ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230409

Extracting and Exploiting Bounds of Numeric Variables for Optimal Linear Numeric Planning

Ryo Kuroiwa[a;*], **Alexander Shleyfman**[b] **and J. Christopher Beck**[a]

[a]Department of Mechanical and Industrial Engineering, University of Toronto
[b]Department of Computer Science, Bar-Ilan University
ORCiD ID: Ryo Kuroiwa https://orcid.org/0000-0002-3753-1644,
Alexander Shleyfman https://orcid.org/0000-0001-9187-2354,
J. Christopher Beck https://orcid.org/0000-0002-4656-8908

Abstract. In numeric AI planning, a state is represented by propositions and numeric variables, actions change the values of numeric variables in addition to adding and deleting propositions, and goals and preconditions of actions may include conditions over numeric variables. While domains of numeric variables are rational numbers in general, upper and lower bounds on variables affected only by constant increase and decrease can sometimes be determined and exploited by a heuristic function. In this paper, we generalize the existing method to variables that are changed by linear effects. We exploit the extracted bounds to improve the numeric LM-cut heuristic, a state-of-the-art admissible heuristic for linear numeric planning. Empirical evaluation shows that our method improves the performance of LM-cut in multiple domains. The proposed method can also detect unsolvability of some numeric tasks in polynomial time.

1 Introduction

While in the past two decades classical planning has been widely studied by the AI planning community, formalisms that can address more realistic planning problems have received much less attention. A case in point is numeric planning, an extension of classical planning with numeric state variables, that has recently had a resurgence in popularity. While early research focused only on the satisficing setting, where the objective is to find an executable plan regardless of its length or cost [8, 3], recent work has developed techniques to solve numeric planning problems optimally using model-based approaches [16, 13] and heuristic search [21, 19, 17, 12, 10]. In particular, heuristic search approaches use A^* search [5] with admissible heuristic functions, which compute a lower bound of the optimal cost, and achieve state-of-the-art performance.

Although the domains of numeric variables are unbounded in general, Coles at al. [3] proposed a method to identify lower and upper bounds in some special cases. The extracted bounds are exploited by heuristic functions in both satisficing and optimal settings [3, 17]. The proposed method, however, is applicable only when the changes in numeric variables are given by constants.

In this paper, we generalize the existing method of extracting the bounds of numeric variables to linear numeric planning, where both conditions and effects on numeric variables are represented by linear formulas. We obtain bounds on numeric variables using inductive equations and prove that their fixed point yields valid bounds. While such bounds can be useful for both model-based and heuristic search approaches, we incorporate them in LM-cut [10], a state-of-the-art admissible heuristic for linear numeric planning, without loss of the admissibility. We empirically show that the use of the bounds improves the performance of LM-cut in almost all of the existing numeric planning domains. We also introduce a new linear planning domain to highlight the benefits of our method.

* Corresponding Author. Email: ryo.kuroiwa@mail.utoronto.ca.

2 Numeric Planning

Linear numeric planning is a subset of PDDL 2.1 [4] (see the PICKUP domain explained in Sec. 5.1 and supplementary materials [11]). A task is represented by a 5-tuple $\langle \mathcal{F}, \mathcal{N}, \mathcal{A}, s^0, G \rangle$, where $\mathcal{F}$ is the set of *propositions*, $\mathcal{N}$ is the set of *numeric variables*, s^0 is the *initial state*, and G is the set of *goal conditions*. A *state* s is a pair $\langle s_p, s_n \rangle$, where $s_p \subseteq \mathcal{F}$ is a set of propositions, and $s_n : \mathcal{N} \to \mathbb{Q}$ is a value assignment to the numeric variables. We denote the value of a numeric variable v in s by $s[v]$. The initial state s^0 is a state. A *numeric condition* ψ is a linear formula $\sum_{v \in \mathcal{N}} w_v^\psi v \trianglerighteq w_0^\psi$ with $\forall v \in \mathcal{N}, w_v^\psi \in \mathbb{Q}$, $w_0^\psi \in \mathbb{Q}$, and $\trianglerighteq \in \{\geq, >\}$. The condition ψ is *satisfied* in a state s if $\sum_{v \in \mathcal{N}} w_v^\psi s[v] \trianglerighteq w_0^\psi$, denoted by $s \models \psi$. The goal conditions $G = \langle G_p, G_n \rangle$ is a pair of a set of propositions $G_p \subseteq \mathcal{F}$ and a set of numeric conditions G_n, and a state s is a *goal state* if $G_p \subseteq s_p$ and $\forall \psi \in G_n, s \models \psi$.

An action $a \in \mathcal{A}$ is a triplet $\langle \mathsf{pre}(a), \mathsf{eff}(a), \mathsf{cost}(a) \rangle$, where $\mathsf{pre}(a) = \langle \mathsf{pre}_p(a), \mathsf{pre}_n(a) \rangle$ is the set of preconditions, $\mathsf{eff}(a) = \langle \mathsf{add}(a), \mathsf{del}(a), \mathsf{num}(a) \rangle$ is the set of effects, and $\mathsf{cost}(a) \in \mathbb{Q}_0^+$ is the cost. Propositional preconditions $\mathsf{pre}_p(a) \subseteq \mathcal{F}$ are a set of propositions, numeric preconditions $\mathsf{pre}_n(a)$ are a set of numeric conditions, and a is *applicable* if $\mathsf{pre}_n(a) \subseteq s_p$ and $\forall \psi \in \mathsf{pre}_n(a), s \models \psi$. In the set of effects, $\mathsf{add}(a)$ is a set of propositions to add, $\mathsf{del}(a)$ is a set of propositions to remove, and $\mathsf{num}(a)$ is a set of numeric effects on variables $u \in \mathcal{N}$ in the form of $u \mathrel{+}= \sum_{v \in \mathcal{N}} w_v^{a,u} v + w_0^{a,u}$ with $\forall v \in \mathcal{N}, w_v^{a,u} \in \mathbb{Q}$ and $w_0^{a,u} \in \mathbb{Q}$. Alternately, we can represent the effect as $u := (w_u^{a,u} + 1)u + \sum_{v \in \mathcal{N} \setminus \{u\}} w_v^{a,u} v + w_0^{a,u}$. We assume that an action has at most one numeric effect on a given numeric variable. When a is applied in a state s, it transitions to a *successor state* $s[\![a]\!]$, where $s[\![a]\!]_p = (s_p \setminus \mathsf{del}(a)) \cup \mathsf{add}(a)$ and $s[\![a]\!][u] = s[u] + \sum_{v \in \mathcal{N}} w_v^{a,u} s[v] + w_0^{a,u}$.

Given a state s, an *s-plan* is a sequence of actions $\pi = \langle a_1, \ldots, a_m \rangle$ such that for $i \in [m] := \{1, \ldots, m\}$, action a_i is applicable in state s^{i-1} where $s^i = s^{i-1}[\![a_i]\!]$ with $s^0 := s$, and $s^m \models G$ is a goal state. The cost of π is $\mathsf{cost}(\pi) = \sum_{i=1}^{m} \mathsf{cost}(a_i)$, and an *optimal s-plan* minimizes its cost. A solution for a linear numeric planning task is an s^0-plan, and *optimal planning* is to find an optimal s^0-plan. The cost of an optimal s-plan is denoted $h^*(s)$.

For the effect of an action a on a numeric variable u, when $\forall v \in \mathcal{N}, w_v^{a,u} = 0$, i.e., in the form of $u \mathrel{+}= w_0^{a,u}$, it is called a *simple effect*. Numeric variables affected only by simple effects are called *simple numeric variables*. When all numeric effects are simple effects, a task is called a *simple numeric planning task*. When no numeric variables exist, a task is called a *classical planning task*.

