Structurally Restricted Fragments of Numeric Planning – a Complexity Analysis

Alexander Shleyfman[1], **Daniel Gnad**[2], **Peter Jonsson**[2]

[1]The Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel
[2]Department of Computer and Information Science, Linköping University, Linköping, Sweden
alexash@biu.ac.il, daniel.gnad@liu.se, peter.jonsson@liu.se

Abstract

Numeric planning is known to be undecidable even under severe restrictions. Prior work has investigated the decidability boundaries by restricting the expressiveness of the planning formalism in terms of the numeric functions allowed in conditions and effects. We study a well-known restricted form of Hoffmann's simple numeric planning, which is undecidable. We analyze the complexity by imposing restrictions on the causal structure, exploiting a novel method for bounding variable domain sizes. First, we show that plan existence for tasks where all numeric variables are *root* nodes in the causal graph is in **PSPACE**. Second, we show that for tasks with only numeric *leaf* variables the problem is decidable, and that it is in **PSPACE** if the propositional state space has a fixed size. Our work lays a strong foundation for future investigations of structurally more complex tasks. From a practical perspective, our method allows to employ heuristics and methods that are geared towards finite variable domains (such as pattern database heuristics or decoupled search) to solve non-trivial families of numeric planning problems.

Introduction

In recent years, significant progress has been made in developing methods for planning with numeric variables (Hoffmann 2002; Shin and Davis 2005; Gerevini, Saetti, and Serina 2008; Eyerich, Mattmüller, and Röger 2009; Coles et al. 2013; Illanes and McIlraith 2017; Scala et al. 2017; Aldinger and Nebel 2017; Li et al. 2018; Piacentini et al. 2018a,b; Kuroiwa et al. 2021; Shleyfman, Kuroiwa, and Beck 2023). From a theoretical perspective, the success of these methods raises a conundrum since even very simple forms of numeric planning are known to be undecidable (Helmert 2002; Gnad et al. 2023). The important question in this context is obvious: how to restrict numeric planning to render it decidable? Helmert's seminal paper presents decidable cases based on restricting the formalism itself, e.g. the allowed mathematical functions and relations. Our approach is different: we identify decidable fragments based on their structural properties. To this end, we generalize *causal graphs* (CG) to numeric planning. The CG approach has been a major source of complexity results for classical planning (Jonsson and Bäckström 1998; Brafman and Domshlak 2003; Helmert 2004; Giménez and Jonsson 2008). We consider a restricted but still expressive variant of the *simple numeric planning* (SNP) formalism (Hoffmann 2003; Scala et al. 2016). While SNP only allows linear expressions as numeric conditions and additive constants as action effects, it is undecidable. We disallow linear conditions involving several variables; the resulting formalism is still undecidable. We study this restricted formalism because the standard method to compile away such conditions introduces cyclic dependencies between numeric variables and these cannot be handled by our proposed approaches. This formalism equals the class $(\mathcal{C}_c, \mathcal{C}_c, \mathcal{E}_{\pm c})$ by Helmert (2002).

We focus completely on providing decidability results in this paper and leave hardness results for future work. One should note that almost every decidability result that we present is in fact a **PSPACE**-membership result; such a result implies that the problem under consideration is not harder than classical propositional planning. We start with tasks with a single numeric variable x, which provides a necessary platform for more interesting structures. Our main tool is Thm. 2: for every solvable one-variable task Π_x one can compute a bounded interval I s.t. there exists a plan π that stays inside I during execution. The proof is based on a nontrivial action reordering technique. To solve Π_x, it is thus sufficient to explore a finite part of the underlying infinite state space, and this finite part is so small that it proves membership in **PSPACE**.

We continue our analysis with instances containing two variables x and y s.t. y *depends* on x but not vice versa, i.e. instances with CG $x \to y$. If x or y is propositional, we show that there exists a bounding interval for the numeric variable implying a finite search space. Our proof technique is based on guessing action sequences and analysing them via integer programming and Thm. 2. This allows us to show that SNP with an arbitrary number of numeric root or leaf variables is decidable, and lies in **PSPACE** if all numeric variables are (1) roots, or (2) leaves and the state space induced by the propositional variables has a fixed size. This can be used and extended in various ways. For instance, numeric planning with a fork or inverted-fork CG is decidable and often can even be solved in polynomial space. Such CG structures were first studied by Domshlak and Dinitz (2001) and their tractability result has been highly influential when studying more complex structural properties in classical plan-

ning (Brafman and Domshlak 2003; Bäckström and Jonsson 2013; Domshlak and Nazarenko 2013).

We conclude the paper by discussing our findings. In particular, we demonstrate how our methods can be applied to broader fragments of numeric planning and discuss how our results can be utilized by techniques known from classical planning that require bounded domain sizes.

Numeric Planning

We consider *simple numeric planning* (SNP) as introduced by Hoffmann (2003) and refined by Scala et al. (2016). SNP was originally defined in the STRIPS formalism (Fikes and Nilsson 1971) but we present it (in a more general way) as an extension of the finite-domain planning (FDR) formalism (Bäckström and Nebel 1995; Helmert 2009).

A *simple numeric task* (SNT) is a 4-tuple $\Pi = \langle \mathcal{V}, \mathcal{A}, s_0, G \rangle$, where $\mathcal{V} = \mathcal{V}_n \cup \mathcal{V}_p$ is a finite set of *numeric* and *propositional variables*, $\mathcal{A}$ is the set of *actions*, s_0 is the *initial state*, and G is the set of *goal conditions*. Numeric variables $\mathcal{V}_n$ have domain $\mathbb{Q}$; each propositional variable $v \in \mathcal{V}_p$ has a finite domain $\mathcal{D}(v)$. The set of *states* of Π is $\mathcal{S} := \times_{v \in \mathcal{V}_p} \mathcal{D}(v) \times \times_{v \in \mathcal{V}_n} \mathbb{Q}$, i.e. all assignments over all variables $\mathcal{V}$. We refer to a state $s \in \mathcal{S}$ as a set of numeric and propositional facts of the form $\langle v, f \rangle$, $s = s_n \cup s_p$, we say that a fact is numeric if $v \in \mathcal{V}_n$ and $f \in \mathbb{Q}$, or propositional if $v \in \mathcal{V}_p$ and $f \in \mathcal{D}(v)$. We say that $s \models (v = f)$ iff $\langle v, f \rangle \in s$, and write $s[v] = f$, i.e., $s[v]$ indicates the value of $v \in \mathcal{V}$ in state s. We say that s' is a *partial state* if there is a state $s \in \mathcal{S}$ s.t. $s' \subset s$.

Conditions can be either propositional or numeric. Propositional conditions are partial propositional states, i.e., ψ is a *propositional condition* if there is $s \in \mathcal{S}$ s.t. $\psi \subseteq s_p$. A *linear numeric condition* over the variables $V \subseteq \mathcal{V}_n$ is written as $\psi : \sum_{v \in V} w_v v \bowtie w_0$ where $\bowtie \in \{>, \geq, <, \leq\}$, $w_v, w_0 \in \mathbb{Q}$. We say that s *satisfies* ψ, denoted $s \models \psi$, if $\sum_{v \in V} w_v s[v] \bowtie w_0$. We extend this to sets of conditions Ψ by $s \models \Psi$. We let $\Psi(v)$ denote all numeric conditions where the variable $v \in \mathcal{V}_n$ appears. We assume that all numbers are given in binary encoding.

