Monotonic Variants of Saturated Cost Partitioning

Mauricio Salerno[1], Raquel Fuentetaja[1], Jendrik Seipp[2]

[1]Universidad Carlos III de Madrid, Leganés, Madrid, Spain
[2]Linköping University, Linköping, Sweden

Abstract

One of the strongest approaches for optimal classical planning is to compute multiple abstractions and combine them admissibly with cost partitioning. Ideally, a cost partitioning algorithm should guarantee that adding abstractions or refining existing ones can only improve heuristic estimates. However, due to its greedy nature, the state-of-the-art saturated cost partitioning (SCP) algorithm lacks both of these monotonicity properties. We introduce a simple ordering strategy for SCP to obtain monotonicity under adding abstractions and a general cost partitioning scheme to obtain monotonicity under refinements. Empirically, we show that our monotonic SCP variants outperform standard SCP for Cartesian abstractions and are competitive for pattern databases.

Introduction

A state-of-the-art approach to optimal classical planning uses A^* search (Hart, Nilsson, and Raphael 1968) with abstractions that yield informative admissible heuristics. Single abstractions usually cannot provide sufficient information (Seipp and Helmert 2018), so modern methods decompose the task into subtasks, compute one abstraction per subtask, and combine their estimates admissibly with cost partitioning (Katz and Domshlak 2008a,b; Pommerening, Röger, and Helmert 2013; Pommerening et al. 2015; Seipp, Keller, and Helmert 2020).

Saturated cost partitioning (SCP) is a state-of-the-art cost partitioning scheme (Seipp, Keller, and Helmert 2020). However, due to its greedy nature, SCP is not *monotonic under adding abstractions*: extending the set of abstractions considered by SCP can yield less informative heuristics.

A second notion of monotonicity is *monotonicity under refinements*, which arises when some abstractions are refinements or coarsenings of others. In this setting, replacing coarsenings by their refinements should not decrease heuristic estimates. Because abstractions preserve paths, a refinement is always at least as informative as any coarsening under the same cost function. One might therefore want to discard the coarsenings, since any cost assigned to them could instead be used by the refinement to obtain a dominating heuristic. Coarsenings and refinements arise naturally in many abstraction schemes: for pattern databases (PDBs) (Culberson and Schaeffer 1998; Edelkamp 2001), if a PDB is created using a set of variables V, and another PDB is created using a superset of variables $V' \supset V$, then the former PDB is a coarsening of the latter. Similarly, when merging two Cartesian abstractions (Salerno et al. 2025), the resulting merged abstraction is a refinement of the two original abstractions.

Unfortunately, SCP also fails this second notion of monotonicity. Sievers et al. (2020) show that the SCP heuristic obtained after merging two merge-and-shrink factors can be less informative than the one before merging. Likewise, Salerno et al. (2025) show that when merging Cartesian abstractions, both post-hoc optimization and saturated cost partitioning can yield less informative heuristics if the coarser abstractions are discarded, and, even without discarding them, saturated cost partitioning can yield less informative heuristics. This behavior is highly undesirable, as the merging operation is computationally expensive and should always be beneficial.

To address these issues, we make two main contributions. First, we propose a simple ordering for saturated cost partitioning that guarantees monotonicity under adding abstractions. Second, we introduce a general cost partitioning scheme tailored to the common setting where some abstractions are refinements of other abstractions. Under this scheme, the cost partitioning heuristic obtained over a set of refinements is guaranteed to be at least as informative as the one obtained using only the corresponding coarsenings, yielding monotonicity under refinements. Empirically, our schemes outperform standard saturated cost partitioning for Cartesian abstractions and are competitive for PDBs.

Background

For this paper, the only relevant aspect of planning tasks is that they induce weighted transition systems. A *transition system* is a tuple $\mathcal{T} = \langle S, \mathcal{L}, T, s_0, S_\star \rangle$ defining a directed graph with states S, labels $\mathcal{L}$, and transitions $T \subseteq S \times \mathcal{L} \times S$. The initial state is $s_0 \in S$, and $S_\star \subseteq S$ is the set of goal states. A *path* is a sequence of transitions. A *goal path* leads from s_0 to a state in $S_\star$. The sequence of labels in a goal path is a *plan*. A *weighted* transition system is defined by $\langle \mathcal{T}, cost \rangle$, where $cost : \mathcal{L} \to \mathbb{R}_0^+$. The cost of a plan is the sum of the costs of its labels, and a plan is optimal if there is no plan with lower cost. A plan for $\mathcal{T}$ is a plan for the corresponding planning task.

