfman 2012, 2015). In classical planning, symmetry reduction methods compute equivalence classes of states, where the equivalence relation of these classes is based on a precomputed symmetry group. The search exploits these classes by replacing all states in this class with some representative state. Domshlak et al. have shown that given a “path” where each consequent state was replaced by a representative state, one may efficiently reconstruct a corresponding path in the original state space. Hence, the expanded search tree must contain at most one representative of each class at all times.

In this work, we show that the graph-based symmetry detection method proposed by Pochter et al. can be adapted for the numeric setting. We extend the notion of structural symmetries proposed by Shleyfman et al. (2015) to account for linear numeric formulas and demonstrate the equivalence of the obtained symmetries. We also established that from the theoretical standpoint computing these numeric symmetries is not harder than computing the symmetries for classical planning. By grounding these symmetries to the state space level, we enable the use of both DKS and OSS practically as is. Finally, our experimental evaluation demonstrates that in presence of symmetries in the planning task the symmetry breaking algorithms compete favorably with A∗, even if equipped with a strong heuristic such as numeric LM-cut.

Preliminaries

We consider a fragment of numeric planning restricted to the FDR formalism (Bäckström and Klein 1991; Bäckström and Nebel 1995; Helmert 2009) with the addition of numeric state variables, where the conditions and effects on numeric variables are restricted to linear formulas, and unsurprisingly called linear numeric planning task (LT). Formally, LT is de-
defined as a 4-tuple $\Pi = (\mathcal{V}, \mathcal{A}, s_f, G)$, where $\mathcal{V} = \mathcal{V}_p \cup \mathcal{V}_n$, with $\mathcal{V}_n$ being a finite set of propositional variable, where each variable $v \in \mathcal{V}_n$ is associated with a finite domain of values $\mathcal{D}(v)$, and $\mathcal{V}_p$ is a set of numeric variables. Numeric variables $v \in \mathcal{V}_n$ have rational values, i.e., $\mathcal{D}(v) = \mathbb{Q}$. If $\Pi$ does not have any numeric state variable ($\mathcal{V}_n = \emptyset$), we have a classical (FDR) planning task $\Pi_{FDR}$. Assignment of a valid value $d$ to a variable $v \in \mathcal{V}$ is called a fact and denoted by $(v, d)$. For a subset of variables $V \subseteq \mathcal{V}$ we define its joint domain to be $\mathcal{D}[V] = \times_{v \in V} \mathcal{D}(v)$. The state space is $\mathcal{S} = \mathcal{D}[\mathcal{V}]$. A state $s \in \mathcal{S}$ is a full assignment over all variables, and can be seen as a tuple $(s_p, s_n)$, where $s_p \in \mathcal{D}[\mathcal{V}_p]$ and $s_n \in \mathcal{D}[\mathcal{V}_n]$; $s[v]$ indicates the value of the variable $v$ over the state $s$. $s_f$ is a state. Note that $s, s_p, s_n$ may be presented via vector representation, or as a set of facts $s = s_p \cup s_n$. In the latter case, there are no two facts that involve the same variable, and $|s| = |\mathcal{V}|$. The value of a variable $v$ in $s$ is given by $s[v] = d$, and is equivalent to $(v, d) \in s$. A partial state is a subset $s^{pt} \subseteq s$ of some $s \in \mathcal{S}$.

A linear expression $\xi$ has the form $\xi = \sum_{v \in \mathcal{V}} w_v^x v + w_0^x$, where $V \subseteq \mathcal{V}_n$, with $\forall v \in V, w_v^x \in \mathbb{Q}$, and $w_0^x \in \mathbb{Q}$. The value of $\xi$ in a state $s$ is given by the expression $s[\xi] = \sum_{v \in \mathcal{V}} w_v^x s[v] + w_0^x$. For simplicity, we assume that there is always a variable $v_0 \in \mathcal{V}_n$ such that for each state $s$ it holds $s[v_0] = 1$. This assumption allows a more convenient expression $\xi = \sum_{v \in \mathcal{V}} w_v^x v$. We assume that there are no redundant factors in the condition representation, i.e., for all $v \in V$ it holds $w_v^x \neq 0$. $\Xi$ is the set of all linear expressions in $\Pi$. The set of all constants that appear in $\xi$ is denoted by $\mathit{num}(\xi)$, and the set of all variables that appear in $\xi$ is denoted by $\mathit{vars}(\xi)$. Similarly, for a partial state $s^{pt}$ we denote by $\mathit{vars}(s^{pt}) \subseteq \mathcal{V}$ the variables involved in $s^{pt}$.

Conditions can be either propositional or numeric. A propositional condition $\psi$ is a partial state over the variables $\mathit{vars}(\psi) \subseteq \mathcal{V}_p$. We say that $\psi$ is satisfied by $s$, $s \models \psi$, if $\psi \subseteq s_p$. A propositional condition $\psi$ has the form $\xi \geq 0$, where $\xi$ is a linear expression. In this case, we say that $\psi$ is satisfied by $s$ if $s[\xi] \geq 0$ holds. The set of conditions $\Psi$ is satisfied by $s$, if for each $\psi \in \Psi$ it holds that $s \models \psi$. The goal condition $G = G_p \cup G_n$ is a union over sets of propositional and numeric conditions, respectively.

