Managing Infinite Abstractions in Numeric Pattern Database Heuristics

Markus Fritzsche[1], **Daniel Gnad**[1,2], **Mikhail Gruntov**[3], **Alexander Shleyfman**[4]
[1]Linköping University, Sweden
[2]Heidelberg University, Germany
[3]Technion - Israel Institute of Technology, Israel
[4]Bar-Ilan University, Israel
markus.fritzsche@liu.se, daniel.gnad@uni-heidelberg.de, gruntovm@campus.technion.ac.il, alexash@biu.ac.il

Abstract

Pattern Database (PDB) heuristics are an established approach in optimal classical planning that is used in state-of-the-art planning systems. PDBs are based on projections, which induce an abstraction of the original problem. Computing all cheapest plans in the abstraction yields an admissible heuristic. Despite their success, PDBs have only recently been adapted to numeric planning, which extends classical planning with numeric state variables. The difficulty in supporting numeric variables is that the induced abstractions, in contrast to classical planning, are generally infinite. Thus, they cannot be explored exhaustively to compute a heuristic. The foundational work that introduced numeric PDBs employed a simple approach that computes only a finite part of the abstraction. We analyze this framework and identify cases where it necessarily results in an uninformed heuristic. We propose several improvements over the basic variant of numeric PDBs that lead to enhanced heuristic accuracy.

Code — https://github.com/dgnad/numeric-fast-downward
Datasets — https://doi.org/10.5281/zenodo.17592540

Introduction

Numeric planning concerns the algorithmic generation of action sequences that transition a state space from a given initial state to a goal state. In the past, research predominantly focused on classical planning, characterized by finite-domain state variables. However, practical applications often require reasoning over numerical quantities, such as resource consumption or spatial distances. Numeric planning, where state variables may assume numerical values, enables such scenarios. As a consequence of allowing numeric variables, state spaces are generally infinite, which makes numeric planning undecidable (Helmert 2002). Despite the high complexity, effective algorithms for numeric planning have been developed (Shin and Davis 2005; Gerevini, Saetti, and Serina 2008; Illanes and McIlraith 2017; Cardellini, Giunchiglia, and Maratea 2024), which are often based on heuristic search (Eyerich, Mattmüller, and Röger 2009; Coles et al. 2013; Scala et al. 2016; Li et al. 2018; Scala et al. 2017; Aldinger and Nebel 2017; Piacentini et al. 2018; Kuroiwa et al. 2022; Kuroiwa, Shleyfman, and Beck 2023).

Our work extends a recently proposed heuristic-search approach to optimal numeric planning using pattern database (PDB) heuristics (Gnad et al. 2025). We adopt the established simple numeric planning (SNP) formalism, where numeric variables are subject to changes by a constant, and preconditions and goals are defined by linear inequalities (Hoffmann 2003; Scala et al. 2016). PDB heuristics are known for their effectiveness in classical planning and are based on a projection of the state space onto a subset of variables, the pattern. The abstract state space induced by the pattern is explored exhaustively, and shortest goal distances are computed and stored for all abstract states (Culberson and Schaeffer 1996, 1998; Edelkamp 2002; Holte et al. 2004; Felner, Korf, and Hanan 2004; Haslum, Bonet, and Geffner 2005; Holte et al. 2006; Anderson, Holte, and Schaeffer 2007; Haslum et al. 2007; Katz and Domshlak 2009).

While PDBs are widely used in classical planning, their application to numeric planning remains limited. Gnad et al. (2025) introduced numeric PDBs recently, addressing the challenge of infinite abstract state spaces by only partially exploring the abstractions. Their results showed the potential of PDBs for SNP, but limitations remain: (1) their methods may explore irrelevant abstract states, (2) when no abstract goal is found their PDBs may perform no better than a blind heuristic, (3) their heuristic is completely uninformed in unexplored parts of the abstraction. We advance the numeric PDB framework to address these limitations. We tackle (1) and (2) with a targeted exploration of the abstraction via A^* search with an alternative heuristic. We combine PDBs with admissible heuristics for abstract states in the fringe of the explored state space as well as unseen states to address (3). Moreover, we adopt a technique known from cartesian abstractions to refine the heuristic during search (Eifler and Fickert 2018) where the estimates are missing due to the partial exploration. Our enhancements significantly improve the heuristic quality compared to the numeric PDBs developed by Gnad et al. (2025), leading to a substantial performance increment on common numeric planning benchmarks.

Preliminaries

We build on a common numeric planning formalism, termed *restricted numeric tasks* (RT) (Hoffmann 2003), that extends the finite-domain representation (FDR) (Bäckström and Nebel 1995; Helmert 2009) to include numeric fluents.

The plan existence problem for RT is undecidable (Helmert 2002), even for highly restricted cases (Gnad et al. 2023).

An RT is given as a tuple $\Pi = \langle \mathcal{V}, \mathcal{A}, s_0, G \rangle$, where $\mathcal{V} = \mathcal{V}_n \cup \mathcal{V}_p$ denotes the set of *numeric* and *finite-domain variables*, $\mathcal{A}$ represents the finite set of *actions*, s_0 is the *initial state*, and G describes the *goal* conditions. Each variable $v \in \mathcal{V}_p$ has a finite *domain* $\mathcal{D}(v)$ if $v \in \mathcal{V}_p$, and $\mathcal{D}(v) = \mathbb{Q}$ otherwise, where $\mathbb{Q}$ are rationals. A *state* $s = \langle s_p, s_n \rangle$ is a complete assignment to the variables in $\mathcal{V}$, where $s_p \in \times_{v \in \mathcal{V}_p} \mathcal{D}(v)$ and $s_n \in \times_{v \in \mathcal{V}_n} \mathbb{Q}$. *Conditions* in RT can be either propositional or numeric. Propositional conditions of the form $\langle v, d \rangle$ are called *facts*, where $v \in \mathcal{V}_p$ and $d \in \mathcal{D}(v)$. Numeric conditions are (in)equalities of the form $v \bowtie w$, where $\bowtie \in \{>, \geq, =, \leq, <\}$, $v \in \mathcal{V}_n$, and $w \in \mathbb{Q}$. A state s satisfies a condition ψ if the corresponding constraint is fulfilled, denoted by $s \models \psi$.

