for Optimal Numeric Planning with Simple Conditions

Ryo Kuroiwa, Alexander Shleyfman, Chiara Piacentini, Margarita P. Castro, J. Christopher Beck
Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada, ON M5S 3G8
rkuroiwa@mie.utoronto.ca, shleyfman.alexander@gmail.com, {chiarap, mpcastro, jcb}@mie.utoronto.ca

Abstract

We consider optimal numeric planning with numeric conditions consisting of linear expressions of numeric state variables and actions that increase or decrease numeric state variables by constant quantities. We build on previous research to introduce a new variant of the numeric \( h^\text{max} \) heuristic based on the delete-relaxed version of such planning tasks. Although our \( h^\text{max} \) heuristic is inadmissible, it yields a numeric version of the classical LM-cut heuristic which is admissible. Further, we prove that our LM-cut heuristic neither dominates nor is dominated by the existing numeric heuristic \( h^\text{max} \). Admissibility also holds when integrating the numeric cuts into the operator-counting (OC) heuristic producing an admissible numeric version of the OC heuristic. Through experiments, we demonstrate that both these heuristics compete favorably with the state-of-the-art heuristics: in particular, while sometimes expanding more nodes than other heuristics, numeric OC solves 19 more problem instances than the next closest heuristic.

Introduction

In this paper, we consider numeric planning with instantaneous actions. The presence of numeric state variables introduces a further degree of complexity over classical planning, making plan existence undecidable in general (Helmert 2002). Nevertheless, since the introduction of numeric variables in PDDL2.1 (Fox and Long 2003), several methods have been proposed to solve satisficing and optimal variants of numeric planning problems (Hoffmann 2003b; Shin and Davis 2005; Gerevini, Saetti, and Serina 2008; Eye-rich, Mattmüller, and Röger 2009; Coles et al. 2013; Scala et al. 2016; Illanes and McIlraith 2017; Li et al. 2018; Scala, Haslum, and Thiébaux 2016; Scala et al. 2017; Aldinger and Nebel 2017; Piacentini et al. 2018a,b). We consider here optimal planning with so-called simple numeric conditions (where numeric variables can be increased or decreased by constant quantities and pre-conditions are inequalities involving linear expressions) and focus on the delete-relaxed version of such problems to develop heuristics.

First, we introduce a variation of the numeric \( h^\text{max} \) heuristic based on sub-goaling relaxations (Scala, Haslum, and Thiébaux 2016). Next, building upon our version of the relaxed \( h^\text{max} \), we extend the well-known classical LM-cut heuristic (Helmert and Domshlak 2009) to account for numeric effects and include it in the operator-counting framework (Pommerening et al. 2015). We prove that while our version of the relaxed \( h^\text{max} \) is inadmissible, our LM-cut and, therefore, operator counting heuristics are admissible and show the theoretical relationship among these heuristics.

The empirical evaluation indicates a good trade-off between informativeness and computational time, in particular, these heuristics favorably compete with several classes of state-of-the-art heuristics and overall achieve higher coverage. The experimental results over classical planning tasks with resources show the potential of explicitly modelling resources as numeric state variables, instead of using their propositional representation.

Preliminaries

Numeric Planning

We consider a fragment of numeric planning restricted to the STRIPS formalism (Fikes and Nilsson 1971) with the addition of numeric state variables. We focus our attention on a restricted class of numeric planning defined by Hoffmann (2003a), called the restricted numeric task (RT). Formally, RT is defined as a 5-tuple \( RT = (F_p, N, A, s_I, G) \), where \( F_p \) is a finite set of facts and \( N \) is a set of numeric variables. Numeric variables \( v \in N \) have rational values in \( Q \). A state \( s \) is a tuple \( (s_p, s_n) \), where \( s_p \subseteq F_p \) is a set of facts and \( s_n \) is a full assignment over the variables in \( N \); \( s[v] \) indicates the value of the variable \( v \in N \).

Conditions can be either propositional \( \psi \in F_p \) or numeric, defined as \( \psi : v \geq w \), with \( \geq \in \{\geq, >\} \), \( v \in N \) and \( w \in Q \). The set of all numeric conditions is denoted by \( F_n \). A propositional condition \( \psi \in F_p \) is satisfied by the state \( s \) if \( \psi \in s_p \). A numeric condition \( \psi : v \geq w \in F_n \) is satisfied by \( s \) if \( s[v] \geq w \).

An action \( a \in A \) is a triplet \( \langle \text{pre}(a), \text{eff}(a), \text{cost}(a) \rangle \), where \( \text{pre}(a) \) are the preconditions, \( \text{eff}(a) \) the effects, and \( \text{cost}(a) \in R^{0+} \) is the cost. Preconditions are defined as \( \text{pre}_p(a) \cup \text{pre}_n(a) \), with propositional and numeric conditions, respectively. Effects are a triplet \( \text{eff}(a) = \ldots \)

---

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
\(\text{add}(a), \text{del}(a), \text{num}(a)\), where \(\text{add}(a), \text{del}(a) \subseteq \mathcal{F}_n\) are added and deleted facts, and \(\text{num}(a)\) is the set of additive numeric effects of the from \((v \leftarrow k)\), where \(v \in \mathcal{N}\) and \(k \in \mathbb{Q}\). We assume that actions have at most one numeric effect on each numeric variable. We say that action \(a\) is applicable in state \(s\) if \(s \models \text{pre}_a(a) \cup \text{pre}_n(a)\). The result of applying \(a\) in \(s\) is denoted by \(s[a] = \langle s'_p, s'_n \rangle\), with \(s'_p = s_p \setminus \text{del}(a) \cup \text{add}(a)\), while for each \(v \in \mathcal{N}\), \(s_n'[v] = s_n[v] + k\) if \((v \leftarrow k) \in \text{num}(a)\), and \(s_n'[v] = s_n[v]\) otherwise. We replace \(\psi: v > w\) with \(\psi': v \geq w + \varepsilon\), where \(\varepsilon > 0\) is a sufficiently small constant, assuming only one conditions.

