Journal of Artificial Intelligence Research 78 (2023) 619-639 Submitted 04/2023; published 11/2023

Research Note

Maximisation of Admissible Multi-Objective Heuristics

Patrik Haslum PATRIK.HASLUM@ANU.EDU.AU
Ryan Xiao Wang RYAN.WANG@ANU.EDU.AU
The Australian National University, Canberra, Australia

Abstract

In multi-objective (MO) heuristic search, solution costs, as well as heuristic values, are sets of multi-dimensional cost vectors, representing possible non-dominated trade-offs between objectives. The maximum of two or more such vector sets, which is an important operation in creating informative admissible MO heuristics, can be defined in several ways: Geißer et al. recently proposed two MO maximum operators, the component-wise maximum (comax) and the anti-dominance maximum (admax), which represent different trade-offs between informativeness and computational cost. We show that the anti-dominance maximum is not admissibility-preserving, and propose an alternative, the "select one" maximum (somax). We also show that the comax operator is the greatest admissibility-preserving MO maximum, and briefly investigate its efficient implementation. The conclusion of our experimental results is that somax achieves a trade-off similar to that intended with admax – cheaper to compute but less informed – also when compared to an improved comax implementation.

1. Introduction

A multi-objective (MO) optimisation problem has some number k of objective functions, without an *a priori* specified trade-off between them. The cost of a solution is not a single value but a vector of values, one for each objective, and solutions that achieve better values on different objectives – for example, a schedule that is slower but cheaper vs. one that is faster but more expensive – are incomparable. Therefore, the optimal solution to such a problem is a set of solutions representing all *non-dominated* solution cost vectors, known as the *Pareto cover set* (PCS) (Roijers, Vamplew, Whiteson, & Dazeley, 2013). The multi-objective version of deterministic, discrete sequential decision problems, such as classical planning, can be solved by MO heuristic search algorithms, such as NAMOA* (Mandow & Pérez-de-la-Cruz, 2010), which make use of multi-objective admissible heuristics. In the multi-objective setting, the heuristic value of a state is also a set of (k-dimensional) cost vectors, like the solution set it lowerbounds. Recently, Geißer, Haslum, Thiébaux, and Trevizan (2022) proposed MO generalisations of several families of admissible classical planning heuristics.

In single-objective heuristic search, taking the maximum of heuristic values is an important operation: the maximum of two admissible heuristics is also admissible, and at least as informed as either of the two heuristics. Maximisation plays a key role in effective combination of abstraction heuristics (e.g., Holte, Felner, Newton, Meshulam, & Furcy, 2006; Haslum, Helmert, Bonet, Botea, & Koenig, 2007; Seipp, Keller, & Helmert, 2020), and in the definition of planning heuristics such as $h^{\max}$ and h^m (often known as critical path heuristics). In the multi-objective setting, however, it is not obvious how to define the maximum of two heuristic values, as these are sets of cost vectors that are only partially ordered by the dominance relation. Geißer et al. (2022) proposed two different MO maximum operators, the component-wise maximum (comax) and the anti-dominance maximum (admax), representing different trade-offs between complexity of computing the MO maximum and its informativeness. Empirically, heuristics using comax performed better on the majority of domains and problems in their benchmark suite.

In this paper, we show that the admax operator proposed by Geißer et al. (2022) is in fact not admissibility-preserving. We examine the reason for this flaw, and arrive at a new MO maximum operator, the "select one" maximum (somax), which preserves admissibility. Although somax lacks most other theoretical guarantees, canonical MO PDB heuristics using it have lower total runtime compared to their counterparts using comax on a majority of the suite of benchmark MO planning problems used in Geißer et al.'s (2022) evaluation.

We also investigate the comax operator further, showing it is in fact the strongest possible MO maximum. The potential disadvantage of comax, which motivates the continued search for alternative MO maximum operators, is its computational cost. We show that comax can be computed efficiently (in linear time) in the special bi-objective case, and investigate some improvements to its computation in the general case. Still, in the final experimental comparison, neither of heuristics using the improved comax implementation nor somax consistently outperform the other.