An *abstraction* of a transition system $\mathcal{T}$ is a surjective function $\alpha : S \to S^\alpha$ mapping *concrete* states into *abstract* states. The *abstract transition system* induced by α is $\mathcal{T}^\alpha = \langle S^\alpha, \mathcal{L}, T^\alpha, s_0^\alpha, S_\star^\alpha \rangle$, where $s_0^\alpha = \alpha(s_0)$, $S_\star^\alpha = \{\alpha(s) \mid s \in S_\star\}$, and $T^\alpha = \{\langle \alpha(s), \ell, \alpha(s') \rangle \mid \langle s, \ell, s' \rangle \in T\}$. Abstractions *preserve* paths in $\mathcal{T}$. Thus, for a cost function *cost*, the abstraction heuristic $h^\alpha(cost, s)$, defined as the cost of an optimal plan in $\mathcal{T}^\alpha$ from $\alpha(s)$ to a goal state, is admissible.

A *cost partitioning* over a sequence of heuristics $\mathcal{H} = \langle h_1, h_2, \ldots, h_n \rangle$ for a weighted transition system is a sequence of cost functions $C = \langle c_1, c_2, \ldots, c_n \rangle$ such that $c_i : \mathcal{L} \to \mathbb{R}$ and $\sum_{i=1}^n c_i(\ell) \leq cost(\ell)$ for all labels ℓ. The cost partitioning heuristic is defined as $h^C(s) = \sum_{i=1}^n h_i(c_i, s)$ and is admissible (Katz and Domshlak 2010).

Given an admissible heuristic h for a weighted transition system $\langle \mathcal{T}, cost \rangle$, a *saturated cost function* is a cost function *scf* that satisfies: (1) $scf(\ell) \leq cost(\ell)$ for all labels ℓ, and (2) $h(scf, s) = h(cost, s)$, for all $s \in S$. For abstraction heuristics there is a unique minimum *scf*, denoted as $saturate(h, cost)$ (Seipp, Keller, and Helmert 2020). Saturated cost partitioning (SCP) (Seipp, Keller, and Helmert 2020) considers a set of n heuristics $\mathcal{H}$ and an order $\omega = \langle h_1, \ldots, h_n \rangle$ over them, and computes the cost partitioning $C = \langle c_1, c_2, \ldots, c_n \rangle$, where $c_i = saturate(h_i, rem_{i-1})$, with $rem_0 = cost$, and $rem_i = rem_{i-1} - c_i$, for $i \geq 1$. We denote the resulting SCP heuristic as h_ω^{SCP}.

Monotonicity Under Adding Abstractions

We first consider the setting where a set of abstractions $\mathcal{A}$ is extended with additional abstractions $\mathcal{B}$. Ideally, the heuristic obtained from $\mathcal{A} \cup \mathcal{B}$ should be at least as informative as the one obtained from $\mathcal{A}$ alone.

Definition 1 (Monotonicity Under Adding Abstractions)**.** *Let $\langle \mathcal{T}, cost \rangle$ be a weighted transition system, let $\mathcal{A}$ and $\mathcal{B}$ be sets of abstractions of $\mathcal{T}$, and let a cost partitioning method compute heuristics $h_\mathcal{A}$ and $h_{\mathcal{A}\cup\mathcal{B}}$ from $\mathcal{A}$ and $\mathcal{A} \cup \mathcal{B}$, respectively. The method is* monotonic under adding abstractions *if, for every state s, $h_{\mathcal{A}\cup\mathcal{B}}(cost, s) \geq h_\mathcal{A}(cost, s)$.*

We now propose a simple ordering strategy for SCP that guarantees monotonicity under adding abstractions.

