Bridging Multi-Valued Heuristics and Dimensionality Reduction in Multi-Objective Search

Maya Wolff,[1] **Ariel Felner,**[2] **Oren Salzman**[1]

[1]Department of Computer Science, Technion – Israel Institute of Technology, Haifa, Israel
[2]Faculty of Computer and Information Science, Ben-Gurion University of the Negev, Beer-Sheva, Israel
wo@cs.technion.ac.il, felner@bgu.ac.il, osalzman@cs.technion.ac.il

Abstract

Multi-objective shortest-path (MOSP) algorithms traditionally rely on single-valued heuristics (SVHs), which associate each state with a single admissible cost vector. While SVHs provide safe lower bounds, they fail to capture the trade-off structure of the Pareto frontier and often yield weak search guidance. Multi-valued heuristics (MVHs) address this limitation by mapping states to sets of cost estimates, enabling a richer approximation of possible trade-offs.

Modern MOSP algorithms are highly dependent on dimensionality reduction (DR) techniques to efficiently perform dominance checks. However, integrating MVHs with DR introduces subtle correctness challenges. We show that naively combining DR with MVHs destroys the ordering invariants required for DR, leading to unsound and incomplete search. To address this issue, we develop the first theoretical frameworks for safely integrating MVHs with DR.

First, we introduce NAMOA*dr-mvh, a theoretical baseline that restores search correctness by enforcing heuristic consistency. Recognizing the practical limitations of this approach, we then introduce our primary contribution L-NAMOA*dr-mvh. This algorithm employs a "lazy," optimistic approach to DR, preserving exact correctness with only an *admissible* MVH by dynamically detecting and repairing local ordering violations. Across a range of benchmarks, L-NAMOA*dr-mvh matches or improves over state-of-the-art MOSP algorithms, and achieves speedups of over 10x in instances where the additional guidance provided by the MVH translates into stronger pruning.

Introduction & Related Work

In the multi-objective shortest-path (MOSP) problem we are interested in finding paths between two vertices of a graph while considering multiple, often conflicting objectives. Applications of MOSP range from transporting hazardous materials considering travel distance and risk (Bronfman et al. 2015) and inspecting a region of interest using cameras placed on-board robotic platforms (Fu et al. 2023). This family of problems is not new, with results dating back decades (Vincke 1976; Hansen 1980; Clímaco and Pascoal 2012; Current and Marsh 1993; Skriver 2000; Ulungu and Teghem 1991). Nevertheless, there has been renewed interest and significant progress in the field of heuristic search for multi-objective search (MOS) (Salzman et al. 2023), reflecting the reality that real-world systems rarely optimize a single measure (Salzman et al. 2026).

State-of-the-art (SOTA) heuristic MOS algorithms rely primarily on two mechanisms to improve their efficiency. First, they utilize *heuristics* to guide the search. Most heuristic MOS algorithms use a single-valued heuristic (SVH), which estimates the cost to the goal from every state (see, e.g., (Ahmadi et al. 2024; Ren et al. 2025)). However, there are a handful of works that considered the more-informative notion of multi-valued heuristics (MVH), which map states to sets of cost estimates, enabling a richer approximation of possible trade-offs (see, e.g., (Mandow and Pérez-de-la Cruz 2005; Geißer et al. 2022; Zhang et al. 2023)). Second, modern algorithms employ *dimensionality reduction* (DR) techniques to improve the efficiency of dominance checking operations (i.e., assessing if one solution is strictly better than another across all problem objectives). Improving these dominance checks is crucial, as they are often the key computational bottleneck of many SOTA algorithms for MOSP (Pulido, Mandow, and Pérez-de-la Cruz 2015).

Each mechanism (namely, MVHs and DR) by itself is a powerful algorithmic tool. Yet, utilizing both simultaneously is highly non-trivial and has been identified as a key challenge in the field (Salzman et al. 2023). To this end, in this paper we pinpoint the precise source of this incompatibility and develop two algorithmic frameworks to overcome it.

First, we introduce NAMOA*dr-mvh which serves as a theoretical baseline. It demonstrates that if we impose constraints on the MVH (specifically, requiring it to be consistent) a relaxed approach of DR can be used safely. However, while theoretically sound, this approach may introduce computational inefficiencies as the algorithm may need to generate redundant paths. Moreover, the approach relies on the difficult task of constructing consistent MVHs.

To overcome these practical bottlenecks, we introduce our primary contribution: L-NAMOA*dr-mvh. Rather than forcing the aforementioned constraints upfront, this algorithm adopts a "lazy," optimistic approach. It performs fast DR by default, but dynamically monitors the search to detect cases when the heuristic guidance might cause an error. Only when a potential error is detected does the algorithm fall back to a rigorous, standard dominance check.

Consequently, L-NAMOA*dr-mvh preserves soundness

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

and completeness while requiring only a standard, admissible MVH. By avoiding the potential overhead of NAMOA*dr-mvh, it successfully combines the stronger guidance of MVHs with the computational efficiency of DR. We empirically evaluate our algorithms on a range of benchmarks, demonstrating that L-NAMOA*dr-mvh matches or improves upon NAMOA*dr, with speedups of over $10\times$ when MVH guidance enables stronger pruning.

Notation & Algorithmic Background

Notation

We follow standard notation in MOS (Salzman et al. 2023). Boldface font indicates vectors, and lower-case and upper-case symbols indicate elements and sets, respectively. For an N-dimensional vector $\mathbf{v}$, v_i denotes its i-th component. Vector addition is defined as element-wise summation.

Let $\mathbf{p}$ and $\mathbf{q}$ be N-dimensional vectors. We say that $\mathbf{p}$ *weakly dominates* $\mathbf{q}$, denoted as $\mathbf{p} \preceq \mathbf{q}$, if $p_i \leq q_i$ for all $i = 1, \ldots, N$. We say that $\mathbf{p}$ *dominates* $\mathbf{q}$, denoted as $\mathbf{p} \prec \mathbf{q}$, if $\mathbf{p}$ weakly dominates $\mathbf{q}$ and there exists an index j such that $p_j < q_j$. When $\mathbf{p} \not\prec \mathbf{q}$ and $\mathbf{q} \not\prec \mathbf{p}$, we say that $\mathbf{p}$ and $\mathbf{q}$ are mutually non-dominated. Furthermore, we say that $\mathbf{p}$ is lexicographically smaller than $\mathbf{q}$, denoted as $\mathbf{p} <_{\text{lex}} \mathbf{q}$, if $p_k < q_k$ for the first index k such that $p_k \neq q_k$.

Let X be a set of N-dimensional vectors. We denote by $\text{Nd}(X)$ the cost-unique subset of X containing only mutually non-dominated vectors. The truncation function Tr maps a vector $\mathbf{v} = (v_1, \ldots, v_N)$ to the $(N-1)$-dimensional vector obtained by removing its first component, i.e., $\text{Tr}(\mathbf{v}) = (v_2, \ldots, v_N)$. With a slight abuse of notation, we define the truncated set of X as $\text{Tr}(X) = \text{Nd}(\{\text{Tr}(\mathbf{x}) \mid \mathbf{x} \in X\})$, similarly written as X^{Tr}.