2. Background

We adopt the following statement of the multi-objective planning problem from Geißer et al. (2022): A multi-objective planning (MOP) task is a STRIPS planning task with k cost functions, $c_1, \ldots, c_k$. Each action a is associated with a k-dimensional cost vector, $\vec{c}(a) \in \mathbb{N}^k$, where the ith component, denoted $\vec{c}(a)^i$, is a's contribution to c_i. The cost of a plan $\pi = a_1, \ldots, a_n$ is also a vector, $\vec{c}(\pi) = \sum_{i=1}^{n} \vec{c}(a_i)$, where the sum is taken with vector addition. We apply $\vec{c}$ also to sets of plans: if Π is a set of plans, then $\vec{c}(\Pi) = \{\vec{c}(\pi) \mid \pi \in \Pi\}$ is the set of cost vectors of plans in Π.

Given two cost vectors $\vec{v}_1, \vec{v}_2$, $\vec{v}_1$ *Pareto dominates* $\vec{v}_2$, denoted $\vec{v}_1 \prec \vec{v}_2$, iff for all $i = 1 \ldots k$: $\vec{v}_1^i \leq \vec{v}_2^i$ and $\vec{v}_1 \neq \vec{v}_2$. Dominance is a strict partial order, i.e., it is transitive and asymmetric. Given a set of cost vectors, V, $\mathsf{ND}(V)$ denotes the set of vectors in V that are not dominated by any other in V, i.e., $\mathsf{ND}(V) = \{\vec{v} \in V \mid \nexists \vec{v}' \in V \ \vec{v}' \prec \vec{v}\}$. We say $\vec{v}_1$ *dominates or equals* $\vec{v}_2$, denoted $\vec{v}_1 \preceq \vec{v}_2$, iff $\vec{v}_1 \prec \vec{v}_2$ or $\vec{v}_1 = \vec{v}_2$. A plan π_1 dominates (resp. dominates or equals) plan π_2 if $\vec{c}(\pi_1) \prec \vec{c}(\pi_2)$ (resp. $\vec{c}(\pi_1) \preceq \vec{c}(\pi_2)$). With slight abuse of notation, we use $\mathsf{ND}(\cdot)$ to denote the subset of non-dominated plans in a set of plans as well.

A MOP task can have multiple solution plans whose cost vectors are mutually non-dominating, and not equal, representing different possible trade-offs between the k objectives. The solution to a MOP task is defined as computing a set of plans including one representative of each such non-dominated cost vector; this known as a *Pareto coverage set*. If $\mathcal{T}$ is a MOP task and $\Pi(\mathcal{T})$ the set of all plans for $\mathcal{T}$, a Pareto coverage set, $\mathsf{PCS}(\mathcal{T})$, is a set of plans such that: (*i*) $\mathsf{PCS}(\mathcal{T}) \subseteq \mathsf{ND}(\Pi(\mathcal{T}))$; and (*ii*) $\forall \pi' \in \Pi(\mathcal{T}) \ \exists \pi \in \mathsf{PCS}(\mathcal{T}) \ \vec{c}(\pi) \preceq \vec{c}(\pi')$. An alternative solution concept is to compute the *Pareto front*, $\mathsf{PF}(\mathcal{T}) = \mathsf{ND}(\Pi(\mathcal{T}))$, i.e., the set of all non-dominated plans.

We will use $\mathsf{PCS}(s)$ and $\mathsf{PF}(s)$, where s is any state in $\mathcal{T}$, to denote the Pareto coverage set and Pareto front, respectively, starting from the given state s instead of the initial state of $\mathcal{T}$.

The Pareto coverage set of a task (or state) does not have to be unique, since there may be different plans for each non-dominated cost vector. However, $\vec{c}(\mathsf{PCS}(\mathcal{T}))$ is unique, i.e., whatever representatives are chosen they collectively have the same set of cost vectors. In the special case that $k = 1$, computing the PCS reduces to finding a plan with minimum cost, i.e., classical optimal planning.

3. MO Heuristics

A multi-objective heuristic function maps states of the planning problem to sets of k-dimensional cost vectors. Like a heuristic in the classical, single-objective case, it represents an estimate of the cost of solution plans from the state, but since there can exist several plans, with distinct non-dominated cost vectors, the heuristic estimate is also a set. To compare such cost vector sets, we extend the notion of dominance:

Definition 1. *Let V and U be sets of k-dimensional cost vectors.*
$V \preceq U$ *iff* $\forall \vec{u} \in \mathsf{ND}(U)\ \exists \vec{v} \in V\ \vec{v} \preceq \vec{u}$.
$V \prec U$ *iff* $V \preceq U$ *and* $\exists \vec{u} \in \mathsf{ND}(U)\ \exists \vec{v} \in V\ \vec{v} \prec \vec{u}$.