Let $\mathcal{A}$ and $\mathcal{B}$ be two sets of abstractions. To guarantee that the new abstractions in $\mathcal{B}$ never hurt saturated cost partitioning, we need to produce an order $\omega_{\mathcal{A}\cup\mathcal{B}}$ over $\mathcal{A} \cup \mathcal{B}$ such that $h^{\text{SCP}}_{\omega_{\mathcal{A}\cup\mathcal{B}}}(cost, s) \geq h^{\text{SCP}}_{\omega_\mathcal{A}}(cost, s)$ for a given order $\omega_\mathcal{A}$ and every state $s \in S$.

A simple way to achieve this is to compute the cost partitioning in two steps. First, compute an order $\omega_\mathcal{A}$ for $\mathcal{A}$ and an order $\omega_\mathcal{B}$ for $\mathcal{B}$ independently. Then, construct the order $\omega_{\mathcal{A}\cup\mathcal{B}}$ by concatenating the individual orders, $\omega_{\mathcal{A}\cup\mathcal{B}} = \omega_\mathcal{A} \mathbin{+\!\!+} \omega_\mathcal{B}$.

Theorem 1. *Let $\langle \mathcal{T}, cost \rangle$ be a weighted transition system, $s \in S$ a state, and let $\mathcal{A}$ and $\mathcal{B}$ be two sets of abstraction heuristics for $\mathcal{T}$. If $\omega_\mathcal{A}$ and $\omega_\mathcal{B}$ are orders over $\mathcal{A}$ and $\mathcal{B}$, respectively, then for order $\omega_{\mathcal{A}\cup\mathcal{B}} = \omega_\mathcal{A} \mathbin{+\!\!+} \omega_\mathcal{B}$ it holds that $h^{\text{SCP}}_{\omega_{\mathcal{A}\cup\mathcal{B}}}(cost, s) \geq h^{\text{SCP}}_{\omega_\mathcal{A}}(cost, s)$.*

Proof. Under the order $\omega_{\mathcal{A}\cup\mathcal{B}} = \omega_\mathcal{A} \mathbin{+\!\!+} \omega_\mathcal{B}$, the abstractions in $\mathcal{A}$ are processed exactly as in $\omega_\mathcal{A}$. Therefore, they contribute the same heuristic value as before, and processing the additional abstractions in $\mathcal{B}$ afterwards can only increase the total heuristic value. □

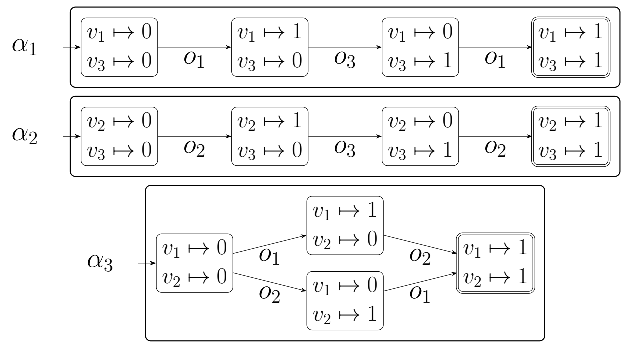


Figure 1: Three abstractions showing that for SCP concatenating partial orders can be the only way to guarantee monotonicity under adding abstractions. In label order $\langle o_1, o_2, o_3 \rangle$, the saturated cost functions are $\langle 1, 0, 1 \rangle$, $\langle 0, 1, 1 \rangle$, and $\langle 1, 1, 0 \rangle$. For $\mathcal{A} = \{\alpha_1, \alpha_2\}$ and $\mathcal{B} = \{\alpha_3\}$, $h^{\text{SCP}}_{\omega_\mathcal{A}}(cost, s_0) = 5$ for either order, but inserting α_3 first or second yields 3 or 4.

This naive strategy is suboptimal when the abstractions in $\mathcal{B}$ are more informative, because they receive only the costs left after saturating $\mathcal{A}$. However, finding a different order that guarantees $h^{\text{SCP}}_{\omega_{\mathcal{A}\cup\mathcal{B}}}(cost, s) \geq h^{\text{SCP}}_{\omega_\mathcal{A}}(cost, s)$ is more challenging and, in some cases, impossible (see Figure 1).