An *action* $a \in \mathcal{A}$ is a tuple $\langle \mathsf{pre}(a), \mathsf{eff}(a) \rangle$, where $\mathsf{pre}(a)$ are the *preconditions*, and $\mathsf{eff}(a)$ the *effects* of a. Preconditions are defined as $\mathsf{pre}(a) := \mathsf{pre}_p(a) \cup \mathsf{pre}_n(a)$, with propositional and linear numeric conditions, respectively. Effects $\mathsf{eff}(a) := \mathsf{eff}_p(a) \cup \mathsf{eff}_n(a)$ are similarly defined as sets of propositional and numeric effects. For SNT, numeric effects have the form $(v \mathrel{+}= c)$, where $v \in \mathcal{V}_n$ and $c \in \mathbb{Q} \setminus \{0\}$. Actions have at most one effect on each numeric variable. We say that action a is *applicable* in state s if $s \models \mathsf{pre}(a)$. The result of applying a in s is denoted by $s[\![a]\!] := s'_p \cup s'_n$, with $s'_p[v] = d$ if $\langle v, d \rangle \in \mathsf{eff}_p(a)$, $s'_n[v] = s_n[v] + c$ if $(v \mathrel{+}= c) \in \mathsf{eff}_n(a)$, and $s[\![a]\!][v] = s[v]$ otherwise.

The goal condition $G = G_p \cup G_n$ denotes propositional and numeric conditions, respectively. We say that s_* is a *goal state* if $s_* \models G$. An *s-plan* is an action sequence π that can be applied successively in s and results in a goal state $s_* \models G$. A plan for Π is an s_0-plan.

The computational problem that we consider is the *plan existence* problem (PE), i.e. given a task Π, is there a plan for Π? Given a set of tasks $\mathbf{A}$, we let PE$\mathbf{A}$ denote PE with inputs restricted to $\mathbf{A}$. We let $\|X\|$ denote the number of bits needed to represent an object X such as a planning task. Clearly, the number of variables and actions does not exceed $\|\Pi\|$. The number of elements in a propositional variable domain $\mathcal{D}(v)$ does not exceed $\|\Pi\|$ either. We may, without loss of generality, assume that each domain value is used in the precondition or the effect of at least one action, and we may assume that at least one bit is needed to describe that, e.g. "a precondition of action a is that variable v has value d".

A *restricted task* (RT) is a variant of SNT where all numeric conditions are of the form: $\psi : v \bowtie w_0$, with $w_0 \in \mathbb{Q}$, $v \in \mathcal{V}_n$, and $\bowtie \in \{>, \geq, <, \leq\}$. Similar to SNT, actions can only increase or decrease variables by constant quantities (Hoffmann 2003; Scala, Haslum, and Thiébaux 2016). Hoffmann points out that SNT can be reduced to an RT with a simple translation. Given an SNT $\Pi = \langle \mathcal{V}, \mathcal{A}, s_0, G \rangle$, we construct a corresponding Π^{RT}. For every distinct linear expression $\xi : \sum_{v \in V} w_v v$ that appears in Π, we introduce a numeric variable v^ξ with $s_0^{\text{RT}}[v^\xi] = \sum_{v \in V} w_v s_0[v]$. We replace every condition $\xi \bowtie w_0$ with $v^\xi \bowtie w_0$. For every action a with an effect $v \mathrel{+}= c_v^a$, an effect $v^\xi \mathrel{+}= \sum_{v \in V} w_v c_v^a$ is added. This transformation is polynomial in $\|\Pi\|$, and it immediately proves undecidability of the plan existence problem for RT. Here we assume all tasks to be in RT form.

Causal Graphs for Numeric Planning

Planning tasks are typically structurally complex, and *causal graphs* (CG) are a common means to study this structure (e.g., Knoblock 1994; Bacchus and Yang 1994; Brafman and Domshlak 2003). We adopt the compact definition by Helmert (2004): the CG of a classical planning task $\Pi = \langle \mathcal{V}, \mathcal{A}, I, G \rangle$ is a digraph $CG(\Pi) = \langle \mathcal{V}, \mathcal{E} \rangle$, where $(u, v) \in \mathcal{E}$ if $u \neq v$ and there exists $a \in \mathcal{A}$, s.t. $u \in vars(\mathsf{pre}(a)) \cup vars(\mathsf{eff}(a))$ and $v \in vars(\mathsf{eff}(a))$, where $vars(s)$ denotes the set of variables defined in the (partial) state s. Intuitively, the CG contains an edge from a variable v to a variable v', if changing the value of v' might require v to change its value, too, so v' *depends* on v.

It is not clear how to adapt this definition for general numeric planning. Let us first consider RTs. Every condition and effect of an action involves at most one numeric variable so one can construct the CG of an RT in the same manner as for propositional tasks, i.e., we can treat numeric variables as propositional ones. Linear conditions $\psi : \sum_{x \in V} w_x s[x] \bowtie w_0 \in \mathsf{pre}_n(a)$ in SNT are more intricate. If numeric variables x_1 and x_2 both appear in V, then they are co-dependent in terms of ψ. Note that the case $x_1 + x_2 \geq w_0$ differs from the case when $x_1 \geq w_1$ and $x_2 \geq w_2$. This co-dependency is also seen in the translation to RT, which introduces a new variable v^ψ and cycles in the CG between x_1, x_2, and v^ψ. We leave it to future work to define a suitable CG for SNT and focus on the simpler RT case in this paper.

In the sequel, we focus on RTs with numeric variables that are *root* or *leaf* nodes in the CG: roots have only outgoing edges and leaves have only incoming edges. We call numeric variables that are leaves (roots) *numeric leaves (roots)*. A CG has a *fork structure* if there is a variable y s.t. all edges

in the CG are of the form $y \to x$, and there is an edge for each $x \in \mathcal{V}$. A CG has an *inverted fork structure* if $y \leftarrow x$ instead. These important notions were introduced in classical planning by Domshlak and Dinitz (2001).

Integer Restricted Tasks

To simplify the forthcoming proofs we present an integer form of RT tasks. It is similar to the "domain simplification" of Helmert (2002), but additionally normalises initial state values to 0 and all numeric conditions to be integer.

Suppose we have an RT $\Pi = \langle \mathcal{V}, \mathcal{A}, s_0, G \rangle$. Any condition can be seen as a test whether $x \in \mathcal{V}_n$ belongs to a given rational interval (which is not necessarily bounded). Let $[\![l^-, l^+]\!]$ denote an interval where $l^- \in \{-\infty\} \cup \mathbb{Q}$, $l^+ \in \mathbb{Q} \cup \{+\infty\}$, and $l^- \leq l^+$. The numeric precondition of each action a has the form $\mathsf{pre}_n(a) = \{x \in [\![l^-, l^+]\!] \mid x \in \mathcal{V}_n\}$. Note that (1) $x \in [\![l^-, l^+]\!]$ is a semantic notation meaning $s \models x \in [\![l^-, l^+]\!]$ iff $s[x] \in [\![l^-, l^+]\!]$, and (2) we define conditions on *all* numeric variables replacing empty conditions with $x \in (-\infty, \infty)$. The numeric goal conditions G_n have the same form. Let $\mathcal{W}_x$ denote the set of numbers that appear in the numeric conditions on x, i.e. $\mathcal{W}_x := \{l^-, l^+ \mid (x \in [\![l^-, l^+]\!]) \in \Psi(x)\} \setminus \{-\infty, \infty\}$. Note that $\mathcal{W}_x$ is a finite set. Each action a has numeric effects of the form $x \mathrel{+}= c$ with $c \neq 0$. We let $\mathcal{C}_x := \{c \mid x \mathrel{+}= c \in \mathsf{eff}_n(a), a \in \mathcal{A}\} \subseteq \mathbb{Q}$ denote the set of all additive constants affecting x.