An *action* $a \in \mathcal{A}$ is specified by $\langle \mathsf{pre}(a), \mathsf{eff}(a) \rangle$, where $\mathsf{pre}(a)$ and $\mathsf{eff}(a)$ are the *preconditions*, a set of conditions, and *effects*, a set of effects, respectively. Effects can be propositional, i.e., facts $\langle v, d \rangle$, where $v \in \mathcal{V}_p$ and $d \in \mathcal{D}(v)$, or numeric. Numeric effects in RT are expressed as $(v \mathrel{+}= m)$, with $v \in \mathcal{V}_n$ and $m \in \mathbb{Q} \setminus \{0\}$. Such effects are often called *simple*. An action a is *applicable* in a state s if $s \models \mathsf{pre}(a)$, and its application results in a new state denoted $s[\![a]\!]$. Each action modifies each variable at most once. Actions have non-negative costs $\mathsf{cost}(a) \in \mathbb{R}^{0+}$.

The *goal* G consists of propositional and numeric conditions. A state s_* is a goal state if it satisfies G, i.e., $s_* \models G$. An *s-plan* is a sequence of actions leading from the state s to a goal state, with the plan cost being the sum of action costs. An *optimal* s-plan minimizes this cost, denoted by $h^*(s)$. If no goal state is reachable from s, then $h^*(s) = \infty$. A (optimal) plan for Π is an (optimal) s_0-plan.

The *state space* of Π is a labeled transition system (LTS) which is defined as $\mathcal{T} = \langle S, L, \mathsf{cost}, T, s_0, S_* \rangle$, where S is the set of states of Π, L are the action labels $\mathcal{A}$, cost is the action cost function, and T is the set of transitions $(s, s', a) \in S \times S \times L$ with a being applied in s leading to s'. The set S_* consists of the goal states. The state space of an RT is generally infinite due to numeric variables. The path cost between states s and s' is $\mathsf{cost}(s, s')$, with the minimum path cost denoted as $\mathsf{cost}^*(s, s')$, and $\mathsf{cost}^*(s, s) = 0$.

SNP to RT Following Hoffmann (2003), Scala, Haslum, and Thiébaux (2016) introduced simple numeric planning (SNP), where numeric effects are simple and conditions are linear. Hoffmann (2003) showed that SNP can be compiled to RT in polynomial time w.r.t. active numeric conditions, replacing each formula with a numeric variable.

PDB Heuristics for Optimal Classical Planning

Pattern database (PDB) heuristics h^P project the task Π on a subset of its variables, called the *pattern* $P \subseteq \mathcal{V}$, ignoring variables $\mathcal{V} \setminus P$. This projection $\Pi\big|_P$ induces an *abstraction*, which maps each state in the (*concrete*) state space of Π to an abstract state in the *abstract state space* by restricting all expressions to the variables in P. The set of abstract states in a projection is given by $S\big|_P := \{s\big|_P \mid s \in S\}$. The PDB heuristic h^P is defined as the perfect heuristic in the projection $\Pi\big|_P$. PDB heuristics are usually precomputed once by determining the optimal costs, $h^*\big|_P(s\big|_P)$, for all abstract states $s\big|_P \in S\big|_P$. Thus, for each state $s \in S$ we define the PDB heuristic as $h^P := h^*\big|_P(s\big|_P)$.

A heuristic $h : S \to \mathbb{R}^{0+} \cup \{\infty\}$ is *admissible* if $h(s) \leq h^*(s)$ for all states $s \in S$. PDB heuristics are admissible and can be used for optimal planning within an A^* search (Hart, Nilsson, and Raphael 1968) to estimate the cost to goal.

PDBs for Numeric Variables

In classical planning, PDBs are typically computed by generating the entire transition system for a given projection. However, since numeric variables have infinite domains, projecting the task onto a pattern that includes at least one numeric variable generally results in an infinite abstraction.

To address this challenge, Gnad et al. (2025) proposed an approach to handle potentially infinite transition systems, essentially expanding only a finite fragment of the abstraction.

Bounding Infinite Transition Systems Let $\mathcal{T}$ be a transition system with initial state s_0. We are interested only in the states that can be reached from s_0. Consider a sub-transition system $\bar{\mathcal{T}}$ over the states $\bar{S} := S_E \cup S_F$, s.t.

1. $\bar{T} \subseteq T$ and $\bar{S}_* \subseteq S_*$;
2. states in $\bar{S}$ are reachable from s_0;
3. if $s \in S_E$ and $(s, s', l) \in T$ then $(s, s', l) \in \bar{\mathcal{T}}$, i.e., all successors of states in S_E are in $\bar{\mathcal{T}}$,
4. if s is in S_F than either none of its successors are in $\bar{\mathcal{T}}$, i.e., s has no outgoing edges in $\bar{\mathcal{T}}$, or $s \in S_*$.

Gnad et al. (2025) propose uniform cost search (UCS) to gradually traverse the transition system by expanding nodes according to their distance from the abstract initial state s_0. The expanded fragment of $\bar{\mathcal{T}}$ corresponds to the graph explored by UCS. Here, S_E denotes the set of expanded nodes, and S_F denotes the fringe nodes. Moreover, $S_E \cap S_F \subseteq \bar{S}_*$.

The heuristic is then defined for the states in $S_E \cup S_F$:

$$\tilde{d}(s) := \min_{s' \in S_F} \{\mathsf{cost}^*(s, s') + d(s')\}, \text{ with}$$

$d(s') = 0$ if $s' \in S_*$ or the minimum action cost otherwise.

Homomorphisms of Infinite LTS We use the standard definition of LTS homomorphisms (or abstractions) to argue that the distance-to-goal of the image yields an admissible heuristic (Libkin 2004; Gnad et al. 2025).

Definition 1. *Let* $\mathcal{T} = \langle S, L, \mathsf{cost}, T, s_0, S_* \rangle$ *and* $\mathcal{T}' = \langle S', L', \mathsf{cost}', T', s_0', S_*' \rangle$. *A map* $\alpha : S \cup L \to S' \cup L'$ *is an* ***LTS homomorphism*** *if:*

1. $\alpha(S) \subseteq S'$, $\alpha(L) \subseteq L'$,
2. $\alpha(T) \subseteq T'$, *where* $\alpha((s, s', l)) = (\alpha(s), \alpha(s'), \alpha(l))$,
3. $\mathsf{cost}'(\alpha(l)) \leq \mathsf{cost}(l)$ *for all* $l \in L$,
4. $\alpha(s_0) = s_0'$, $\alpha(S_*) \subseteq S_*'$.