A vector $\mathbf{v} \in \mathbb{R}^N$ is *t-discarded* by a set $X \subseteq \mathbb{R}^N$ if there exists a vector $\mathbf{u} \in X$ such that $\text{Tr}(\mathbf{u}) \in \text{Tr}(X)$ and one of the following holds: (i) $u_1 < v_1$ and $\text{Tr}(\mathbf{u}) \preceq \text{Tr}(\mathbf{v})$, or (ii) $u_1 = v_1$ and $\text{Tr}(\mathbf{u}) \prec \text{Tr}(\mathbf{v})$.

A *multi-objective search graph* is a tuple $G = \langle S, E, c \rangle$, where S is a finite set of states, $E \subseteq S \times S$ is a finite set of edges, and $c : E \to \mathbb{R}^N_{\geq 0}$ is a cost function that associates a vector of N non-negative cost components (namely, the objectives) with each edge. The successor function is defined as $\text{Succ}(s) = \{s' \in S \mid (s, s') \in E\}$. A path π from state s_1 to state s_ℓ is a sequence of states $[s_1, \ldots, s_\ell]$ such that $(s_j, s_{j+1}) \in E$ for all $j = 1, \ldots, \ell - 1$. The cost of a path π is $\mathbf{c}(\pi) = \sum_{j=1}^{\ell-1} c(s_j, s_{j+1})$.

A *multi-objective search instance* is a tuple $P = \langle S, E, c, s_{\text{start}}, s_{\text{goal}} \rangle$, where $s_{\text{start}} \in S$ and $s_{\text{goal}} \in S$ are the start and goal states, respectively. A path connecting s_{start} to s_{goal} is called a *solution*. A solution is *Pareto-optimal* if its cost is not dominated by any other solution. The *Pareto-optimal solution set*, denoted Π^*, is the set of all Pareto-optimal solutions.

Finally, multi-objective search algorithms often use a heuristic function to guide the search.

A *single-valued heuristic* (SVH) is a function $h : S \to \mathbb{R}^N_{\geq 0}$ that estimates the cost to s_{goal} from every state s. We say that h is admissible if $h(s) \preceq \mathbf{c}(\pi)$ for every state s and for all path π from s to s_{goal}. We say that h is consistent if $h(s_{\text{goal}}) = \mathbf{0}$ and $h(s) \preceq c(s, s') + h(s')$ for all $(s, s') \in E$.

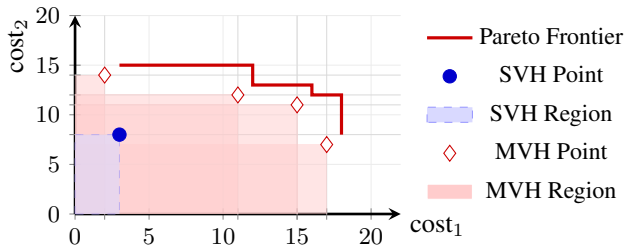


Figure 1: Comparison of pruning regions by SVH (blue) and MVH (red). The solid red line represents the Pareto-optimal solution frontier of the problem.

A *multi-valued heuristic* (MVH) $H : S \to 2^{\mathbb{R}^N_{\geq 0}}$ maps each state to a set of mutually non-dominated vectors. We say that H is admissible if for every state s and for all path π from s to s_{goal}, $H(s)$ contains a cost vector that weakly dominates $\mathbf{c}(\pi)$. We say that H is consistent if $H(s_{\text{goal}}) = \{\mathbf{0}\}$ and for every $(s, s') \in E$ and for every $\mathbf{h} \in H(s)$, there exists $\mathbf{h}' \in H(s')$ such that $\mathbf{h} \preceq c(s, s') + \mathbf{h}'$.

The Pruning Power of MVHs

Intuitively, the advantage of an MVH over an SVH lies in its "informativeness". While an SVH provides one lower-bound estimate, an MVH can capture a set of non-dominated trade-offs that more accurately reflects the costs to the goal.

This is illustrated in Fig. 1 where an SVH (blue) provides a "loose" lower bound. While this point dominates a large area of the cost space, its proximity to the origin makes it a weak filter by the actual costs of known solutions. In contrast, the MVH provides a frontier of "tighter" estimates (red) that sit further from the origin. Since MVH estimates are tighter while remaining admissible, they make it more likely that a path cost will be dominated by an existing solution. This allows the search to prune suboptimal branches early, whereas a looser bound would be forced to explore them.

Algorithmic Background

Multi-Objective A* (MOSA*). Best-first search MOS algorithms, often generalized as MOSA* (Skyler et al. 2024), compute Π^* by maintaining a priority queue OPEN of generated but not yet extracted paths. A fundamental difference between standard single-objective A* and MOSA* is path tracking: while standard A* typically expands a state only once via its shortest path, MOSA* must maintain a separate search node for every distinct, mutually non-dominated cost vector $\mathbf{g}$ reaching that state. When utilizing a traditional single-valued heuristic (SVH), each such node n is evaluated by a single vector $\mathbf{f}(n) = \mathbf{g}(n) + \mathbf{h}(s(n))$. Prominent MOSP algorithms, including NAMOA* and NAMOA*dr, are specific instantiations of this MOSA* framework. However, as we will demonstrate, this straightforward evaluation where one $\mathbf{g}$-value maps to exactly one $\mathbf{f}$-value becomes more complex when integrating MVHs.

To ensure efficiency, MOSA* algorithms rely on *dominance checks* to determine whether a newly generated or extracted node can still contribute to the Pareto-optimal solution set. This typically involves two layers of pruning: (i) *local dominance checking*, where a node's cost $\mathbf{g}$ is compared against previously discovered paths to the same state to prune locally sub-optimal routes, and (ii) *global dominance checking*, where a node's evaluation $\mathbf{f}$ is compared against known solutions at the goal to prune paths that are already pruned. Because a node must be compared against entire sets of mutually non-dominated vectors, these dominance checks constitute the primary computational bottleneck of MOS, directly motivating the need for DR.

NAMOA* NAMOA* (Mandow and Pérez-de-la Cruz 2005) is an instantiation of the MOSA* framework that natively supports MVHs. Instead of mapping to a single $\mathbf{f}$-value, a path reaching state s with cost $\mathbf{g}$ under an MVH H is evaluated using a *set* of mutually non-dominated $\mathbf{f}$-values, defined as $F(s,\mathbf{g}) = \mathrm{Nd}(\{\mathbf{g}+\mathbf{h} \mid \mathbf{h} \in H(s)\})$. Consequently, each node in OPEN is represented as a triplet $n = \langle s, \mathbf{g}, F(s,\mathbf{g})\rangle$.

To execute dominance checks, NAMOA* maintains the following sets. For *global dominance checking*, it maintains Sols, the non-dominated cost vectors of solutions found so far. For *local dominance checking*, it maintains two sets of $\mathbf{g}$-values for each state s: $G_{\mathrm{op}}(s)$ (paths generated but not yet extracted) and $G_{\mathrm{cl}}(s)$ (paths already extracted).[1]

At each iteration, NAMOA* extracts a node whose F-set contains an $\mathbf{f}$-value that is not dominated by any other $\mathbf{f}$-value in OPEN. When a successor node is generated, it is pruned globally if all its $\mathbf{f}$-values are dominated by Sols, and pruned locally if its $\mathbf{g}$-value is dominated by $G_{\mathrm{op}}(s')$ or $G_{\mathrm{cl}}(s')$. While NAMOA* is optimal in the number of path extractions when using a consistent heuristic (Mandow and Pérez de la Cruz 2010), performing full dominance checks is computationally expensive.