This definition of domainance between cost vector sets mirrors the defintion of dominance between cost vectors: $V \preceq U$ iff for every cost vector in U there is one in V that is at least as good ($\vec{v} \preceq \vec{u}$), and $V \prec U$ iff in addition there is at least one essential cost vector in U (not dominated by another vector in U) such that there is a strictly better cost vector in V ($\vec{v} \prec \vec{u}$). In Section 3.2 below, we show that dominance is indeed a partial order over a suitably defined set of cost vector sets.

Definition 2. *A MO heuristic H is* admissible *iff $H(s) \preceq \vec{c}(\mathsf{PCS}(s))$ for all states s.*

The definition of admissibility given by Mandow and Pérez-de-la-Cruz (2010), and also used by Geißer et al. (2022), is slightly different: $\forall s\ \forall \pi \in \mathsf{PF}(s)\ \exists \vec{v} \in H(s)\ \vec{v} \preceq \vec{c}(\pi)$, but equivalent since $\vec{c}(\mathsf{PF}(s)) = \vec{c}(\mathsf{PCS}(s))$.

Admissibility of an MO heuristic requires the weak dominance relation ($\preceq$) between the heuristic set and the true cost set, while strict dominance between admissible heuristic sets implies that the dominated set is, in a sense, more informed. This mirrors the case in classcial single-objective heuristics, where admissibility requires $h(s) \leq h^\star(s)$ and $h(s) < h'(s)$ implies that h' is a stronger heuristic than h when both are admissible.

Because dominance is transitive (cf. Section 3.2), $H(s) \preceq \vec{c}(\mathsf{PCS}(s))$ implies $\mathsf{ND}(H(s)) \preceq \vec{c}(\mathsf{PCS}(s))$. That is, we can remove from the heuristic set for any state any dominated cost vectors without compromising admissibility.

Conversely, because admissibility is defined as the existence of some cost vector $\vec{v} \in H(s)$ that dominates or equals each solution cost vector, an admissible heuristic set also remains admissible when enlarged by any set of cost vectors, i.e., if $H(s) \preceq \vec{c}(\mathsf{PCS}(s))$ then $H(s) \cup V \preceq \vec{c}(\mathsf{PCS}(s))$ for any set V; this holds even if V contains cost vectors that do not dominate or equal any solution cost in $\vec{c}(\mathsf{PCS}(s))$.

3.1 Size of the Heuristic Set

In general, there is no relationship between the size of the Pareto cover set and that of an admissible heuristic set. A small (PCS) set can be dominated-or-equalled by a much larger (heuristic) set. For $k \geq 3$, a set of integer cost k-vectors V such that $V \preceq \{\vec{v}\}$, for a given k-vector $\vec{v} > \vec{0}$, and such that V contains no cost vector dominated by another in V, i.e., $V = \mathsf{ND}(V)$, can be arbitrarily large, though finite. Conversely, any cost vector set V is dominated-or-equalled by its so-called ideal point, $\langle \min_{\vec{v} \in V} \vec{v}^1, \ldots, \min_{\vec{v} \in V} \vec{v}^k \rangle$, and therefore by a singleton set.

There are some trivial limits: $m^{(2^n)}$, where m and n are the number of actions and propositions, respectively, in the problem, bounds the number of possible non-redundant plans, and therefore the size of the PCS. This applies also to abstraction heuristics, where the bound is determined by the number of abstract actions and states, respectively. Geißer et al. (2022) gave a similar bound on the size of the heuristic set for MOIP operator counting heuristics.

3.2 Vector Set Dominance

In this section, we show that dominance between cost vector sets has the properties of a partial order.

Lemma 3. *Both weak and strict vector set dominance are transitive.*

Proof. Follows from transitivity of the vector dominance relations. Suppose $V \preceq U \preceq W$, and $\vec{w} \in W$: there must exist $\vec{u} \in U$ such that $\vec{u} \preceq \vec{w}$, and $\vec{v} \in V$ such that $\vec{v} \preceq \vec{u}$; this implies $\vec{v} \preceq \vec{w}$.