The map α is extended to sequences and sets elementwise, and we write $\alpha(\mathcal{T})$ for the image of $\mathcal{T}$. Since existential-positive formulas are homomorphism-invariant, plans are preserved, with non-increasing costs:

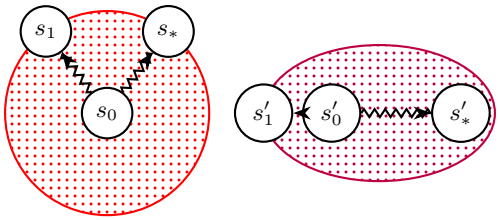

Figure 1: Illustration of the heuristic values for two initial states s_0 and s'_0 in uninformed (left) and informed (right) partial expansion of the projection.

Theorem 1. *For any homomorphism α of an LTS $\mathcal{T}$ and a state s of S we have that $h^*(\alpha(s)) \leq h^*(s)$, i.e., the map $\alpha \circ h^* : S \to \mathbb{R}^{0+} \cup \{\infty\}$ is an admissible heuristic for $\mathcal{T}$.*

A projection on a pattern forms an LTS homomorphism. We remark that $\alpha(s)$ and $s|_P$ are not the same, since all projections are abstractions, but not vice versa.

Targeted Projection Exploration using A*

Gnad et al. (2025) proposed partial exploration of abstract transition systems induced by selected patterns. We analyze weaknesses of this approach and suggest improvements.

Why Exploring the Right States Matters

For a state s a PDB heuristic h^P approximates the goal distance by its exact abstract goal distance $h^*(s|_P)$ in the projected task $\Pi|_P$. But this cannot be computed on a partially explored abstraction, in particular if no abstract goal state was reached. Figure 1 demonstrates that having explored an abstract goal state in the PDB does not guarantee that the actual goal distance is reported for most of the explored states. Let s_1 and s'_1 be states on the fringe S_F of the explored regions, and assume that the Euclidean distance in the figure corresponds to the actual distance between states in the abstraction. On the left, we have $\tilde{d}(s_0) = \mathsf{cost}^*(s_0, s_*) < \mathsf{cost}^*(s_0, s_1) + d(s_1)$. On the other hand, $\tilde{d}(s'_0)$ reports $\mathsf{cost}^*(s'_0, s'_1) + d(s'_1)$, since s'_1 may lie on a path with cost lower than the cost to the explored goal. This intuition suggests that we always want some "padding" around each state to increase its heuristic value. It also indicates why exploring the abstraction using a greedy search or random walks yields poor results in this setting – most states in the explored area lie on or very close to the fringe. We suggest an A* algorithm guided by an admissible heuristic to improve this process, aiming to increase the "padding" for the explored states and reporting the actual goal distances rather than the distance to the fringe.

Abstraction Exploration using A*

Progressing the abstract state space using UCS does not guarantee that abstract goal states are included in the PDB. The informative value of such a PDB is limited, as the heuristic values reported are merely distances to the fringe. Consider for illustration the extreme case where the abstract state space and the concrete state space are identical and the cost function is uniform. Without reached abstract goal such a PDB heuristic is entirely uninformed (see Proposition 2).

To counteract this, we propose adapted progression and regression algorithms of the PDB generation to increase the chance of covering abstract goals. Using an admissible heuristic h and a slightly altered A^* algorithm, we search for the closest goal state in the abstract state space. Contrary to regular A^* we do not stop the search once the first goal state is retrieved from the open list. Instead, we continue the exploration until the number of generated unique states reaches a predetermined bound B. The choice of B is determined empirically. Preliminary experiments with B values from 5.000 to 100.000 showed that our approach is robust with noticeable difference only in the sailing domain. In the regression step, we initialize every non-goal, non-closed state s in the open list (these are the fringe states – S_F), with its heuristic value $h(s)$.

Estimating Goal Distances

Consider an LTS homomorphism $\alpha : \mathcal{T} \to \mathcal{T}'$ between two (possibly infinite) transition systems. If h is an admissible heuristic for $\mathcal{T}'$, then $\alpha \circ h$ is also admissible for $\mathcal{T}$. We propose to use admissible goal estimates for a finite sub-transition system of $\mathcal{T}'$, denoted $\bar{\mathcal{T}}$. For the purpose of constructing a PDB heuristic, we may assume that $\mathcal{T}'$ originates from a pattern defined over numeric variables, although this assumption is not required for the discussion that follows.

Let $\bar{\mathcal{T}} = \langle \bar{S}, L, \mathsf{cost}, \bar{T}, s'_0, \bar{S}_* \rangle$ be a finite sub-transition system of $\mathcal{T}'$, and let $\bar{S} := S_E \cup S_F$ be the states that correspond to the nodes expanded by a search within the Best First Search paradigm that does not stop after encountering goal states, where S_E corresponds to the expanded nodes and S_F to the open nodes in the fringe and goal states.

For a heuristic h on $\bar{S}$ we define the following refinement:

$$\tilde{h}(s) := \begin{cases} \min_{s' \in S_F} \{\mathsf{cost}^*(s, s') + h(s')\} & \text{if } s \in S_E \\ h(s) & \text{otherwise.} \end{cases}$$

Note that $\tilde{h}$ is defined only on the set $\bar{S}$.

Proposition 1. *If h is admissible so is $\tilde{h}(s)$, moreover if h is consistent, then so is $\tilde{h}$. $\tilde{h}$ dominates h.*

Proof. Note that for each $s \notin S_E$ we have $\tilde{h}(s) = h(s)$. It remains to show admissibility for $s \in S_E$.