Action $a = (\mathit{pre}(a), \mathit{eff}(a), \mathit{cost}(a)) \in \mathcal{A}$ has preconditions $\mathit{pre}(a) = \mathit{pre}_p(a) \cup \mathit{pre}_n(a)$, effects $\mathit{eff}(a) = \mathit{eff}_p(a) \cup \mathit{eff}_n(a)$, and cost $\mathit{cost}(a) \in \mathbb{R}^{>0}$. $\mathit{pre}_p(a)$ is a partial state over propositional variables and $\mathit{pre}_n(a)$ is a set of linear numeric conditions. The set of all numeric conditions of the task is denoted by $\Psi_n$, i.e., $\Psi_n = G_n \cup \bigcup_{a \in \mathcal{A}} \mathit{pre}_n(a)$. $a$ is applicable to $s$ if $s \models \mathit{pre}(a)$. Similarly, the propositional effect $\mathit{eff}_p(a)$ is a partial state on a subset of $\mathcal{V}_p$. The effect $\mathit{eff}_n(a)$ is a set of numeric effects of the form $(v \leftarrow \xi)$, where $v \in \mathcal{V}_n$ and the value of $\xi \in \Xi$. We assume that assignment effect $v := \xi$ and subtractive effect $v := \xi + c$ are normalized to the additive forms $v := \xi - v$ and $v := \xi - c$, and one action has at most one effect on the same numeric variable. The result of applying $a$ in $s$ is denoted by $s[a] = s'_{p} \cup s'_{n}$, where the resulting state is defined as $s'_{p}[v] = \mathit{eff}_p(a)[v]$ for $v \in \mathit{vars}(\mathit{eff}_p(a))$, $s'_{n}[v] = s[v] + \xi[s]$ if $(v \leftarrow \xi) \in \mathit{eff}_n(a)$, and $s'[a][v] = s[v]$ otherwise. $\mathcal{A}(s)$ is the set of all actions applicable to $s$.

An $s$-plan is an action sequence $\pi$ that can be applied successively in $s$ and results in a goal state $s_f \models G$. A plan for $\Pi$ is an $s_f$-plan. The cost of an $s$-plan $\pi$ is the sum of all its action costs and an optimal $s$-plan has minimal cost among all possible $s$-plans.

A state transition graph is a labeled digraph $\Pi = (S, E)$, whose vertexes $S$ are the states of $\Pi$, the set of labeled arcs $E = \{(s, s[a]; a) \mid s \in S, a \in \mathcal{A}(s)\}$ is induced by the actions of $\Pi$, where $\mathit{cost}(s, s[a]; a) = \mathit{cost}(a)$. A plan for $\Pi$ is equivalent to a path from $s_f$ to $s_a$ in $\Pi$. 

### Structural Symmetries and PDG

This subsection defines the notion of structural symmetries (Shleynman et al. 2015), which captures previously proposed concepts of symmetries in classical planning. In short, structural symmetries relabel a given planning task. Variables are mapped to variables, values to values (preserving the $\langle \mathit{var}, \mathit{val}\rangle$ structure), and actions are mapped to actions. In this work, we follow the definition of structural symmetry for FDR planning tasks as defined by Wehrle et al. (2015). For a planning task $\Pi_{FDR} = (\mathcal{V}, \mathcal{A}, I, G)$, let $P$ be the set of $\Pi$’s facts, and let $P_{col} := \{(v, d) \mid d \in \mathcal{D}(v)\} \setminus V$ be the set of sets of facts attributed to each variable in $\mathcal{V}$. We say that a permutation $\sigma : P \cup \mathcal{A} \rightarrow P \cup \mathcal{A}$ is a structural symmetry if the following holds:

1. $\mathit{cost}(P_{\sigma}) = P_{\sigma}$,
2. $\sigma(\mathcal{A}) = \mathcal{A}$, and, for all $a \in \mathcal{A}$, $\sigma(\mathit{pre}(a)) = \mathit{pre}(\sigma(a))$,
3. $\sigma(\mathit{eff}(a)) = \mathit{eff}(\sigma(a))$, and $\mathit{cost}(\sigma(a)) = \mathit{cost}(a)$.

$\sigma(G) = G$.

We define the application of $\sigma$ to a set $X$ by $\sigma(X) := \{\sigma(x) \mid x \in X\}$, where $\sigma$ is applied recursively up to the level of action labels and facts. For example, let a be a partial state, since $a$ can be represented as a set of facts, applying $\sigma$ to $a$ results in a partial state $a'$, s.t. for all facts $(a, d) \in a$ it holds that $\sigma(a, d') \in a'$ and $a'[\sigma(v)] = d'$. This implies that $\sigma$ uniquely defines the values of each variable, i.e., if $d \in \mathcal{D}(v)$, we can write $\sigma(d) = d'$, where $d' \in \mathcal{D}(\sigma(v))$. Note that the set of all structural symmetries of an FDR task is a finite set of bijections closed under composition. Thus, we have that structural symmetries form a group over the task $\Pi_{FDR}$, denoted by $\mathit{Aut}(\Pi_{FDR})$.

### Symmetries and Problem Description Graphs

The problem description graph (PDG) was introduced by Pochter et al. (2011), and later on reformulated by Domshlak et al. (2012), and Shleynman et al. (2015).

**Definition 1.** Let $\Pi_{FDR}$ be a FDR planning task. The problem description graph $\mathit{PDG}_{\Pi_{FDR}}$ is the colored digraph $(N, E, \mathit{col})$ with nodes

$$N = \mathcal{V}_n \cup \bigcup_{v \in \mathcal{V}} \mathcal{D}(v) \cup \mathcal{A},$$

where $\mathcal{V}_n = \{v \mid v \in \mathcal{V}\}$, $\mathcal{D}(v) = \{d \mid d \in \mathcal{D}(v)\}$, and...
\[ N_A = \{n_a \mid a \in A\}; \text{ node colors} \]

\[
\text{col}(n) = \begin{cases} 
0 & \text{if } n \in N_V \\
1 & \text{if } n^d_v \in \bigcup_{v \in V} N_{D(v)} \land \langle v, d \rangle \in G \\
2 & \text{if } n^d_v \in \bigcup_{v \in V} N_{D(v)} \land \langle v, d \rangle \not\in G \\
3 + \text{cost}(a) & \text{if } n_a \in N_A
\end{cases}
\]

and edges

\[
E = \bigcup_{v \in V} E_v \cup \bigcup_{a \in A} E_{a}^{\text{pre}} \cup E_{a}^{\text{eff}},
\]

where \( E_v = \{\langle n_v, n^d_v \mid d \in D(v)\}, E_{a}^{\text{pre}} = \{\langle n_a, n^d_v \mid \langle v, d \rangle \in \text{pre}(a)\}, \text{ and } E_{a}^{\text{eff}} = \{\langle n_v, n_a \mid \langle v, d \rangle \in \text{eff}(a)\}. \)