NAMOA*dr To alleviate the computational bottleneck of full-vector dominance checks, Pulido, Mandow, and Pérez-de-la Cruz (2015) proposed NAMOA*dr. This algorithm restricts the search to use a consistent SVH, enabling the use of dimensionality reduction (DR). In the context of MOS algorithms, DR hinges on the following property:

Property 1. *Assume that a best-first search MOS algorithm (i) uses a consistent SVH h and (ii) orders* OPEN *lexicographically according to the $\mathbf{f}$-value of nodes (i.e., $\mathbf{g}+\mathbf{h}$). Let n, n' be two search nodes corresponding to the same state s. If n is extracted before n' from* OPEN *then both $f_1(n) \le f_1(n')$ and $g_1(n) \le g_1(n')$.*

To understand why Property 1 holds, note that nodes are extracted from OPEN in lexicographically non-decreasing order of $\mathbf{f}$-values. Because the $\mathbf{h}$-value is fixed for a given state s when using an SVH, the nodes of state s are extracted in lexicographically non-decreasing order of $\mathbf{g}$-values.

The implication of Property 1 is that the first cost component (g_1 or f_1) of a newly extracted node will always be greater than or equal to that of all previously extracted nodes for the same state. This guarantees monotonic growth in the first dimension, eliminating the need to check dominance for that cost element.

Leveraging this, NAMOA*dr replaces standard dominance checks with more efficient *t-discarding* tests (Sec.). Instead of maintaining full open and closed sets, NAMOA*dr maintains only a *truncated* set of closed nodes for each state, denoted by $G^{\mathrm{Tr}}_{\mathrm{cl}}(s)$.

Alg. 1 illustrates this control flow (red annotations should be ignored at this point). Blue annotations highlight the specific operations where DR replaces standard NAMOA* dominance tests, utilizing the truncated vectors $\mathrm{Tr}(\cdot)$ and the reduced per-state sets $G^{\mathrm{Tr}}_{\mathrm{cl}}(\cdot)$.

After initialization, nodes are extracted from OPEN in lexicographically non-decreasing $\mathbf{f}$ order (Line 8). Due to Property 1, dominance checks at both extraction time and generation time are performed using the truncated vectors $\mathrm{Tr}(\cdot)$ and the reduced per-state sets $G^{\mathrm{Tr}}_{\mathrm{cl}}(\cdot)$. Specifically, an extracted node n is pruned if its truncated cost vector $\mathrm{Tr}(\mathbf{g}(n))$ is dominated by $G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n))$ (Line 9) or if its truncated evaluation vector $\mathrm{Tr}(\mathbf{f}(n))$ is dominated by $G^{\mathrm{Tr}}_{\mathrm{cl}}(s_{\mathrm{goal}})$ (Line 11).

When a node passes these t-discarding checks upon ex-

Algorithm 1 NAMOA*dr (NAMOA*dr-mvh)

Input: A MOS instance $(S, E, \mathbf{c}, s_{\mathrm{start}}, s_{\mathrm{goal}})$; a consistent SVH $\mathbf{h}$. ▷ MVH H
Output: A cost-unique Pareto-optimal solution set Sols.

1: Sols $\leftarrow \emptyset$
2: **for all** $s \in S$ **do** ▷ **for all** $\mathbf{h} \in H(s)$ **do**
3: $G^{\mathrm{Tr}}_{\mathrm{cl}}(s) \leftarrow \emptyset$ ▷ $G^{\mathrm{Tr}}_{\mathrm{cl}}(s,\mathbf{h}) \leftarrow \infty$
4: $n \leftarrow$ new node with $s(n) = s_{\mathrm{start}}$
5: parent$(n) \leftarrow$ NULL; $\mathbf{g}(n) \leftarrow \mathbf{0}$; $\mathbf{f}(n) \leftarrow \mathbf{h}(s_{\mathrm{start}})$
6: OPEN $\leftarrow \{n\}$
7: **while** OPEN $\neq \emptyset$ **do**
8: $n \leftarrow$ OPEN.Pop ▷ lex smallest $\mathbf{f}$-value
9: **if** $G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n)) \preceq \mathrm{Tr}(\mathbf{g}(n))$ **then** ▷ $G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n),\mathbf{h}(n))$
10: **continue**
11: **if** $G^{\mathrm{Tr}}_{\mathrm{cl}}(s_{\mathrm{goal}}) \preceq \mathrm{Tr}(\mathbf{f}(n))$ **then**
12: **continue**
13: $G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n)) \leftarrow \mathrm{Nd}\big(G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n)) \cup \mathrm{Tr}(\mathbf{g}(n))\big)$ ▷ $G^{\mathrm{Tr}}_{\mathrm{cl}}(s(n),\mathbf{h}(n))$
14: **if** $s(n) = s_{\mathrm{goal}}$ **then**
15: Sols $\leftarrow$ Sols $\cup \{n\}$
16: **continue**
17: **for all** $s' \in \mathrm{Succ}(s(n))$ **do**
18: $n' \leftarrow$ a new node with $s(n') = s'$
19: parent$(n') \leftarrow n$; $\mathbf{g}(n') \leftarrow \mathbf{g}(n) + \mathbf{c}(s(n), s')$
20: $\mathbf{f}(n') \leftarrow \mathbf{g}(n') + \mathbf{h}(s')$ ▷ $\forall \mathbf{h}' \in H(s')$ s.t. $\mathbf{h}(s(n)) \preceq \mathbf{c}(s(n),s') + \mathbf{h}'$
21: **if** $G^{\mathrm{Tr}}_{\mathrm{cl}}(s') \preceq \mathrm{Tr}(\mathbf{g}(n'))$ **then** ▷ $G^{\mathrm{Tr}}_{\mathrm{cl}}(s',\mathbf{h}')$
22: **continue**
23: **if** $G^{\mathrm{Tr}}_{\mathrm{cl}}(s_{\mathrm{goal}}) \preceq \mathrm{Tr}(\mathbf{f}(n'))$ **then**
24: **continue**
25: OPEN.Insert(n') ▷ ordered lex. according to $\mathbf{f}$-value
26: **return** Sols

[1]Here 'op' and 'cl' correspond to open and closed, respectively.

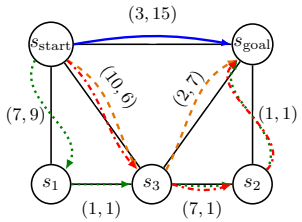


Figure 2: Search graph.

State	$H(s)$
s_{start}	{(2,14), (11,12), (15,11), (17,7)}
s_1	{(2,7), (8,2)}
s_2	{(1,1)}
s_3	{(1,6), (7,1)}
s_{goal}	{(0,0)}

Table 1. MVH H values.

traction, its truncated $\mathbf{g}$-value is inserted into $G_{\text{cl}}^{\text{Tr}}$ (Line 13), ensuring that subsequent dominance tests can be done using $N-1$ dimensions. This same t-discarding logic is applied to filter new generated successor nodes (Lines 21 and 23).

The Incompatibility of MVH and DR

In the previous section, we established that DR via t-discarding relies entirely on Property 1: nodes corresponding to the same state must be extracted in lexicographically non-decreasing order of their $\mathbf{g}$-values. In the context of NAMOA*dr, this ordering invariant is naturally guaranteed when using a consistent SVH.