The goal conditions \(G = G_p \cup G_n\) denote propositional and numeric conditions, respectively. The set of numeric conditions \(\mathcal{F}_n = \{\psi \mid \psi \in G_n \vee \exists a \in A : \psi \in \text{pre}_n(a)\}\) is called active numeric facts. Note that \(\mathcal{F}_n\) is finite. When a state \(s\) satisfies a condition \(\psi\) or a set of conditions \(\Psi\), we write \(s \models \psi\) and \(s \models \Psi\), respectively. The goal condition \(G\) is a subset of \(\mathcal{F}_p \cup \mathcal{F}_n\). We say that \(s_n\) is a goal state if \(s_n \models G\).

An \(s\)-plan is an action sequence \(\pi\) that can be applied successively in \(s\) and results in a goal state \(s_n \models G\). A plan for \(\Pi\) is an \(s_1\)-plan. The cost of an \(s\)-plan \(\pi\) is the sum of all its action costs and an optimal \(s\)-plan has minimal cost among all possible \(s\)-plans. The optimal cost of an \(s\)-plan is denoted by \(h^*(s)\). The cost of \(\Pi\) is \(h^*(s_1)\).

A disjunctive fact landmark \(L_F \subseteq \mathcal{F}_p \cup \mathcal{F}_n\) is a set of facts such that in the execution of any plan \(\pi\), there is a state \(s\) such that \(s \models \psi\) for some \(\psi \in L_F\). A disjunctive action landmark \(L \subseteq A\) is a set of actions such that set-wise it holds \(L \cap \pi = \emptyset\), for every plan \(\pi\) for \(\Pi\).

If a \(\Pi^{RT}\) does not have any numeric state variable \((\mathcal{N} = \emptyset)\), we have a classical (STRIPS) planning task \(\Pi\). A numeric variable \(v \in \mathcal{N}\) is called a resource variable if it has a non-negative domain \([0, R_v]\), where \(R_v \in \mathbb{Q}^{\geq 0}\) is the maximum capacity, and \(v\) is affected only by actions. When a numeric state variable \(v \in \mathcal{N}\) has a non-negative domain \([0, R_v]\), where \(R_v \in \mathbb{Q}^{\geq 0}\) is the maximum capacity and is affected only by actions which are applicable in \(s\) if \(s[a][v] \in [0, R_v]\), it is called resource variable. If all the numeric variables are resources and the effects are the type \(v \leftarrow k^a\) with \(k^a < 0\) for all \(a \in A, v \in \mathcal{N}\), we have a resource-constrained planning task (RCP) (Nahost, Hoffmann, and Müller 2012), while if there exists at least one action with \(k^a > 0\), we have a planning task with resources (RP) (Wilhelm, Steinmetz, and Hoffmann 2018). RT generalizes RP by allowing numeric variables to have domain \(\mathbb{Q}\) and numeric conditions of the form: \(\psi: v > w^0\), with \(w^0, w^0 \in \mathbb{Q}\).

Scala, Haslum, and Thiebaux (2016) introduce planning with simple numeric conditions (SC), an extension of RT where numeric preconditions are \(\psi: \exists v \in \mathcal{N} v \cdot w^\psi \geq w^0\) with \(w^\psi, w^0 \in \mathbb{Q}\). Tasks with SC only are called SC tasks (SCT). They can be reduced to RT by introducing a new numeric variable for each SC and modifying numeric effects so that they change the variable by the net effects on the SC, as shown in the supplementary material (Kuroiwa et al. 2021).

**Delete-free RT.** We say that an RT is delete-free if for each action \(a \in \mathcal{A}\) it holds that \(\text{del}(a) = \emptyset\) and all numeric effects are of the form \((v \leftarrow + k^a)\) with \(k^a > 0\). The support function \(supp: \mathcal{F}_p \cup \mathcal{F}_n \rightarrow 2^A\) for such tasks is defined to be \(\text{supp}(\psi) = \{a \in \mathcal{A} : \psi \in \text{add}(a)\}\) if \(\psi \in \mathcal{F}_p\) and \(\text{supp}(v \geq w_0) = \{a \in \mathcal{A} : v \leftarrow + k^a \in \text{num}(a)\}\) if \(v \geq w_0 \in \mathcal{F}_n\). The optimal cost of the delete-free RT is denoted by \(h^+\).

The delete-relaxed version of an RT \(\Pi^{RT}\), \(\Pi^+\), is a delete-free RT, where the effects \(\text{del}(a)\) and \(\forall (v \leftarrow + k) \in \text{num}(a)\) if \(k \leq 0\) are ignored for all \(a \in \mathcal{A}\). The optimal cost of \(\Pi^+\), \(h^+\), is an admissible estimate of the optimal cost of \(\Pi^{RT}\), since all plans for \(\Pi^{RT}\) are also plans for \(\Pi^+\). In what follows, we consider heuristics for delete-relaxed versions of RTs.

**LM-Cut in Classical Planning**

Helmert and Domshlak (2009) introduce the LM-cut heuristics for classical planning and show that it produces excellent estimates of \(h^+\) for many tasks. The heuristic is based on disjunctive action landmarks, computed as a cut in a labelled weighted digraph called justification graph.

A labelled weighted digraph is formally defined by a triplet \(G = (N, E, W)\), where \(N\) are the vertices of the graph, \(E \subseteq N \times N \times A\) are labelled edges of the graph, where \(A\) denotes the label set, and \(W: E \rightarrow \mathbb{R}^{\geq 0}\) is the weight function on edges. An interleaving sequence of vertices and labels \((n_0, a_0, n_1, \ldots, a_m, n_{m+1})\) (that can be viewed as a sequence of edges) is called a path if for each \(i \in \{0, \ldots, m\}\) it holds that \((n_i, n_{i+1}; a_m) \in E\). When the labels are evident from the context we denote such path by \(\alpha(n_0, n_{m+1})\) and say that \(n_{m+1}\) is reachable from \(n_0\) if such a path exists. Given two disjoint sets \(N_1, N_2 \subseteq N\), we define a directed cut to be \((N_1, N_2) = \{(n_1, n_2; a) : a \in E \land n_1 \in N_1, n_2 \in N_2\}\). We also assume that \(N_1 \cup N_2 = N\). The weights of the path \(\pi\) and the cut \(L\) are denoted respectively as \(W(\pi) = \sum_{(n \in \pi)} W(e)\) and \(W(L) = \min_{(n \in \pi)} W(e)\).