Monotonicity Under Refinements

We now consider the setting where some abstractions are coarsenings of others.

Definition 2 (Coarsening and Refinement)**.** *Let $\mathcal{T}$ be a transition system, and let α and β be two abstractions of $\mathcal{T}$, inducing the abstract transition systems $\mathcal{T}^\alpha$ and $\mathcal{T}^\beta$, respectively. We say that α is a* coarsening *of β and β a* refinement *of α, denoted by $\alpha \preceq \beta$, if there exists an abstraction $\gamma : S^\beta \to S^\alpha$ such that $\alpha = \gamma \circ \beta$.*

Ideally, replacing coarsenings by their refinements should preserve or improve heuristic estimates.

Definition 3 (Monotonicity Under Refinements)**.** *Let $\langle \mathcal{T}, cost \rangle$ be a weighted transition system, let $\mathcal{A}$ and $\mathcal{B}$ be sets of abstractions of $\mathcal{T}$ such that every $\alpha \in \mathcal{A}$ is a coarsening of at least one abstraction in $\mathcal{B}$, and let a cost partitioning method compute heuristics $h_\mathcal{A}$ and $h_\mathcal{B}$ from $\mathcal{A}$ and $\mathcal{B}$, respectively. The method is* monotonic under refinements *if, for every state s, $h_\mathcal{B}(cost, s) \geq h_\mathcal{A}(cost, s)$.*

We propose a cost partitioning scheme based on the path-preserving property of abstractions that guarantees monotonicity under refinements. If $\alpha \preceq \beta$, then for any cost function *cost* and state $s \in S$, we have $h^\alpha(cost, s) \leq h^\beta(cost, s)$. This observation leads to the following theorem:

Theorem 2. *Let $\langle \mathcal{T}, cost \rangle$ be a weighted transition system, let $s \in S$ be a state, let $\mathcal{A} = \langle \alpha_1, \alpha_2, \ldots, \alpha_n \rangle$ be a sequence of abstractions of $\mathcal{T}$, and let β be an abstraction of $\mathcal{T}$ such that $\alpha_i \preceq \beta$ for all i. If $\langle c_1, c_2, \ldots, c_n \rangle$ is a cost partitioning for $\langle \mathcal{T}, cost \rangle$ over $\mathcal{A}$, then $\sum_{i=1}^n h^{\alpha_i}(c_i, s) \leq h^\beta(\sum_{i=1}^n c_i, s)$.*

Proof. Since $\alpha_i \preceq \beta$ for all i, each α_i is also an abstraction of $\mathcal{T}^\beta$. Therefore, $\langle c_1, c_2, \ldots, c_n\rangle$ is a cost partitioning over $\mathcal{A}$ for the weighted transition system $\langle \mathcal{T}^\beta, \sum_{i=1}^n c_i\rangle$. The corresponding cost partitioning heuristic $\sum_{i=1}^n h^{\alpha_i}(c_i, s)$ is admissible for this transition system, so it is bounded above by its optimal cost, namely $h^\beta(\sum_{i=1}^n c_i, s)$. □

Given a cost partitioning $\langle c_1, c_2, \ldots, c_n\rangle$ over a set of n abstractions $\mathcal{A}$ of $\mathcal{T}$ and a refinement β of every abstraction in $\mathcal{A}$, Theorem 2 guarantees that the heuristic $h^\beta(\sum_{i=1}^n c_i, s)$ is at least as informative as the cost partitioning heuristic over $\mathcal{A}$. However, if there are *multiple* refinements $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_m\}$ of some abstractions in $\mathcal{A}$, distributing the combined costs across $\mathcal{B}$ may yield higher heuristic values than assigning all costs to a single refinement. We now show how to split costs among the refinements of a coarsening α while preserving dominance.