Admissibility: Note that for a goal state $s_* \in S_F$ we have $\tilde{h}(s_*) = 0$. Let π^*_s be an optimal s-plan. Since, by definition of $\bar{\mathcal{T}}$, for each $s \in S_E$, along each s-plan π there is at least one state s' such that $s' \in S_F$. Take the first such state. Let s^*_F be such a state for π^*_s. Then,

$$\begin{aligned} h^*(s) &= \mathsf{cost}^*(s, s^*_F) + h^*(s^*_F) \\ &\geq \min_{s' \in S_F} \{\mathsf{cost}^*(s, s') + h(s')\} = \tilde{h}(s). \end{aligned}$$

Consistency: Since h is consistent, we have $h(s) \leq \mathsf{cost}^*(s, s') + h(s')$ for every s' reachable from s. Let $s, s' \in \bar{S}$ such that $(s, s', l) \in T$. We aim to prove that $\tilde{h}(s) \leq \mathsf{cost}(l) + \tilde{h}(s')$. This inequality holds trivially for each $s \in \bar{S}_*$ by admissibility of $\tilde{h}$, since $\tilde{h}(s) = 0$ for all goal

states. Therefore, the condition also holds for all $s \in S_F$, as each such state there is either a goal or has no successors.

Assume then that $s \in S_E \setminus S_F$, the successor of s denoted s' can be either in S_F or in S_E. If s' is in S_F we have that:

$$\begin{aligned}\tilde{h}(s) &= \min_{s'' \in S_F} \{\mathsf{cost}^*(s, s'') + h(s'')\} \\ &\leq \mathsf{cost}^*(s, s') + h(s') \leq \mathsf{cost}(l) + \tilde{h}(s'),\end{aligned}$$

since $h(s') = \tilde{h}(s')$. Now, assume that $s' \in S_E$. Since h is consistent we have:

$$\begin{aligned}\tilde{h}(s) &= \min_{s'' \in S_F} \{\mathsf{cost}^*(s, s'') + h(s'')\} \\ &\leq \mathsf{cost}^*(s, s') + \min_{s'' \in S_F} \{\mathsf{cost}^*(s', s'') + h(s'')\} \\ &\leq \mathsf{cost}^*(s, s') + \tilde{h}(s') \leq \mathsf{cost}(l) + \tilde{h}(s'). \qquad \square\end{aligned}$$

Is Reaching an Abstract Goal Important?

We demonstrate that PDBs can provide informative guidance even when no goal states are reached in the abstraction (Example 1). This challenges the naive assumption that goal-unreached abstractions are necessarily uninformed. Proposition 2, in contrast, serves as a kind of "no free lunch" observation: it shows that within a fully explored region that contains no goal states, the PDB refined using the blind heuristic offers no advantage over the blind heuristic itself in terms of node expansions. We employ the commonly used blind heuristic, where $d(s) = 0$ if s is a goal state, and the minimal cost of the actions applicable in s otherwise.

Example 1. *Consider the problem with $\mathcal{V} = \mathcal{V}_n = \{x, y, z\}$ and three unit-cost actions with no preconditions and a single effect each. The effects have the form $x \mathrel{+}= 1$, $y \mathrel{+}= 1$, and $z \mathrel{+}= 1$. The initial state is given by $\forall v : s_0[v] = 0$, for simplicity we will write it as $(0, 0, 0)$. The goal condition is $G = \{x = 4, y = 1, z = 1\}$, or $(4, 1, 1)$ in short.*

Consider the pattern $P = \{x\}$. Since we assume no goal state is reached in the projection of our TS to P, say only the abstract states $x \in \{0, 1, 2\}$ are explored, where $S_E = \{x = 0, x = 1\}$ and $S_F = \{x = 2\}$. Since $x = 2$ is not the goal, we get $d(x = 2) = 1$. Thus, the PDB heuristic is

$$\tilde{h}(x = 2) = 1, \tilde{h}(x = 1) = 2, \tilde{h}(x = 0) = 3,$$

and $\tilde{h} = d$ otherwise.

Let g be the distance of the goal from the initial state. Let us look at A^* with the following priority functions $f_1 = g + \tilde{h}$ and $f_2 = g + d$. f_1 corresponds to the PDB heuristic, and f_2 to the "improved" blind heuristic.

For the single goal state we have $f_1(4, 1, 1) = f_2(4, 1, 1) = g(4, 1, 1) = 6$. So, A^* will expand all states with f-value smaller than 6 and will not expand states with f-value greater than 6. Let us look at the state $(0, 2, 2)$:

$$f_1(0, 2, 2) = 4 + 3 = 7 \text{ and } f_2(0, 2, 2) = 4 + 1 = 5.$$

Thus, $(0, 2, 2)$ will be expanded by blind search, but not by A^* with the PDB heuristic.

The example illustrates that even if the PDB heuristic does not reach a goal state, it can still provide more informative guidance than a blind heuristic. This is particularly relevant when using partial pattern projections that cover only a limited number of variables. To better understand the practical implications of such heuristics in early layers of the search, we now analyze how the refined heuristic $\tilde{h}$ compares to the improved blind heuristic d within a bounded search depth. Specifically, we consider the structure of states within a fixed cost-radius around the initial state. We show that, under assumptions that no abstract goal states are reached and the expanded PDB region preserves the original distances, A* guided by $\tilde{h}$ expands the same set of nodes up to the f-layer $r + 1$ as A* guided by the blind heuristic d.

Proposition 2. *Consider a unit-cost* RT *task Π with induced TS $\mathcal{T}$, and denote by S the states of Π. Let $g(s)$ be the cost of getting from s_0 to a state $s \in S$, and assume there is an $r > 0$ such that for each goal state s_* it holds $g(s_*) > r+1$.*

Let $\alpha : \mathcal{T} \to \mathcal{T}'$ be an LTS homomorphism, and let $\bar{\mathcal{T}}$ be a bounded sub-TS of $\mathcal{T}'$, such that $\bar{S} := S_E \cup S_F$ form a ball of radius $r > 0$ around $\alpha(s_0)$, i.e.,

$$\bar{S} := \{\alpha(s) \mid s \in S, \mathsf{cost}^*(\alpha(s_0), \alpha(s)) \leq r\}.$$

Assume that for each state $s \in S$ such that $g(s) \leq r$ it holds that $g(s) = \mathsf{cost}^*(\alpha(s_0), \alpha(s))$. Then, up to the f-layer $r+1$, A^* with $\alpha \circ \tilde{d}$ expands the same nodes as A^* with the "improved" blind heuristic d.[1]

Proof. We aim to show that for each s with $g(s) \leq r$ it holds

$$\begin{aligned}f_1(s) := g(s) + \tilde{d}(\alpha(s)) \leq r + 1 \iff \\ f_2(s) := g(s) + d(s) \leq r + 1.\end{aligned}$$

Since, there are no goal states within distance $r + 1$ of s_0 and we assume a unit-cost, we can write $d(s) = 1$ for each s with $g(s) \leq r + 1$. Thus, $g(s) \leq r$ iff $f_2(s) \leq r + 1$.