Lastly, for a vertex \(n_0 \in N\), we define the set of edges incident to \(n_0\) as \(in(n_0) = \{(n, n_0; a) \in E \mid n \in N, a \in A\}\) and the in-border of \(E^\prime \subseteq E\) as \(\partial^m(E^\prime) = \{(n_0 \in N \mid in(n_0) \cap E^\prime = \emptyset)\}\).

The computation of the LM-cut heuristic is performed in rounds, leveraging the concept of cost partitioning (see Definition 1) to guarantee the admissibility of the heuristic. Starting with \(h^{LM-cut}(s) = 0\), each round proceeds as follows:

1. Compute the \(h^{max}\) values (Bonet and Geffner 2001) of all relevant facts. If \(max_{a \in G} h^{max}(g) = 0\) return the computed heuristic value. If there is \(g \in G\) such that \(h^{max}(g) = \infty\) return \(\infty\).
2. Use the \(h^{max}\) values to construct a Justification Graph (JG) and using this graph compute a disjunctive action landmark \(L\) with nonzero cost \(c\). Update \(h^{LM-cut}(s) = c\).
3. Reduce the costs of all actions in \(L\) by \(c\).
4. Go to Step 1, using the updated action costs.

A formal definition of the computation can be found in the literature (Helmert and Domshlak 2009; Bonet and Helmert 2010).
LM-cut in RT Planning

We can use the exact same schema as described in the previous section to compute an admissible heuristic for RT planning. To prove that the heuristic remains admissible when considering numeric state variables, we need the following building blocks: (1) an assurance that the cost-partitioning concept is valid in RT, (2) an $h^\text{max}$ heuristic that handles numeric conditions, and (3) a JG with numeric preconditions and effects. These building blocks and the appropriate proofs are discussed below.

Cost-Partitioning in RT Planning

In their works, Katz and Domshlak (2008) and Yang et al. (2008) independently proposed an approach to additively combine individual admissible heuristic estimates.

Definition 1. Given a planning task $\Pi$, its cost-partitioning is a family of planning tasks $\{\Pi_i\}_{i=1}^n$ where each task differs from $\Pi$ only by its cost function $c$, and it holds that $\forall a \in A : \sum_{i=1}^n c_i(a) \leq c(a)$.

Proposition 1. Given a planning task $\Pi$, cost partitioning $\{\Pi_i\}_{i=1}^n$, and an admissible heuristic function $h_i$ for each $\Pi_i$, the heuristic function $h(s) = \sum_{i=1}^n h_i(s)$ is admissible.

Here we give a simplified version of the claim proposed by Katz and Domshlak. The proof of this, less general, claim is given in the supplementary material (Kuroiwa et al. 2021). Note that the proof does not rely on any formalism properties and depends only on the transition system. Therefore, Prop. 1 holds for numeric planning.

$h^\text{max}$ in Numeric Planning

While the computation of $h^\text{max}$ in classical planning (Bonet and Geffner 2001) can be done in polynomial time, with numeric state variables, the problem of calculating $h^\text{max}$ is NP-hard even if we are restricted to RTs.

Let $h^\text{max}(s) = h(s, G)$ be defined as a maximal fixed-point of the following recursive equations: $h(s, F) = \max_{\psi \in F} h(s, \psi)$ for $F \subseteq F_p \cup F_n$ and

\[
\hat{h}(s, \psi) = \begin{cases} 
\min_{a \in \text{supp}(\psi)} h(s, \psi(a)) + \text{cost}(a) & \text{if } s \models \psi \\
0 & \text{otherwise.}
\end{cases}
\]

Here, pre$_\psi(a)$ is pre$(a)$ if $\psi \in F_p$, or pre$(a) \cup \{v \geq w - k^a\}$ if $\psi : v \geq w \in F_n$. Note that $h^\text{max}$ can be computed in a finite number of steps (Scala, Haslum, and Thiébaux 2016).

Proposition 2. Given an RT and a state $s$, the problem of computing $h^\text{max}$ is NP-hard (Kuroiwa et al. 2021).

As in the classical case, the RT heuristic can be interpreted using both interval (Aldinger and Nebel 2017) and subgoal negation relaxations (Scala, Haslum, and Thiébaux 2016), but because of its intractability, a further relaxation is needed. Scala et al. (2016) modified $h^\text{max}$ by introducing a function $m_a(s, \psi)$ that, intuitively, accounts for the number of times action $a$ must be applied in state $s$ to reach fluent $\psi$.

Definition 2. Given a delete-free RT, a state $s$, a fact $\psi \in F$, and action $a \in \text{supp}(\psi)$, We define $m_a$ as

\[
m_a(s, \psi) = \begin{cases} 
0 & \text{if } s \models \psi \\
\frac{\text{cost}(a)}{\text{num}(a)} & \text{if } s \not\models \psi \land \psi : v \geq w \in F_n \\
1 & \text{otherwise.}
\end{cases}
\]

where $k^a \in \mathbb{Q}^{\geq 0}$ is the numeric effect of action $a$ on variable $v$, i.e., $(v + = k^a) \in \text{num}(a)$. If $a \not\in \text{supp}(\psi)$, we say $m_a(s, \psi) = \infty$.

This definition allows us to restrict our fluents to the set of active numeric facts, $F_n$. Note that in contrast to $F_n$, $F_p$ is a finite set. This notation allows us to define another version of $h^\text{max}$ that we denote by $h_{\text{cri}}^\text{max}$.