3 Extracting Bounds in Numeric Planning

Bounds of numeric variables can be useful in heuristic search and model-based methods. For example, the LM-cut heuristic for numeric planning (see Sec. 4) underestimates the cost to achieve a numeric condition $\psi : \sum_{v \in \mathcal{N}} w_v^\psi v \geq w_0^\psi$ using an action a [10], and more accurate estimation results in a better heuristic. If a has effect $u \mathrel{+}= \sum_{v \in \mathcal{N}} w_v^{a,u} v + w_0^{a,u}$ on variable u in ψ with $w_u^\psi > 0$, LM-cut needs to overestimate the effect to underestimate the cost to achieve ψ using a. Unless the effect is simple, i.e., $u \mathrel{+}= w_0^\psi$, it needs to compute an upper bound on a linear formula $\sum_{v \in \mathcal{N}} w_v^{a,u} v$. LM-cut can obtain an upper bound for some special cases, e.g., where all variables v with $w_v^{a,u} \neq 0$ are simple variables. However, in the worst case, the linear formula is overestimated by ∞, assuming that ψ is achieved by applying a only once. If we know finite upper bounds $\overline{v}$ for $w_v^{a,u} > 0$ and lower bounds $\underline{v}$ for $w_v^{a,u} < 0$, we can obtain a better estimation $\sum_{v \in \mathcal{N}: w_v^{a,u} > 0} w_v^{a,u} \overline{v} + \sum_{v \in \mathcal{N}: w_v^{a,u} < 0} w_v^{a,u} \underline{v}$. In another example, in a mixed-integer programming model for numeric planning, $s^i[v]$, the value of variable v in the i th state achieved by a plan, is represented by a decision variable [16]. A tighter bound for a decision variable results in a tighter linear relaxation.

We say that $\overline{v} \in \mathbb{Q}$ is an upper bound of v, if for any state s that lies along any plan π it holds that $s[v] \leq \overline{v}$. Similarly, $\underline{v} \in \mathbb{Q}$ is a lower bound of v if $s[v] \geq \underline{v}$ always holds. Below, we introduce an existing method to extract the upper and lower bounds of simple numeric variables [3]. We adapt this method to a more general setting.

In what follows, we only consider numeric conditions in the form $\psi : \sum_{v \in \mathcal{N}} w_v^\psi s[v] \geq w_0^\psi$; if all $v \in \mathcal{N}$ with $w_v^\psi \neq 0$ are simple variables, we can normalize $\sum_{v \in \mathcal{N}} w_v^\psi s[v] > w_0^\psi$ to $\sum_{v \in \mathcal{N}} w_v^\psi s[v] \geq w_0^\psi + \frac{1}{M}$, where M is an integer such that $M w_0^\psi$, $M w_v^\psi$ for all v, and $M w_0^{a,v}$ for all $a \in \mathcal{A}$ and $v \in \mathcal{N}$ with $w_v^\psi \neq 0$ are integer. If there exists a non-simple variable v with $w_v^\psi \neq 0$, we relax the condition to $\sum_{v \in \mathcal{N}} w_v^\psi v \geq w_0^\psi$. This condition is always satisfied when the original condition is satisfied. In what follows, when a numeric condition is satisfied, we assume that its normalized or relaxed counterpart is satisfied and use $\geq$ instead of $>$.

3.1 Bounds on Simple Numeric Variables

For simple numeric variables, Coles et al. [3] defined the notion of a producer and a consumer and proposed a method to extract bounds of numeric variables.

Definition 1 (Bounded producer and consumer [3]) *Let an action a have a simple effect on v, $v \mathrel{+}= w_0^{a,v}$. If the effect is positive, i.e., $w_0^{a,v} > 0$, and a has a precondition in the form of $-v \geq w_0^\psi$, then a is a bounded producer of v. If the effect is negative, i.e., $w_0^{a,v} < 0$, and a has a precondition in the form of $v \geq w_0^\psi$, then a is a bounded consumer of v.*

Let $\overline{\mathcal{A}}(v)$ and $\underline{\mathcal{A}}(v)$ be the sets of bounded producers and consumers of a variable v, respectively. When all actions having positive effects on v are bounded producers, then the upper bound of v is

$$\overline{v} = \max \left\{ s^0[v], \max_{a \in \overline{\mathcal{A}}(v)} \min_{\psi \in \mathsf{pre}_n(a): -v \geq w_0^\psi} \{-w_0^\psi + w_0^{v,a}\} \right\}.$$

When all actions having negative effects on v are bounded consumers, then the lower bound of v is

$$\underline{v} = \min \left\{ s^0[v], \min_{a \in \underline{\mathcal{A}}(v)} \max_{\psi \in \mathsf{pre}_n(a): v \geq w_0^\psi} \{-w_0^\psi + w_0^{v,a}\} \right\}.$$

3.2 Bounds in Linear Numeric Planning

In this section, we extend the above method to linear numeric planning tasks. Since we aim to compute bounds on the numeric variables of the task Π, we ignore its propositional component. As all numeric preconditions are linear inequalities, we can view the numeric domain where each action a is applicable as a polytope, which may not necessarily be unbounded. Consequently, each linear effect of a may have an upper or lower bound. We propose a method to approximate these bounds when they exist. To avoid the need to solve linear programs at each step, we restrict the LP constraints to *axis-parallel boxes*, or "boxes" for short. Boxes are polytopes where each face is defined by inequalities of the form $x \leq c$ or $x \geq c$. Computing a max/min of a linear function on a box is linear in the number of numeric variables involved.

We start by setting the upper bound and lower bounds on each numeric variable $v \in \mathcal{N}$ to be $\overline{v}^0 := \infty$ and $\underline{v}^0 := -\infty$. We intend to update these bounds iteratively. We assume that $\infty + c = \infty$ and $-\infty + c = -\infty$, $c \cdot \infty = \infty$ if $c > 0$, and $c \cdot (-\infty) = \infty$ if $c < 0$. The $-\infty$ behaves similarly under multiplication by a constant. In what follows we do not use $-\infty + \infty$. We define the bounds of the numeric variables of the task in iteration i as a box $\mathsf{B}^i = \times_{v \in \mathcal{N}} [\underline{v}^i, \overline{v}^i]$, where $\underline{v}^i$ and $\overline{v}^i$ are an upper and a lower bound on v. For completeness, we assume that 0 times infinity is always 0.[1]

Let $\overline{v}_a$ and $\underline{v}_a$ be upper and lower bounds on the domain of v where the action a can be applied. For example, in the case discussed by Coles et al. [3], the action a with precondition $\mathsf{pre}(a) = \{v \geq 1\}$ has the bounds $\underline{v}_a = 1$ and $\overline{v}_a = \infty$. Since our bounds are computed iteratively we denote them by $\underline{v}_a^i$ and $\overline{v}_a^i$ for each iteration $i \in \mathbb{N}$. We define $\mathsf{preB}_a^i := \times_{v \in \mathcal{N}} [\underline{v}_a^i, \overline{v}_a^i]$ to be the minimal $|\mathcal{N}|$-dimensional box that contains the intersection of the polytopes $\mathsf{pre}_n(a)$ and B^{i-1}. We aim to derive tighter bounds on $\overline{v}_a$ and $\underline{v}_a$, using the linear preconditions of a. Let $\psi \in \mathsf{pre}_n(a)$ be a precondition of the form $\sum_{v \in \mathcal{N}} w_v^\psi v \geq w_0^\psi$. Let $u \in \mathcal{N}$ be a numeric variable s.t. $w_u^\psi \neq 0$. For a to be applicable, the following condition on u must hold

$$w_u^\psi u + \sum_{v \neq u: w_v^\psi > 0} w_v^\psi \overline{v}_a^{i-1} + \sum_{v \neq u: w_v^\psi < 0} w_v^\psi \underline{v}_a^{i-1} \geq w_0^\psi.$$

[1] The intervals belong to $[-\infty, \infty]$ – the extended real line – a compactification of $\mathbb{R}$. $[-\infty, \infty] = \{-\infty\} \cup \mathbb{R} \cup \{\infty\}$, where for each $x \in \mathbb{R}$ it holds that $-\infty < x < \infty$.