However, this invariant collapses when introducing MVHs. When a state s is associated with a set of heuristic estimates $H(s)$, two nodes n and n' of the same state can be evaluated using different heuristic vectors, $\mathbf{h} \in H(s)$ and $\mathbf{h}' \in H(s)$. As we will see shortly, in certain settings, $\mathbf{f}(n) \prec_{\text{lex}} \mathbf{f}(n')$ even if $\mathbf{g}(n) \succ_{\text{lex}} \mathbf{g}(n')$. When extracting nodes by their $\mathbf{f}$-values, a node with a larger first-objective cost g_1 might be extracted before a node with a smaller g_1 cost, violating monotonic growth in the first dimension.

Lemma 1. *Property 1 does not hold generally for best-first MOS algorithms employing an MVH H, even if H is admissible and consistent.*

Proof. We prove this by counter-example. Consider the bi-objective search graph shown in Fig. 2. The search query is $s_{\text{start}} \rightarrow s_{\text{goal}}$, and the Pareto-optimal solution set Π^* consists of four solution paths (colored in Fig. 2, also shown in Fig. 1): π_1, π_2, π_3 and π_4 with costs $(3, 15), (12, 13), (16, 12)$ and $(18, 8)$, respectively.

Table 1 specifies an admissible and consistent MVH H for this query (also shown in Fig. 1). We execute a naive adaptation of NAMOA*dr using this MVH. Node entries in OPEN are represented as $n = (s, \mathbf{g}, F(s, \mathbf{g}))$, where $F(s, \mathbf{g}) = \text{Nd}\{\mathbf{g}+\mathbf{h} \mid \mathbf{h} \in H(s)\}$. The node selected for extraction has the lexicographically-smallest $\mathbf{f}$-value among all available $\mathbf{f} \in F$ in OPEN. For DR, we maintain a truncated closed-set $G_{\text{cl}}^{\text{Tr}}(s)$ for each state. As this is a bi-objective graph, truncated vectors are scalars (as they contain only the second cost component).

Table 2 traces the execution. A node trace is denoted as $\langle s, \mathbf{g}, \mathbf{h}, \mathbf{f}_{\text{min lex}} \rangle$.

- **Iteration 0:** Extracts $\langle s_{\text{start}}, (0, 0), (2, 14), (2, 14) \rangle$. We initialize $G_{\text{cl}}^{\text{Tr}}(s_{\text{start}}) \leftarrow 0$. Successors $(s_{\text{goal}}, s_1, s_3)$ are generated and inserted into OPEN.
- **Iteration 1:** Extracts $\langle s_{\text{goal}}, (3, 15), (0, 0), (2, 14) \rangle$. $G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}})$ updates to 15.
- **Iteration 2:** Extracts $\langle s_1, (7, 9), (2, 7), (9, 16) \rangle$. Since $\text{Tr}((9, 16)) = 16 \succ 15$ (the value of $G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}})$), the $\mathbf{f}$-value $(9, 16)$ is pruned. The next lexicographically-smallest $\mathbf{f} \in F(s_1, (7, 9))$ is $(15, 11)$, created with $\mathbf{h} = (8, 2)$. The updated node remains in OPEN as $\langle s_1, (7, 9), (8, 2), (15, 11) \rangle$.
- **Iteration 3:** Extracts $\langle s_3, (10, 6), (1, 6), (11, 12) \rangle$. $G_{\text{cl}}^{\text{Tr}}(s_3)$ updates to 6. Successors (s_{goal}, s_2) are generated.
- **Iteration 4:** Extracts $\langle s_{\text{goal}}, (12, 13), (0, 0), (12, 13) \rangle$. $G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}})$ updates to 13.
- **Iteration 5:** Re-extracts the updated node $\langle s_1, (7, 9), (8, 2), (15, 11) \rangle$. It generates a successor s_3 with cost $\mathbf{g} = (8, 10)$. The algorithm checks if $\text{Tr}(8, 10) = 10 \prec 6$ (the value of $G_{\text{cl}}^{\text{Tr}}(s_3)$). As it is not, the algorithm incorrectly assumes the path is dominated and prunes it.

It.	OPEN candidates $\langle s, \mathbf{g}, \mathbf{h}, \mathbf{f} \rangle$	DR action
0	$\rightarrow \langle s_{\text{start}}, (0, 0), (2, 14), (2, 14) \rangle$	$G_{\text{cl}}^{\text{Tr}}(s_{\text{start}}) \leftarrow 0$
1	$\rightarrow \langle s_{\text{goal}}, (3, 15), (0, 0), (2, 14) \rangle$ $\langle s_1, (7, 9), (2, 7), (9, 16) \rangle$ $\langle s_3, (10, 6), (1, 6), (11, 12) \rangle$	$G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}}) \leftarrow 15$
2	$\rightarrow \langle s_1, (7, 9), (2, 7), (9, 16) \rangle$ $\langle s_3, (10, 6), (1, 6), (11, 12) \rangle$	$F(s_1, (7, 9)) \leftarrow$ $F(s_1, (7, 9)) \setminus \{(9, 16)\}$
3	$\rightarrow \langle s_3, (10, 6), (1, 6), (11, 12) \rangle$ $\langle s_1, (7, 9), (8, 2), (15, 11) \rangle$	$G_{\text{cl}}^{\text{Tr}}(s_3) \leftarrow 6$
4	$\rightarrow \langle s_{\text{goal}}, (12, 13), (0, 0), (12, 13) \rangle$ $\langle s_1, (7, 9), (8, 2), (15, 11) \rangle$ $\langle s_2, (17, 7), (1, 1), (18, 8) \rangle$	$G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}}) \leftarrow 13$
5	$\rightarrow \langle s_1, (7, 9), (8, 2), (15, 11) \rangle$ $\langle s_2, (17, 7), (1, 1), (18, 8) \rangle$	node is pruned, $G_{\text{cl}}^{\text{Tr}}(s_1) \prec \text{Tr}(8, 10)$

Table 2. Execution trace of naively combining NAMOA*dr with an admissible and consistent MVH on Figure 2.

At Iteration 5, the true full cost vector of the previously closed s_3 node is $(10, 6)$. The new $\mathbf{g}$-value is $(8, 10)$. Clearly, $(10, 6)$ does *not* weakly dominate $(8, 10)$. The pruning is an artifact of the truncated comparison, which assumes $g_1 \geq 10$. However, the new node's $g_1 = 8 < 10$. This proves that lexicographic extraction of $\mathbf{f}$-values under an MVH does not guarantee lexicographic expansion of $\mathbf{g}$-values, breaking Property 1. □

Because the t-discarding framework relies entirely on the invariant established by Property 1, applying it without modification under an MVH causes the search to incorrectly prune Pareto-optimal paths (as demonstrated in Iteration 5), breaking completeness and optimality. Preserving correctness therefore requires re-establishing suitable ordering invariants, which motivates the design of the safe search frameworks presented in the following sections.

NAMOA*dr-mvh

As proven in Sec. , the variability of heuristic estimates within an MVH destroys the ordering invariant required for DR. However, if we group nodes not merely by their state s, but by the specific state-heuristic pair $(s, \mathbf{h})$, we can theoretically isolate the variance and restore the invariant.