We say that an RT $\bar{\Pi}$ is *integer* if for each $x \in \mathcal{V}_n$ it holds that $\mathcal{C}_x \cup \mathcal{W}_x \subseteq \mathbb{Z}$ and $s_0[x] = 0$. We say that it has a *bounded goal condition* if for each $x \in \mathcal{V}_n$: $(x \in [\![l^-, l^+]\!]) \in G_n$ implies that $l^-, l^+ \in \mathbb{N}_0$.

We will show that every RT Π has a corresponding integer instance $\bar{\Pi}$ (computable in polynomial time) that is solvable iff Π is. The basic idea is to apply a linear transformation to the rational values appearing in Π: these are multiplied with a suitable coefficient to become integers and another coefficient is added to ensure that the initial condition is 0. The full proofs of the results below are in the supplementary material (Shleyfman, Gnad, and Jonsson 2023). Assume $C \in \mathbb{Q} \setminus \{0\}$ and $B \in \mathbb{Q}$. We define the map $(C \cdot x + B)(\Pi)$ on Π and an $x \in \mathcal{V}_n$ as follows: each condition of the form $x \in [\![a, b]\!]$ in Π is replaced with $x \in [\![C \cdot a + B, C \cdot b + B]\!]$, and the effect $x \mathrel{+}= c$ is replaced with $x \mathrel{+}= C \cdot c$. The initial state $s_0[x] = x_0$ is replaced with $s_0[x] = C \cdot x_0 + B$.

Lemma 1. *Every plan for Π is also a plan for $(C \cdot x + B)(\Pi)$ and vice-versa, where $C \in \mathbb{Q} \setminus \{0\}$ and $B \in \mathbb{Q}$.*

Let $\mathrm{LCD}(X)$ denote the *least common denominator* of a finite set X of rational numbers, i.e., $\mathrm{LCD}(X)$ is the smallest number s.t. $\mathrm{LCD}(X) \cdot x$ is an integer for every $x \in X$. To obtain the integer RT we repeatedly apply Lemma 1 on each variable x using the coefficients $C = \mathrm{LCD}(\mathcal{C}_x \cup \mathcal{W}_x)$ and $B = -C \cdot s_0[x]$.

Corollary 1. *For each RT Π, there exists an integer RT $\bar{\Pi}$ s.t. $\bar{\Pi}$ is solvable iff Π is solvable. Moreover, $\bar{\Pi}$ can be computed in polynomial time, $\|\bar{\Pi}\| \in O(\|\Pi\|^2)$, and the size of each number in $\bar{\Pi}$ is at most $\|\Pi\|$ bits.*

Note that the transformation to integer RTs does not change the CG. Next, we show that we can bound the numeric goal conditions to a closed interval in some cases.

We introduce some useful notation. Note that in an integer RT it holds that $\mathcal{C}_x, \mathcal{W}_x \subseteq \mathbb{Z}$. We define $C_x^{\max} := \max_{c \in \mathcal{C}_x} |c|$ to be the bound on the maximal change that can occur in the value of x due to application of one action. We also define the bounds on the explicit numbers in numeric conditions on x: $M_x^- := \min(\mathcal{W}_x \cup \{0\}) - 1$ and $M_x^+ := \max(\mathcal{W}_x \cup \{0\}) + 1$. We call $[M_x^-, M_x^+]$ the *explicit numeric interval* (ENI) of x. In the sequel, we are interested in bounding ENIs in a solvability-preserving way.

Lemma 2. *Let Π be an integer RT where $(x \geq g) \in G_n$ and each action that affects $x \in \mathcal{V}_n$ affects no other variables. Then, there is a $g' \geq g$ s.t. each plan for Π contains a subsequence of actions π' that solves Π and for each state s along π' it holds that $s[x] < g'$.*

Proof. Let $C_x^+ = \max(\mathcal{C}_x)$. Let π be a plan for Π, and let $g' = M_x^+ + C_x^+$. If π terminates within the ENI, the last state of π lies within $[g, g']$. Otherwise, let $\tilde{s}$ be the last state along π s.t. $\tilde{s}[x] \leq M_x^+$. Since π starts at $s_0[x] = 0$ and each action changes x by at most C_x^+, there is at least one state s along π after $\tilde{s}[x]$ s.t. $s[x] \in [g, M_x^+ + C_x^+)$. Thus, π has a prefix that achieves $x \in [g, g')$. As actions that affect x do not affect any other variables and satisfy $x \geq M_x^+$. Hence, we obtain π' by omitting all actions after $\tilde{s}$ that affect x. □

If the conditions of Lem. 2 hold for an $x \in \mathcal{V}_n$, we can replace the goal condition $x \geq g$ with $x \in [g, g + C_x^+]$.

Bounding the Reachable State Space

We will now prove decidability and **PSPACE** membership for various classes of RT. These results are based on proving that, to preserve solvability, it suffices to consider only a part of the state space reachable from the initial state. For decidability, it is sufficient to prove (1) that this space is finite and (2) that each state has a finite description. Then, we can explicitly generate the state space and perform search in it. For **PSPACE** membership, we need more restrictive conditions: (1^+) the reachable space is of size $2^{p(||\Pi||)}$ and (2^+) each state can be described using $q(||\Pi||)$ bits (where p and q are polynomials). Recall that **NSPACE**$(f(n))$ is the class of decision problems that can be solved by nondeterministic algorithms using space $O(f(n))$, while **DSPACE**$(f(n))$ is defined analogously but for deterministic algorithms.

Theorem 1 (Savitch 1970)**.** *If $f(n) \in \Omega(\log n)$, then **NSPACE**$(f(n)) \subseteq$ **DSPACE**$(f(n)^2)$.*

For tasks satisfying (1^+) and (2^+), there is a nondeterministic planning algorithm that guesses the next state in the plan (which can be represented using a polynomial number of bits), and where no plans that are longer than $2^{p(\|\Pi\|)}$ steps are considered; such an algorithm can be implemented using only polynomial space. Thm. 1 implies that there is a corresponding deterministic polynomial-space algorithm and that the planning problem is in **PSPACE**.

We start by presenting our basic criteria for bounding the size of the reachable state space. Intuitively, we want each numeric action to be applicable in just a finite number of states. Thus, let $\mathbf{X}$ denote the set of integer RTs with

the following property: for each $a \in \mathcal{A}$ and for each effect $x \mathrel{+}= k_x^a \in \mathsf{eff}_n(a)$, the action a has the precondition $x \in [\![l^-, l^+]\!] \in \mathsf{pre}_n(a)$, where $l^-, l^+ \in \mathbb{Z}$.

Lemma 3. PEX *is in* ***PSPACE****.*

Proof. Choose $\bar{\Pi} \in \mathbf{X}$ and let $x \in \mathcal{V}_n$ be a numeric variable with ENI $[M_x^-, M_x^+]$. The subdomain of x that is reachable from the initial state is bounded by $I := [M_x^- - C_x^{\max}, M_x^+ + C_x^{\max}]$. The proof is by contradiction. Pick a state s s.t. $s[x] \notin I$. Assume there is an action a and a state s' s.t. $s = s'[\![a]\!]$. Then, $s[x] = s'[x] + k_x^a$, where k_x^a is the effect of a on x. But $|k_x^a| \leq C_{x_-}^{\max}$ so a cannot be applied in s' since (by our assumption on $\bar{\Pi}$) the precondition of a on x is an interval bounded by $[M_x^-, M_x^+]$. Since every numeric variable has a bounded domain, the search space is finite, which automatically implies bounded goal conditions.