Theorem 3. *Let $\langle \mathcal{T}, cost\rangle$ be a weighted transition system, α an abstraction of $\mathcal{T}$ and $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_m\}$ a set of abstractions of $\mathcal{T}$ such that $\alpha \preceq \beta_i$ for all $\beta_i \in \mathcal{B}$. For any state $s \in S$ and any sequence of factors $\langle f_1, f_2, \ldots, f_m\rangle$, where $f_i \in \mathbb{R}_0^+$ and $\sum_{i=1}^m f_i = 1$, we have $h^\alpha(cost, s) \leq \sum_{i=1}^m h^{\beta_i}(f_i \cdot cost, s)$.*

Proof. Since $\alpha \preceq \beta_i$ for all i, we have $h^\alpha(cost, s) \leq h^{\beta_i}(cost, s)$. Multiplying by $f_i \geq 0$ yields $f_i \cdot h^\alpha(cost, s) \leq f_i \cdot h^{\beta_i}(cost, s)$ for all i. Summing over all i gives $\sum_{i=1}^m f_i \cdot h^\alpha(cost, s) \leq \sum_{i=1}^m f_i \cdot h^{\beta_i}(cost, s)$. Since $\sum_{i=1}^m f_i = 1$, the left-hand side is $h^\alpha(cost, s)$. Finally, scaling all label costs by f_i scales all plan costs, and therefore also the optimal cost, by the same factor, so $f_i \cdot h^{\beta_i}(cost, s) = h^{\beta_i}(f_i \cdot cost, s)$ for all i. Hence, $h^\alpha(cost, s) \leq \sum_{i=1}^m h^{\beta_i}(f_i \cdot cost, s)$. □

Theorem 3 allows us to split the costs assigned to a coarsening α among its refinements, ensuring that the combined heuristic is at least as informative as h^α alone, without allocating any cost to α.

Definition 4 (Cost Partitioning Over Refinements)**.** *Given a weighted transition system $\langle \mathcal{T}, cost\rangle$ and two sets of abstractions $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ and $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_m\}$, such that for each $\alpha \in \mathcal{A}$ there exists at least one $\beta \in \mathcal{B}$ with $\alpha \preceq \beta$, and for each $\beta_j \in \mathcal{B}$ there exists a non-empty set of abstractions $\mathcal{A}_j \subseteq \mathcal{A}$ such that $\alpha_i \preceq \beta_j$ for all $\alpha_i \in \mathcal{A}_j$. If $\langle c_1, c_2, \ldots, c_n\rangle$ is a cost partitioning for $\langle \mathcal{T}, cost\rangle$ over $\mathcal{A}$, a* cost partitioning over refinements *over $\mathcal{B}$ is a cost partitioning $\langle c'_1, c'_2, \ldots, c'_m\rangle$ for $\langle \mathcal{T}, cost\rangle$ over $\mathcal{B}$ where each c'_j is computed as $c'_j = \sum_{\alpha_i \in \mathcal{A}_j} c_i \cdot f_{ij}$, with each $f_{ij} \in \mathbb{R}_0^+$, such that for each $1 \leq i \leq n$, $\sum_{\alpha_i \preceq \beta_j} f_{ij} = 1$.*

A cost partitioning over refinements guarantees that the heuristic obtained using only abstractions in $\mathcal{B}$ is at least as informative as the one obtained using only abstractions in $\mathcal{A}$.

Theorem 4. *Let $\langle \mathcal{T}, cost\rangle$ be a weighted transition system, let $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ and $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_m\}$ be two sets of abstractions for $\mathcal{T}$, and let $s \in S$ be a state of $\mathcal{T}$. If $\langle c_1, c_2, \ldots, c_n\rangle$ is a cost partitioning for $\langle \mathcal{T}, cost\rangle$ over $\mathcal{A}$, and $\langle c'_1, c'_2, \ldots, c'_m\rangle$ is a cost partitioning over refinements for $\mathcal{B}$, then $\sum_{i=1}^n h^{\alpha_i}(c_i, s) \leq \sum_{j=1}^m h^{\beta_j}(c'_j, s)$.*