In their work, Pochter et al. observed that PDG symmetry is a symmetry of \( \mathcal{T}_\Pi \) that is induced by a graph automorphism of the PDG of \( \Pi_{FDR} \).

\[ \text{symmetry of } \Pi_{FDR} \]

they induce the same transition graph symmetry, i.e., the isomorphism. When the groups \( \text{Aut}(PDG(\Pi_{FDR})) \) and \( \text{Aut}(\Pi_{FDR}) \) are isomorphic, namely \( \text{Aut}(PDG(\Pi_{FDR})) = \text{Aut}(\Pi_{FDR}) \).

Group homomorphism \( f \) is a function between two groups \( f : \Gamma \rightarrow \hat{\Gamma} \) that respects the group operations. i.e., given \( \sigma, \sigma' \in \Gamma \) it holds \( f(\sigma \circ \sigma') = f(\sigma) \circ f(\sigma') \). Isomorphism is a bijective (one-to-one and onto) homomorphism. Note that all isomorphisms are invertible, where the inverse is also an isomorphism. When the groups \( \Gamma \) and \( \hat{\Gamma} \) are isomorphic, we write \( \Gamma = \hat{\Gamma} \). If homomorphism \( f : \Gamma \rightarrow \hat{\Gamma} \) is injective (one-to-one) we say that \( \Gamma \) is a subgroup of \( \hat{\Gamma} \) and write \( \Gamma \leq \hat{\Gamma} \).

Symmetries in Numeric Domains

Shleymfan and Jonnson (2021) show that determining whether two states are symmetric in a transition system induced by an FDR planning task is PSPACE-hard. This result automatically grants us that PSPACE-hardness also holds for numeric planning, since it, trivially, contains the FDR formalism. What is more troublesome, is the fact that the presence of numeric fluents makes the search space infinite, which may lead to a lot of unpleasant properties of its symmetry group. Consider, for example, that the permutation group of a countable number of elements, contains an uncountable number of symmetries, infinitely many of which have infinite order.

Since identifying the whole symmetry group of the transition task is infeasible, we would like to obtain a manageable subgroup and use it for symmetry breaking in the search over the state space. To this end, we extend the Structural Symmetries and the PDG by numeric fluents. Then, we show that Numeric Structural Symmetries (NSS) can be mapped into symmetries of the state transition graph (Thm. 2), and that the symmetry group of the numeric version of PDG is isomorphic to the NSS group of the task (Thm. 3). For the flowchart of the proof of the grounding process see Figure 1.

We exploit the grounded, effectively computed subgroup \( \Gamma_{\Pi} \) of \( \text{Aut}(\mathcal{T}_\Pi) \) by plugging it into the symmetry breaking searches DKS (Domshlak, Katz, and Shleymfan 2012), OSS (Domshlak, Katz, and Shleymfan 2015).

Let us give here a motivational example. Suppose we have a planning task where homogeneous trucks deliver various cargo across a city, where the city map is represented via digraph. A truck \( T \) in this task is represented with three variables: \( \text{loc}(T) \) the locations of the truck (finite domain variable), \( \text{fuel}(T) \) the amount of fuel in the truck, and \( \text{load}(T) \) the current load of the truck (both numeric variables). Since the trucks are homogeneous, we would like to have a symmetry \( \sigma \) to capture this information, i.e., given two trucks \( T_1 \) and \( T_2 \) a map that replaces only the labels of the trucks should induce an automorphism of the transition graph. In what follow we would like to capture this behavior. Instead of saying that \( T_1 \) is symmetric to \( T_2 \), we want a bijection that switches between the variables associated with \( T_1 \) and \( T_2 \), as follows: \( \sigma(\text{loc}(T_1)) = \text{loc}(T_2) \), \( \sigma(\text{fuel}(T_1)) = \text{fuel}(T_2) \), \( \sigma(\text{load}(T_1)) = \text{load}(T_2) \), and vise versa.

\[ N_A = \{n_a \mid a \in A\}; \text{ node colors} \]

\[
\text{col}(n) = \begin{cases} 
0 & \text{if } n \in N_V \\
1 & \text{if } n^d_v \in \bigcup_{v \in V} N_{D(v)} \land \langle v, d \rangle \in G \\
2 & \text{if } n^d_v \in \bigcup_{v \in V} N_{D(v)} \land \langle v, d \rangle \not\in G \\
3 + \text{cost}(a) & \text{if } n_a \in N_A
\end{cases}
\]

\[ E = \bigcup_{v \in V} E_v \cup \bigcup_{a \in A} E_{a}^{\text{pre}} \cup E_{a}^{\text{eff}},\]

where \( E_v = \{\langle n_v, n^d_v \mid d \in D(v)\}, E_{a}^{\text{pre}} = \{\langle n_a, n^d_v \mid \langle v, d \rangle \in \text{pre}(a)\}, \text{ and } E_{a}^{\text{eff}} = \{\langle n_v, n_a \mid \langle v, d \rangle \in \text{eff}(a)\}. \)
Figure 1: Schematic representation of detection and grounding of structural symmetries for linear numeric planning.

**Numeric Structural Symmetries**

For an LT $\Pi = \langle V, A, I, G \rangle$, we say that a permutation $\sigma$ over $V \cup A \cup (\bigcup_{v \in V_p} D(v))$ is a numeric structural symmetry (NSS) if the following holds:\footnote{Example of disjoint union: $\{5\} \cup \{5\} = \{(5, 0), (5, 1)\}$}:

1. $\sigma(V_p) = V_p$, $\sigma(V_n) = V_n$.
2. For all $v \in V_p$ holds $\sigma(D(v)) = D(\sigma(v))$.
3. $\sigma(A) = A$.