Definition 3. Given an RT and a state $s$, we define heuristic function $h_{\text{cri}}^\text{max}(s) := h_{\text{cri}}^\text{max}(s, G)$ as follows. For a set of facts $F \subseteq F_p \cup F_n:

\[
h_{\text{cri}}^\text{max}(s, F) = \max_{\psi \in F} h_{\text{cri}}^\text{max}(s, \psi).
\]

For a fact $\psi \in F_p \cup F_n$, $h_{\text{cri}}^\text{max}(s, \psi) = 0$ if $s \models \psi$, or otherwise

\[
h_{\text{cri}}^\text{max}(s, \psi) = \min_{a \in \text{supp}(\psi)} h_{\text{cri}}^\text{max}(s, \text{pre}(a)) + m_a(s, \psi) \cdot \text{cost}(a).
\]

Unfortunately, as we show in Ex. 1, $h_{\text{cri}}^\text{max}$ is not an admissible heuristic. In contrast, in their work, Scala et al. (2016) propose another level of relaxation, decoupling preconditions and numeric effects of actions, that assures the admissibility of their $h^\text{max}$ version (i.e., $h_{\text{cri}}^\text{max}$).

Definition 4. Given an RT and a state $s$, the heuristic function $h_{\text{hb}}^\text{max}$ is defined exactly as $h_{\text{cri}}^\text{max}$, except when $\psi \in F_n$ is a numeric fact, in this case

\[
h_{\text{hb}}^\text{max}(s, \psi) = \min_{a \in \text{supp}(\psi)} h_{\text{hb}}^\text{max}(s, \text{pre}(a)) + \min_{a \in \text{supp}(\psi)} m_a(s, \psi) \cdot \text{cost}(a).
\]

Note that while these definitions address all subsets of $F_p \cup F_n$, in practice we need to compute $h_{\text{cri}}^\text{max}$ only for the fluents in the finite set $F_p \cup F_n$, which allows us to compute both $h_{\text{cri}}^\text{max}$ and $h_{\text{hb}}^\text{max}$ in polynomial time in the size of the RT. It is also important to note that for any state $s$ in a given RT, $h_{\text{hb}}^\text{max}(s, \psi) \leq h_{\text{cri}}^\text{max}(s, \psi)$ because $h_{\text{hb}}^\text{max}$ decouples preconditions and effects of actions that have numeric fluents, while $h_{\text{cri}}^\text{max}$ does not.

The Justification Graph and LM-cut

To follow the scheme of Helmert and Domshlak (2009) for the numeric version of LM-cut, we need to construct a JG using our $h_{\text{cri}}^\text{max}$ estimates.

Helmert and Domshlak (2009) defined a JG to have a singleton goal set and all operators to have exactly one precondition, one add effect, and no delete effects. This defines a directed weighted graph $G$ whose vertices correspond to the facts of the planning task and labeled weighted edges correspond to actions. In the classical case, the weight of the shortest path from a vertex representing some fact in $s$ to another fact $\psi$, corresponds to the $h_{\text{cri}}^\text{max}(s, \psi)$ value, hence the graph is said to justify $h_{\text{cri}}^\text{max}$.

To meet the requirement of having exactly one precondition, the $h_{\text{cri}}^\text{max}$-values need to be preserved. Intuitively speaking, the precondition choice function defined by Bonet

\footnote{The notation “cri” is chosen since this version of $h^\text{max}$ admits the critical path value in our version of the justification graph.}
and Helmert (2010) maps each action to one of its preconditions, which in classical planning is done via maximizing $h_{\text{cri}}^{\text{max}}$. Thus, for $h_{\text{cri}}^{\text{max}}$, we can write as follows.

**Definition 5.** Given an RT, a precondition choice function $\text{pcf}_{\text{cri}} : S \times A \rightarrow \mathcal{F}_p \cup \mathcal{F}_n$ is the function that satisfies the condition

$$\text{pcf}_{\text{cri}}(s, a) \in \arg \max_{\psi \in \text{pre}(a)} h_{\text{cri}}^{\text{max}}(s, \psi).$$

For actions with no precondition, we add an artificial fact denoted by $\emptyset$. If $\text{pre}(a) = \emptyset$, we write $\text{pcf}_{\text{cri}}(s, a) = \emptyset$.

In contrast to $h_{\text{cri}}^{\text{max}}$, $h_{\text{cri}}^{\text{max}}$ decouples numeric effects of an action from its preconditions. To make $\text{pcf}_{\text{cri}}$ justify $h_{\text{cri}}^{\text{max}}$, $\text{pcf}_{\text{cri}}$ is defined to map each action/effect pair $\langle a, \psi \rangle$, not each action, to one of the preconditions of some $\hat{a} \in \text{supp}(\psi)$. Thus, it is possible that $\text{pcf}(s, a, \psi) \neq \text{pcf}(s, a, \psi')$ for $\psi \neq \psi'$ and $\text{pcf}(s, a, \psi) \notin \text{pre}(a)$. The result is a precondition choice function defined as follows:

$$\text{pcf}_{\text{cri}}(s, a, \psi) \in \arg \max_{\psi' \in \text{pre}(a)} h_{\text{cri}}^{\text{max}}(s, \psi').$$

where $\hat{a} = a$ if $\psi \in \mathcal{F}_p$, or otherwise

$\hat{a} \in \arg \min_{a' \in \text{supp}(\psi)} h_{\text{cri}}^{\text{max}}(s, \text{pre}(a')).$

Our definitions of JG for $h_{\text{cri}}^{\text{max}}$ and $h_{\text{cri}}^{\text{max}}$ differ only on the precondition chosen by the $\text{pcf}$ version. We use the generalised term $\text{pcf}$ as a short notation for both function $\text{pcf}_{\text{cri}}$ and $\text{pcf}_{\text{cri}}$, where $\text{pcf}_{\text{cri}}$ is extended with a mute variable $\psi$.

**Definition 6.** Given an RT and a state $s$, the justification graph is a labelled weighted digraph $G = (N, E, W)$ with

1. a set of vertices $N = \{ n_\psi | \psi \in \mathcal{F}_p \cup \mathcal{F}_n \cup \{ \emptyset \} \}$;
2. a set of labeled edges:
   $$E = \hat{E} \cup \{(n_\psi, n_{\psi'}, a) | a \in \text{supp}(\psi'), \psi = \text{pcf}(s, a, \psi') \};$$
   where for simplicity we include the following zero-cost edges
   $$\hat{E} = \{(n_0, n_0; a_0) | s \models \psi \};$$
3. and a weight function $W : E \rightarrow \mathbb{R}^+$ defined as
   $$(n_\psi, n_{\psi'}; a) \mapsto m_a(s, \psi') \cdot \text{cost}(a).$$

The function $\text{lbl} : E \rightarrow A$ is defined as $(n_\psi, n_{\psi'}; a) \mapsto a$.