On the other hand,

$$\begin{aligned}f_1(s) &:= g(s) + \tilde{d}(\alpha(s)) \\ &= g(s) + r + d(s) - \mathsf{cost}^*(\alpha(s_0), \alpha(s)) \\ &= g(s) + r + 1 - g(s) = r + 1.\end{aligned}$$

Thus, $g(s) \leq r$ iff $f_1(s) = r + 1$. Hence, the result follows from the layer-wise expansion of A^*. $\square$

The key insight here is that $\tilde{d}$ uses the lowest cost to a fringe state in the projection, plus the heuristic estimate at the fringe, to improve estimates within the explored region. The heuristic remains admissible and preserves consistency. The example demonstrates that $\alpha \circ \tilde{d}$ can avoid expanding certain sub-optimal states that a blind heuristic would explore. Furthermore, under restrictive assumptions (i.e., unit-cost, no goal within a bounded depth), our proposition shows that A^* guided by $\alpha \circ \tilde{d}$ expands exactly the same states up to a certain cost-layer as when guided by the improved blind heuristic d. This suggests that early-stage search behavior with refined PDBs mimics blind search, but gains potential advantages beyond the initial layers.

[1] The expansions happen not necessarily in the same order.

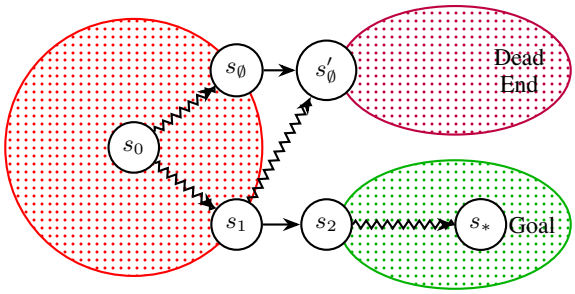

Figure 2: Illustration of a setting where detected dead ends are not reported by the PDB heuristic due to the partial expansion of the projection.

Handling Unexplored States

Using a partially explored projection as a PDB heuristic does not guarantee that for every concrete state s generated during the search we have reached the abstract counterpart $s|_P$, i.e., the lookup of the heuristic value for s can *fail*. If that happens, a simple backup strategy is to resort to the blind heuristic (e.g., the minimum action cost for non-goal states). We suggest two alternatives that are strictly more informative: (1) an admissible heuristic computed for $s|_P$, which we call the *failed-lookup heuristic*, and (2) an abstraction refinement that *extends* the projection by invoking another exploration from $s|_P$ and augmenting the goal distances.

Employing a failed-lookup heuristic is particularly important when dead ends are present and the abstract state space was constructed using a heuristic capable of detecting dead ends. In such settings, detected dead ends are never expanded by the adapted A* algorithm. This implies that only a small fraction of the states will be explored in abstract spaces containing dead ends. Without a failed-lookup heuristic, $\tilde{h}(s)$ reports the minimum action cost if $s\big|_P$ corresponds to an unexplored state. This results in misleadingly optimistic heuristic values and in the search wasting effort in regions filled with dead ends that were actually identified as such. See Figure 2 for an illustration. While dead ends can be detected on the fringe of the sub-transition system, this does not necessarily lead to pruning of the entire dead end region. In the example, we have a state $s_\emptyset$ that is reached in the projection and identified as a dead end by the heuristic h. Although we can prune $s_\emptyset$ and avoid its expansion, it is not the only entry point to the dead end region. In particular, state s_1 is not a dead end, but it lies on the fringe and has paths both to a goal state and into the dead end region. Thus, we cannot prune s_1, and, unless we use a failed-lookup heuristic capable of detecting dead ends for states outside the PDB, the search may still reach and explore the dead end region.

When extending the projection, $s|_P$ is treated as a temporary abstract initial state, and the abstract state space is explored as if it was from the original initial state, updating heuristic values for all visited states up to a predefined limit on the number of abstract states. Previously explored abstract states retain their existing heuristic values. Note that this form of failed-lookup handling can introduce inconsistencies into the heuristic, as explained in detail below.

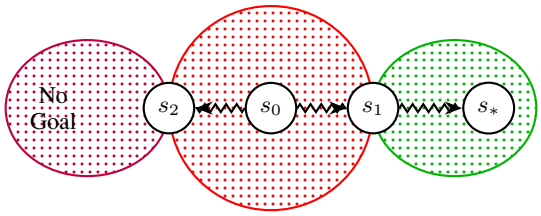

Figure 3: Illustration to indicate the importance of a backup heuristic h_{Fl} for failed PDB lookups.

Putting the Pieces Together

Our proposed framework integrates three distinct admissible heuristics within a PDB: the **exploration heuristic** h_{Ex}, which guides the abstract state-space exploration toward goal states; the **fringe heuristic** h_{Fr}, which refines distance estimates for fringe states S_F beyond the minimal action cost; and the **failed-lookup heuristic** h_{Fl}, serving as a fallback for states s whose abstract counterparts $s|_P$ were not reached ($s|_P \notin S_E \cup S_F$). We denote heuristic instantiations by *XYZ*, where *X*, *Y*, and *Z* correspond respectively to h_{Ex}, h_{Fr}, and h_{Fl}. Each position takes one of the following heuristics: *B* for the blind heuristic, *L* for the LM-cut heuristic (Kuroiwa et al. 2022), and, by slight abuse of notation, *E* for the state-space extension used exclusively as h_{Fl}. The baseline by Gnad et al. (2025) is represented by *BBB*.

We next analyze our framework formally, leading to the conclusion that only certain instantiations are meaningful. For this purpose, we introduce a wildcard character *X* in an instantiation name (e.g., *BXB*) to signify the use of an arbitrary admissible heuristic other than the blind heuristic. For instance, *BXB* denotes an instantiation where the blind heuristic is used for both h_{Ex} and h_{Fl}, while an unspecified heuristic is used for h_{Fr}.