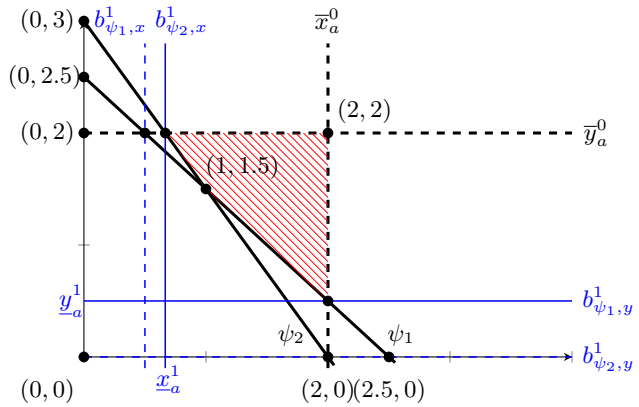

Figure 1. Suppose that an action a has as its preconditions the linear inequalities $\psi_1 : x + y \geq 2.5$ and $\psi_2 : 3x + 2y \geq 6$. The points that obey this inequalities lie to the right of the black lines in the picture. Suppose that in the previous iteration we determined that $\overline{x}^0_a = 2$, $\overline{y}^0_a = 2$, $\underline{x}^0_a = -\infty$, and $\underline{y}^0_a = -\infty$. The dashed lines represent the respective upper bounds $\overline{x}_a$ and $\overline{y}_a$. Using this we can compute that $b^1_{\psi_1,y} = 0.5$, $b^1_{\psi_2,y} = 0$ and that $b^1_{\psi_1,x} = 0.5$, $b^1_{\psi_2,x} = 2/3$. Thus, using these values we can determine that $\underline{x}^1_a = 2/3$ and $\underline{y}^1_a = 0.5$ (denoted by blue lines). Note that the box $\mathsf{preB}^1_a = [2/3, 2] \times [0.5, 2]$ is the smallest box possible to include the set of all points where a can be applied – the red area on the graph.

Thus, if the appropriate $\overline{v}^{i-1}_a$'s and $\underline{v}^{i-1}_a$'s are finite we can derive the upper or the lower bound on u, depending on the sign of w^ψ_u, i.e.,

$$b^i_{\psi,u} = \frac{\sum_{v \neq u : w^\psi_v > 0} w^\psi_v \overline{v}^{i-1}_a + \sum_{v \neq u : w^\psi_v < 0} w^\psi_v \underline{v}^{i-1}_a - w^\psi_0}{-w^\psi_u}. \quad (1)$$

To apply a, we need the bounds to hold altogether, thus

$$\overline{v}^i_a = \min\left\{\overline{v}^{i-1}, \min_{\psi \in \mathsf{pre}_n(a) : w^\psi_v < 0} b^i_{\psi,v}\right\}, \quad (2)$$

$$\underline{v}^i_a = \max\left\{\underline{v}^{i-1}, \max_{\psi \in \mathsf{pre}_n(a) : w^\psi_v > 0} b^i_{\psi,v}\right\}. \quad (3)$$

Thus, we recursively defined preB^i_a using preB^{i-1}_a and B^{i-1} (for a brief example see Fig. 1).

We now calculate the numerical bounds for the effects of a based on preB^i_a. For the purpose of bound computation we transfer all numeric effects into their assignment form. For the general linear effect of a on $u \in \mathcal{N}$: $u \mathrel{+}= \xi \iff u := u + \xi$, where $\xi = \sum_{v \in \mathcal{N}} w^{a,u}_v v + w^{a,u}_0$. If action a does not effect the variable u – for the purpose of bound computation – we add the nominal effect $u := u$. Transforming increment effects into assignment effects may enable tighter bounds. For example, if we did not obtain any bounds on u under the application of a, i.e., $\overline{u}^i_a = \infty$ and $\underline{u}^i_a = -\infty$, the increment effect $u \mathrel{+}= -u + 2$ will result the bounds on u being ∞ and a $-\infty$. However, the assignment effect $u := 2$ will result in the upper and lower bounds of 2. Thus, for the rest of this section we replace all additive linear effects with assignment effects.

For the effect $(u := u + \xi) \in \mathsf{eff}_n(a)$, we compute the bounds

$$\overline{\xi}^i_u = \sum_{v \neq u : w^{a,u}_v > 0} w^{a,u}_v \overline{v}^i_a + \sum_{v \neq u : w^{a,u}_v < 0} w^{a,u}_v \underline{v}^i_a + w^{a,u}_0$$

$$\underline{\xi}^i_u = \sum_{v \neq u : w^{a,u}_v > 0} w^{a,u}_v \underline{v}^i_a + \sum_{v \neq u : w^{a,u}_v < 0} w^{a,u}_v \overline{v}^i_a + w^{a,u}_0.$$

For brevity we transform these into bounds on assigment effects. If $w^{a,u}_u \geq -1$, an upper and a lower bound on u made by are given by:

$$\overline{asn}^i_{a,u} = (w^{a,u}_u + 1)\overline{u}^i_a + \overline{\xi}^i_u, \quad (4)$$

$$\underline{asn}^i_{a,u} = (w^{a,u}_u + 1)\underline{u}^i_a + \underline{\xi}^i_u. \quad (5)$$

If $w^{a,u}_u < -1$, $\overline{u}^i_a$ and $\underline{u}^i_a$ are swapped in the equations above.

Furthermore, if the linear formula of the effect is used in a precondition of the action, we can exploit the precondition to derive bounds of the effect. For example, the precondition $2x + 2y \leq 3$ imposes an upper bound of $\frac{3}{2}$ on u under the effect $u := x + y$.

Thus, let us assume that some $\psi \in \mathsf{pre}(a)$ is of the form $\psi : r \sum_{v \in \mathcal{N}} w^{a,u}_v v \leq w^\psi_0$, with the effect $u := \sum_{v \in \mathcal{N}} w^{a,u}_v v + w^{a,u}_0$ is one of the effects of a. If $r > 0$, we replace the bound on the effect of a on u with $\overline{asn}^i_{a,u} := \min\left\{\overline{asn}^i_{a,u}, w^\psi_0/r + w^{a,u}_0\right\}$, if $r < 0$ we multiply the inequality by -1 and replace the bound $\underline{asn}^i_{a,u}$.[2]

Lastly, we set the upper bound on the result of the application of a to be $\overline{l}^i_{a,u} = \max\{s^0[u], \overline{asn}^i_{a,u}\}$. The lower bound on the application is defined in a similar fashion $\underline{l}^i_{a,u} = \min\{s^0[u], \underline{asn}^i_{a,u}\}$. Using these bounds we define the following effect box $\mathsf{effB}^i_a := \times_{v \in \mathcal{N}}[\underline{l}^i_{a,u}, \overline{l}^i_{a,u}]$. Note that for each $i \in \mathbb{N}$ and each $a \in \mathcal{A}$ it holds that $s^0 \in \mathsf{effB}^i_a$.

We define the next iteration of the bounds: $\overline{v}^i = \max_{a \in \mathcal{A}} \overline{l}^i_{a,u}$ and $\underline{v}^i = \min_{a \in \mathcal{A}} \underline{l}^i_{a,u}$. This gives us the box $\mathsf{B}^i := \times_{v \in \mathcal{N}}[\underline{v}_i, \overline{v}^i]$. Geometrically speaking, B^i is the smallest possible box s.t. $\bigcup_{a \in \mathcal{A}} \mathsf{effB}^i_a \subseteq \mathsf{B}^i$.

We first compute $\overline{v}^i_a$ and $\underline{v}^i_a$, bounds on variable v when action a is applicable. Using $\overline{v}^i_a$ and $\underline{v}^i_a$, we compute bounds on effects of a. Using bounds on effects of all actions, we compute $\overline{v}^i$ and $\underline{v}_i$, bounds on numeric variables. Then, we compute $\overline{v}^{i+1}_a$ and $\underline{v}^{i+1}_a$ using $\overline{v}^i_a$ and $\underline{v}^i_a$ and repeat the process. We sketch the algorithm we use to compute the bounds.

1. Initialize the initial upper bounds with ∞ and the lower bounds with $-\infty$.
2. For $i \in \mathbb{N}$ repeat 3–5 until there is no change or until i exceeds some given i_0.
3. Compute $\overline{v}^i_a$ and $\underline{v}^i_a$ for each variable v and action a in an arbitrary order (Equations (1) – (3)).
4. Compute $\overline{asn}^i_{a,v}$ and $\underline{asn}^i_{a,v}$ for each action a and variable v in an arbitrary order (Equations (4) and (5)).
5. Compute $\overline{v}^i$ and $\underline{v}^i$ for each $v \in \mathcal{N}$ in an arbitrary order.