To achieve this, we can force an MVH to behave locally like an SVH. When a node n expands state s, instead of evaluating the successor s' with the entire set $H(s')$, the algorithm must generate a *distinct* successor node for every valid $\mathbf{h}' \in H(s')$. By assigning a separate search node for each heuristic value, the search space explodes into a Cartesian product of all non-dominated $\mathbf{g}$-values and all available $\mathbf{h}$-values for each state.

Unfortunately, splitting the nodes is not enough and we must also ensure that the sequence of heuristic estimates chosen along a path preserves the monotonic growth of evaluation vectors. To recover the monotonic growth required for DR, the search must enforce *path-consistent heuristic selection* (Due to lack of space, the formal definition is available in Wolff, Felner, and Salzman (2026)). This means that when a node n generates a successor n', it can only bind to an $\mathbf{h}' \in H(s')$ that satisfies the consistency triangle inequality: $\mathbf{h}(n) \preceq \mathbf{c}(s, s') + \mathbf{h}'$.

If the MVH is consistent, and the algorithm enforces this path-consistent selection, we can formally guarantee that t-discarding becomes safe again.

Theorem 1 (Restored $\mathbf{g}$-value ordering)**.** *Assume H is a consistent MVH and the algorithm employs path-consistent heuristic selection. Let n and n' be two extracted nodes that share the same state s and the exact same selected heuristic vector $\mathbf{h}$. If n is extracted before n', then $\mathbf{g}(n) \leq_{\text{lex}} \mathbf{g}(n')$.*

Proof is provided in Wolff, Felner, and Salzman (2026).

Limitations of NAMOA*dr-mvh. Thm. 1 implies that t-discarding remains sound if the algorithm maintains a separate local truncated closed-set $G_{\text{cl}}^{Tr}(s, \mathbf{h})$ for each valid $\mathbf{h} \in H(s)$ (as shown in the red annotations of Algorithm 1). However, while theoretically sound, this framework is practically crippled by three major limitations:

- **L1 State-Space Explosion:** As established, generating a successor for every consistent $\mathbf{h}'$ forces the algorithm to generate an overwhelming number of redundant nodes representing the exact same physical path, severely cluttering OPEN.
- **L2 Fragmentation of Pruning Power:** By partitioning the closed-sets by $\mathbf{h}$, the algorithm performs dominance checks against much smaller sets, drastically reducing the likelihood of successful t-discarding.
- **L3 The Consistency Burden:** Constructing tight, strictly consistent MVHs is both difficult and computationally expensive (see also Sec.).

We therefore seek an alternative framework that avoids the Cartesian product explosion entirely, preserves the efficiency of sound DR, and only requires *admissible* MVHs.

L-NAMOA*dr-mvh: Lazy DR

We now present our primary algorithmic contribution, L-NAMOA*dr-mvh (where the "L" explicitly stands for *Lazy*). This framework enables the sound use of DR together with MVHs while dropping the consistency requirement. It requires only the *admissibility* of the MVH.

The key idea is to use DR optimistically: assume the required ordering invariants hold, dynamically detect when they are violated, and fall back to a full Pareto dominance test only when absolutely necessary. Similar to NAMOA*dr, the algorithm maintains a priority queue OPEN sorted by lexicographically-smallest $\mathbf{f}$-values. However, it maintains *two* closed-sets for every state s: (i) a truncated closed-set $G_{\text{cl}}^{\text{Tr}}(s)$ for fast t-discarding, and (ii) a full non-dominated set $G_{\text{cl}}(s)$ of discovered $\mathbf{g}$-values, serving as a correctness-critical fallback.

Algorithmic Mechanics: The Node Lifecycle

To understand how L-NAMOA*dr-mvh outlined in Alg. 2, overcomes the state-space explosion (Sec. , **L1**), we trace the lifecycle of a node as it is generated, extracted, and potentially re-evaluated.

Generation & Lazy Binding (CHOOSEH). Instead of eager node-splitting (which creates a Cartesian product of $\mathbf{g}$-values and $\mathbf{h}$-values), L-NAMOA*dr-mvh generates *one* search node $n = (s, \mathbf{g}, \mathbf{f})$ per physical path. When a successor is generated, it is evaluated *lazily*. The CHOOSEH function (Line 34) scans $H(s)$ and binds the path to the lexicographically-smallest $\mathbf{h}$-value that is not already t-discarded by the current known solutions. This collapses the combinatorial explosion back to a single node in OPEN.

Extraction & Lazy Re-evaluation. If an extracted node is later found to be globally dominated by a newly discovered solution, it is not immediately discarded. Instead, the algorithm performs a *lazy re-evaluation* (Line 9): it calls CHOOSEH again to find the *next* best non-dominated $\mathbf{h}$-value and re-inserts the updated node into OPEN. This simulates exploring all valid heuristics without inserting multiple copies of the same path into the queue.

Optimistic DR & Fallback (LOCALDOMCHECK). As L-NAMOA*dr-mvh does not require a consistent MVH, the first objective is not guaranteed to grow monotonically. Lexicographic extraction of $\mathbf{f}$-values might expand nodes out of $\mathbf{g}$-value order, making naive t-discarding unsafe.

To resolve this, the LOCALDOMCHECK function (Line 41) first optimistically checks if $\text{Tr}(\mathbf{g})$ is dominated by $G_{\text{cl}}^{\text{Tr}}(s)$. If it is, the algorithm validates the invariant: it checks whether the new node's g_1 is greater than or equal to the maximum g_1 seen so far for state s. If $g_1 \geq \max g_1'$, the monotonic growth assumption holds locally, t-discarding is sound, and the node is safely pruned. However, if $g_1 < \max g_1'$, a sequence violation is detected. The algorithm falls back to a standard, full Pareto dominance check against $G_{\text{cl}}(s)$, pruning the node only if it is truly dominated.

Algorithm 2 L-NAMOA*dr-mvh

Input: A MOS instance $(S, E, \mathbf{c}, s_{\text{start}}, s_{\text{goal}})$; an admissible MVH H.
Output: A cost-unique Pareto-optimal solution set Sols.