We define an application of $\sigma$ to a partial state $s^{pt}$ over the propositional variables $vars(s^{pt})$, as $\sigma(s^{pt})[v] := \sigma(s^{pt}[v])$ for all $v \in vars(s^{pt})$. Note that by points 1 and 2 the result of this application, $\sigma(s^{pt})$, is also a partial state.

Let $\xi = \sum_{v \in V} w_v^\xi \cdot v$ be a linear formula over the variables $V \subseteq V_n$. We define the application of $\sigma$ to $\xi$ to be $\sigma(\xi) = \sum_{v \in V} w_v^\xi \cdot \sigma(v)$. The application of $\sigma$ to numeric effect $v \mapsto \xi$ is written as $\sigma(v) \mapsto \sigma(\xi)$, where $v \in V_n$. Using these notations we establish the following conditions on $\sigma$:

4. For all $a \in A$, $\sigma(\text{pre}_p(a)) = \text{pre}_p(\sigma(a))$, $\sigma(\text{pre}_n(a)) = \text{pre}_n(\sigma(a))$, $\sigma(\text{eff}_p(a)) = \text{eff}_p(\sigma(a))$, $\sigma(\text{eff}_n(a)) = \text{eff}_n(\sigma(a))$, and $\sigma(\text{cost}(a)) = \text{cost}(a)$.
5. $\sigma(G_n) = G_n$ and $\sigma(G_p) = G_p$.

Note that immediate consequence of this definition is that for any structural symmetry $\sigma$, it holds that $\sigma(\Xi) = \Xi$. Note also that that for two structural symmetries $\sigma_1, \sigma_2$ the composition $\sigma_1 \circ \sigma_2$ is also a structural symmetry.

- This is straightforward for properties 1, 3, and 5: $\sigma_1 \circ \sigma_2(X) = \sigma_1(\sigma_2(X)) = \sigma_1(X) = X \text{ where } X \in \{V_n, V_p, A, G_n, G_p\}$.
- Property 4 holds for each $a \in A$ since for each $h \in \{\text{pre}_p, \text{pre}_n, \text{eff}_n, \text{eff}_p\}$ we have $\sigma_1 \circ \sigma_2(h(a)) = \sigma_1(\sigma_2(h(a))) = \sigma_2(\sigma_1(h(a))) = h(\sigma_1(\sigma_2(a))) = h(h_1(\sigma_2(a))) = h_1(h_2(a)) = h(\sigma_1 \circ \sigma_2(a))$. By replacing $a$ by $v \in V_p$ and $h$ by $D$, we get property 2. The check for cost is trivial and repeats the previous point.

A well-known result in group theory (Herstein 1975) states that a finite set closed under an operation forms a group. Here we denote the group of all structural symmetries as $\Aut(\Pi)$. This definition leads us to the observation below.

**Theorem 2.** There is a natural injection from $\Aut(\Pi)$ to $\Aut(\Pi')$, i.e., for each $\sigma \in \Aut(\Pi)$ there is a unique $\tilde{\sigma} \in \Aut(\Pi')$.

**Proof.** Let $\Pi' = \langle S, E \rangle$ be a transition system induced by a LT $\Pi = \langle V, A, s_0, G \rangle$, and let $\sigma \in \Aut(\Pi)$ be a structural symmetry. For a state $s = s_p \cup s_n \in S$ we define the application of $\sigma$ to $s$ as $\tilde{\sigma}(s_p \cup s_n) = \tilde{\sigma}(s_p) \cup \tilde{\sigma}(s_n)$, where

$\tilde{\sigma}(s_p) = \{\sigma((v, d_v)) \mid (v, d_v) \in s_p\}$

and $\tilde{\sigma}(s_n) = \{(v, q_v) \mid (v, q_v) \in s_n\}$ and $\tilde{\sigma}(a) = \sigma(a)$ for $a \in A$. By properties 1-3, we have that $\tilde{\sigma}$ is a permutation over $S \cup A$, and the map $\sigma \mapsto \tilde{\sigma}$ is injective.

Let $(s, s'; a) \in E$, and let $\sigma$ be a structural symmetry. We aim to show that $\tilde{\sigma}(s), \tilde{\sigma}(s'; \tilde{\sigma}(a)) \in E$. Let us decouple the states $s$ and $s'$ to their propositional and numeric components $s_p \cup s_n$ and $s'_p \cup s'_n$, respectively. The proof of the propositional part was already done by Shlyefman et al. (2020), but we present it here for the sake of completeness. Note that by definition of $\tilde{\sigma}$ we have $\tilde{\sigma}(s_p) = \sigma(s_p)$. Let $\psi$ be a partial assignment over the propositional variables such that $s_p \models \psi$, note that set wise it can be written as $v \models s_p$. By applying $\sigma$ to both sides we have $\sigma(s_p) \models \sigma(\psi)$, thus $\sigma(s) \models \sigma(\psi)$. This claim grants us: $s \models p \Rightarrow \sigma(s) \models \sigma(p)$ implies that $\tilde{\sigma}(s) \models \text{pre}_p(\sigma(a))$, and $s' \models \text{eff}_p(\sigma(a))$ implies that $\tilde{\sigma}(s') \models \text{eff}_p(\sigma(a))$. Since $\sigma$ is a permutation over $V_p$, its application to the unaffected variables $V_p \setminus vars(\text{eff}_p(\sigma(a)))$ can be written as $V_p \setminus vars(\text{eff}_p(\sigma(a)))$, and $s' \models \psi$ for each $v \in V_p \setminus vars(\text{eff}_p(\sigma(a)))$ we have that $\tilde{\sigma}(s') \models \sigma(s') \models \overline{\sigma}(s') \models \sigma(s') \models \overline{\sigma}(s')$ for each $v \in V_p \setminus vars(\text{eff}_p(\sigma(a)))$, resulting in $\tilde{\sigma}(s) \models \overline{\sigma}(a)$. Note, that the fact that for any partial assignment $\psi$, it holds $s_p \models \psi \iff \tilde{\sigma}(s) \models \sigma(\psi)$ which directly implies that $s_p \models G_p \iff \tilde{\sigma}(s) \models G_p$.