For the A* search, the f-value of state s is given as:

$$f_{\text{XYZ}}(s) = g(s) + \begin{cases} \tilde{Y}(\alpha(s)) & \text{if } \alpha(s) \in S_E \cup S_F, \\ Z(\alpha(s)) & \text{otherwise.} \end{cases}$$

where $\tilde{Y}$ is the PDB heuristic using Y on the fringe.

Instantiations *XBB* and *BXB* will generally result in limited overall performance compared to the baseline *BBB*.

Proposition 3. *Let BBB and BXB be two instantiations for the same pattern P, where B is the blind heuristic and X is an admissible heuristic that dominates B. For any two concrete states s and s', where $s\big|_P \in S_E \cup S_F$ (the explored or fringe sets) and $\alpha(s')$ is not in either of these sets, it holds:*

$$f_{BBB}(s) > f_{BBB}(s') \implies f_{BXB}(s) > f_{BXB}(s').$$

Yet, the converse implication, $f_{BXB}(s) > f_{BXB}(s') \implies f_{BBB}(s) > f_{BBB}(s')$, does not generally hold.

Proof. The construction of the pattern database heuristic, i.e., the expansion of abstract states, is determined by the h_{Ex} heuristic. Since both *BBB* and *BXB* use the blind heuristic for h_{Ex}, they expand the exact same abstract states.

Consider the states s and s' as specified in the claim. For state $\alpha(s') \notin S_E \cup S_F$, the f-value is determined by the h_{Fl}

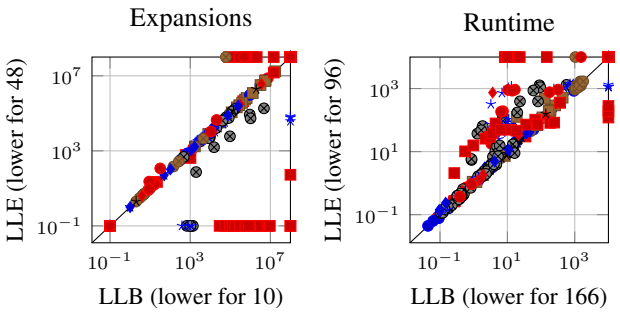

Figure 4: Per-instance comparison of PDB construction with the LM-cut heuristic for exploration and fringe, without (x-axis) and with (y-axis) refinement upon lookup failures. We compare search-space size (left) and runtime (right).

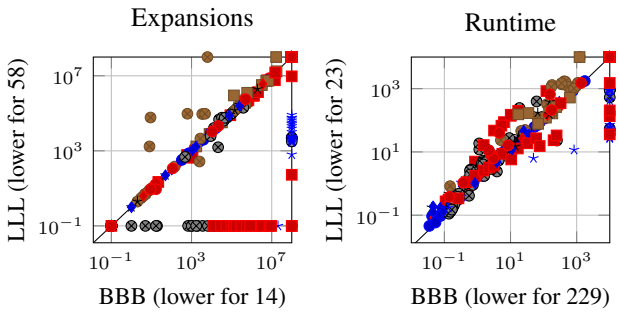

Figure 5: Per-instance comparison of PDB construction between BBB (x-axis) and LLL (y-axis). We compare search-space size (left) and runtime (right).

heuristic. In both *BBB* and *BXB* instantiations, the h_{Fl} heuristic is the blind heuristic. So we have $f_{\text{BBB}}(s') = f_{\text{BXB}}(s')$.

For state $\alpha(s') \in S_E \cup S_F$ the f-value is determined by the expanded states and the h_{Fr} heuristic. Since X represents an admissible heuristic which is generally more informed (i.e., provides greater or equal heuristic values) than the blind heuristic, it follows that $f_{\text{BBB}}(s) \leq f_{\text{BXB}}(s)$.

Combining these observations: Given $f_{\text{BBB}}(s) > f_{\text{BBB}}(s')$ (Premise 1). We know $f_{\text{BBB}}(s') = f_{\text{BXB}}(s')$ (from s''s definition and h_{Fl}). Substituting this into Premise 1 gives $f_{\text{BBB}}(s) > f_{\text{BXB}}(s')$. We also know $f_{\text{BXB}}(s) \geq f_{\text{BBB}}(s)$. Therefore, $f_{\text{BXB}}(s) \geq f_{\text{BBB}}(s) > f_{\text{BXB}}(s')$, which implies $f_{\text{BXB}}(s) > f_{\text{BXB}}(s')$. This proves the first implication.

For a counter argument, consider unit action cost and state s with $\alpha(s) \in S_F$. Assume state s' with $\alpha(s') \notin S_E \cup S_F$ that is not a goal state, hence $B(\alpha(s')) = 1$. Assume $f_{BBB}(s) = g(s) + B(\alpha(s)) > f_{BBB}(s') = g(s') + B(\alpha(s'))$. For the same states, we have $f_{BXB}(s) = g(s) + B(\alpha(s))$ and $f_{BXB}(s') = g(s') + X(\alpha(s'))$. Since $X(\alpha(s')) \geq B(\alpha(s'))$, the converse does not necessarily follow. □

This shows that, for *BXB* instantiations, states whose abstract projections are already in $S_E \cup S_F$ tend to have higher h-values than those that are not. This occurs because h_{Fl} is identical in both *BXB* and *BBB*, while h_{Fr} in *BXB* is more informed. As a result, A* in the concrete space tends to prefer states with failed lookups (i.e., those not in $S_E \cup S_F$), lacking any search guidance. This illustrates a general weakness of using a blind h_{Fl}, as illustrated in Figure 3. Here, consider a *BXB* instantiation where the PDB uniformly explores a ball around $s_0|_P$ of some radius r, and then computes the heuristic X on the fringe of this ball. For all failed lookups, the PDB heuristic reports the "blind" heuristic value. Consider two fringe states s_1 and s_2 such that $X(s_1) > X(s_2)$. Suppose there are no goal states in the vicinity of s_2, but the heuristic X is mistaken and reports a larger value for s_1 than for s_2. In the concrete state space, we have $f(s_1) = g(s_1) + X(s_1) > f(s_2) = g(s_2) + X(s_2)$. Since all failed lookups fall back to the blind heuristic, it follows that all descendants s' of s_2 with $g(s') < X(s_1) - X(s_2)$ will be explored before A* proceeds to s_1.