The **PSPACE** membership follows by Thm. 1. Let $\left\|\bar{\Pi}\right\| = n$, so $\bar{\Pi}$ may have at most n variables, and each number has at most n bits. Thus, the size of the reachable search space is at most $(2^n)^n = 2^{n^2}$ and each state can be described using at most n^2 bits. □

Thus, undecidability of RT instances is caused by conditions of the form $x \leq l^+$ or $x \geq l^-$, i.e. conditions that lack a lower and/or an upper bound. It is now interesting to identify cases of RT where such conditions can be replaced by conditions based on bounded intervals. This can be done, for example, by showing that for each plan, there is an equivalent plan where variable values do not exceed given precomputed bounds. We introduce the following notation.

Let π be a plan for an integer RT Π, represented by the following state/action sequence:

$$s_0, a_1, s_1, a_2, \ldots \vec{s}_{k_1}, \vec{a}_{k_1+1}, \vec{s}_{k_1+1}, \ldots \vec{a}_{k_2}, \vec{s}_{k_2+1}, \ldots, \\ \overleftarrow{a}_{k_3}, \ldots \overleftarrow{a}_{k_4}, \ldots, \vec{a}_{k_5}, \ldots \vec{a}_{k_6}, \ldots, s_n \models G,$$

and let $[M_x^-, M_x^+]$ be the ENI for x. We say that an action a that labels a transition $s \xrightarrow{a} s'$ *exceeds* M_x^+ for x if $s[x] \geq M_x^+$ and $s'[x] \geq M_x^+$. The definition for exceeding M_x^- is the same, but here $x \leq M_x^-$ holds instead. Suppose that the sub-sequences of π marked by $\vec{}$ are the ones that exceed M_x^+ and the $\overleftarrow{}$ ones exceed M_x^-. Since $M_x^- < M_x^+$, between each $\vec{}$ and $\overleftarrow{}$ sub-sequence there must be at least one unmarked action (neither $\vec{}$ nor $\overleftarrow{}$). The first and the last actions of π are also unmarked, since, both $s_0[x] = 0$ and the interval $[g_1, g_2]$ lie inside the ENI $[M_x^-, M_x^+]$, where $G = \{x \in [g_1, g_2]\}$.

To obtain an equivalent instance with bounded interval preconditions, one needs to reason about reordering of plans (see, for instance, Thm. 2). We require the following result:

Lemma 4. *Let $\mathcal{C} \subseteq \mathbb{Z}$ be a finite set and let $C^{\max} = \max_{c \in \mathcal{C}} |c|$. Let $a, b \in \mathbb{Z}$ be s.t. $|a - b| \leq 2C^{\max}$, and let $\{c_i\}_{i=1}^n$ be a sequence of numbers s.t. each $c_i \in \mathcal{C}$ and $a + \sum_{i=1}^n c_i = b$. Then, the following holds for every $y \in \mathbb{Z}$ satisfying $a, b \in [y, y + 2C^{\max}]$: there is a permutation $\sigma : [n] \to [n]$ s.t. for each $k \in [n]$ it holds that $a + \sum_{i=1}^k c_{\sigma(i)} \in [y, y + 2C^{\max}]$.*

Proof. The proof is by induction. Suppose that for $k-1$ it holds that $a_{k-1} := a + \sum_{i=1}^{k-1} c_{\sigma(i)} \in [y, y + 2C^{\max}]$. For the element with index $\sigma(k)$, we need to chose one of the indices from the set $[n] \setminus \sigma([k-1])$. We have three cases:

Case 1. There is $i \in [n] \setminus \sigma([k-1])$ s.t. $c_i = 0$. Then, set $\sigma(k) := i$, and $a_{k-1} = a_k$.

Case 2. All elements with indices in $[n] \setminus \sigma([k-1])$ are of the same sign. Then, for each $i \in [n] \setminus \sigma([k-1])$ it holds that $a_{k-1} + c_i \in [a_{k-1}, b] \subseteq [y, y + 2C^{\max}]$. Here, $a_{k-1} \in [y, y + 2C^{\max}]$ by induction, and $b \in [y, y + 2C^{\max}]$ by definition. Thus, we can set $\sigma(k) := i$ for any $i \in [n] \setminus \sigma([k-1])$.

Case 3. There are two indices $i, j \in [n] \setminus \sigma([k-1])\}$ s.t. c_i and c_j are of different sign. By definition, it holds that $|c_i|, |c_j| \leq C^{\max}$. Note that since $a_{k-1} \in [y, y + 2C^{\max}]$, it either holds that $a_{k-1} - y \leq C^{\max}$ or that $y + 2C^{\max} - a_{k-1} \leq C^{\max}$, i.e., the distance from a_{k-1} to one of the endpoints of the interval exceeds $C^{\max}$. Thus, either $a_{k-1} + c_i$ or $a_{k-1} + c_j$ lies in the interval $[y, y + 2C^{\max}]$. Without loss of generality, assume that $a_{k-1} + c_j \in [y, y + 2C^{\max}]$. Then, we set $\sigma(k) := j$, and repeat the process. □

Single Numeric Variable

Suppose we have an RT with a Single Numeric Variable (SNVT), $\Pi_x = \langle \mathcal{V}, \mathcal{A}, s_0, G \rangle$, where $\mathcal{V} = \mathcal{V}_n = \{x\}$. We prove that PESNVT can be solved by search in a finite subset of the state space. This is not clear a priori from the problem formulation, and forms a basis for our forthcoming complexity results. Cor. 1 and Lem. 2 imply that Π_x can be transformed into an integer RT with a bounded goal condition in polynomial time.

Theorem 2. PESNVT *is in* ***PSPACE****.*

Proof. Let Π_x be an integer SNVT, where x has the ENI $[M_x^-, M_x^+]$. We aim to show that there exists a bounded interval $I = [M_x^- - 2C_x^{\max}, M_x^+ + 2C_x^{\max}]$ s.t. each plan π for Π_x can be reordered into a plan π' for Π_x^I. Here, Π_x^I denotes Π_x where each precondition $\mathsf{pre}(a) = \{x \in [\![l^-, l^+]\!]\}$ of each $a \in \mathcal{A}$ is replaced with the precondition $\mathsf{pre}(a) = \{x \in [\![l^-, l^+]\!] \cap I\}$. Then, the claim follows by Lem. 3.

Let π be a plan for Π_x and $\langle s_0, \ldots, s_n \rangle$ be the sequence of states traversed by π. We aim at reordering the parts of the plan that exceed M_x^+ from above and M_x^- from below. The definition of *exceed* assures that these parts do not overlap, and that there is at least one action that separates them.

We now calculate $I = [L_x^-, L_x^+]$. First, we obtain the L_x^- bound. Let $\mathcal{A}^{-\infty} := \{a \in \mathcal{A} \mid \forall s[x] \leq M_x^- : s \models \mathsf{pre}(a)\}$. So $\mathcal{A}^{-\infty}$ is the set of all actions with preconditions either $x \in \mathbb{Q}$ (i.e., no preconditions) or $x \leq b$ for some $b \in \mathbb{Q}$, i.e., the actions that can be applied with an arbitrarily small x.