We aim to show that $\tilde{\sigma}(s) \models \sigma(a)$ and $\tilde{\sigma}(s') \models \sigma(a)$. Let $\psi \in \Pi_n$ be a numeric condition $\psi := \sum_{v \in V} w_v^\psi \cdot v \psi \geq 0$ such that $s_n \models \psi$. By definition of $\tilde{\sigma}$ we have $s[\psi] = \tilde{\sigma}(s)[\sigma(\psi)]$, thus

$\sum_{v \in V} s[v] \cdot w_v^\psi = \sum_{v \in V} \tilde{\sigma}(s)[\sigma(\psi)] \cdot w_v^\psi \geq 0$ implies that $s_n \models \psi \iff \tilde{\sigma}(s) \models \sigma(\psi)$.

This statement combined with the previous paragraph has two immediate consequences:

1. $s \models G \iff \tilde{\sigma}(s) \models G$, and
2. $s \models pre(a) \iff \tilde{\sigma}(s) \models pre(\sigma(a))$.

Now, to show that $\tilde{\sigma}(s), \tilde{\sigma}(s'; \tilde{\sigma}(a)) \in E$, it is enough ensure that for any $v \in V_n$ it holds that $s'[v] = \tilde{\sigma}(s')[\sigma(v)]$. Here we have two cases, either $v$ is affected by $\text{eff}_n(a)$ or not. If $v$ is not affected, i.e., $v \not\in vars(\text{eff}_n(\sigma(a)))$, then since $\sigma$ is a permutation it holds that $\sigma(v) \not\models \tilde{\sigma}(s)[\sigma(\psi)]$. Otherwise, let $v = \xi$
be the numeric effect of $a$ on $v$. Let $\xi = \sum_{v' \in V} w_{\xi} v'$. By definition we have that $\sigma(v) = \sigma(\xi) \in \text{eff}_a(\sigma(a))$. Thus,$$
s'=[s][v] + \sum_{v' \in V} w_{\xi} \tilde{\sigma}(s)(\sigma(v)) + \sum_{v' \in V} w_{\xi} \tilde{\sigma}(s)(\sigma(v')) = \tilde{\sigma}(s)(\sigma(v)),
$$since $s[v] = \tilde{\sigma}(s)(\sigma(v))$ for each $v \in V_n$ by definition of $\tilde{\sigma}$.

Hence, we have that $\sigma$ is the symmetry of the transition graph $\mathcal{T}_\Pi$, since it satisfies the following three requirements:

- $(s, s'; a) \in E$ iff $(\sigma(s), \sigma(s'); \sigma(a)) \in E$,
- cost($\sigma(a)$) = cost($a$), and
- $s$ is a state goal iff $\sigma(s)$ is a state goal.

This result establishes that structural symmetries induce transition graph symmetries. Next, we show how to compute these symmetries using a variation of the problem description graph modified for numerical planning.

**Numerical Problem Description Graph**

As the state transition graph $\mathcal{T}_\Pi$ of a planning task I is usually too large to be given explicitly, symmetries must be inferred from a compact description. Pochter et al. introduced a method for deducing some symmetries of the planning task from automorphisms of a certain graphical structure, the *problem description graph* (PDG) of II. Later, Domshlak et al. slightly modified the definition, mainly to add support for general-cost actions. As observed by Pochter et al., every automorphism of the PDG of II induces an automorphism of $\mathcal{T}_\Pi$, and the former can be found using off-the-shelf tools for the discovery of automorphisms in explicit graphs, such as *bliss* (Junttila and Kaski 2007).

Note that in contrast to the propositional FDR the transition system defined by its numeric counterpart may be infinite. We group numeric elements of conditions and effects into a set of linear formulas. Recall, that $\Xi$ denotes the set of all linear formulas that appear in II (both in conditions and effects). For each $v \in V_n$, we define $W(v) = \{w_{\xi}^v \in \mathbb{Q} \mid \exists \xi \in \Xi : w_{\xi}^v \in \text{nums}(|\xi|)\}$, the set of all numeric coefficients associated with $v$. Let $C_\Pi$ be the set of unique costs of actions and constants in numeric variables in the task II, i.e.,

$$C_\Pi = \{\text{cost}(a) \mid a \in A\} \cup \bigcup_{v \in V_n} W(v).$$

We define a function $\text{ord} : C_\Pi \rightarrow \mathbb{Z}$ to be an order preserving function with respect to the lexicographic order over the set $C_\Pi$, i.e., for $c_1, c_2 \in C_\Pi$ holds that $c_1 \leq c_2$ implies that $\text{ord}(c_1) \leq \text{ord}(c_2)$. The main properties of the function $\text{ord}$ that we are interested in are

1. for $c_1, c_2 \in C_\Pi$ it holds $c_1 \leq c_2$ iff $\text{ord}(c_1) = \text{ord}(c_2)$.
2. for each $c \in C_\Pi$ it holds that $\text{ord}(c) \leq |C_\Pi|_n$.

In the following definition, the function $\text{ord}$ allows us to distinguish between the task constants while operating within a reasonable number of vertex colors.

**Definition 2.** Let $\Pi = \langle V, A, s_I, G \rangle$ be a LT. The **numerical problem description graph** (NPDG$_\Pi$) of II is the colored digraph $(N, E, \text{col})$ with nodes (e.g., Figure 2)
least one neighbor of color 0 and an in-degree that is at least one. All other sets differ due to different colors. Note also that since α is edge preserving we have that if α(n_v) = n_u then α(N_D(v)) = N_D(u). Vertices in N_v must differ from the ones in N_u under α, since each n in N_v has at least one outgoing neighbor of color 0, and the vertices in N_u have none of these. Hence, we have that f(α) obeys the properties 1-3 of the definition of NSS.