The following example demonstrates that this procedure **does not necessarily terminate**.

Example 1 *Consider a planning task* Π *with two numeric variables* $\mathcal{N} = \{x, y\}$, $\mathcal{F} = \emptyset$, *initial state* $s^0[x] = s^0[y] = 0$, *and actions* $\mathcal{A} = \{a_1, a_2\}$, *where* $\mathsf{pre}(a_1) = \{x \leq 1, x \leq 0.3y\}$, $\mathsf{num}(a_1) = \{x \mathrel{+}= 1\}$, $\mathsf{pre}(a_2) = \{y \leq x\}$ *and* $\mathsf{num}(a_2) = \{y \mathrel{+}= 0.5x\}$. *The goal does not affect our computation, thus we set it* $G = \{y \geq 2\}$.

Note that $\underline{x}^i = \underline{x}^i_a = \underline{y}^i = \underline{y}^i_a = 0$. *In this example, we are interested only in the upper bounds. Also,* $\overline{x}^i = \overline{x}^i_{a_2}$ *and* $\overline{y}^i = \overline{y}^i_{a_1}$. *We show the bounds after the first to fifth iterations.*

i	$\overline{x}^i_{a_1}$	$\overline{x}^i$	$\overline{y}^{i+1}_{a_2}$	$\overline{y}^{i+1}$
1	*1*	*2*	*2*	*3*
3	*0.9*	*1.9*	*1.9*	*2.85*
5	*0.855*	*1.855*	*1.855*	*2.7825*

[2] Since all linear conditions in $\mathsf{pre}(a)$ are represented in their linear normal form (LNF) [8], we do at most one update per bound.

In general for $i \geq 2$, $\overline{x}^{i+1} = 1 + 0.3\overline{y}^i$ *and* $\overline{y}^{i+1} = 1.5\overline{x}^i$, *so* $\overline{y}^{i+2} = 1.5 + 0.45\overline{y}^i$. *These series are infinite, but their fixed points are* $\lim_{i\to\infty} \overline{y}^i = 30/11$ *and* $\lim_{i\to\infty} \overline{x}^i = 20/11$ *since* $(\overline{y}^{i+2} - 30/11) = 0.45(\overline{y}^i - 30/11)$.

Next, we show that while our algorithm does not necessarily terminate, we can end it at any point and still get feasible bounds.

3.3 Correctness of the Algorithm

A function f is called *increasing* (*decreasing*) if for all x and y s.t. $x \leq y$ one has $f(x) \leq f(y)$ ($f(x) \geq f(y)$). A linear function $\mathbb{R}^n \to \mathbb{R}, \vec{x} \to \sum_{j=1}^n c_j x_j + c_0$ is increasing in x_j if $c_j \geq 0$, and decreasing if $c_j \leq 0$. Constant functions are both increasing and decreasing. Let both f_1 and f_2 be increasing (decreasing) functions and let $g \in \{\min, \max\}$, then $g(f_1, f_2)$ is also increasing (decreasing).

We start the proof of correctness with the following lemma.

Lemma 1 *For each* $i \in \mathbb{N}$ *it holds that* $\mathsf{B}^{i+1} \subseteq \mathsf{B}^i$.

Proof Sketch: The claim is proven by induction. We assume that for each $k < i$ it holds that $\mathsf{B}^{k+1} \subseteq \mathsf{B}^k$ and $\mathsf{preB}_a^{k+1} \subseteq \mathsf{preB}_a^k$ for each $a \in \mathcal{A}$. Then, using monotonicity of compositions of linear and max and min functions, we prove that $\mathsf{preB}_a^{i+1} \subseteq \mathsf{preB}_a^i$. Next, once again using monotonicity, we prove that $\mathsf{effB}_a^{i+1} \subseteq \mathsf{effB}_a^i$. This observation grants us $\bigcup_{a\in\mathcal{A}} \mathsf{effB}_a^{i+1} \subseteq \bigcup_{a\in\mathcal{A}} \mathsf{effB}_a^i$. Thus,

$$\bigcup_{a\in\mathcal{A}} \mathsf{effB}_a^{i+1} \subseteq \mathsf{B}^i := \bigtimes_{v\in\mathcal{N}} [\underline{v}_i, \overline{v}^i].$$

Since B^{i+1} is the smallest box that contains $\bigcup_{a\in\mathcal{A}} \mathsf{effB}_a^{i+1}$, we have that $\mathsf{B}^{i+1} \subseteq \mathsf{B}^i$. The full proof can be found in [11]. □

We proved that $\mathsf{B}^{i+1} \subseteq \mathsf{B}^i \subseteq \cdots \subseteq \mathsf{B}^0$, and by construction $(s^0)_n \in \mathsf{B}^i$ for all $i \in \mathbb{N}$. Thus, we can define $\mathsf{B}^* := \bigcap_{i=1}^\infty \mathsf{B}^i$, which we call the bounding of the task, and we know that $\mathsf{B}^* \neq \emptyset$. Moreover, since each sequence of bounds $\{\overline{v}^i\}_{i=1}^\infty$ is a monotonic sequence bounded by $s^0[v]$, we know that $\lim_{i\to\infty} \overline{v}^i = \overline{v}^*$. Thus, B^* forms a box. To finish the proof, we need to show that for every consequently applicable sequence of actions $\pi = \langle a_1, \ldots, a_m\rangle$ applied from s^0 and resulting in a state s it holds that $s_n \in \mathsf{B}^*$. Given we are interested in numeric bounds, it is enough to show this for a relaxation where we ignore the propositional part, i.e., we set $\mathcal{F} = \emptyset$.

Lemma 2 *Let* $s_n \in \mathsf{B}^*$ *be a proper numeric state*[3] *s.t.* $s_n \models \mathsf{pre}_n(a)$. *Then,* $s_n[\![a]\!] \in \mathsf{B}^*$.

Proof Sketch: Let $\psi \in \mathsf{pre}_n(a)$. Since we know that $s_n \models \mathsf{pre}_n(a)$, we know that $\sum_{v\in\mathcal{N}} w_v^\psi s[v] \geq w_0^\psi$. We also know that $s_n \in \mathsf{B}^*$, hence $s_n \in \mathsf{B}^i$ for each $i \in \mathbb{N}$. Next, we prove by induction that $s_n \in \mathsf{preB}_a^i$ for each i. Then, for each effect $u \mathrel{+}= \xi \in \mathsf{num}(a)$ the result of its application $s_n[u] + s_n[\xi]$ lies in effB_a^i. Since each action affects each variable exactly once (recall the dummy effects $u \mathrel{+}= 0$) we have that $s_n[\![a]\!] \in \mathsf{effB}_a^i$. Thus, $s_n[\![a]\!] \in \mathsf{effB}_a^i \subseteq \mathsf{B}^i$ for each i. Hence, we have that $s_n[\![a]\!] \in \mathsf{B}^*$. Once again, the full proof is found in the supplementary material. □

These two lemmas provide us with the following result.

Theorem 1 *Let* Π *be a linear numeric planning task, let* B^* *be the bounding of* Π. *Let* $\pi = \langle a_1, \ldots, a_m\rangle$ *be a plan for* Π, *and let* $\langle s^0, \ldots, s^m\rangle$ *be the sequence of states that corresponds to* π. *Then,* $s_n^k \in \mathsf{B}^*$ *for each* $k \in [m]$.

[3] We assume that all values of s_n are finite.

Since $\{\mathsf{B}^i\}_{i=1}^\infty$ is a sequence of nested boxes s.t. $\mathsf{B}^* \subseteq \mathsf{B}^i$ for each i, we can compute $\mathsf{B}^j \supseteq \mathsf{B}^*$ for some large enough j.

Lastly, we present an unsatisfaibility criteria for the linear numeric planning task. Let A_G be the polytope defined by G_n, the linear conditions given in the numeric goal part of the task.