The construction of the JG is at most quadratic in the size of the RT problem and, when no numeric conditions are present, the construction is identical to that of the JG used in classical planning formalisms such as STRIPS or FDR (Bäckström and Nebel 1995; Helmert 2006).

**Admissibility**

We now turn to the admissibility status of the four heuristics discussed above, as summarized in Table 1.

**h_{\text{cri}}^{\text{max}} Heuristics.** While Scala et al. (2016) proved the admissibility of $h_{\text{cri}}^{\text{max}}$, we show in Ex. 1 that $h_{\text{cri}}^{\text{max}}$ is inadmissible.

**Example 1.** Let $(\mathcal{F}_p, N, A, s_1, G)$ be an RT with $\mathcal{F}_p = \emptyset$ and $N = \{ v \}$. Let $s_1 = \{ v = 0 \}$, $G = \{ v \geq 6 \}$, and $A = \{ a_1, a_2 \}$, where

<table>
<thead>
<tr>
<th>action</th>
<th>pre</th>
<th>eff</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>$\emptyset$</td>
<td>$v = +1$</td>
<td>1</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$v \geq 2$</td>
<td>$v = +2$</td>
<td>1</td>
</tr>
</tbody>
</table>

Note, that in this case $\mathcal{F}_n = \{ v \geq 2, v \geq 6 \}$. The JG of the first and second iterations can be seen in Figure 1, where the critical path is indicated in red and the landmarks of numeric LM-cut are denoted by blue lines. Thus, we have:

$h_{\text{cri}}^{\text{max}}(s_1) = 3 < h_{\text{cri}}^{\text{LM-cut}}(s_1) = h^*(s_1) = 4 < h_{\text{cri}}^{\text{max}}(s_1) = 5.$

The example also shows that $h_{\text{cri}}^{\text{max}}$ does not dominate $h_{\text{cri}}^{\text{LM-cut}}$ even in the case where all variables are numeric. This complements the fact that $h_{\text{cri}}^{\text{max}}$ is dominated by $h_{\text{cri}}^{\text{LM-cut}}$ in classical planning (Helmert and Domshlak 2009).

**h_{\text{cri}}^{\text{LM-cut}} Heuristics.** The JGs based on $\text{pcf}_{\text{cri}}$ and $\text{pcf}_{\text{cri}}$ admit the property of justifying $h_{\text{cri}}^{\text{max}}$ and $h_{\text{cri}}^{\text{max}}$, correspondingly. However, it is important to note the following.

**Proposition 3.** The LM-cut heuristic based on the $\text{pcf}_{\text{cri}}$ JG, $h_{\text{cri}}^{\text{LM-cut}}$, is inadmissible.

The intricate example of inadmissibility of $h_{\text{cri}}^{\text{LM-cut}}$ is shown in the supplementary material (Kuroiwa et al. 2021).

We now turn to the proof of admissibility of $h_{\text{cri}}^{\text{LM-cut}}$. Since $h_{\text{cri}}^{\text{LM-cut}}$ is inadmissible, we construct the JG using the $\text{pcf}_{\text{cri}}$ function. To simplify the notation, in what follows, we drop the subscript and write $h_{\text{cri}}^{\text{LM-cut}}$ instead of $h_{\text{cri}}^{\text{LM-cut}}$.

Without compromising the admissibility, we restrict the vertices of $G$ to the following: for each $n_\psi \in N$ it holds that there are facts $\psi' \in s$ and $\psi'' \in G$ such that there
exist a path \((n_\psi, n_\psi)\) and a path \((n_\psi, n_{\psi'})\). Note that both checks are polynomial, and assure that \(h_{\text{cut}}^{\max}(s, \psi) < \infty\). To establish a goal vertex for our computation we choose
\[
g \in \arg\max_{\psi \in G} h_{\text{cut}}^{\max}(s, \psi).
\]

**Definition 7.** Given an RT, a state \(s\), and the JG \(G = \langle N, E, W \rangle\), the **goal zone** of the graph \(G\) is the set of vertices that can reach the goal fact \(g\) at zero cost and is defined as:
\[
N^g = \{ n_\psi \in N \mid \exists \text{path}(n_\psi, n_g) : W(\text{path}(n_\psi, n_g)) = 0 \}.
\]

The **before-goal zone** is a set of vertices that can be reached from the vertex \(n_0\) without passing through \(N^g\):
\[
N^0 = \{ n_\psi \in N \mid \exists \text{path}(n_0, n_\psi) : \text{path}(n_0, n_\psi) \cap N^g = \emptyset \}.
\]

Lastly, the **beyond-goal zone** is \(N^b = (N \setminus N^g) \setminus N^0\).

The RT problem is unsolvable if there are no directed paths from \(n_0\) to \(n_g\). Whereas, if \(n_0 \in N^g\) there is a path from \(n_0\) to \(n_g\) such that \(W(\text{path}(n_0, n_g)) = 0\), which in turn means that heuristic value for \(s\) is 0.

Scala et al. (2017) defined landmarks for numeric planning. Following Helmert and Domshlak (2009), we show below how to extract such landmarks from JGs.

**Lemma 1.** Assume an RT of a non-zero optimal cost. Let \(G\) be the JG corresponding to \(\Pi^{RT}\), and let \(L\) be a directed cut in \(G\) that separates \(n_0\) from \(n_g\), such that \(W(L) > 0\). Then,
\[1. \ \partial^{\text{m}}(L)\text{ is a disjunctive fact landmark.}
\]
\[2. \ \text{lb}(L)\text{ is a disjunctive action landmark.}\]

The proof of this claim is presented in the supplementary material (Kuroiwa et al. 2021). We also note that this property does not necessarily hold for a JG that is constructed based on pcfbd.

Now, we are ready to state and prove our main theoretical claim that states that numeric LM-cut is admissible.