Sols $\leftarrow \emptyset$
for all $s \in S$ **do**
  $G_{\text{cl}}^{\text{Tr}}(s) \leftarrow \emptyset; G_{\text{cl}}(s) \leftarrow \emptyset$
$n \leftarrow$ new node with $s(n) = s_{\text{start}}$
parent$(n) \leftarrow$ NULL; $\mathbf{g}(n) \leftarrow \mathbf{0}$; $\mathbf{f}(n) \leftarrow H_{lex}^{\min}(s_{\text{start}})$
OPEN $\leftarrow \{n\}$
while OPEN $\neq \emptyset$ **do**
  $n \leftarrow$ OPEN.Pop
  if $\exists\, \mathbf{f}_{\text{goal}} \in G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}})\ s.t.\ \mathbf{f}_{\text{goal}} \preceq \text{Tr}(\mathbf{f}(n))$ **then**
    $\mathbf{h}' \leftarrow$ CHOOSEH$(s(n), \mathbf{g}(n), G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}}))$
    if $h' \neq \perp$ **then**
      $\mathbf{h}(n) \leftarrow \mathbf{h}'$;
      $\mathbf{f}(n) \leftarrow \mathbf{g}(n) + \mathbf{h}'$
      OPEN.Insert(n)
    continue
  if LOCALDOMCHECK$(s(n), \mathbf{g}(n))$ **then**
    continue
  $G_{\text{cl}}^{\text{Tr}}(s(n)) \leftarrow \big(G_{\text{cl}}^{\text{Tr}}(s(n)) \cup \{\text{Tr}(\mathbf{g}(n))\}\big)$;
  $G_{\text{cl}}(s(n)) \leftarrow G_{\text{cl}}(s(n)) \cup \{\mathbf{g}(n)\}$
  if $s(n) = s_{\text{goal}}$ **then**
    Sols $\leftarrow$ Sols $\cup \{n\}$
    continue
  for all $s' \in \text{Succ}(s)$ **do**
    $n' \leftarrow$ a new node with $s(n') = s'$
    parent$(n') \leftarrow n$; $\mathbf{g}(n') \leftarrow \mathbf{g}(n) + \mathbf{c}(s(n), s')$
    $\mathbf{h}' \leftarrow$ CHOOSEH$(s', \mathbf{g}(n'), G_{\text{cl}}^{\text{Tr}}(s_{\text{goal}}))$
    if $\mathbf{h}' = \perp$ **then**
      continue
    if LOCALDOMCHECK$(s', \mathbf{g}(n'))$ **then**
      continue
    $\mathbf{f}(n') \leftarrow \mathbf{g}(n') + \mathbf{h}'$
    OPEN.Insert(n')
return Sols

function CHOOSEH$(s, \mathbf{g}, T)$
  for all $\mathbf{h} \in H(s)$ in lexicographic order **do**
    $\mathbf{f} \leftarrow \mathbf{g} + \mathbf{h}$
    if $T \preceq \text{Tr}(\mathbf{f})$ **then** ▷ t-discarding
      continue
    return $\mathbf{h}$
  return $\perp$

function LOCALDOMCHECK$(s, \mathbf{g})$
  if $\exists\, \mathbf{g}' \in G_{\text{cl}}^{\text{Tr}}(s) : \text{Tr}(\mathbf{g}') \preceq \text{Tr}(\mathbf{g})$ **then**
    if $\max_{\mathbf{g}' \in G_{\text{cl}}^{\text{Tr}}(s)} g_1' \leq g_1$ **then** ▷ t-discarding
      return true
    if $\exists\, \mathbf{g}' \in G_{\text{cl}}(s) : \mathbf{g}' \preceq \mathbf{g}$ **then**
      return true
  return false

Theoretical Guarantees

Correctness of L-NAMOA*dr-mvh relies on the soundness of these lazy evaluations and fallback mechanisms. As CHOOSEH only discards heuristic evaluations that are globally pruned by known solutions, and LOCALDOMCHECK detects local ordering violations using full dominance checks, the algorithm never incorrectly prunes a Pareto-optimal path.

Theorem 2 (Correctness of L-NAMOA*dr-mvh)**.** *If H is an admissible MVH then L-NAMOA*dr-mvh returns the complete Pareto-optimal solution set Π^*.*

Formal proofs for the correctness of the CHOOSEH *and* LOCALDOMCHECK *procedures, as well as the full proof of Thm. 2, are provided in Wolff, Felner, and Salzman (2026).*

While inconsistent heuristics may trigger repeated full Pareto fallbacks (adding subsequently dominated cost vectors to $G_{\text{cl}}(s)$), we demonstrate in Sec. that these effects remain limited in practice, allowing L-NAMOA*dr-mvh to achieve dramatic speedups when the MVH provides strong search guidance.

Construction of Multi-Valued Heuristics

In this section we describe an approach to generate admissible MVHs as well as how to modify them into consistent MVHs. Recent work by Geißer et al. (2022) investigated the construction of admissible and consistent MVHs in the context of domain-independent planning. They extended several classes of classical planning heuristics to the MOS setting. However, while their work provides theoretical insights, these constructions rely heavily on the structural properties of factored planning domains (such as STRIPS representations and delete-relaxation reasoning) and are not directly applicable to general graph-based MOS problems.

To evaluate our algorithms, we require domain-agnostic MVHs. We achieve this through a pre-processing phase that performs an approximate backward MOS from the goal.

A*pex-MVH (Admissible MVHs). To generate an admissible MVH, we use an adaptation of the A*pex algorithm (Zhang et al. 2022). A*pex is an approximate MOS algorithm designed to efficiently compute an ε-approximation of the exact Pareto-optimal solution set. It achieves this by grouping similar paths and representing each group using a single multi-dimensional vector called an *apex*.

An apex is constructed by taking the component-wise minimum cost across all paths within its respective group. Consequently, an apex is guaranteed to weakly dominate the cost vector of every actual path it represents. By executing A*pex as a backward search from s_{goal} with an ε-approximation factor, we compute a set of mutually non-dominated apexes for every state s. Because each apex weakly dominates the true path costs to the goal, this resulting set $H(s)$ serves as an admissible MVH.

Consistency-Fixed MVHs. Unfortunately, the A*pex-MVH approach is not guaranteesd to be consistent. Thus, we apply a *consistency-fix* procedure modeled after Bellman-Ford label updates to create consistent MVHs based on existing admissible MVHs to use in NAMOA*dr-mvh.

Intuitively, as long as there is an inconsistency in a state's heuristic set, this procedure lowers the violating heuristic values to resolve it. Specifically, if there exist states s, s' and

a heuristic vector $\mathbf{h}' \in H(s')$ such that $\mathbf{c}(s, s') + \mathbf{h}' \prec \mathbf{h}$ for some existing estimate $\mathbf{h} \in H(s)$, the estimate $\mathbf{h}$ violates consistency. We resolve this by inserting the smaller vector $\mathbf{c}(s, s') + \mathbf{h}'$ into $H(s)$ and discarding any newly-dominated values in $H(s)$, including the original $\mathbf{h}$.

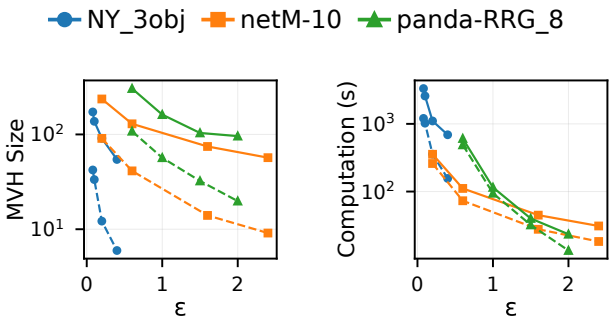
This refinement propagates backward through the graph until a fixed point is reached and all local inconsistencies are resolved. Because this procedure only lowers bounds that were originally admissible, the resulting MVH remains admissible and becomes, by construction, consistent. However, the procedure tends to result in more complex, less-informed MVH, as we will see in Section .

Evaluation

In this section we evaluate our two new algorithms NAMOA*dr-mvh and L-NAMOA*dr-mvh, comparing them to the baselines NAMOA* and NAMOA*dr. Experiments were run on three different benchmarks (`netM-10` `NY-3obj` and `panda-RRG-8` which have three, three and eight objectives, respectively).[2] All experiments were conducted on an AWS computing cluster, with 32 GB of memory and runtime limit of 7200s per query instance (except `netM10` where we used 3600s), and executed in a single-threaded CPU configuration on Intel Xeon processors (2.1–2.4 GHz). We implemented all algorithms in C++, using a common codebase as much as possible[3]. For each benchmark we compared 50 different search queries. We used NAMOA*dr where the SVH is the ideal-point heuristic SVH $\mathbf{h}^{\text{IP}}$ (Salzman et al. 2026)[4].