Corollary 1 *Suppose we have a linear numeric planning task* Π, *with its bounding box denoted by* B^*, *and its goal polytope denoted by* A_G. *The condition* $\mathsf{B}^* \cap A_G = \emptyset$ *is sufficient but not necessary to determine that* Π *is unsolvable.*

Since $\mathsf{B}^* \subseteq \mathsf{B}^i$ for each i, the task is unsolvable if $\mathsf{B}^i \cap A_G = \emptyset$ for some i, and since both B^i and A_G are represented as sets of linear inequalities, finding if $\mathsf{B}^* \cap A_G$ is empty can be done in polynomial time [9]. For example, a task is unsolvable if $x \geq 2 \in G$, $s^0[x] = 0$, and the only action has effect $x \mathrel{+}= 1$ with precondition $x \leq 0$.

4 Exploiting Bounds in a Heuristic Function

Next, we show how to exploit the pre-computed bounds of numeric variables in linear numeric planning tasks. The state-of-the-art method for optimal linear numeric planning is A^* [5] with $h_2^{\text{LM-cut}}$ [10], the LM-cut heuristic generalized from classical planning [7] and simple numeric planning [12]. LM-cut maps a state s to a non-negative rational value, $h_2^{\text{LM-cut}}(s)$, that is less than or equal to the cost of an optimal s-plan. The latter property is called the *admissibility*, which ensures the optimality of A^*. LM-cut obtains the lower bound on the optimal cost by computing a lower bound on the cost to achieve each condition ψ using a single action a. We tighten this estimation by using bounds of numeric variables: intuitively, we compute an upper bound on the net effect of a on the linear formula ψ using the individual bounds of each variable. These bounds, described in Sec 3, are computed once, prior to the execution of the search.

4.1 LM-Cut for Linear Numeric Planning

LM-cut first transforms a numeric planning task by *one-variable compilation* (OVC), which replaces a numeric condition $\psi : \sum_{v\in\mathcal{N}} w_v^\psi \trianglerighteq w_0^\psi$ with $x^\psi \trianglerighteq w_0^\psi$, where x^ψ is an auxiliary numeric variable and $\trianglerighteq \in \{\geq, >\}$. In OVC, a larger numeric variable value is always better because more numeric conditions are possibly satisfied. Then, LM-cut relaxes the transformed task by ignoring negative effects that decrease the values of numeric variables. In the relaxation, the value of a numeric variable monotonically increases. The relaxation is called the *delete-relaxation* as it is similar to the delete-relaxation of classical planning, where delete effects are ignored, and the set of satisfied propositions monotonically expands. Here we only describe the key features of $h_2^{\text{LM-cut}}$ related to our method. For a full definition, please refer to the original paper [10].

4.1.1 One-Variable Compilation

In OVC, for a numeric condition $\sum_{v\in\mathcal{N}} w_v^\psi v \trianglerighteq w_0^\psi$, where $\trianglerighteq \in \{\geq, >\}$, an auxiliary variable x^ψ is introduced, and ψ is represented by $x^\psi \trianglerighteq w_0^\psi$. Let us now look at the effects of a on x^ψ. Assume that $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u}$ is the effect of a on each original variable $u \in \mathcal{N}$ (here, for brevity, we allow the constants to be zeroes, but obviously in the implementation these can be removed). Then, the effect of a on x^ψ is given by

$$x^\psi \mathrel{+}= \sum_{u\in\mathcal{N}} w_u^\psi \left(\sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u} \right),$$

where the constants can be rewritten as $w_v^{a,\psi} = \sum_{u\in\mathcal{N}} w_u^\psi w_v^{a,u}$ and $w_0^{a,\psi} = \sum_{u\in\mathcal{N}} w_u^\psi w_0^{a,u}$. Thus, the effect of a on x^ψ is written as $x^\psi \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,\psi} v + w_0^{a,\psi}$, which is in a linear formula. Therefore, after moving to OVC we still have linear effects on the variables, but all conditions in OVC are of the from $u \trianglerighteq w_0^\psi$. Next, we will see how these linear effects are treated in the delete-relaxation.

4.1.2 First- and Second-Order Delete-Relaxations

In the delete-relaxation of OVC, propositional delete effects $\mathsf{del}(a)$ and numeric effects that decrease the value of a numeric variable are ignored. In simple numeric planning, simple effects $u \mathrel{+}= w_0^{a,u}$ with $w_0^{a,u} \leq 0$ are relaxed to $u \mathrel{+}= 0$ [12]. In the first-order delete-relaxation for linear numeric planning (see [10]), non-simple effects are relaxed to conditional effects that increase numeric variables to infinity if certain conditions are satisfied. For each non-simple effect $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u}$ on a non-auxiliary variable u, numeric conditions $x_+^{a,u} > 0$ and $x_-^{a,u} > 0$ are introduced, corresponding to $\sum_{v\in\mathcal{N}} w_v^{a,u} v > 0$ and $\sum_{v\in\mathcal{N}} w_v^{a,u} v < 0$. Effects on $x_+^{a,u}$ and $x_-^{a,u}$ are introduced according to effects on x^ψ. Then, $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u}$ is relaxed to conditional effect $u \mathrel{+}= \infty$ if $x_+^{a,u} > 0$. For auxiliary variables x^ψ corresponding to numeric condition ψ, if $w_u^\psi > 0$, the effect of a on x^ψ is relaxed to conditional effect $x^\psi \mathrel{+}= \infty$ if $x_+^{a,u} > 0$. Accordingly, if $w_u^\psi < 0$, the effect is relaxed to conditional effect $x^\psi \mathrel{+}= \infty$ if $x_-^{a,u} > 0$.

In the second-order delete-relaxation [10], if a non-simple effect is identified as a *second-order simple effect (SOSE)*, it is not relaxed to the above conditional effect. A non-simple effect on an auxiliary or a non-auxiliary variable u, $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u}$, is a SOSE if it is only affected by simple effects of actions that do not change u. In other words, if the effect is a SOSE, each action a' satisfies $w_v^{a,u} \neq 0 \rightarrow \forall v' \in \mathcal{N}, w_{v'}^{a',v} = 0$ (only simple effect on a) and either of the following conditions:

1. $\sum_{v\in\mathcal{N}} w_v^{a,u} w_0^{a',v} \leq 0$ (no positive effect on a).
2. $w_0^{a',u} = 0$ and $\forall v \in \mathcal{N} : w_v^{a',u} = 0$ (no effect on u).

When a' satisfies only the second condition, it is called a *second-order supporter* for a on u. For a SOSE $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} v + w_0^{a,u}$, an auxiliary variable $x_+^{a,u}$, corresponding to $\sum_{v\in\mathcal{N}} w_v^{a,u} v$, is introduced even if u is an auxiliary variable, and the effect is replaced with $u \mathrel{+}= x_+^{a,u} + w_0^{a,u}$. The effect of action a' on this variable is $x_+^{a,u} \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u} w_0^{a',v}$, which is a simple effect.

4.1.3 Action Multiplicators

In the computation of $h_2^{\text{LM-cut}}$, the number of applications of action a to achieve numeric condition ψ from state s is estimated by an *action multiplicator* $\mathsf{m}_a(s,\psi)$. While there can be multiple choices for $\mathsf{m}_a(s,\psi)$, the admissibility is guaranteed as long as $\mathsf{m}_a(s,\psi)\mathsf{cost}(a)$ is a lower bound of the minimum cost to achieve ψ from s using only a in the delete-relaxation [12]. Let e be the effect of action a on variable x^ψ for condition ψ in the delete-relaxation. We assume that numeric conditions are normalized or relaxed to $x^\psi \geq w_0^\psi$ without loss of admissibility and $s \not\models \psi$, i.e., $s[x^\psi] < w_0^\psi$. If e is not a SOSE,

$$\mathsf{m}_a(s,\psi) = \begin{cases} \infty & \text{if } e : x^\psi \mathrel{+}= 0 \\ 1 & \text{else if } e : x^\psi \mathrel{+}= \infty \\ \frac{w_0^\psi - s[x^\psi]}{w_0^{a,\psi}} & \text{otherwise.} \end{cases} \tag{6}$$

If a has no effect on ψ, we cannot achieve ψ using a. If a has a non-simple effect, we need to apply a at least once, so $\mathsf{m}_a(s,\psi)$ is underestimated by 1. If a has a simple effect, we estimate the number of applications by division.