**Theorem 1.** Let \(\Pi^{RT} = \{ F_p, N, A, s, g \}\) be a solvable RT with a non-zero optimal cost. Let \(G = \langle N, E, W \rangle\) be the JG corresponding to \(\Pi^{RT}\), and let \(N^0, N^g\) and \(N^b\) be before-, beyond- and goal zones, as defined above. Let \(L = (N^0, N^g \cup N^b)\) be a directed cut in \(G\). We define the weight of the cut to be the weight of the minimal edge in the cut, i.e.
\[W(L) = \min_{e \in L} W(e).
\]
\[h_1(s) = W(L)\text{ is admissible for }\Pi^{RT}, \text{ where }\Pi^{RT}\text{ is a copy of }\Pi^{RT}\text{, with the augmented cost function }\]
\[\text{cost}_1(a) = \left\{
\begin{array}{l}
\frac{W(L)}{m_a^L} & \text{if } a \in \text{lb}(L) \\
0 & \text{otherwise},
\end{array}
\right.
\]
\text{where } m_a^L = \min_{(n_\psi, n_{\psi'}) \in L} m_a(s, \psi'), \text{ i.e., the minimum over all } m's \text{ of the same action edges in the cut.}

**Proof.** First, we show that \(\text{cost}_1(a) \leq \text{cost}(a)\) for all \(a \in A\). The claim is clear for \(a \notin \text{lb}(L)\), otherwise there is \((n_\psi, n_{\psi'}; a) \in L\) such that \(W(n_\psi, n_{\psi'}; a) = m_a^L \cdot \text{cost}(a)\), thus
\[\text{cost}_1(a) = \frac{W(L)}{m_a^L} \leq \frac{m_a^L \cdot \text{cost}(a)}{m_a^L} = \text{cost}(a).
\]

To finish the claim, all is left to show that the weight of \(L\) is an admissible estimate for the solution of \(\Pi^{RT}\). Assume in contradiction that it is not, i.e., there is a plan \(\pi\) such that
\[\text{cost}_1(\pi) < W(L).
\]

By Lem. 1 point 1, \(\partial^{\text{m}}(L)\) is a disjunctive fact landmark. Thus, there is at least one fluent in \(\partial^{\text{m}}(L)\) that is achieved by the plan \(\pi\).\(^4\) We denote by \(\psi_0\) the first fluent in \(\partial^{\text{m}}(L)\) that is achieved by \(\pi\). Note that \(\text{lb}(\text{in}(n_\psi)) = \text{supp}(\psi)\); we write \(L_\psi = L \cap \text{in}(n_\psi)\). By construction of the JG, we have that
\[\min_{a \in \text{lb}(L_\psi)} m_a(s, \psi_0) \cdot \text{cost}_1(a) \leq \text{cost}_1(\pi) < W(L).
\]

The fact that \(m_a^L \leq m_a(s, \psi_0)\) allows us to conclude with the following contradiction
\[W(L) \leq \min_{a \in \text{lb}(L_\psi)} \frac{W(L)}{m_a^L} \leq \min_{a \in \text{lb}(L_\psi)} m_a(s, \psi_0) \cdot \text{cost}_1(a).
\]

Using Prop. 1 and Thm. 1, we finalize our claim.

**Corollary 1 (Admissibility).** The LM-cut heuristic for RT is admissible, and can be computed in polynomial time.

**Proof.** Admissibility follows directly from Prop. 1 and Thm. 1. The computation of \(h_{\text{cut}}^{\max}(s, \psi)\) and the construction of the JG are both polynomial in RT and the cuts \(L\) are produced in polynomial time in the size of JG. Thus, if we show that the number of such cuts does not exceed \(|A|\) we can prove our claim. We show that for each \(L\) there is at least one action \(a \in \text{lb}(L)\) where the cost is reduced to zero.

Let \((\psi, \psi'; a) \in L\) be the edge where \(W(L)\) achieves its minimum \((W(L) > 0)\). By definition of \(W\) we have that
\[W(L) = W(\psi, \psi'; a) = m_a(s, \psi') \cdot \text{cost}(a) = m_a^L \cdot \text{cost}(a).
\]
Thus, the updated cost of \(a \in A\), \(\text{cost}_1(a)\), is
\[\text{cost}(a) - \frac{W(L)}{m_a^L} = \text{cost}(a) - \frac{m_a^L \cdot \text{cost}(a)}{m_a^L} = 0.
\]

Finally, we show that \(h_{\text{cut}}^{\max}\) does not dominate \(h_{\text{cut}}^{\max}\).

**Example 2.** Let \(\{ F_p, N, A, s_1, g \}\) be an RT with \(F_p = \{ p, g \}\) and \(N = \{ v \}\). Let, the rest of the elements in the tuple be \(s_1 = \{ v = 0 \}, G = \{ g \}, \text{ and } A = \{ a_1, a_2, a_3, a_4 \}\), where
\[\\]
\(^4\)This is a minor abuse of notation: fluent \(\psi\) corresponds to the node \(n_\psi \in \partial^{\text{m}}(L)\).
LM-Cut for Operator-Counting Constraints

Given a disjunctive action landmark \( L \), the landmark constraint is as follows (Bonet and Helmer 2010):

\[
\sum_{a \in \text{lbl}(L)} X_a \geq 1. 
\]  

The above inequality is a valid OC constraint. In classical planning, the landmarks extracted by \( h_{\text{LM-cut}} \) can be used to generate the landmark constraints (Pommerening et al. 2014). Here, we generalize this approach.

Theorem 2. Given an RT, let \( L \) be a cut set obtained by the LM-cut heuristic. Let \( \text{count}(\pi,a) \) be the number of times action \( a \) appears in a plan \( \pi \). The following holds for any \( \pi \)

\[
\sum_{a \in \text{lbl}(L)} \frac{\text{count}(\pi,a)}{m_a^L} \geq 1. 
\]

Proof. Let \( \pi \) be a plan for the RT, and let \( L = (N^0, N^a \cup N^b) \) be the cut in the JG. Recall that \( \partial^a(L) \) is a disjunctive fact landmark. Thus, there is at least one fluent in \( \partial^a(L) \) that is achieved by the plan \( \pi \). Let \( \psi_0 \) be the first fluent in \( N^a \cup N^b \) that is achieved by \( \pi \), and let \( a_0 \) be the action in \( \pi \) that achieves \( \psi_0 \), i.e., \( a_0 \in \text{lbl}(L_{\psi_0}) \cap \pi \), where \( \text{lbl}(L_{\psi_0}) = \text{supp}(\psi_0) \cap \text{lbl}(L) \).