Property 5 is preserved also due to edge preservation and the fact that α(n_G) = n_G. Note that if (v,d) ∈ G_p then, by construction of NPDG_Π, we have that (n_v^+, n_G) ∈ E, thus (α(n_v^+), α(n_G)) = n_G ∈ E, which means that f(α)((v,d)) ∈ G_p. Now, let ψ : Ξ ≥ 0 ∈ G_n. Then, (n_ξ^+, n_G) ∈ E, and once again (α(n_ξ^+), n_G) ∈ E. Note that for each v ∈ vars(Ξ) there is w_v^+ ∈ nums(Ξ), and by construction of NPDG_Π there are edges (n_v, n_w^+, v_ξ, n_ξ^+). Since α is edge preserving, we have the path (α(n_v), α(n_w^+), α(n_ξ^+), α(n_ξ^+), α(n_G)) ∈ E, thus the summand w_v^+ f(α)(v) is a part of the expression f(α)(ξ) ≥ 0, since α(n_w^+) and n_ξ^+ must be of the same color, v and f(α)(v) have the same multiple in Ξ and f(α)(ξ), respectively. And since α is in-degree preserving, the number of summands in Ξ is equal to the number of summands in f(α)(ξ). Hence, f(α)(ξ) is well-defined in terms of NSS, and f(α)(ξ) ≥ 0 ∈ G. Therefore, f(α)(G) ⊆ G. To show the converse we need to show that α^{-1} ∈ Aut(NPDG_Π) and f is a homomorphism.

To prove property 4, we need to show that f(α)(h(α)) = h(f(α)(a)) for each h ∈ {pre, pre}, eff, eff, cost}. This property on preconditions is obtained exactly as Property 5, where the node n_G is replaced with the nodes n_a and n_{f(α)(a)}. The proof for effects is similar, but requires us to look at the nodes n_ξ^+, where the only outgoing edge is pointed towards a node that represents a numeric variable.

We omit the full proof here, since it repeats almost verbatim the one given in the previous paragraph.

For the injectivity of f, consider α, β ∈ Aut(NPDG_Π):

\[ f(α) = f(β) \implies l^{-1} α l = l^{-1} β l \implies α(n) = β(n) \text{ for all } n \in N_V \cup N_A \cup \bigcup_{v \in V_p} N_D(v), \]

where the last implication follows from the fact that l is bijective. To show that f is injective, we need to show that α(n) = β(n) for all n ∈ N. Thus, assume in contradiction that there is n ∈ N such that α(n) ≠ β(n). These n is either n_G or lies in N_G ∪ \bigcup_{v} N_D(v). Since n_G has its own color, it is a fixed under permutations in Aut(NPDG_Π). Since f is well-defined we have that f(α), f(β) ∈ Aut(Π), thus for n_w ∈ \bigcup_{v} N_D(v) we have

\[ α(n_w) = n_w = f(α)(n_w) = f(β)(n_w), \text{ and } \]

\[ α(n_ξ^+) = n_ξ^+ = f(α)(n_ξ^+) = f(β)(n_ξ^+) \text{ for } n_ξ^+ ∈ N_Ξ, \]

Lastly, to show that f is surjective, we need to prove that f^{-1}(σ) is in Aut(NPDG_Π), when σ is an NSS. We extend this inverse as follows

\[ f^{-1}(σ)(n) = \begin{cases} n_{σ(x)} & \text{if } n = n_x \in N_V \cup N_A \cup N_A \\ n_{σ(x,y)} & \text{if } n = n_y \in N_V \cup N_A \cup N_A \\ n_{σ(x)} & \text{if } n = n_{w}^+ \in \bigcup_{v} N_V, \\ n_{σ(ξ)} & \text{if } n = n_ξ^+ \in N_Ξ \wedge ξ \in \{+, ≥\}, \\ n_G & \text{if } n = n_G. \end{cases} \]

The extended function f^{-1}(σ) is well-defined, i.e., all vertices that involve σ in their indices do exist in NPDG_Π. To show this, we note that by definition σ is a permutation on variables, actions, and π(σ(v)) ∈ D(σ(v)). Moreover, for each n_w^+, there is ξ ∈ Ξ, such that w ∈ nums(ξ). We saw that for an NSS it holds that σ(Ξ) = Ξ. Hence, σ(ξ) ∈ Ξ,
Thus, \( w(\alpha) \) is a summand in \( \sigma(\xi) \). Lastly, to show that \( n_{\sigma(\xi)}^1 \in NPDG_{H_1} \), note that \( \sigma(\Xi) = \Xi \) and NSS preserves linear conditions and effects.

Next, note that, by construction, \( f^{-1} \) is a homomorphism, since for \( \sigma, \sigma' \in Aut(\Pi) \) it holds \( f^{-1}(\sigma \circ \sigma') = f^{-1}(\sigma) \circ f^{-1}(\sigma') \). Moreover, for the identity element \( id_\Pi \in Aut(\Pi) \) it holds that \( e := f^{-1}(id_\Pi) \) is an identity element in \( Aut(\Pi) \). Let \( \alpha := f^{-1}(\sigma) \); it is invertible, hence, a permutation since:

\[
\alpha \circ f^{-1}(\sigma^{-1}) = f^{-1}(\sigma) \circ f^{-1}(\sigma^{-1}) = f^{-1}(\sigma \circ \sigma^{-1}) = f^{-1}(id_\Pi) = e \implies \alpha^{-1} = f^{-1}(\sigma^{-1}).
\]

Next, we need to show that \( \alpha \) is an edge- and color-preserving. We begin with color preservation.