If no state in $\langle s_0, \ldots, s_n \rangle$ has a value of x below M_x^-, then there is no need for reordering. Otherwise, let s_{k_1} be the first state below M_x^-, and let $\pi_{k_1 \to k_2} := \{a_{k_1+i}\}_{i=1}^{k_2-k_1}$ be the longest sub-sequence of π that starts in s_{k_1} s.t. all actions in $\pi_{k_1 \to k_2}$ exceed M_x^-. Let s_{k_2} be the last state along the application of $\pi_{k_1 \to k_2}$. Note that it may happen that $s_{k_1} = s_{k_2}$, in this case we set $\pi_{k_1 \to k_2} = \emptyset$. By definition, $s_{k_1} \neq s_0$ and $s_{k_2} \not\models G$. Since an action cannot increase or decrease the value of x by more than $C_x^{\max}$, it holds that $s_{k_1}[x], s_{k_2}[x] \in [M_x^- - C_x^{\max}, M_x^-]$. Thus,

$|s_{k_1}[x] - s_{k_2}[x]| \leq C_x^{\max}$. Moreover, since $\pi_{k_1 \to k_2}$ is a sequence of action applications:

$$s_{k_1}[x] + \sum_{i=1}^{k_2-k_1} c_{k_1+i} = s_{k_2}[x],$$

where c_j is the effect of applying a_j in s_{j-1}. By construction, all actions in $\pi_{k_1 \to k_2}$ belong to $\mathcal{A}^{-\infty}$, and thus can be applied within the interval $(-\infty, M_x^-]$. By Lem. 4, there exists a permutation σ of indices of $\pi_{k_1 \to k_2}$ s.t.

$$s_{k_1}[x] + \sum_{i=1}^{k_2-k_1} c_{k_1+\sigma(i)} \in [M_x^- - 2C_x^{\max}, M_x^-].$$

Set $L_x^- := M_x^- - 2C_x^{\max}$. Note that there is a finite number of such disjoint prefixes, and each prefix can be reordered such that for each state s along the reordered plan, it holds that $s[x] \geq L_x^-$. The reordering that bounds the value of x along π from above is obtained by using the upper bound $L_x^+ := M_x^+ + 2C_x^{\max}$. The claim follows by Lem. 3. □

This result applies to one-variable SNTs, too. Consider the SNT-to-RT transformation: if the given SNT is one-variable, then the resulting RT is identical to the SNT.

Numeric Roots

Let NRRT be the class of RT where all numeric variables are CG-roots. NRRT includes, for example, many tasks with inverted fork structure. We prove that PENRRT is in **PSPACE**. Let $x \in \mathcal{V}_n$ be a numeric variable. We say that each action a that affects x is an *inner action* if it affects only x and has no preconditions on other variables, i.e., $\mathsf{pre}(a) = \{x \in [\![l^-, l^+]\!]\}$. Such actions do not add edges to the causal graph.

Lemma 5. *Let Π be an integer* RT *with a numeric variable $x \in \mathcal{V}_n$ which is a root node in the CG of Π. Then, there is an* RT Π^I *where x is restricted to the interval $I := [M_x^- - 2C_x^{\max}, M_x^+ + 2C_x^{\max}]$, and Π^I is solvable iff Π is solvable.*

Proof. The proof is similar to that of Thm. 2. Note that all actions that affect x are inner actions since x is a root. Let π be a plan for Π, let $S = \langle s_0, \ldots, s_n \rangle$ be the sequence of states traversed by π, and let $[M_x^-, M_x^+]$ be the ENI of x. We aim at producing an interval $I = [L_x^-, L_x^+]$ s.t. $[M_x^-, M_x^+] \subseteq I$ and there is a reordering of π s.t. the values of x along its application do not exceed I. If no state in S achieves a value below M_x^- or above M_x^+, then there is no need in reordering the plan. Assume that some state in S achieves a value below M_x^-. We let s_{k_1} be the first state with x below M_x^-, and let $\pi_{k_1 \to k_2} := \{a_{k_1+i}\}_{i=1}^{k_2-k_1}$ be the longest sub-sequence of π that starts in s_{k_1} s.t. all actions in $\pi_{k_1 \to k_2}$ exceed M_x^-. Let us look at this sequence:

$$\ldots, \overleftarrow{s}_{k_1}, \overleftarrow{a}_{k_1+1}, \overleftarrow{s}_{k_1+1}, \mathbf{a_{k_1+2}}, \ldots, \overleftarrow{s}_{k_2}, \mathbf{a_{k_2}}, \overleftarrow{s}_{k_2+1}, \ldots.$$

The sequence $\pi_{k_1 \to k_2}$ contains two kinds of actions: (1) inner actions of x, marked in **bold**, and (2) actions that do not affect x but may have a precondition on x, marked as $\overleftarrow{\cdot}$. By the definition of M_x^- and since type (2) actions do not affect x, we move them to the very front of $\pi_{k_1 \to k_2}$ while keeping their relative order. We then reorder the inner actions by Lem. 4 and append them to the prefix of type (2) actions. This gives us the bound $L_x^- := M_x^- - 2C_x^{\max}$. The bound $L_x^+ := M_x^+ + 2C_x^{\max}$ is obtained analogously. □

Combining this result with Lem. 3 proves the following.

Corollary 2. PENRRT *is in* ***PSPACE****.*

Forks

We first consider a single numeric leaf that depends on a single propositional root. This result is the cornerstone for our fork complexity result. It is presented as a decidability result and we will refine it into a **PSPACE**-membership result in Theorem 3. Our approach has some similarities with Helmert's (2002) Algorithm 22. Both approaches rely on guessing action sequences, and both employ an ILP that encodes the number of times certain actions are applied. The details of how these approaches are used differ significantly, though, and while Helmert only requires the number of guesses to be bounded to show decidability, we derive bounds that show **PSPACE**-membership.

Lemma 6. *Let* **A** *denote the set of* RT*s with two variables $\mathcal{V} = \{v, x\}$, where (1) $v \in \mathcal{V}_p$ is a propositional variable, (2) $x \in \mathcal{V}_n$ is a numeric variable, and (3) the CG of the task has exactly one edge (v, x). Then,* PE**A** *is decidable.*

Proof. Assume without loss of generality (via Cor. 1 and Lem. 2) that $\Pi_{v,x}$ is an integer task in **A** with bounded numeric goal condition. No action a affects both v and x due to the CG. Hence, if an action affects the propositional variable v, then $\mathsf{pre}(a) = \{\langle v, u \rangle\}$ and $\mathsf{eff}(a) = \{\langle v, u' \rangle\}$ with $u, u' \in \mathcal{D}(v)$ and actions affecting v are inner actions. View the values of v as a directed graph with nodes $\mathcal{D}(v)$ and each action a with $\mathsf{pre}(a) = \{\langle v, u \rangle\}$ and $\mathsf{eff}(a) = \{\langle v, u' \rangle\}$ inducing a directed edge (u, u'). We denote the $k \in \mathbb{N}$ strongly connected components (SCC) of this graph by $\{\mathcal{C}_j\}_{j=1}^k$. Recall that the SCCs of a graph form a DAG.