Suppose that e is a SOSE $x^\psi \mathrel{+}= y + w$ where y is a variable and w is a constant. If we only use a to achieve ψ,

$$\mathsf{m}_a(s,\psi) = \begin{cases} \infty & \text{if } s[y] + w \leq 0 \\ \frac{w_0^\psi - s[x^\psi]}{s[y]+w} & \text{otherwise.} \end{cases} \tag{7}$$

Consider achieving ψ using a and its second-order supporter $y \mathrel{+}= w' \in \mathsf{eff}_n(a')$. Let $\mathsf{m}^y_{a',a}(s,\psi)$ and $\mathsf{m}^\psi_{a',a}(s,\psi)$ be action multiplicators for a' and a. The admissibility is guaranteed if $\mathsf{m}^y_{a',a}(s,\psi)\mathsf{cost}(a') + \mathsf{m}^\psi_{a',a}(s,\psi)\mathsf{cost}(a)$ is a lower bound to achieve ψ from s using only a' and a in the second-order relaxation.

If $\mathsf{cost}(a') = 0$, then $\mathsf{m}^y_{a',a}(s,\psi) = 0$ and $\mathsf{m}^\psi_{a',a}(s,\psi) = 1$ because a' can be applied an arbitrary number of times, and a must be applied at least once. Similarly, if $\mathsf{cost}(a) = 0$, then $\mathsf{m}^\psi_{a',a}(s,\psi) = 0$ while $\mathsf{m}^y_{a',a}(s,\psi)$ depends on $s[y] + w$: if $s[y] + w > 0$, then a is unnecessary, so $\mathsf{m}^y_{a',a}(s,\psi) = 0$; if $s[y] + w < 0$, then a' is applied multiple times to make $s[y] + w$ positive, so $\mathsf{m}^y_{a',a}(s,\psi) = \frac{-s[y]-w}{w'}$; if $s[y] + w = 0$, then a' must be applied at least once, so $\mathsf{m}^y_{a',a}(s,\psi) = 1$.

Now, assume that $\mathsf{cost}(a) > 0$ and $\mathsf{cost}(a') > 0$. We first increase the value of y to Y using a' and then increase the value of x^ψ by applying a X times. Relaxing the domain of X to continuous values, the minimum cost to achieve ψ from s using a' and a is lower bounded by the optimal cost of the following optimization problem:

$$\begin{aligned} &\min X\mathsf{cost}(a) + \frac{Y - s[y]}{w'}\mathsf{cost}(a') \\ &\text{s.t. } w_0^\psi - s[x^\psi] = X(Y + w). \end{aligned}$$

By solving the problem using Lagrange multipliers, the optimal solution $(X, Y) = (X^*, Y^*)$ is given as follows:

$$X^* = \frac{w_0^\psi - s[x^\psi]}{Y^* + w}.$$

$$Y^* = \sqrt{\frac{(w_0^\psi - s[x^\psi])w'\mathsf{cost}(a)}{\mathsf{cost}(a')}} - w.$$

If $Y^* < s[y]$, we should use only a to achieve ψ, so $\mathsf{m}^y_{a',a}(s,\psi) = 0$ and $\mathsf{m}^\psi_{a',a}(s,\psi) = \mathsf{m}_a(s,\psi)$. Therefore,

$$\mathsf{m}^y_{a',a}(s,\psi) = \begin{cases} 0 & \text{if } Y^* < s[y] \vee \mathsf{cost}(a') = 0 \\ 0 & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w > 0 \\ 1 & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w = 0 \\ \frac{-s[y]-w}{w'} & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w < 0 \\ \frac{Y^* - s[y]}{w'} & \text{otherwise.} \end{cases} \tag{8}$$

$$\mathsf{m}^\psi_{a',a}(s,\psi) = \begin{cases} \mathsf{m}_a(s,\psi) & \text{if } Y^* < s[y] \\ 1 & \text{if } \mathsf{cost}(a') = 0 \\ 0 & \text{if } \mathsf{cost}(a) = 0 \\ X^* & \text{otherwise.} \end{cases} \tag{9}$$

4.2 Improving LM-Cut with Bounds

Using the method proposed in Section 3.2, we can compute $\overline{inc}^i_{a,v}$ and $\underline{inc}^i_{a,v}$, upper and lower bounds of the increment effect of action a on variable v. We omit i in what follows, assuming that the

computation is stopped with a fixed number of iterations.

$$\overline{inc}_{a,v} := \sum_{v:w_u^{a,v}>0} w_u^{a,v}\overline{u}_{a,u} + \sum_{v:w_u^{a,v}<0} w_u^{a,v}\underline{u}_{a,u} + w_0^{a,v},$$
$$\underline{inc}_{a,v} := \sum_{v:w_u^{a,v}<0} w_u^{a,v}\overline{u}_{a,u} + \sum_{v:w_u^{a,v}>0} w_u^{a,v}\underline{u}_{a,u} + w_0^{a,v}.$$

These bounds can be used to overestimate the effect of a on a numeric condition ψ, which satisfies the admissibility. For the auxiliary variable x^ψ for a numeric condition ψ, the upper bound of the increment effect of action a is

$$\overline{inc}_{a,\psi} := \sum_{v:w_v^\psi>0} w_v^\psi \overline{inc}_{a.v} + \sum_{v:w_v^\psi<0} w_v^\psi \underline{inc}_{a,v}. \quad (10)$$

We use the following action multiplicator instead of Equation (6):

$$\mathsf{m}_a(s,\psi) = \begin{cases} \infty & \text{if } \overline{inc}_{a,\psi} \le 0 \\ 1 & \text{if } \overline{inc}_{a,\psi} = \infty \\ \frac{w_0^\psi - s[x^\psi]}{\overline{inc}_{a,\psi}} & \text{if } 0 < \overline{inc}_{a,\psi} < \infty. \end{cases} \quad (11)$$

The cost to achieve ψ using only a from s in the delete-relaxation is lower bounded by $\mathsf{m}_a(s,\psi)\mathsf{cost}(a)$.

Furthermore, we possibly identify more SOSEs based on the bounds. Suppose that an action a has an effect $u \mathrel{+}= \sum_{v\in\mathcal{N}} w_v^{a,u}v + w_0^{a,u}$. Recall that the effect is a SOSE if it is only affected by simple effects of actions that do not change u. With upper bounds, we can overestimate non-simple effects by simple effects. Now, we extend the definition of SOSE: the effect of a on u is a SOSE if each action a' with effect $u \mathrel{+}= \xi$ satisfies either of the following conditions:

1. $\sum_{v:w_v^{a,u}>0} w_v^{a,u}\overline{inc}_{a',v} + \sum_{v:w_v^{a,u}<0} w_v^{a,u}\underline{inc}_{a',v} \le 0.$
2. $\sum_{v:w_v^{a,u}>0} w_v^{a,u}\overline{inc}_{a',v} + \sum_{v:w_v^{a,u}<0} w_v^{a,u}\underline{inc}_{a',v} < \infty$ and $\overline{inc}_{a',u} \le 0.$

Now, a' is a second-order supporter if it satisfies only the second condition. The upper bound of the effect of a' on the auxiliary variable $x_+^{a,u}$, which is used in the SOSE $u \mathrel{+}= x_+^{a,u} + w_0^{a,u}$, is

$$\overline{inc}_{a',x_+^{a,u}} := \sum_{v:w_v^{a,u}>0} w_v^{a,u}\overline{inc}_{a',v} + \sum_{v:w_v^{a,u}<0} w_v^{a,u}\underline{inc}_{a',v}.$$

The upper bound on $x_+^{a,u}$ when a is applicable is

$$\overline{x_+^{a,u}} := \sum_{v:w_v^{a,u}>0} w_v^{a,u}\overline{v}_a + \sum_{v:w_v^{a,u}<0} w_v^{a,u}\underline{v}_a.$$

Based on these bounds, we extend the action multiplicators.