- \( \alpha(N_{V_p}) = N_{V_p}, \alpha(N_{V_n}) = N_{V_n}, \) and \( \alpha(N_A) = N_A \) since \( \sigma(V_p) = V_p, \sigma(V_n) = V_n, \) and \( \sigma(A) = A \), respectively. All vertices in \( N_{V_p} \) are of the same color. On action vertices, \( n_{\alpha} \in N_A \), the colors are preserved due to:

\[
\text{col}(\alpha(n_{\alpha})) = \text{col}(n_{\sigma(a)}) = \text{ord}(\text{cost}(\sigma(a))) = \text{ord}(\text{cost}(a)) = \text{col}(n_{\alpha}).
\]

- For a variable \( v \in V_p \) we have \( \sigma(D(v)) = D(\sigma(v)) \). Thus, \( \alpha(N_{D(v)}) = N_{D(\sigma(v))} \). Thus, \( \alpha \) fixes the color on the propositional domains vertices.

- Let \( v \in V_n \) and \( n_{\xi}^w \in N_{D(v)} \), then

\[
\text{col}(\alpha(n_{\xi}^w)) = \text{col}(n_{\xi}^{\sigma(\alpha)}) = \text{ord}(w) = \text{col}(n_{\xi}^w).
\]

- Note that for \( n_{\xi}^1 \in N_{\Xi} \) it holds that \( \alpha(n_{\xi}^1) = n_{\sigma(\xi)}^1 \in N_{\Xi}, \) and all vertices in \( N_{\Xi} \) are of the same color.

- Trivially, \( \text{col}(\alpha(n_{\xi})) = \text{col}(n_{\xi}). \)

Thus, we have that \( \alpha \) is color preserving vertex permutation. The last property for \( \alpha \) being a color-preserving automorphism, is edge-preservation, \( (n, n') \in E \iff (\alpha(n), \alpha(n')) \in E. \)

The “\( \Rightarrow \)” part of this property follows from a meticolous application of NSS properties 1-5 to:

\[
E = E_{V_p} \cup E_{V_n} \cup E_{\Xi} \cup E_A \cup E_G.
\]

We omit the full proof here since it is straightforward and repetitive. Overall, the edges of \( NPDG_{H_1} \) correspond to element membership in sets, e.g., the edge \( (n_{\xi}^w, n_{\zeta}^v) \) corresponds to \( (v, d) \in G \), \( (n_{\xi}^w, n_{\zeta}^v) \in nms(\xi) \) and \( u \equiv \xi \in eff_u(a) \) for some \( a \in A \) and \( v \in V_n \). This part of the proof is similar to the proofs of Thm. 5 in Sievers et al. (2019) and Thm. 4.4 in Shleyfman (2020).

Briefly, the edges in \( E_{V_p} \) are preserved due to properties 1 and 2, \( E_{\Xi} \) and \( E_A \) are preserved due to properties 3-4, and \( E_G \) is preserved due to property 5. \( E_{\Xi} \) is preserved since \( \sigma(\Xi) = \Xi \). The “\( \Leftarrow \)” part holds since \( \alpha \) is invertible.

To summarise, in this section we proved that the groups \( Aut(NPDG_{H_1}) \) and \( Aut(\Pi) \) are isomorphic, i.e., structurally identical, and that there is an injection that maps these groups into a subgroup of \( Aut(\Pi) \), we name this group \( \Gamma_{H_1} \).

The generators of \( \Gamma_{H_1} \) are the ones used by DKS and OSS for symmetry breaking. Note that if we have the generators of \( Aut(NPDG_{H_1}) \), we can compute the generators of \( \Gamma_{H_1} \) in linear time. In the next section we discuss the computational complexity of computing the generators of \( Aut(NPDG_{H_1}) \).

**Computational Complexity**

The graph isomorphism (GI) decision problem that gets as input two finite graphs and determines if these graphs are isomorphic. This problem is neither known to be NP-complete nor tractable. Since many related problems appeared to be polynomial-time equivalent to the GI problem (Mathon 1979), it gave its name to a complexity class. As usual for complexity classes within the polynomial-time hierarchy, a problem \( X \) is called GI-hard if there is a polynomial-time reduction from GI to \( X \). A problem \( X \) lies in GI if it can be reduced to the GI problem. A problem \( X \) is GI-complete if it both lies in GI and GI-hard, i.e., polynomial-time equivalent to the GI problem.

Shleyfman (2019) showed that computing the generators for the groups of Structural Symmetries and the automorphisms of PDG is GI-hard. Since Numeric Planning contains FDR in the trivial case when \( V_n = \emptyset \), and in this case \( NPDG \) and PDG are equivalent, computing the automorphism group of NPDG and NSS is also GI-hard.

Shleyfman and Jonsson (2021), in turn, proved that computing the generators for the automorphism group of a colored is GI-complete. Since NPDG is a colored graph, the problem of computing generators of its automorphism group lies in GI. Thus, we have that the computation of generators of NPDG and NSS is GI-complete.

Note that while the computation on solutions to the GI problem is suspected to have a quasi-polynomial complexity (Babai 2015, 2016), there are some off-the-shelf symmetry detection packages such as bliss (Junttila and Kaski 2007) or Saucy (Darga, Sakallah, and Markov 2008), that while being a worst-case exponential in time, perform well in practice. For the experimental evaluation of the method proposed in this paper, we chose bliss.

**Experimental Evaluation**

We implement the symmetry detection method in Numeric Fast Downward (Aldinger and Nebel 2017). All experiments are run on an Intel Xeon Gold 6148 processor with a 30-minutes time limit and 4 GB memory limit using GNU parallel (Tange 2011). We evaluate A*, DKS and OSS using the blind heuristic, operator-counting heuristic with LM-cut and state equations constraints, \( h_{L^M}^{LM-cut, SEQ} \) (Kuroiwa et al. 2021), for SCT domains, which is a subset of LT, and LM-cut heuristic adapted for LT planning, \( h_{L^M}^{LM-cut} \) (Kuroiwa, Shleyfman, and Beck 2022), for LT domains. We only show instances where symmetries are detected, since the overhead of detecting no symmetries is usually less than 5 sec.