We start by outlining the properties of the MVH we construct. We then continue with a comparison of the different algorithms across the benchmarks and conclude by pinpointing the source of the efficiency of L-NAMOA*dr-mvh, i.e., how effective is the lazy approach we employ.

MVH construction. To obtain admissible and consistent MVHs, we follow the approach described in Sec. . Specifically, we run A*pex using different approximation factors ε to obtain the so-called "A*pex-MVH" and then apply the consistency-fixed procedure.[5] We refer to these MVHs as (A)-type and (C)-type, respectively. Fig. 3 reports the MVH size and computation time as a function of ε. As expected, both size and computation time decrease as ε increases. However, as we will see shortly, the extra running time invested in creating larger, more informed MVHs will allow, when the MVH provides strong search guidance, a larger speedup across multiple queries.



(a) Average MVH size as a function of ϵ.

(b) Average computation time as a function of ϵ.

Figure 3: MVH size (a) and MVH computation time (b) as a function of ε for each benchmark. Solid and dashed lines correspond to (C)-type and (A)-type MVHs, respectively.

Algorithms comparison. To compare the different algorithms, we start (Fig. 4) by plotting the runtime of each algorithm (for NAMOA* and L-NAMOA*dr-mvh we use both (A)-type and (C)-type) as a function of the MVH size. For `netM-10`, we see that as the MVH size increases, the obtainable speedup also increases, with a clear advantage for the larger, admissible-only (A)-type MVH over the more complex (C)-type. This suggests that `netM-10` is an environment where stronger heuristic guidance can successfully narrow the search space without being overwhelmed by objective correlation or density. For `NY-3obj`, there is no speedup. We attribute this to the fact that this benchmark includes highly-correlated objectives (Halle et al. 2025). The high correlation implies that the dominance relationship is more frequently satisfied, resulting in sparse Pareto sets that effectively collapse the search space toward a lower-dimensional frontier where pruning using regular SVH (like NAMOA*dr with $\mathbf{h}^{\text{IP}}$) remains highly efficient. Thus, the additional guidance of an MVH has little room to improve over the already effective SVH-based pruning. Finally for `panda-RRG-8`, we see a slight speedup for smaller MVHs. We suggest that the topological density of `panda-RRG-8` generates a Pareto frontier with high cardinality, so that the objective space becomes effectively saturated with non-dominated solutions. This density dilutes the pruning power of any heuristic, including MVHs. A relatively small MVH or a SVH is able to cover most of the different trade-offs just by exploring the graph. Together, these results suggest that the benefit of L-NAMOA*dr-mvh depends on whether the additional MVH guidance can translate into meaningful pruning. This happens clearly in `netM-10`, while in `NY-3obj` and `panda-RRG-8` the baseline pruning is already competitive, leaving less room for improvement.

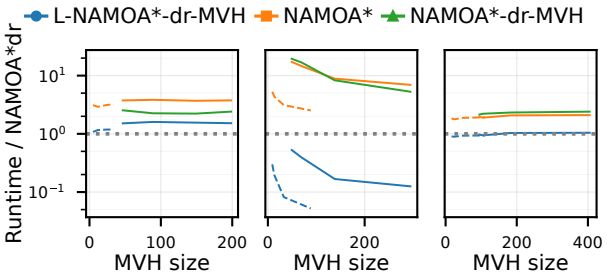
We continue in Fig. 6 to illustrate the possible speedup for each algorithm variant, together with their number of generated and expanded nodes. Every algorithm is selected with

[2] https://github.com/CRL-Technion/Multi-Objective-Search-Benchmarks.git.

[3] https://github.com/CRL-Technion/bridging-mvh-dr.

[4] $\mathbf{h}^{\text{IP}}$ combines a set of N single-objective heuristics $h_1, \ldots, h_N$. Here, $h_i : S \to \mathbb{R}_{\geq 0}$ corresponds to the shortest path from each state according to the i'th objective and $\forall s \in S$, $\mathbf{h}^{\text{IP}}(s) := (h_1(s), \ldots, h_N(s))$. The ideal point heuristic, which is admissible, is easily computed by running N (single-objective) instances of Dijkstra's algorithm starting from s_{goal} (i.e., one instance for each objective).

[5] We selected 4 ε-approximation factors. For `netM-10` we used $\varepsilon \in \{0.2, 0.6, 1.6, 2.4\}$ For `NY-3obj` we used $\varepsilon \in \{0.08, 0.1, 0.2, 0.4\}$; For `panda-RRG-8` we used $\varepsilon \in \{0.6, 1, 1.5, 2\}$.



(a) NY-3obj (b) netM-10 (c) panda-RRG-8

Figure 4: Runtime relative to NAMOA*dr as a function of MVH size for each algorithm. Solid and dashed lines correspond to (C)-type and (A)-type MVHs, respectively. NY-3obj: 264,346 nodes, 733,846 edges, 3 objectives, netM-10: 10,000 nodes, 59,743 edges, 3 objectives, panda-RRG-8: 1,000 nodes, 10,260 edges, 8 objectives.

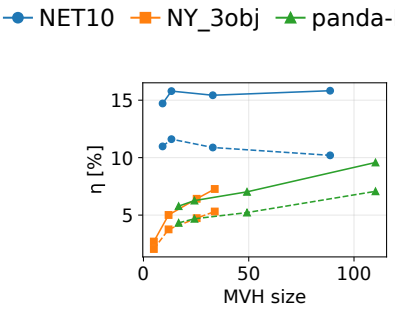
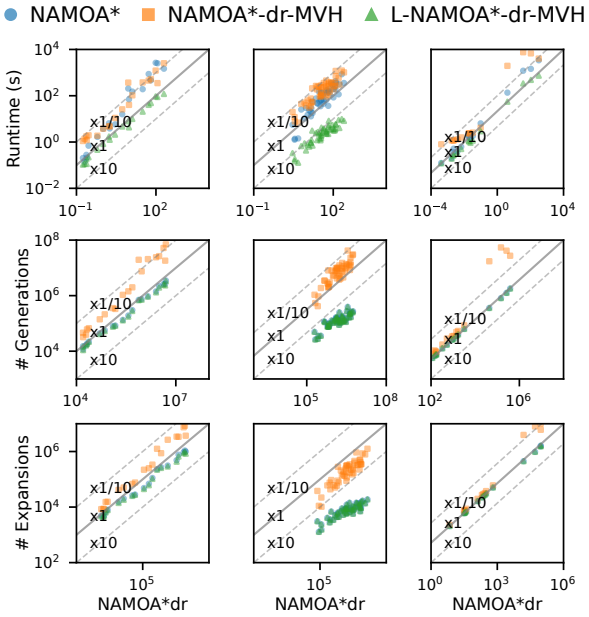


Figure 5: Frequency η, measured in percentage for which L-NAMOA*dr-mvh performs full dominance checks out of all local (solid line) and global (dashed line) dominance checks, as a function of (A)-type MVHs.

its best-performing MVH. L-NAMOA*dr-mvh provides a consistent speedup or similar runtime to NAMOA*dr, often exceeding one order of magnitude ($> 10\times$) in the netM-10 benchmark. In contrast, NAMOA* and NAMOA*dr-mvh perform significantly slower, emphasizing the necessity of DR optimizations upon its lack or relaxed adaptation.