XBB instantiations introduce another issue. When an informed heuristic is used only for h_{Ex}, abstract dead ends may not be sufficiently explored or marked. If a concrete dead end state is projected to such an abstract state during search, the blind h_{Fl} may return a low value, failing to report the dead end and causing unnecessary exploration. This can be mitigated by using a more informed h_{Fl}.

Finally, instantiations such as *BBE* can lead to inconsistencies. Suppose a state s is a successor of a non-goal state in S_F, where h_{Fr} assigns minimal cost due to being blind. If the true goal distance from s exceeds this value, $h(s)$ will only correctly reflect this if all of its successors are fully expanded in the abstract space.

Experimental Evaluation

Our implementation is based on Numeric Fast Downward (Aldinger and Nebel 2017), extending the planner by Gnad et al. (2025). We conducted our experiments on a cluster of Intel Xeon Gold 6130 CPUs using Downward Lab 8.2 (Seipp et al. 2017), adopting the runtime and memory limits of 30 min and 8GiB from the recent International Planning Competition (IPC) 2023 (Taitler et al. 2024). We use the benchmark set as proposed by Gnad et al. (2025), consisting of the IPC'23 and benchmarks from the literature (Scala, Haslum, and Thiébaux 2016; Scala et al. 2017, 2020; Shleyfman, Kuroiwa, and Beck 2023; Benyamin et al. 2024). Code, experimental data and an extended version of the paper are publicly available (Fritzsche et al. 2025).

Experimental Setup As general baselines we compare our approach to blind search (**B**) and A* with the LM-cut heuristic (**L**) (Kuroiwa et al. 2022) and Hrmax (**H**) (Scala et al. 2020). We include the plain numeric PDBs without our enhancements for comparison (**BBB**) (Gnad et al. 2025).

For all PDB-based approaches, we build on the pattern collection generation called iPDB, which uses a hill-climbing approach and combines multiple PDBs using the canonical heuristic (Haslum et al. 2007). We limit the size of candidate patterns by an approximated domain-size product of 100.000, adopting the mechanism suggested by Gnad et al. (2025) to approximate this range using the *explicit nu-*

	Domain	#	B	H	L	BBB	BBE	LLE	LLB	BBL	BLL	LLL
IPC 2023	delivery	20	2	2	**3**	2	2	2	2	2	2	2
	drone	20	3	3	3	**4**	**4**	**4**	**4**	**4**	**4**	**4**
	expedition	20	5	**6**	**6**	**6**	**6**	**6**	**6**	**6**	**6**	**6**
	farmland-ipc23	15	4	**15**	**15**	8	8	**15**	**15**	**15**	**15**	**15**
	hydropower	20	9	**11**	**11**	9	8	1	9	8	10	10
	mprime	20	6	9	**15**	12	12	12	12	12	12	12
	rover-ipc23	20	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
	sailing-ipc23	20	0	5	**8**	0	0	0	0	0	0	0
	sugar	20	2	2	**12**	3	3	3	3	3	3	3
	zenotravel-ipc23	20	6	6	**8**	6	6	6	6	6	6	6
from literature	counters	20	3	4	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**
	counters-sym	11	2	3	**11**	8	8	8	8	9	8	9
	depots	20	4	5	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**
	depots-sym	20	4	4	**7**	6	6	6	6	6	6	6
	farmland	30	12	**30**	**30**	12	12	**30**	26	**30**	**30**	**30**
	fn-counters-small	8	6	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**
	forestfire	20	10	10	**11**	10	10	10	10	10	10	10
	minecraft-pogo	20	14	0	5	**18**	**18**	17	**18**	**18**	17	17
	minecraft-sword	20	**20**	0	9	**20**	**20**	**20**	**20**	**20**	**20**	**20**
	petri-net	20	2	6	8	**9**	**9**	8	8	**9**	**9**	8
	plant-watering	63	**63**	**63**	**63**	**63**	**63**	**63**	**63**	**63**	**63**	**63**
	rover-unit	20	4	4	**7**	6	6	6	6	6	6	6
	sailing	40	10	28	**40**	15	15	18	15	17	17	18
	satellite	20	1	1	**2**	**2**	1	1	1	**2**	**2**	**2**
	zenotravel	23	6	7	**13**	10	10	9	10	10	10	10
	others	72	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	$\sum$	622	203	236	**311**	253	251	269	272	280	280	281

Table 1: Coverage (# solved instances) of the baseline planners (columns "B"–"BBB") and instantiations of our framework.

meric interval (ENI) (Shleyfman, Gnad, and Jonsson 2023). For every pattern, we generate 10.000 unique states when exploring the abstract state space. We limit our pattern collection size by the overall of 1 million unique abstract states. The limit of the hill-climbing procedure for iPDB is 900 sec. Gnad et al. (2025) showed that iPDB is mostly superior to systematic PDBs (Pommerening, Röger, and Helmert 2013). In preliminary experiments, we adapted the genetic algorithm by Franco et al. (2017), but with limited success.

For the PDB heuristics, we transform the SNP encoding generated with the translator component of numeric Fast Downward to the RT formalism required for the heuristics (we refer to Gnad et al. 2025, for the details).

In our new framework, we experiment with different combinations, all of which are based on the LM-cut heuristic or the extension of the projection.

Results Discussion

Table 1 shows the number of solved instances per domain. Domains with equal performance across all configurations are grouped under "others". Baselines appear on the left, our configurations on the right. All variants except *BBE* outperform the *BBB* baseline, confirming the effectiveness of our enhancements. For *BBE*, it turns out that the overhead of further exploring the projection from abstract states where the lookup fails, in combination with doing so in a blind way, does not pay off. This is confirmed by Figure 4, the reduced search space does not yield runtime gains.