Let π be a plan for $\Pi_{v,x}$, $\langle s_0, \ldots, s_n \rangle$ be the sequence of states traversed by π, and $[M_x^-, M_x^+]$ be the ENI of x. If no state in $\langle s_0, \ldots, s_n \rangle$ achieves a value below M_x^-, there is no need for reconstruction. Otherwise, let s_{k_1} be the first state below M_x^-, and let $\pi_{k_1 \to k_2} := \{a_{k_1+i}\}_{i=1}^{k_2-k_1}$ be the longest sub-sequence of π that starts in s_{k_1} s.t. all actions in $\pi_{k_1 \to k_2}$ exceed M_x^-. Let s_{k_2} be the last state along the application of $\pi_{k_1 \to k_2}$. Note that it may happen that $s_{k_1} = s_{k_2}$, in this case we set $\pi_{k_1 \to k_2} = \emptyset$. Otherwise, by definition, $s_{k_1} \neq s_0$ and $s_{k_2} \not\models G$. By construction of the sub-sequence it holds that $s_{k_1-1}[x], s_{k_2+1}[x] > M_x^-$, thus $s_{k_1}[x], s_{k_2}[x] \in [M_x^- - C_x^{\max}, M_x^-]$. Define $a := s_{k_1}[x]$ and $b := s_{k_2}[x]$. The actions in $\pi_{k_1 \to k_2}$ traverse the values of v in some order over its SCCs, we assume this order to be $\vec{\mathcal{C}} : \mathcal{C}_1 \to \mathcal{C}_2 \to \cdots \to \mathcal{C}_m$, for some $m \leq k$. We next show how to compute a bound $\mathcal{M}_x^{-,\vec{\mathcal{C}},a,b}$ for all possible value pairs $a, b \in [M_x^- - C_x^{\max}, M_x^-] \cap \mathbb{Z}$ and all possible SCC-chains $\vec{\mathcal{C}}$. Thus, for each sub-sequence $\pi_{k_1 \to k_2}$ of the plan π that exceeds M_x^-

from below there is a bound

$$L_x^- = \min_{\vec{\mathcal{C}};a,b\in[M_x^- - C_x^{\max}, M_x^-]} \mathcal{M}_x^{-,\vec{\mathcal{C}},a,b}$$

that bounds a reconstructed plan from below. We can compute L_x^- by iteratively solving an optimization problem which we explain next.

To check for existence of an action sequence that changes the value of x from a to b without exceeding M_x^-, we enumerate all possible sequences of SCCs. Let $\vec{\mathcal{C}}$ be such a sequence. For each $i \in [m]$ we define a set of actions:

$$\mathcal{A}_x^j = \mathcal{A}_x^0 \cup \{a \in \mathcal{A} \mid \mathsf{pre}(a) = \{\langle v, u\rangle, x \in (-\infty, w_a]\}, M_x^- \le w_a, \langle v, u\rangle \in \mathcal{C}_j\}, \text{where}$$

$$\mathcal{A}_x^0 = \{a \in \mathcal{A} \mid \mathsf{pre}(a) = \{x \in (-\infty, w_a]\}, M_x^- \le w_a\}.$$

Actions in $\mathcal{A}_x^j$ affect x and can be applied interchangeably when variable v has a value in $\mathcal{C}_j$. The ILP $\heartsuit$ below is formulated as a maximization problem on the number of times n_a^j an action a with additive effect c_a is applied while v has a value in $\mathcal{C}_j$:

$$\max f(n) := \sum_{j=1}^{m} \sum_{a\in\mathcal{A}_x^j : c_a < 0} n_a^j c_a,$$

$$\text{s.t.} \sum_{j=1}^{m} \sum_{a\in\mathcal{A}_x^j} n_a^j c_a = b - a, \qquad (\heartsuit)$$

$$\sum_{j=1}^{k} \sum_{a\in\mathcal{A}_x^j} n_a^j c_a \le M_x^- \qquad (\forall k \in [m-1]),$$

$$n_a^j \in \mathbb{N} \qquad (\forall a \in \mathcal{A}, j \in [m]).$$

We can apply the actions in $\mathcal{A}_x^i$ in any order (modulo the applications of inner actions of v). By Lem. 4, if the maximization problem has a solution, there is a sequence of actions that leads from $a := s_{k_1}[x]$ to $b := s_{k_2}[x]$, where the value of x at states along this sequence is below M_x^-. We maximize $f(n)$ since all considered c_a's are negative.

Let n^* be the optimal solution for the ILP problem. By construction, any ordering of action applications can reach a point that is less than $f(n^*)$ so we set $\mathcal{M}_x^{-,\vec{\mathcal{C}},a,b} := f(n^*)$. By solving this problem for all possible $\vec{\mathcal{C}}$, a, and b, we obtain the lower bound L_x^-. The number of optimization problems we need to solve to obtain L_x^- is then $2^k(C_x^{\max})^2$, where k is the number of SCCs in the domain of v.

The solution for L_x^+ is almost the same but we replace $\le$ with $\ge$ and $\max$ with $\min$ in the optimization problems. □

This result lets us analyze more general planning tasks with multiple numeric leaves. Let $\Pi_{v,x_1,\ldots,x_n}$ be an RT with a propositional variable v and numeric leaf variables x_i. One can transform $\Pi_{v,x_1,\ldots,x_n}$ into an integer RT (Cor. 1) with bounded numeric goal conditions (Lem. 2) in polynomial time (Cor. 1). We denote such transformed tasks by FRT. To show that PEFRT is in **PSPACE**, we need the following result by Papadimitriou (1981, Lem. 4). We assume that the coefficients in integer linear programs (ILP) are integers. The lemma holds for both maximisation and minimisation problems via multiplication of the objective function by -1.

Lemma 7 (Papadimitriou 1981)**.** *Assume that the following ILP is feasible and bounded:* $\max\ c' \cdot x$ *s.t.* $Ax \le b, x \in \mathbb{N}^t$. *Then, its optimal solution* z^* *satisfies* $|z^*| \le M \cdot \sum_{i=1}^{t} |c_i|$. *Here,* $M = t^2(ma^2)^{2m+3}$, *where* $m \times t$ *is the size of the integer program, and* $a = \max_{i\in[t],j\in[m]}\{|a_{i,j}|, |b_i|\}$ *is a bound on the sizes of numbers in the program.*

Theorem 3. *PEFRT is in* ***PSPACE****.*

Proof. We employ the same machinery as in the proof of Lem. 6. In particular, we reuse the ILP $\heartsuit$ and its associated notation. Let $\Pi_{v,x_1,\ldots,x_n}$ be an integer RT with bounded goal conditions. Let $\|\Pi_{v,x_1,\ldots,x_n}\| = n$. Thus, $|\mathcal{D}(v)| \le n$. Let further $\vec{\mathcal{C}} : \mathcal{C}_1 \to \mathcal{C}_2 \to \cdots \to \mathcal{C}_m$ be a path through the SCCs in the domain of v. To show the claim we iterate over all such paths – there are at most 2^n of them. We fix one $\vec{\mathcal{C}}$ at a time, bound the domains of the numeric variables using the ILP from Lem. 6, and check the solvability of the bounded task. If $\Pi_{v,x_1,\ldots,x_n}$ has a solution, it must traverse a $\vec{\mathcal{C}}$. Such a plan π needs to respect $\vec{\mathcal{C}}$, so $\langle v, u\rangle \in s_0 \cap \mathcal{C}_1$, and, if $\langle v, u'\rangle \in G$, we also require $\langle v, u'\rangle \in \mathcal{C}_m$. For a given $\vec{\mathcal{C}}$ we ignore the inner actions of v that do not obey $\vec{\mathcal{C}}$.