Next, note that if \( m_{a_0}(s,\psi_0) \leq 1 \), it holds that

\[
1 \leq \frac{\text{count}(\pi,a_0)}{m_{a_0}(s,\psi_0)} \leq \frac{\text{count}(\pi,a_0)}{m_a^L} \leq \sum_{a \in \text{lbl}(L)} \frac{\text{count}(\pi,a)}{m_a^L}. 
\]

Thus, assume that \( m_{a_0}(s,\psi_0) > 1 \) and \( \psi_0 \in \bar{F}_n \) is a numeric fluent. Assume that \( \psi_0 \) is of the form \( v \geq w_0 \). For each \( a \in \text{supp}(\psi_0) \) it holds that \( (v + = k^a) \in \text{num}(a) \). Since \( \pi \) achieves \( \psi_0 \) we can write

\[
w_0 \leq s[v] + \sum_{a \in \text{lbl}(L_{\psi_0})} k^a \cdot \text{count}(\pi,a).
\]

Now, subtract from both sides of the inequality \( s[v] \) and subsequently divide by \( w_0 - s[v] > 0 \):

\[
1 \leq \sum_{a \in \text{lbl}(L_{\psi_0})} \frac{k^a}{w_0 - s[v]} \cdot \text{count}(\pi,a) = \sum_{a \in \text{lbl}(L_{\psi_0})} \frac{\text{count}(\pi,a)}{m_{a_0}(s,\psi_0)} \leq \sum_{a \in \text{lbl}(L)} \frac{\text{count}(\pi,a)}{m_a^L}. \tag{\ref{eq:oc-constraint}}
\]

From Thm. 2, we derive the following OC constraint:

\[
\sum_{a \in \text{lbl}(L)} \frac{X_a}{m_a^L} \geq 1, \tag{\ref{eq:oc-constraint}}
\]

where \( L \) is a cut obtained by the LM-cut heuristic.

Experimental Evaluation

In all the experiments, we evaluate the heuristics by using them in \( A^* \) search imposing a 30-minute time-limit and 4 GB memory limit on an Intel(X) Xeon(R) CPU E5-2620 @
2.00GHz processor. We implemented the heuristics in Numeric Fast Downward (NFD) (Aldinger and Nebel 2017)\(^5\) using C++11 with GCC 7.5.0 on Ubuntu 18.04 and using CPLEX 12.10 as a mathematical programming solver.

Domains with simple numeric conditions are taken from the literature (Scala et al. 2016, 2017, 2020). We exclude ZENOTRAVEL because some conditions are not simple numeric conditions (Piacentini et al. 2018b). From COUNTERS, we exclude three instances that are in SMALLCOUNTERS. For SAILING, in addition to the original instances (Scala et al. 2016), we include the instances with a single boat (Scala et al. 2017), removing duplicates. Since multiple configurations solve all instances in FARMLAND, GARDENING, and SAILING, we also added satisfying versions of these domains (FARMLAND-SAT, GARDENING-SAT, and SAILING-SAT) excluding instances appearing in the optimal versions. A task is translated into an RT when computing the numeric LM-cut. We implemented the numeric LM-cut and LP/IP-based heuristics. For the numeric heuristics, we add the redundant constraints to the goal conditions and preconditions of actions in the same way as Scala et al. (2016).

**Comparison with Propositional LM-Cut**

To evaluate the benefit of considering numeric conditions in LM-cut, we compare the numeric and propositional versions. In the numeric domains, the propositional LM-cut ignores numeric conditions, i.e., all numeric conditions are achieved with zero cost when computing heuristic values.

We omit the domains COUNTERS, FARMLAND-SAT, and CHILDSNACK because both versions of LM-cut solve no instances. The results in the top of Table 2 show that the numeric version of $h^{LM-cut}$ outperforms the propositional.

In addition, we consider a set of domains containing resource variables from the optimal IPC track, translated into RP tasks (Wilhelm, Steinmetz, and Hoffmann 2018). If there are multiple versions for the same domain, we use the latest one. Search is performed on the original space, while the numeric heuristic is computed using the transformed task. The results of this comparison are shown in the bottom of Table 2. The translation into numeric version increases the coverage on three domains while reducing the number of the expanded states and search time on seven domains. However, the propositional version solves more instances than the numeric version on four domains.

**Comparison with Numeric Heuristics**

We compare the numeric LM-cut heuristic ($h^{LM-cut}$) with the following numeric heuristics: the interval relaxation based heuristic $h^{\text{irmax}}$ (Aldinger and Nebel 2017), the numeric max heuristic $h^{\text{max}}_{\text{hbd+}}$ (Scala et al. 2020), the numeric landmark heuristic $h^{\text{lmax}}_{\text{lbd}}$ (Scala et al. 2017), and the generalized sub-goaling heuristic $h^{\text{gen}}_{\text{hbd}}$ (Scala et al. 2020). For $h^{\text{max}}_{\text{hbd+}}$, $h^{\text{lmax}}_{\text{lbd}}$, and $h^{\text{gen}}_{\text{hbd}}$, in addition to our implementations in NFD, we evaluate the original implementations in ENHSP-19\(^6\) using OpenJDK 11.0.9.1.