LLB, which uses LM-cut to guide the search in the abstraction, improves coverage by +19 over *BBB*, supporting our hypothesis that reaching abstract goals yields better heuristics. Adding the failed-lookup heuristic on top (*LLL*) adds 9 more instances. Figure 5 compares *LLL* with *BBB* on expansions (until the last f-layer) and runtime. *LLL* requires fewer expansions except in Minecraft, where LM-cut is ineffective due to a large branching factor. The runtime overhead in this domains is noticeable for easy instances, but many others are solvable only by *LLL*.

Conclusion

We generalize an established framework that adapts PDB heuristics to simple numeric planning and address the challenge of handling an infinite abstract state space by proposing several solutions. Our approach integrates heuristics in a novel way, incorporating techniques such as numeric LM-cut (Kuroiwa et al. 2022) *within the abstraction* to guide PDB exploration toward promising regions and improve goal-distance estimates.

Empirical results on common benchmarks show that our methods significantly improve over the existing variant of numeric PDBs. A major question for future work is how to adapt cost partitioning techniques from classical planning to admissibly combine multiple heuristics. This is a key component in classical planning, without which PDBs are not competitive with state-of-the-art planners.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. Alexander Shleyfman's work was partially supported by ISF grant 2443/23.

References

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

Anderson, K.; Holte, R.; and Schaeffer, J. 2007. Partial Pattern Databases. In *SARA*, 20–34.

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

Benyamin, Y.; Mordoch, A.; Shperberg, S.; Piotrowski, W.; and Stern, R. 2024. Crafting a Pogo Stick in Minecraft with Heuristic Search. In *SOCS*, 261–262.

Cardellini, M.; Giunchiglia, E.; and Maratea, M. 2024. Symbolic Numeric Planning with Patterns. In *AAAI*, 20070–20077.

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

Culberson, J. C.; and Schaeffer, J. 1996. Searching with Pattern Databases. In *Proceedings of the Eleventh Biennial Conference of the Canadian Society for Computational Studies of Intelligence (CSCSI-96)*, volume 1081 of *LNAI*, 402–416. Springer-Verlag.

Culberson, J. C.; and Schaeffer, J. 1998. Pattern Databases. *Computational Intelligence*, 14(3): 318–334.

Edelkamp, S. 2002. Symbolic Pattern Databases in Heuristic Search Planning. In *AIPS*, 274–283.

Eifler, R.; and Fickert, M. 2018. Online Refinement of Cartesian Abstraction Heuristics. In *SOCS*, 46–54. AAAI Press.

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

Felner, A.; Korf, R.; and Hanan, S. 2004. Additive Pattern Database Heuristics. *JAIR*, 22: 279–318.

Franco, S.; Torralba, Á.; Lelis, L. H.; and Barley, M. 2017. On creating complementary pattern databases. In *IJCAI*, 4302–4309.

Fritzsche, M.; Gnad, D.; Gruntov, M.; and Shleyfman, A. 2025. Code and experiment data of the AAAI 2026 paper "Managing Infinite Abstractions in Numeric Pattern Database Heuristics". https://doi.org/10.5281/zenodo.17592540.

Gerevini, A.; Saetti, A.; and Serina, I. 2008. An approach to efficient planning with numerical fluents and multi-criteria plan quality. *AIJ*, 172(8-9): 899–944.

Gnad, D.; Alon, L.-o.; Weiss, E.; and Shleyfman, A. 2025. PDBs Go Numeric: Pattern-Database Heuristics for Simple Numeric Planning. In *AAAI*.

Gnad, D.; Helmert, M.; Jonsson, P.; and Shleyfman, A. 2023. Planning over Integers: Compilations and Undecidability. In *ICAPS*, 148–152. AAAI Press.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

Haslum, P.; Bonet, B.; and Geffner, H. 2005. New Admissible Heuristics for Domain-Independent Planning. In *AAAI*, 1163–1168.

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

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

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

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

Holte, R.; Felner, A.; Newton, J.; Meshulam, R.; and Furcy, D. 2006. Maximizing over Multiple Pattern Databases Speeds up Heuristic Search. *AIJ*, 170(16–17): 1123–1136.

Holte, R.; Newton, J.; Felner, A.; Meshulam, R.; and Furcy, D. 2004. Multiple Pattern Databases. In *ICAPS*, 122–131.

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

Katz, M.; and Domshlak, C. 2009. Structural-Pattern Databases. In *ICAPS*, 186–193.

Kuroiwa, R.; Shleyfman, A.; and Beck, J. C. 2023. Extracting and Exploiting Bounds of Numeric Variables for Optimal Linear Numeric Planning. In *ECAI*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, 1332–1339.

Kuroiwa, R.; Shleyfman, A.; Piacentini, C.; Castro, M. P.; and Beck, J. C. 2022. The LM-Cut Heuristic Family for Optimal Numeric Planning with Simple Conditions. *J. Artif. Intell. Res.*, 75: 1477–1548.

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

Libkin, L. 2004. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer.

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

Pommerening, F.; Röger, G.; and Helmert, M. 2013. Getting the Most Out of Pattern Databases for Classical Planning. In *IJCAI*, 2357–2364. IJCAI/AAAI.

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

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

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2020. Subgoaling Techniques for Satisficing and Optimal Numeric Planning. *JAIR*, 68: 691–752.

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

Seipp, J.; Pommerening, F.; Sievers, S.; and Helmert, M. 2017. Downward Lab. https://doi.org/10.5281/zenodo.790461.

Shin, J.; and Davis, E. 2005. Processes and continuous change in a SAT-based planner. *AIJ*, 166(1-2): 194–253.

Shleyfman, A.; Gnad, D.; and Jonsson, P. 2023. Structurally Restricted Fragments of Numeric Planning – a Complexity Analysis. In *AAAI*, volume 37, 12112–12119.

Shleyfman, A.; Kuroiwa, R.; and Beck, J. C. 2023. Symmetry Detection and Breaking in Linear Cost-Optimal Numeric Planning. In Koenig, S.; Stern, R.; and Vallati, M., eds., *ICAPS*, 393–401. AAAI Press.

Taitler, A.; Alford, R.; Espasa, J.; Behnke, G.; Fiser, D.; Gimelfarb, M.; Pommerening, F.; Sanner, S.; Scala, E.; Schreiber, D.; Segovia-Aguas, J.; and Seipp, J. 2024. The 2023 International Planning Competition. *AI Mag.*, 45(2): 280–296.