With a fixed $\vec{\mathcal{C}}$, we can use Lem. 6 to individually bound the domains of x_i's. For each $x \in \{x_i\}_{i=1}^n$ the bound from below for a $\vec{\mathcal{C}}$ is given by

$$L_x^{-,\vec{\mathcal{C}}} = \min_{a,b\in[M_x^- - C_x^{\max}, M_x^-]} \mathcal{M}_x^{-;\vec{\mathcal{C}},a,b},$$

and this bound can be computed by solving $(C_x^{\max})^2$ ILPs of the type $\heartsuit$. We have $C_x^{\max} \le 2^n$ since the input has n bits. Iterating over all possible a and b we solve the ILP problem $\heartsuit$. Note that all constants except $b-a$ in the ILP come from the definition of the problem, and $|b-a| \le C_x^{\max} \le 2^n$. The solution of the problem is bounded from above by 0 ($c_a < 0$ and $n_a \ge 0$) so if the program is feasible it has an optimal solution $z^* \in \mathbb{Q}$. Now, we apply Lem. 7. The number of constraints in $\heartsuit$ equals the number of SCCs traversed by $\vec{\mathcal{C}}$, and is bounded by n. The number of variables in $\heartsuit$ is bounded by $|\vec{\mathcal{C}}| \cdot |\mathcal{A}| \le n^2$. Lem. 7 now implies that

$$|z^*| \le n^4(n2^{2n})^{2n+3} \cdot \sum_{i=1}^{n} 2^n = n^{2n+8}2^{4n^2+7n}.$$

Thus, we can bound $|z^*|$ by 2^{4n^2+8n} for sufficiently large n.

To obtain an upper bound on the values taken by x, we compute $L_x^{+,\vec{\mathcal{C}}}$ in the same fashion. Hence, the domain of the numeric variable x under the path $\vec{\mathcal{C}}$ is of the size 2^{4n^2+8n+1}. We conclude that the size of the whole state space given the path $\vec{\mathcal{C}}$ can be bounded by

$$|\mathcal{D}(v)| \cdot \Pi_{i=1}^{n} |\mathcal{D}(x_i)| \le n2^{\sum_{i=1}^{n} 4n^2+8n+1} = n2^{4n^3+8n^2+n}$$

To account for $\vec{\mathcal{C}}$, we multiply by the overall number of directed paths $|\vec{\mathcal{C}}| \le 2^n$. Hence, the state space contains less than 2^{5n^3} vertices for large n. Non-deterministic search combined with Thm. 1 implies that a solution to the task $\Pi_{v,x_1,\ldots,x_n}$ can be computed in polynomial space. □

Numeric Leaves

We finally consider tasks where all numeric variables are CG leaves. We can transform such a task into its integer form with bounded goal condition for all numeric variables. We denote this set of instances by NLRT.

Theorem 4. *PENLRT is decidable.*

Proof. Checking plan existence for an RT with a fork-structured CG with numeric leaves is in **PSPACE** by Thm. 3. We show that we can transform an NLRT to an FRT using exponential space. This does not preserve **PSPACE** membership but it shows that PENLRT is decidable.

Let us look at all partial propositional states of the task Π, $\mathcal{S}(\mathcal{V}_p) := \times_{v \in \mathcal{V}_p} \mathcal{D}(v)$. We can replace all propositional variables in Π with a single variable $\tilde{v}$ s.t. $\mathcal{D}(\tilde{v}) = \mathcal{S}(\mathcal{V}_p)$. Each action that affects a $\tilde{v} \in \mathcal{V}_p$ is transformed to an inner action of $\tilde{v}$. The transformation may produce an exponential number of actions, since $\mathcal{S}(\mathcal{V}_p)$ is a projection of the state transition graph on $\mathcal{V}_p$, and thus may have exponential number of partial states and inner actions in the size of $|\mathcal{V}_p|$.

The actions that affect a variable $x \in \mathcal{V}_n$ do not affect other variables in $\mathcal{V}$, since this was already the case in the original task. Hence, the transformed task has one propositional CG root variable $\tilde{v}$ with a potentially exponential size domain, and numeric variables $x_1, \ldots, x_n$ that may depend on $\tilde{v}$. Hence, each $x \in \mathcal{V}_n$ is either a CG-leaf, or a singleton. Since separated components of a CG can be solved separately, we invoke Thm. 3 to check for the decidability of the fork, and Thm. 2 for the decidability of singletons. □

Our last result shows **PSPACE**-membership if we fix the number of propositional variables. We let k-PENLRT denote PENLRT with at most k propositional variables.

Theorem 5. *k-PENLRT is in **PSPACE**.*

Proof. Let Π be an NLRT where $\|\Pi\| = n$ and $|\mathcal{V}_p| = k$. Let $\mathcal{S}(\mathcal{V}_p)$ be the domain of the new CG-root variable as introduced in the proof of Thm. 4. Note that since the domains of our propositional variables are not bounded we have that $|\mathcal{S}(\mathcal{V}_p)| \in O(n^k)$. This is due to the fact that for each $v \in \mathcal{V}_p$ we have $|\mathcal{D}(v)| \leq n$. The number of actions in the new task is $O(|A| \cdot n^k) = O(n^{k+1})$, since every new action may have at most one precondition in $\mathcal{S}(\mathcal{V}_p)$. By plugging this number into the last part of the proof of Thm. 3, we have that the state space of the transformed task is $2^{O(n^{k+c})}$ for some small universal constant $c \in \mathbb{N}$. The overall state space of the bounded numeric RT is bounded by $|\mathcal{S}(\mathcal{V}_p)| \cdot \max_{i \in [n]} |\mathcal{D}(x_i)|^n$ so we need to bound $|\mathcal{D}(x_i)|$ for $x_i \in \mathcal{V}_n$. As before, we apply Lem. 7 to the ILP $\heartsuit$ from the proof of Lem. 6. In our case, the number of constraints is equal to the number of SCCs of $\mathcal{S}(\mathcal{V}_p)$, $m := n^k$. The number of variables is the number of actions in the new task times the number of SCCs, that is $t := n^{2k+1}$. The maximal constant of the problem is $a := 2^n$. Thus, the size of the state space is bounded by

$$2t^2(ma^2)^{2m+3} = 2 \cdot n^{2k+1}(n^k 2^{2n})^{2n^k+3} \subseteq 2^{O(n^{k+1})}.$$

The multiplication by 2 takes into account that we solve two ILPs: one for the positive bound and one for the negative bound. Take this bound to the power of n so that we cover all numeric variables and we see that the universal constant $c = 2$, since $(2^{O(n^{k+1})})^n = 2^{nO(n^{k+1})} = 2^{O(n^{k+2})}$. Thus, by Thm. 1, k-PENLRT is in **PSPACE**. □

The generalization to multiple propositional variables has important practical implications. It allows us to augment arbitrary classical planning tasks with many numeric leaf variables, without affecting the computational complexity.

Discussion

We analyzed the impact of the causal structure of planning tasks on the complexity of simple numeric planning (SNP). As a basis for our more advanced results, we investigated the case of a single numeric variable, showing that it is in **PSPACE**. With this, we make three important contributions: (1) we show that plan existence for SNP is in **PSPACE** if all numeric variables are CG root nodes; (2) the problem is decidable if all numeric variables are leaf nodes in the CG, and (3) if additionally the number of propositional variables is fixed, it also is in **PSPACE**. Hence, we identified substantial fragments of SNP that are no harder than classical planning.