Theorem 2 *Given action a with a SOSE $u \mathrel{+}= y+w$ and its second-order supporter a', let $\overline{inc}_{a',y}$ be an upper bound on the effect of a' on y and $\overline{y}_a$ be an upper bound of y when a is applicable. The optimal cost of the following optimization problem is a lower bound of the cost to achieve numeric condition $\psi : x^\psi \ge w_0^\psi$ from state s using only a' and a.*

$$\min X\mathsf{cost}(a) + \frac{Y - s[y]}{\overline{inc}_{a',y}}\mathsf{cost}(a'),$$
$$\text{s.t. } w_0^\psi - s[x^\psi] = X(Y+w)$$
$$0 \le X, 0 \le Y \le \overline{y_a}.$$

The optimal solution $(X,Y) = (X^*,Y^*)$ for the above problem is given as follows:

$$X^* = \frac{w_0^\psi - s[x^\psi]}{Y^* + w},$$
$$Y^* = \min\left\{\overline{y}_a, \sqrt{\frac{(w_0^\psi - s[x^\psi])\overline{inc}_{a',y}\mathsf{cost}(a)}{\mathsf{cost}(a')}} - w\right\}.$$

Proof: The optimal solution (X^*,Y^*) is from the analytical solution of the minimization problem. We note that linear function achieves its extrema (if such exist) on the boundary of the domain, and then apply the method of Lagrange Multipliers to compute its minimum. The full proof appears in the supplementary material. □

With Theorem 2, we can use the following action multiplicator instead of Equation (8).

$$\mathsf{m}^y_{a',a}(s,\psi) = \begin{cases} 0 & \text{if } Y^* < s[y] \vee \mathsf{cost}(a') = 0 \\ 0 & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w > 0 \\ 1 & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w = 0 \\ \frac{-s[y]-w}{\overline{inc}_{a',y}} & \text{if } \mathsf{cost}(a) = 0 \wedge s[y] + w < 0 \\ \frac{Y^* - s[y]}{\overline{inc}_{a',y}} & \text{otherwise.} \end{cases} \quad (12)$$

We call our version of $h_2^{\text{LM-cut}}$ extended with the bounds $h_{2b}^{\text{LM-cut}}$.

4.3 Rounding Up Action Multiplicators

It is possible that the use of bounds decreases the action multiplicator and results in a less informative heuristic: comparing Equations (6) and (11), if action a has a non-simple effect on ψ and $\overline{inc}_{a,\psi} > w_0^\psi - s[x^\psi]$, then $\mathsf{m}_a(s,\psi) < 1$ in Equation (11) while $\mathsf{m}_a(s,\psi) = 1$ in Equation (6). To address this issue, we round up $\mathsf{m}_a(s,\psi) < 1$ to 1 following LM-cut in simple numeric planning [12]. LM-cut does not lose admissibility with the rounded action multiplicators because a must be applied at least once to achieve ψ. We apply this method in Equations (6), (7), (9), and (11). In Equations (8) and (12), it is applied only when $\mathsf{m}^y_{a',a}(s,\psi) > 0$. We call the resulting heuristics $h_{2+}^{\text{LM-cut}}$ and $h_{2b+}^{\text{LM-cut}}$, respectively.

5 Empirical Evaluation

We implemented the method described in Sections 3.2 and 4.2 in Numeric Fast Downward (NFD) [1][4] using Python 2.7.5 and GCC 9.4.0. All experiments are performed on an Intel Xeon Gold 6148 processor with a single-thread, a 30-minute time limit, and a 4GB RAM limit with GNU Parallel [23].

For numeric preconditions and goal conditions, we add redundant constraints in the same way as Scala et al. [20]. When computing bounds and heuristic functions, for numeric conditions $\sum_{v\in\mathcal{N}} w_v^\psi s[v] > w_0^\psi$ where all v is a simple variable if $w_v^\psi \ne 0$, we transform it to $\sum_{v\in\mathcal{N}} w_v^\psi s[v] \ge w_0^\psi + \frac{1}{M}$ as described above. For M, we use the minimum 10^k such that $10^k w_0^\psi$, $10^k w_v^\psi$ for all v, and $10^k w_0^{a,v}$ for all $a \in \mathcal{A}$ and $v \in \mathcal{N}$ with $w_v^\psi \ne 0$ are integer.

We extract bounds of numeric variables once at the pre-processing stage using the initial state and at most 10 iterations, which are enough to converge to the fix point for every task present. The process takes less than 0.1 seconds in domains except for TPP, where

[4] https://github.com/Kurorororo/numeric-fast-downward

	$h_2^{\text{LM-cut}}$			$h_{2b}^{\text{LM-cut}}$			$h_{2+}^{\text{LM-cut}}$			$h_{2b+}^{\text{LM-cut}}$		
All SOSE	c.	time	#exp.	c.	time	#exp.	c.	time	#exp.	c.	time	#exp.
FO-FARMLAND (50)	25	152.5	35044	25	153.0	35044	**29**	**27.6**	**7053**	**29**	28.2	**7053**
FO-COUNT (20)	4	2.5	9237	4	1.6	6221	4	1.9	7036	**5**	**1.1**	**4238**
FO-COUNT-INV (20)	4	79.3	233905	4	44.6	128324	4	66.0	200346	4	**33.8**	**100941**
FO-COUNT-RND (60)	14	0.4	1923	**16**	**0.3**	1651	15	0.4	1684	**16**	**0.3**	**1329**
FO-SAILING (20)	4	373.1	3151304	**7**	**36.6**	653082	4	379.6	3151267	**7**	37.7	**652983**
Some SOSE												
LIN-CAR-EXP (34)	27	53.3	3670787	27	53.3	3670787	27	**52.7**	**3670782**	27	53.1	**3670782**
LIN-CAR-EXP-UNIT (34)	29	72.4	3824756	29	**65.4**	**3291420**	29	71.7	3825158	29	65.5	3292320
No SOSE												
BARMAN (15)	2	2.1	7202	2	2.1	**6640**	2	2.1	7202	2	2.1	**6640**
BARMAN-UNIT (15)	2	**2.1**	9548	2	2.2	9286	2	**2.1**	9539	2	2.2	**9272**
LIN-CAR-POLY (34)	14	30.9	2525965	14	34.5	2527291	14	**30.3**	2525965	14	33.8	**2525307**
LIN-CAR-POLY-UNIT (34)	14	53.5	2642867	14	48.4	1959943	14	51.8	2641905	14	**47.8**	**1948991**
ROVER-METRIC (10)	6	**36.1**	47445	6	**36.1**	47415	6	37.8	49984	6	**36.1**	**47388**
TPP-METRIC (40)	5	18.8	17181	5	6.4	6307	5	10.1	10957	5	**3.7**	**3361**
ZENOTRAVEL-LINEAR (10)	8	46.6	1228	8	41.0	1226	8	**30.1**	**776**	8	30.8	779
PICKUP (20)	14	344.1	5247955	**16**	267.7	3588681	14	326.2	5233143	**16**	**254.2**	**3515057**
TOTAL (416)	172	-	-	179	-	-	177	-	-	**184**	-	-

Table 1. Coverage (c.), wall-clock time (time) mesaured in seconds, and the number of expansions before the last f-layer (exp.). Time and #exp. are averaged over instances solved by all methods.

some instances require up to 2 minutes, when the large instances are unsolved by all methods.

We evaluate the heuristics with A^* using the domains introduced by previous work [14, 13, 10, 22], five of which have only simple effects and SOSEs (All SOSE), two of which have both SOSEs and non-SOSEs (Some SOSE), and seven have no SOSEs (No SOSE).

We show the coverage, the mean wall-clock time to solve an instance, and the mean number of expansions before the last f-layer in Table 1. Using bounds reduces the number of expansions in almost all domains and achieves higher coverage in FO-COUNT-RND and FO-SAILING. In addition, $h_{2b+}^{\text{LM-cut}}$ solves more instances than $h_{2+}^{\text{LM-cut}}$ in FO-COUNT. However, in non-SOSE domains, the improvement does not lead to an increase in coverage, and using bounds slightly increases the number of expansions in a few domains: $h_{2b}^{\text{LM-cut}}$ expands slightly more states than $h_2^{\text{LM-cut}}$ in LIN-CAR-POLY, and $h_{2b+}^{\text{LM-cut}}$ expands more than $h_{2+}^{\text{LM-cut}}$ in ZENOTRAVEL-LINEAR.