\(^5\)https://github.com/Kurororo/numeric-fast-downward
\(^6\)https://sites.google.com/view/enhsp/

<table>
<thead>
<tr>
<th>Domain</th>
<th>Coverage ('c.'), Average Time ('t.'), # of States</th>
<th>Prop</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLCOUNTERS (8)</td>
<td>6 $8.75$ 160932</td>
<td>7 $0.46$ 5738</td>
<td></td>
</tr>
<tr>
<td>COUNTERS-INV (11)</td>
<td>2 $0.04$ 1734</td>
<td>2 $0.00$ 34</td>
<td></td>
</tr>
<tr>
<td>COUNTERS-RND (33)</td>
<td>6 $0.01$ 476</td>
<td>9 $0.00$ 6</td>
<td></td>
</tr>
<tr>
<td>FARMLAND (30)</td>
<td>11 $21.39$ 219468</td>
<td>30 $0.05$ 486</td>
<td></td>
</tr>
<tr>
<td>GARDENING (63)</td>
<td>63 $7.78$ 198429</td>
<td>63 $1.57$ 18622</td>
<td></td>
</tr>
<tr>
<td>GARDENING-SAT (51)</td>
<td>10 $93.12$ 2308994</td>
<td>12 $15.39$ 163362</td>
<td></td>
</tr>
<tr>
<td>SAILING (40)</td>
<td>9 $92.17$ 2283579</td>
<td>40 $0.19$ 1619</td>
<td></td>
</tr>
<tr>
<td>SAILING-SAT (40)</td>
<td>3 $64.76$ 3085470</td>
<td>14 $0.98$ 23241</td>
<td></td>
</tr>
<tr>
<td>DEPARTS (20)</td>
<td>7 $95.36$ 49827</td>
<td>7 $97.37$ 49873</td>
<td></td>
</tr>
<tr>
<td>ROVERS (20)</td>
<td>4 $6.91$ 126592</td>
<td>4 $7.44$ 130760</td>
<td></td>
</tr>
<tr>
<td>SATELLITE (20)</td>
<td>2 $36.70$ 78620</td>
<td>2 $33.34$ 47351</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Coverage (‘c.’), average time (‘t.’), and # of states expanded (‘e.’) by the propositional and numeric LM-cut heuristics. The time and # of states are averaged over instances solved by both versions. The numeric domains are in the upper and the classical domains are in the lower half.

We show the experimental results in the left-hand side of Table 3. On FARMLAND-SAT, since instances are large, NFD runs out of memory when translating PDDL files to SAS+ files and does not solve any instance. Our LM-cut heuristic improves coverage by 17 tasks compared to the next best heuristic and attains the smallest run-time on 8 of the 11 domains, while expanding more states than at least one competitor in all domains except DEPARTS.

**Comparison in the OC Framework**

In classical planning, the combination of the LM-cut constraints and SEQ outperforms the individual components (Pommerening et al. 2014). To examine whether this is also the case with numeric planning, we evaluate the following OC heuristics using LP: $h^{LP}_{LM-cut}$ which uses the LM-cut constraints (Eq. (6)), $h^{SEQ}_{LP}$ which uses the numeric planning version of SEQ, and $h^{LM-cut,SEQ}_{LP}$ which uses both. In addition, we compare these heuristics with $h^c_{LP}$, an OC heuristic that uses LP with the delete-relaxation constraints and SEQ (Piacentini et al. 2018b). The results of this comparison are shown in the right-hand side of Table 3.

While $h^{LM-cut}_{LP}$ and $h^{SEQ}_{LP}$ are complementary on most of the domains, the coverage of $h^{LM-cut,SEQ}_{LP}$ is equal to the max-
### Table 3: Coverage, average time in seconds, and # of states expanded by different numeric heuristics on simple-condition numeric planning problems with simple numeric conditions.

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Coverage</th>
<th>Time (s)</th>
<th>States expanded</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLCOUNTERS (8)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COUNTERS (8)</td>
<td>0.04</td>
<td>3.91</td>
<td>75</td>
</tr>
<tr>
<td>COUNTERS-INV (11)</td>
<td>0.01</td>
<td>4.02</td>
<td>73</td>
</tr>
<tr>
<td>COUNTERS-RND (33)</td>
<td>0.00</td>
<td>4.03</td>
<td>72</td>
</tr>
<tr>
<td>FARMLAND (30)</td>
<td>0.00</td>
<td>4.04</td>
<td>71</td>
</tr>
<tr>
<td>FARMLAND-SAT (20)</td>
<td>0.00</td>
<td>4.05</td>
<td>70</td>
</tr>
<tr>
<td>Gardening (63)</td>
<td>0.00</td>
<td>4.06</td>
<td>71</td>
</tr>
<tr>
<td>Gardening-SAT (51)</td>
<td>0.00</td>
<td>4.07</td>
<td>72</td>
</tr>
<tr>
<td>SAILING (40)</td>
<td>0.00</td>
<td>4.08</td>
<td>73</td>
</tr>
<tr>
<td>SAILING-SAT (40)</td>
<td>0.00</td>
<td>4.09</td>
<td>74</td>
</tr>
<tr>
<td>DEPOTS (20)</td>
<td>0.00</td>
<td>4.10</td>
<td>75</td>
</tr>
<tr>
<td>ROVERS (20)</td>
<td>0.00</td>
<td>4.11</td>
<td>76</td>
</tr>
<tr>
<td>SATELLITE (20)</td>
<td>0.00</td>
<td>4.12</td>
<td>77</td>
</tr>
</tbody>
</table>

Table 3: Coverage, average time in seconds, and # of states expanded by different numeric heuristics on simple-condition domains. ‘N’ and ‘E’ mean the implementations in NFD and ENHSP-19, respectively. The time and # of states are averaged over instances solved by all of the domains. 'N' and 'E' mean the implementations in NFD and ENHSP-19, respectively. The time and # of states are averaged over instances solved by all of the domains. The coverage is highlighted in bold if it is the highest among its heuristics family (right- and left-hand sides of the table).

### Conclusion

This paper presents an extension of the LM-cut heuristic to numeric planning problems with simple numeric conditions. In order to obtain an admissible estimate, we introduce a new variant of $h_{LP}$ and theoretically analyze the effect of different relaxations both on $h_{max}$ and $h_{LM-cut}$. Although our admissible version of $h_{LM-cut}$ does not show any theoretical dominance over the existing numeric heuristic $h_{max}$, empirically it better approximates the plan cost, improving on the coverage. The strong performance is even more evident when combining LM-cut with SEQ constraints using the operator-counting framework, as it achieves state-of-the-art performance in most numeric domains. The explicit use of numeric state variables also produces more accurate LM-cut heuristic estimations in most cases.