Proof. For each i, the definition of cost partitioning over refinements splits the cost function c_i among refinements of α_i using factors that sum to 1. By Theorem 3, this replacement does not decrease the heuristic value. Hence, after applying it to all i, it remains to combine, for each refinement β_j, all costs assigned to β_j. By Theorem 2, combining these costs into the single cost function c'_j does not decrease the heuristic value either. □

	h_1^{SCP}	h_2^{SCP}	h_{naive}^{SCP}	h_{rand}^{SCP}	h_{unif}^{SCP}
h_1^{SCP}	–	172	0	0	0
h_2^{SCP}	**373**	–	**256**	**237**	214
h_{naive}^{SCP}	**185**	210	–	0	0
h_{rand}^{SCP}	**281**	235	**100**	–	16
h_{unif}^{SCP}	**323**	**254**	**183**	**90**	–

Table 1: Per-task comparison of initial heuristic values obtained by different SCP configurations over Cartesian abstractions. Cell (*r*, *c*) shows the number of tasks in which *r* has a higher heuristic value than *c*. The maximum between (*r*, *c*) and (*c*, *r*) is in bold, to show the "winner" between *r* and *c*.

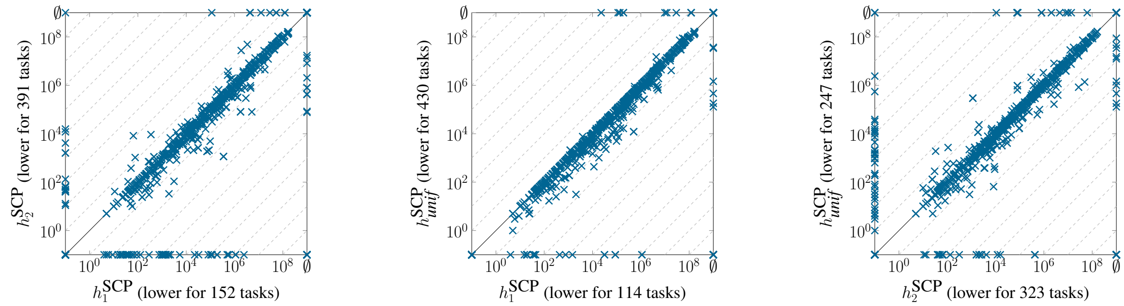
Note that we require each abstraction in $\mathcal{A}$ to be a coarsening of at least one abstraction in $\mathcal{B}$ for simplicity. If that is not the case, we can simply add the abstractions in $\mathcal{A}$ that are not coarsenings of any abstraction in $\mathcal{B}$ to $\mathcal{B}$ with their original cost function, and the resulting cost partitioning will maintain the dominance guarantee.

Experiments

Our experiments address two questions: first, whether the proposed constructions realize their intended monotonicity guarantees in practice; and second, whether these guarantees translate into better search performance.

We implemented the naive ordering and cost partitioning over refinements in the Scorpion planner (Seipp, Keller, and Helmert 2020), an extension of Fast Downward (Helmert 2006). All code and data are available online (Salerno, Fuentetaja, and Seipp 2026). We use A* as the search algorithm and online SCP with greedy orders (Seipp 2017) as the heuristic. For Cartesian abstractions, we consider 1-goal abstractions as the coarsenings of 2-goal abstractions obtained via merging (Salerno et al. 2025). For PDBs, we consider patterns of size 1 as the coarsenings of patterns of size 2 obtained with systematic enumeration (Pommerening et al. 2015). As benchmarks, we use all tasks without conditional effects from IPC editions between 1998 and 2023. All runs have a memory limit of 8 GiB and a time limit of 30 minutes.

As baselines, we consider SCP over 1-goal Cartesian abstractions (h_1^{SCP}) and SCP over the combined set of 1-goal and 2-goal abstractions obtained via merging (h_2^{SCP}). For PDBs, the baselines are single-variable patterns ($h_{\text{PDB-1}}^{\text{SCP}}$) and patterns of size one and two ($h_{\text{PDB-2}}^{\text{SCP}}$). For 1-goal and 2-goal abstractions, we evaluate the naive ordering (h_{naive}^{SCP}) and two variants of cost partitioning over refinements: h_{rand}^{SCP}, which assigns each coarsening to a randomly chosen refinement, and h_{unif}^{SCP}, which distributes each coarsening's cost function uniformly among its refinements. Note that for h_{rand}^{SCP} and h_{unif}^{SCP},



(a) h_1^{SCP} vs h_2^{SCP}. (b) h_1^{SCP} vs h_{unif}^{SCP}. (c) h_2^{SCP} vs h_{unif}^{SCP}.