Symmetries in classical planning vary from domain to domain, and where the GRIPPER has a symmetry group that is exponential in the number of balls in the rooms, the BLOCKSWORLD domain has no symmetries at all due to a specific order imposed on the blocks in the task. The efficiency of symmetry-breaking techniques can be attributed to the vast variety of domains that were introduced to classical planning, within the setting of International Planning Competitions that were held during the last three decades.

As a sanity check, we performed some preliminary experiments on the GRIPPER domain, where the balls have ho-
mogeneous weights, and the gripper-robot has a maximum load. In this setting, the numeric approaches were consistent with the classical ones. From the 20 GRIPPER problems we have, A*, DKS, OSS equipped with blind heuristics solved 7, 20, 20, respectively. The same algorithms equipped with numeric LM-cut solved 6, 20, 20, in the same order. This results in line with those reported by Domshlak et al. for classical planning. We do not include this result in Table 1, is that OSS performs better than DKS. This was not the case on the classical domains, where the algorithms performed on par. We attribute this difference to the fact that DKS stores both original and canonical states, while OSS operates only on canonical states, and thus requires less memory. The fact that numeric states require significantly more memory may play a key role in this behavior.

### Conclusion

In this paper, we extended the notions of structural symmetries and problem description graphs from classical to numeric linear planning. This allowed us to use symmetry-breaking forward search techniques in the numeric setting.

<table>
<thead>
<tr>
<th></th>
<th>A* time expansions</th>
<th>DKS time expansions</th>
<th>OSS time expansions</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEPOT-SYM (20)</td>
<td>4 18.6 1085229</td>
<td>5 11.2 3001317</td>
<td>5 6.2 306229</td>
</tr>
<tr>
<td>Gardening (63)</td>
<td>63 3.8 181199</td>
<td>63 4.2 103376</td>
<td>63 2.2 103405</td>
</tr>
<tr>
<td>Gardening-Sat (51)</td>
<td>10 60.5 2082149</td>
<td>11 71.6 1261201</td>
<td>11 37.3 1265773</td>
</tr>
<tr>
<td>Delivery (20)</td>
<td>2 21.2 1174198</td>
<td>4 0.6 13717</td>
<td>6 0.3 13810</td>
</tr>
<tr>
<td>Rover (19)</td>
<td>4 5.7 164822</td>
<td>4 4.2 70823</td>
<td>4 2.4 70825</td>
</tr>
<tr>
<td>Sailing (20)</td>
<td>0 - -</td>
<td>0 - -</td>
<td>1 - -</td>
</tr>
<tr>
<td>Total (230)</td>
<td>83 - -</td>
<td>87 - -</td>
<td>90 - -</td>
</tr>
<tr>
<td>Linear</td>
<td>Blind</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rover-Metric (10)</td>
<td>4 4.4 154592</td>
<td>4 6.1 110913</td>
<td>4 2.2 110786</td>
</tr>
<tr>
<td>TPP-Metric (40)</td>
<td>5 1.9 40406</td>
<td>5 4.1 39994</td>
<td>5 2.5 55768</td>
</tr>
<tr>
<td>ZENOTRavel-Line (9)</td>
<td>3 1.1 25372</td>
<td>2 2.0 25190</td>
<td>3 1.0 25407</td>
</tr>
<tr>
<td>BARMAN (15)</td>
<td>2 0.7 32854</td>
<td>3 0.2 4388</td>
<td>4 0.1 4417</td>
</tr>
<tr>
<td>BARMAN-UNIT (15)</td>
<td>2 4.0 176334</td>
<td>3 0.6 12618</td>
<td>3 0.3 12708</td>
</tr>
<tr>
<td>Total (105)</td>
<td>16 - -</td>
<td>17 - -</td>
<td>19 - -</td>
</tr>
<tr>
<td>SCT</td>
<td>Blind</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEPOT (6)</td>
<td>1 1096.9 109254</td>
<td>1 831.7 87917</td>
<td>1 799.3 88237</td>
</tr>
<tr>
<td>DEPOT-SYM (20)</td>
<td>6 257.4 80762</td>
<td>6 108.5 45422</td>
<td>7 148.6 61358</td>
</tr>
<tr>
<td>Gardening (63)</td>
<td>63 3.7 17479</td>
<td>63 2.5 10546</td>
<td>63 2.3 10522</td>
</tr>
<tr>
<td>Gardening-Sat (51)</td>
<td>12 66.6 298918</td>
<td>12 46.8 181075</td>
<td>12 40.9 18106</td>
</tr>
<tr>
<td>Delivery (20)</td>
<td>2 0.1 16</td>
<td>6 0.0 15</td>
<td>6 0.0 20</td>
</tr>
<tr>
<td>Rover (19)</td>
<td>4 63.7 154971</td>
<td>4 30.4 66308</td>
<td>4 26.2 66909</td>
</tr>
<tr>
<td>Sailing (20)</td>
<td>20 0.4 109</td>
<td>20 0.4 109</td>
<td>20 0.4 116</td>
</tr>
<tr>
<td>Sailing-Sat (30)</td>
<td>7 0.5 178</td>
<td>6 0.6 178</td>
<td>6 0.6 178</td>
</tr>
<tr>
<td>Total (230)</td>
<td>115 - -</td>
<td>118 - -</td>
<td>119 - -</td>
</tr>
</tbody>
</table>

Table 1: For the search time and the number of expansions, we took the average over instances solved by all of A*, DKS and OSS except for $h_{2}^{LM-cut}$ in FO-SAILING, where A* and OSS solve one instance and DKS solves no instance.
Our experiments demonstrate that these techniques solve more problems than the current state-of-the-art. In the future, one may consider looking for a larger symmetry group.

Acknowledgements

This work is partially supported by the Natural Sciences and Research Council of Canada. Alexander Shleyfman is also partially supported by the Israel Academy of Sciences and Humanities program for Israeli postdoctoral researchers.

References