Also in Fig 6, we can see that the operation counts closely align with the runtime results. In particular, NAMOA*dr-mvh, which creates separate nodes for different heuristic values, causes a state-space blow-up. By contrast, L-NAMOA*dr-mvh's generated and expanded node counts are exactly the same as those of NAMOA*, so the corresponding marks collapse together. This shows that L-NAMOA*dr-mvh preserves the advantage of MVH guidance while avoiding the cost of eagerly materializing all heuristic bindings.

Effectiveness of Laziness. Figure 5 illustrates the frequency for which L-NAMOA*dr-mvh requires performing full dominance check instead of using DR. The very low percentages across all cases (16% at most) highlight the efficiency of applying lazy DR dominance checks, when only occasionally falling back to doing full-dominance check.



(a) NY-3obj (b) netM-10 (c) panda-RRG-8

Figure 6: Per-query comparison against NAMOA*dr across the three benchmarks. Rows show runtime, node generations, and node expansions; columns correspond to benchmarks (see Fig. 4). Runtime results use the best-performing MVH for each algorithm. For node generations and expansions, a single MVH is fixed per benchmark. The solid diagonal marks parity with NAMOA*dr, and dashed diagonals mark $10\times$ differences.

Conclusion & Future Work

In this paper, we addressed the fundamental incompatibility between MVHs and DR in MOS. By tracing how heuristic variance destroys the monotonic growth invariant required for safe t-discarding, we established the first theoretical frameworks to restore search correctness. While our baseline (NAMOA*dr-mvh) proved that consistency restores the invariant at the cost of a state-space explosion, our primary contribution, L-NAMOA*dr-mvh, successfully avoids this bottleneck. By falling back to full dominance checks only when local sequence violations are detected, L-NAMOA*dr-mvh preserves the efficiency of DR while exploiting the stronger guidance of MVHs, achieving order-of-magnitude speedups in instances where this guidance translates into stronger pruning.

This work opens several directions for MOS, including the design of domain-independent MVH construction methods that produce tighter, compact admissible sets for general graphs. Additionally, the optimistic, lazy-validation mechanics introduced in L-NAMOA*dr-mvh are highly modular; future work will explore integrating these techniques into other state-of-the-art multi-objective frameworks, to further extend the reach and efficiency of heuristic-guided MOS algorithms.

References

Ahmadi, S.; Sturtevant, N. R.; Harabor, D.; and Jalili, M. 2024. Exact Multi-objective Path Finding with Negative Weights. In *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling*, volume 34, 11–19.

Bronfman, A.; Marianov, V.; Paredes-Belmar, G.; and Lüer-Villagra, A. 2015. The Maximin HAZMAT Routing Problem. *European Journal of Operational Research*, 241(1): 15–27.

Clímaco, J.; and Pascoal, M. 2012. Multicriteria Path and Tree Problems: Discussion on Exact Algorithms and Applications. *International Transactions in Operational Research*, 19(1-2): 63–98.

Current, J.; and Marsh, M. 1993. Multiobjective Transportation Network Design and Routing Problems: Taxonomy and Annotation. *European Journal of Operational Research*, 65(1): 4–19.

Fu, M.; Kuntz, A.; Salzman, O.; and Alterovitz, R. 2023. Asymptotically Optimal Inspection Planning via Efficient Near-Optimal Search on Sampled Roadmaps. *The International Journal of Robotics Research*, 42(4-5): 150–175.

Geißer, F.; Haslum, P.; Thiébaux, S.; and Trevizan, F. 2022. Admissible Heuristics for Multi-Objective Planning. In *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling*, volume 32, 100–109.

Halle, Y.; Felner, A.; Koenig, S.; and Salzman, O. 2025. A Preprocessing Framework for Efficient Approximate Bi-Objective Shortest-Path Computation in the Presence of Correlated Objectives. In *Proceedings of the International Symposium on Combinatorial Search*, volume 18, 65–73.

Hansen, P. 1980. Bicriterion Path Problems. In Fandel, G.; and Gal, T., eds., *Multiple Criteria Decision Making: Theory and Application*, volume 177 of *Lecture Notes in Economics and Mathematical Systems*, 109–127. Berlin, Heidelberg: Springer.

Mandow, L.; and Pérez-de-la Cruz, J. L. 2005. A New Approach to Multiobjective A* Search. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, 218–223.

Mandow, L.; and Pérez de la Cruz, J. L. 2010. Multiobjective A* Search with Consistent Heuristics. *Journal of the ACM*, 57(5): 27:1–27:25.

Pulido, F.-J.; Mandow, L.; and Pérez-de-la Cruz, J.-L. 2015. Dimensionality Reduction in Multiobjective Shortest Path Search. *Computers & Operations Research*, 64: 60–70.

Ren, Z.; Hernández, C.; Likhachev, M.; Felner, A.; Koenig, S.; Salzman, O.; Rathinam, S.; and Choset, H. 2025. EMOA*: A Framework for Search-Based Multi-Objective Path Planning. *Artificial Intelligence*, 339: 104260.

Salzman, O.; Felner, A.; Hernández, C.; Zhang, H.; Chan, S.-H.; and Koenig, S. 2023. Heuristic-Search Approaches for the Multi-Objective Shortest-Path Problem: Progress and Research Opportunities. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 6759–6768.

Salzman, O.; Hernández Ulloa, C.; Felner, A.; and Koenig, S. 2026. Multi-Objective Search: Algorithms, Applications, and Emerging Directions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 40990–40999.

Skriver, A. J. V. 2000. A Classification of Bicriterion Shortest Path (BSP) Algorithms. *Asia-Pacific Journal of Operational Research*, 17(2): 199–212.

Skyler, S.; Shperberg, S. S.; Atzmon, D.; Felner, A.; Salzman, O.; Chan, S.-H.; Zhang, H.; Koenig, S.; Yeoh, W.; and Hernández Ulloa, C. 2024. Theoretical Study on Multi-objective Heuristic Search. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 7021–7028.

Ulungu, E. L.; and Teghem, J. 1991. Multi-objective Shortest Path Problem: A Survey. In *Workshop on Multicriteria Decision Making: Methods–Algorithms–Applications*, 176–188.

Vincke, P. 1976. A New Approach to Multiple Criteria Decision-Making. In *Multiple Criteria Decision Making*, 341–350. Springer.

Wolff, M.; Felner, A.; and Salzman, O. 2026. Bridging Multi-Valued Heuristics and Dimensionality Reduction in Multi-Objective Search. Doi: 10.48550/arXiv.2606.20644, arXiv:2606.20644.

Zhang, H.; Salzman, O.; Felner, A.; Kumar, T. K. S.; Skyler, S.; Hernández Ulloa, C.; and Koenig, S. 2023. Towards Effective Multi-Valued Heuristics for Bi-objective Shortest-Path Algorithms via Differential Heuristics. In *Proceedings of the International Symposium on Combinatorial Search*, volume 16, 101–109.

Zhang, H.; Salzman, O.; Kumar, T. K. S.; Felner, A.; Hernández Ulloa, C.; and Koenig, S. 2022. A*pex: Efficient Approximate Multi-Objective Search on Graphs. In *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling*, volume 32, 394–403.