Figure 2: Expansions until last f-layer for different SCP configurations. Tasks unsolved within resource limits are marked with $\emptyset$.

	h_1^{SCP}	h_2^{SCP}	h_{naive}^{SCP}	h_{rand}^{SCP}	h_{unif}^{SCP}
h_1^{SCP}	–	**6**	3	1	**6**
h_2^{SCP}	**6**	–	5	3	**6**
h_{naive}^{SCP}	**6**	**11**	–	2	**9**
h_{rand}^{SCP}	**6**	**9**	**3**	–	**9**
h_{unif}^{SCP}	5	5	3	3	–
Coverage	1006	1019	1020	1021.3±1.2	**1037**

Table 2: Per-domain and total coverage comparison of different SCP configurations over Cartesian abstractions. Cell (r, c) holds the number of domains in which r solves more tasks than c. The maximum between (r, c) and (c, r) is in bold, to highlight the “winner” between r and c. The last row shows the total number of solved tasks (average coverage and std. dev. over 10 seeds for h_{rand}^{SCP}).

some costs may still remain after saturating the coarsenings and computing the cost partitioning over refinements. We therefore run a second SCP pass over the refined abstractions to assign any remaining costs among the refinements.

Table 1 illustrates the guaranteed monotonicity of our schemes in terms of initial heuristic values: h_{naive}^{SCP} is monotonic under adding abstractions, whereas h_{rand}^{SCP} and h_{unif}^{SCP} are monotonic under refinements. Consequently, all three methods never produce lower heuristic values than h_1^{SCP}, while h_2^{SCP} produces lower values than h_1^{SCP} in 172 tasks.

Table 2 compares the coverage between the different heuristics. Overall, all three proposed schemes solve more tasks than h_1^{SCP} and h_2^{SCP}. Across individual domains, most approaches are competitive with each other, although h_{unif}^{SCP} performs worst. The difference between its overall and per-domain coverage stems from the fact that h_{unif}^{SCP} solves many Miconic tasks, but solves 1–2 fewer tasks than other approaches in 6–9 other domains.

Figure 2a illustrates the failure of monotonicity under adding abstractions for standard SCP: there are many tasks where h_2^{SCP} expands orders of magnitude more nodes than h_1^{SCP}. Figures 2b and 2c show that h_{unif}^{SCP} is on par with h_2^{SCP} in terms of expansions. Note that the number of expansions by h_{unif}^{SCP} is not always less than or equal to that of h_2^{SCP} because we compute a new SCP heuristic only for every thousandth evaluated state. Both h_{naive}^{SCP} and h_{rand}^{SCP} exhibit similar behavior.

For PDBs, standard SCP is already closer to monotonicity: $h_{\text{PDB-2}}^{\text{SCP}}$ yields lower heuristic values than $h_{\text{PDB-1}}^{\text{SCP}}$ in only 32 tasks and yields higher estimates in 1242 tasks. Accordingly, the gains of our schemes are smaller, so we omit the detailed results for brevity. Overall, when using patterns of size 1–2, $h_{\text{PDB-1}}^{\text{SCP}}$ solves 897 tasks, $h_{\text{PDB-2}}^{\text{SCP}}$ solves 997, h_{naive}^{SCP} solves 996, h_{rand}^{SCP} solves 999, and h_{unif}^{SCP} solves 924.

Perfect Saturated Cost Partitioning. Höft, Speck, and Seipp (2025) propose algorithms for computing all SCP orders more efficiently than enumeration. Their algorithm satisfies monotonicity under adding abstractions, since it can always find the naive order we propose. However, cost partitioning over refinements can compute cost partitionings different from any that SCP alone can find: it can assign costs to a refinement that are lower than its saturated cost function. We omit an empirical comparison because, as the authors report, their approach runs out of memory for tasks with many abstractions (the largest task uses 160). With n goal atoms, there are $\binom{n}{2}$ 2-goal Cartesian abstractions, which makes it infeasible to run the perfect SCP algorithm for most tasks (for $n = 19$, $\binom{n}{2} = 171$ and our tasks have up to 323 goal atoms, averaging 20).