Comparing $h_2^{\text{LM-cut}}$ and $h_{2+}^{\text{LM-cut}}$, the latter increases the coverage in FO-FARMLAND and FO-COUNT-RND and has a better or equal number of expansions in all domains except for ROVER-METRIC. Similarly, $h_{2b+}^{\text{LM-cut}}$ has higher coverage than $h_{2b}^{\text{LM-cut}}$ in FO-FARMLAND, FO-COUNT-RND, and FO-COUNT and expands equal or fewer states in all domains except for LIN-CAR-EXP-UNIT.

5.1 PICKUP *Domain*

To show the advantage of our method in no SOSE domains, we introduce a new domain PICKUP, inspired by the capacitated vehicle routing problem in Operations Research. In this problem, n customers and one depot are given. A worker must pick up a commodity from each customer, while it can carry at most C commodities at a time. The number of commodities carried by the worker is represented by a numeric variable x. At the depot, there is a truck with a capacity Q to deliver commodities to a center. The number of commodities loaded into the truck is represented by y, and the number of commodities delivered to the center is represented by z. In the initial state, $x = y = z = 0$. The goal is to deliver all commodities to the center, i.e., $z \geq n$. The worker can load all commodities to the truck ($y \mathrel{+}= x$ and $x := 0$) if $y + x \leq Q$ at the depot. If $y + x > Q$, the worker can load commodities as much as possible ($y := Q$ and $x \mathrel{+}= y - Q$). The commodities are delivered to the center by driving the truck ($z \mathrel{+}= y$ and $y := 0$), which also returns the truck to the depot. While there are no SOSEs since all numeric variables are affected by non-simple effects, they have upper and lower bounds. We generate 20 instances with different parameters (see the supplementary material). As shown in Table 1, $h_{2b}^{\text{LM-cut}}$ and $h_{2b+}^{\text{LM-cut}}$ solves two more instances than $h_2^{\text{LM-cut}}$ and $h_{2+}^{\text{LM-cut}}$.

6 Conclusion

In this paper, we generalized the method to extract the bounds on numeric variables in linear numeric planning [3]. We argue that these bounds can be used to determine the unsolvability of certain tasks in polynomial time. In addition, we strengthen the numeric LM-cut heuristic [10] with these bounds, maintaining its admissibility. The experimental evaluation demonstrates that the extracted bounds improve the performance of LM-cut in most linear numeric domains.

In future work, we intend to improve the bound extraction using the theory of zonotopes, that were previously employed by Löhr et al. [15] in hybrid systems.

In simple numeric planning, operator-counting (OC) heuristics [18, 17] exploit the bounds in state-equation constraints (SEQ) [2]. LM-cut achieves better performance combined with SEQ in OC [12]. Generalizing SEQ to linear numeric planning and using it with the extracted bounds may be a fruitful direction.

While we use LM-cut as it has state-of-the-art performance, our method of bound extraction is independent of particular solving approaches. For example, the bounds may also improve inadmissible heuristics for linear numeric planning [8, 14]. In model-based approaches, such as mixed-integer programming and satisfiability modulo theories [16, 13], we may derive redundant constraints.

Furthermore, the use of the bounds allows us to reason about the unsolvability of some tasks, a topic that has not been discussed in the literature so far, given that proving that a numeric task is unsolvable is impossible in the general case [6].

Acknowledgements

This work was partially supported by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Johannes Aldinger and Bernhard Nebel, 'Interval based relaxation heuristics for numeric planning with action costs', in *Proc. KI*, pp. 15–28, (2017).

[2] Blai Bonet, 'An admissible heuristic for SAS+ planning obtained from the state equation', in *Proc. IJCAI*, pp. 2268–2274, (2013).

[3] Amanda Jane Coles, Andrew Coles, Maria Fox, and Derek Long, 'A hybrid LP-RPG heuristic for modelling numeric resource flows in planning', *J. Artif. Intell. Res.*, **46**, 343–412, (2013).

[4] Maria Fox and Derek Long, 'PDDL2.1: An extension to PDDL for expressing temporal planning domains', *JAIR*, **20**, 61–124, (2003).

[5] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael, 'A formal basis for the heuristic determination of minimum cost paths', *IEEE Trans. Syst. Sci. Cybern.*, **4**(2), 100–107, (1968).

[6] Malte Helmert, 'Decidability and undecidability results for planning with numerical state variables', in *Proc. AIPS*, pp. 303–312, (2002).

[7] Malte Helmert and Carmel Domshlak, 'Landmarks, critical paths and abstractions: What's the difference anyway?', in *Proc. ICAPS*, pp. 162–169, (2009).

[8] Jörg Hoffmann, 'The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables', *J. Artif. Intell. Res.*, **20**, 291–341, (2003).

[9] Leonid G. Khachiyan, 'A polynomial algorithm in linear programming', *Soviet Mathematics – Doklady*, **20**, 191–194, (1979).

[10] Ryo Kuroiwa, Alexander Shleyfman, and J. Christopher Beck, 'LM-cut heuristics for optimal linear numeric planning', in *Proc. ICAPS*, pp. 203–212, (2022).

[11] Ryo Kuroiwa, Alexander Shleyfman, and J. Christopher Beck. Extracting and exploiting bounds of numeric variables for optimal linear numeric planning – supplementary materials. https://tidel.mie.utoronto.ca/pubs/Supplement_Numeric_Bound_ECAI23.pdf, 2023.

[12] Ryo Kuroiwa, Alexander Shleyfman, Chiara Piacentini, Margarita P. Castro, and J. Christopher Beck, 'The LM-cut heuristic family for optimal numeric planning with simple conditions', *J. Artif. Intell. Res.*, **75**, 1473–1544, (2022).

[13] Francesco Leofante, Enrico Giunchiglia, Erika Ábrahám, and Armando Tacchella, 'Optimal planning modulo theories', in *Proc. IJCAI*, pp. 4128–4134, (2020).

[14] Dongxu Li, Enrico Scala, Patrik Haslum, and Sergiy Bogomolov, 'Effect-abstraction based relaxation for linear numeric planning', in *Proc. IJCAI*, pp. 4787–4793, (2018).

[15] Johannes Löhr, Martin Wehrle, Maria Fox, and Bernhard Nebel, 'Symbolic domain predictive control', in *AAAI*, pp. 2315–2321, (2014).

[16] Chiara Piacentini, Margarita P. Castro, André Augusto Ciré, and J. Christopher Beck, 'Compiling optimal numeric planning to mixed integer linear programming', in *Proc. ICAPS*, pp. 383–387, (2018).

[17] Chiara Piacentini, Margarita P. Castro, André Augusto Ciré, and J. Christopher Beck, 'Linear and integer programming-based heuristics for cost-optimal numeric planning', in *Proc. AAAI*, pp. 6254–6261, (2018).

[18] Florian Pommerening, Gabriele Röger, Malte Helmert, and Blai Bonet, 'Heuristics for cost-optimal classical planning based on linear programming', in *Proc. IJCAI*, pp. 4303–4309, (2015).

[19] Enrico Scala, Patrik Haslum, Daniele Magazzeni, and Sylvie Thiébaux, 'Landmarks for numeric planning problems', in *Proc. IJCAI*, pp. 4384–4390, (2017).

[20] Enrico Scala, Patrik Haslum, and Sylvie Thiébaux, 'Heuristics for numeric planning via subgoaling', in *Proc. IJCAI*, pp. 3228–3234, (2016).

[21] Enrico Scala, Patrik Haslum, Sylvie Thiébaux, and Miquel Ramírez, 'Subgoaling techniques for satisficing and optimal numeric planning', *J. Artif. Intell. Res.*, **68**, 691–752, (2020).

[22] Alexander Shleyfman, Ryo Kuroiwa, and J. Christopher Beck, 'Symmetry detection and breaking in linear cost-optimal numeric planning', in *Proc. ICAPS*, (2022).

[23] Ole Tange, 'GNU parallel - the command-line power tool', *;login: The USENIX Magazine*, **36**, 42–47, (2011).