The complexity landscape is still highly unchartered, though, but our toolbox seems powerful enough for further progress. Such progress is indeed needed in order to capture established numeric benchmarks, both from the IPC and beyond. It turns out that there is no domain that in its current formulation completely falls into our fragments. A common pattern in many domains is pairs of numeric variables that are cross-dependent, often because there exist actions that change both at the same time. Such patterns cannot be captured by our restrictions. On the positive side, parts of established domains can easily be reformulated to fall into a fragment. Consider, e.g., plant watering, where an agent moves on a grid. The movement in its current form cannot be captured due to diagonal moves, which change both the x and y-position of the agent. Dropping diagonal moves solves the issue, and more generally moving on any kind of (higher-dimensional) grid where every action only changes the position along a single axis falls into our fragments. Accordingly, classical domains that contain such structures can be more naturally encoded using numeric planning, e.g. in elevators, floortile, or visitall, avoiding the introduction of multiple "number objects" and having to statically connect them using a "sum" predicate.

Given this, it is currently more reasonable to use our fragments for heuristic computation. For instance, it should be fairly straightforward to produce a pattern database heuristic where the pattern includes variables such that the abstraction falls into a fragment. This can be achieved by starting with all variables in the pattern and greedily removing variables that cause cycles in the causal graph until the pattern satisfies our restrictions. Then, we can bound the size of the abstract state space and potentially obtain an informative heuristic, in particular when combining multiple such patterns additively, as is common in classical planning. There is probably a great potential for heuristics based on bounded fragments of numeric planning.

Acknowledgements

The work of Alexander Shleyfman was partially supported by the Israel Academy of Sciences and Humanities program for Israeli postdoctoral researchers. Daniel Gnad was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and by TAILOR, a project funded by the EU Horizon 2020 research and innovation programme under grant agreement no. 952215. Peter Jonsson was partially supported by the Swedish Research Council (VR) under grant 2021-04371.

References

Aldinger, J.; and Nebel, B. 2017. Interval Based Relaxation Heuristics for Numeric Planning with Action Costs. In *SOCS*, 155–156.

Bacchus, F.; and Yang, Q. 1994. Downward Refinement and the Efficiency of Hierarchical Problem Solving. *AIJ*, 71(1): 43–100.

Bäckström, C.; and Jonsson, P. 2013. A Refined View of Causal Graphs and Component Sizes: SP-Closed Graph Classes and Beyond. *Journal of Artificial Intelligence Research*, 47: 575–611.

Bäckström, C.; and Nebel, B. 1995. Complexity Results for SAS^+ Planning. *Computational Intelligence*, 11(4): 625–655.

Brafman, R. I.; and Domshlak, C. 2003. Structure and Complexity in Planning with Unary Operators. *JAIR*, 18: 315–349.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2013. A Hybrid LP-RPG Heuristic for Modelling Numeric Resource Flows in Planning. *JAIR*, 46: 343–412.

Domshlak, C.; and Dinitz, Y. 2001. Multi-Agent Off-line Coordination: Structure and Complexity. In *ECP*, 277–288.

Domshlak, C.; and Nazarenko, A. 2013. The Complexity of Optimal Monotonic Planning: The Bad, The Good, and The Causal Graph. *JAIR*, 48: 783–812.

Eyerich, P.; Mattmüller, R.; and Röger, G. 2009. Using the Context-Enhanced Additive Heuristic for Temporal and Numeric Planning. In *ICAPS*, 130–137.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artif. Intell.*, 2: 189–208.

Gerevini, A.; Saetti, A.; and Serina, I. 2008. An Approach to Efficient Planning with Numerical Fluents and Multi-Criteria Plan Quality. *Artif. Intell.*, 172(8-9): 899–944.

Giménez, O.; and Jonsson, A. 2008. The Complexity of Planning Problems With Simple Causal Graphs. *JAIR*, 31: 319–351.

Gnad, D.; Helmert, M.; Jonsson, P.; and Shleyfman, A. 2023. Planning over Integers: Compilations and Undecidability. In *ICAPS*.

Helmert, M. 2002. Decidability and Undecidability Results for Planning with Numerical State Variables. In *AIPS*, 303–312.

Helmert, M. 2004. A Planning Heuristic Based on Causal Graph Analysis. In *ICAPS*, 161–170.

Helmert, M. 2009. Concise Finite-Domain Representations for PDDL Planning Tasks. *Artif. Intell.*, 173: 503–535.

Hoffmann, J. 2002. Extending FF to Numerical State Variables. In *ECAI*, 571–575.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating "Ignoring Delete Lists" to Numeric State Variables. *JAIR*, 20: 291–341.

Illanes, L.; and McIlraith, S. A. 2017. Numeric Planning via Abstraction and Policy Guided Search. In *IJCAI*, 4338–4345.

Jonsson, P.; and Bäckström, C. 1998. State-Variable Planning under Structural Restrictions: Algorithms and Complexity. *AIJ*, 100(1–2): 125–176.

Knoblock, C. A. 1994. Automatically Generating Abstractions for Planning. *AIJ*, 68(2): 243–302.

Kuroiwa, R.; Shleyfman, A.; Piacentini, C.; Castro, M. P.; and Beck, J. C. 2021. LM-cut and Operator Counting Heuristics for Optimal Numeric Planning with Simple Conditions. In *ICAPS*, 210–218.

Li, D.; Scala, E.; Haslum, P.; and Bogomolov, S. 2018. Effect-Abstraction Based Relaxation for Linear Numeric Planning. In *IJCAI*, 4787–4793.

Papadimitriou, C. H. 1981. On the Complexity of Integer Programming. *Journal of the ACM*, 28(4): 765–768.

Piacentini, C.; Castro, M. P.; Ciré, A. A.; and Beck, J. C. 2018a. Compiling Optimal Numeric Planning to Mixed Integer Linear Programming. In *ICAPS*, 383–387.

Piacentini, C.; Castro, M. P.; Ciré, A. A.; and Beck, J. C. 2018b. Linear and Integer Programming-Based Heuristics for Cost-Optimal Numeric Planning. In *AAAI*, 6254–6261.

Savitch, W. J. 1970. Relationships Between Nondeterministic and Deterministic Tape Complexities. *Journal of Computer and System Sciences*, 4(2): 177–192.

Scala, E.; Haslum, P.; Magazzeni, D.; and Thiébaux, S. 2017. Landmarks for Numeric Planning Problems. In *IJCAI*, 4384–4390.

Scala, E.; Haslum, P.; and Thiébaux, S. 2016. Heuristics for Numeric Planning via Subgoaling. In *IJCAI*, 3228–3234.

Scala, E.; Ramírez, M.; Haslum, P.; and Thiébaux, S. 2016. Numeric Planning with Disjunctive Global Constraints via SMT. In *ICAPS*, 276–284.

Shin, J.; and Davis, E. 2005. Processes and Continuous Change in a SAT-Based Planner. *Artif. Intell.*, 166(1-2): 194–253.

Shleyfman, A.; Gnad, D.; and Jonsson, P. 2023. Supplementary Material for "Structurally Restricted Fragments of Numeric Planning – a Complexity Analysis". https://zenodo.org/record/7704863. Accessed: 2023-03-07.

Shleyfman, A.; Kuroiwa, R.; and Beck, J. C. 2023. Symmetry Detection and Breaking in Cost-Optimal Numeric Planning. In *ICAPS*.