Conclusions

We distinguished monotonicity under adding abstractions from monotonicity under refinements. For the first notion, we proposed an ordering for SCP that guarantees that adding abstractions never reduces the informativeness of the resulting heuristic. For the second notion, we introduced a cost partitioning scheme tailored for settings where some abstractions are refinements of other abstractions, guaranteeing that the heuristic obtained using only the more fine-grained abstractions is at least as informative as the one obtained using only the coarser abstractions. Empirically, we showed that our schemes outperform standard saturated cost partitioning on Cartesian abstractions and are competitive on pattern databases.

Acknowledgments

This work was partially supported by grant PID2021-127647NB-C21 funded by MICIU/AEI/10.13039/501100011033 and by "ERDF A way of making Europe". Additionally, 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.

References

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

Edelkamp, S. 2001. Planning with Pattern Databases. In Cesta, A.; and Borrajo, D., eds., *Proceedings of the Sixth European Conference on Planning (ECP 2001)*, 84–90. 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.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Höft, P.; Speck, D.; and Seipp, J. 2025. Representing Perfect Saturated Cost Partitioning Heuristics in Classical Planning. In Ortiz, M.; Wassermann, R.; and Schaub, T., eds., *Proceedings of the Twenty-Second International Conference on Principles of Knowledge Representation and Reasoning (KR 2025)*. IJCAI Organization.

Katz, M.; and Domshlak, C. 2008a. Optimal Additive Composition of Abstraction-based Admissible Heuristics. In Rintanen, J.; Nebel, B.; Beck, J. C.; and Hansen, E., eds., *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)*, 174–181. AAAI Press.

Katz, M.; and Domshlak, C. 2008b. Structural Patterns Heuristics via Fork Decomposition. In Rintanen, J.; Nebel, B.; Beck, J. C.; and Hansen, E., eds., *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)*, 182–189. AAAI Press.

Katz, M.; and Domshlak, C. 2010. Optimal admissible composition of abstraction heuristics. *Artificial Intelligence*, 174(12–13): 767–798.

Pommerening, F.; Helmert, M.; Röger, G.; and Seipp, J. 2015. From Non-Negative to General Operator Cost Partitioning. In Bonet, B.; and Koenig, S., eds., *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2015)*, 3335–3341. AAAI Press.

Pommerening, F.; Röger, G.; and Helmert, M. 2013. Getting the Most Out of Pattern Databases for Classical Planning. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, 2357–2364. AAAI Press.

Salerno, M.; Fuentetaja, R.; and Seipp, J. 2026. Code and Data for the ICAPS 2026 Paper "Monotonic Variants of Saturated Cost Partitioning". https://doi.org/10.5281/zenodo.18787651.

Salerno, M.; Fuentetaja, R.; Speck, D.; and Seipp, J. 2025. Merging Cartesian Abstractions for Classical Planning. In Lynce, I.; and Murano, A., eds., *Proceedings of the 28th European Conference on Artificial Intelligence (ECAI 2025)*. IOS Press.

Seipp, J. 2017. Better Orders for Saturated Cost Partitioning in Optimal Classical Planning. In Fukunaga, A.; and Kishimoto, A., eds., *Proceedings of the 10th Annual Symposium on Combinatorial Search (SoCS 2017)*, 149–153. AAAI Press.

Seipp, J.; and Helmert, M. 2018. Counterexample-Guided Cartesian Abstraction Refinement for Classical Planning. *Journal of Artificial Intelligence Research*, 62: 535–577.

Seipp, J.; Keller, T.; and Helmert, M. 2020. Saturated Cost Partitioning for Optimal Classical Planning. *Journal of Artificial Intelligence Research*, 67: 129–167.

Sievers, S.; Pommerening, F.; Keller, T.; and Helmert, M. 2020. Cost-Partitioned Merge-and-Shrink Heuristics for Optimal Classical Planning. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI 2020)*, 4152–4160. IJCAI.