Suppose $V \preceq U \prec W$. Because strict dominance implies weak dominance and from the above, we have $V \preceq W$. There exists a $\vec{w} \in \mathsf{ND}(W)$ such that $\vec{u} \prec \vec{w}$ for some $\vec{u} \in U$. We also have $\vec{v} \preceq \vec{u}$ for some $\vec{v} \in V$, and therefore $\vec{v} \prec \vec{w}$. Hence $V \prec W$. The proof that $V \prec U \preceq W$ implies $V \prec W$ is essentially the same. □

Note that $V \preceq U$ is not equivalent to $(V \prec U) \vee (V = U)$: we can have $V \preceq U$ and $U \preceq V$ with $V \neq U$, if V and U contain different dominated cost vectors. However, their subsets of non-dominated cost vectors must be the same:

Lemma 4. *If $V \preceq U$ and $U \preceq V$ then* $\mathsf{ND}(V) = \mathsf{ND}(U)$.

Proof. Let $\vec{v} \in \mathsf{ND}(V)$. $U \preceq V$ implies $\exists \vec{u} \in U$ such that $\vec{u} \preceq \vec{v}$; $V \preceq U$ then implies $\exists \vec{v}' \in V$ such that $\vec{v}' \preceq \vec{u} \preceq \vec{v}$. This means either $\vec{v}' = \vec{u} = \vec{v}$ or $\vec{v}' \prec \vec{v}$, but the latter contradicts that $\vec{v} \in \mathsf{ND}(V)$. Thus $\vec{v} \in U$, and $\mathsf{ND}(V) \subseteq U$. The proof that $\mathsf{ND}(U) \subseteq V$ is symmetric.

Suppose there is some $\vec{v} \in \mathsf{ND}(V)$ such that $\vec{v} \notin \mathsf{ND}(U)$. Since $\mathsf{ND}(V) \subseteq U$, this means $\vec{v}$ is in U but is dominated by another cost vector $\vec{u} \in \mathsf{ND}(U)$. However, since we also have $\mathsf{ND}(U) \subseteq V$, $\vec{u} \in V$, so $\vec{v}$ is dominated also in V, contradicting that $\vec{v} \in \mathsf{ND}(V)$. Thus $\mathsf{ND}(V) \subseteq \mathsf{ND}(U)$ and, symmetrically, $\mathsf{ND}(U) \subseteq \mathsf{ND}(V)$, leading to $\mathsf{ND}(V) = \mathsf{ND}(U)$. □

Corollary 5. *Let $\mathcal{P}_k = 2^{\mathbb{N}^k}$, i.e., the power set of the set of k-dimensional cost vectors, and $\mathcal{P}_k^{\mathsf{ND}} = \{V \in \mathcal{P}_k \mid \mathsf{ND}(V) = V\}$, i.e., $\mathcal{P}_k^{\mathsf{ND}}$ is the subset of vector sets that do not contain any cost vector dominated by another cost vector in the same set. Then weak set dominance is a partial order over $\mathcal{P}_k^{\mathsf{ND}}$.*

In Section 4.1.1 we will show that comax is the least upper bound operator in this partially ordered set. We end this section by showing that strict vector set dominance is asymmetric, and therefore a strict partial order:

Lemma 6. *$V \prec U$ implies $U \not\preceq V$.*

Proof. Since $V \prec U$ there exists $\vec{u} \in \mathsf{ND}(U)$ such that there exists $\vec{v} \in V$ such that $\vec{v} \prec \vec{u}$. Suppose $U \preceq V$: then there must exist $\vec{u}' \in U$ such that $\vec{u}' \preceq \vec{v}$. By transitivity of cost vector dominance, $\vec{u}' \preceq \vec{v} \prec \vec{u}$ implies $\vec{u}' \prec \vec{u}$, contradicting that $\vec{u} \in \mathsf{ND}(U)$. Hence $U \not\preceq V$. □

3.3 NAMOA*

Finally, we briefly describe two properties of the NAMOA* MO heuristic search algorithm, and their implications for the effectiveness of MO heuristics.

NAMOA* processes an queue of "open", as-yet non-dominated, paths in the search space; each path is represented by a tuple $(s, \vec{g}_s, F_s)$, where s is the state at the end of the path, $\vec{g}_s$ the path cost, and $F_s = \{\vec{g}_s + \vec{h} \mid \vec{h} \in H(s)\}$ a set of estimated cost vectors for the completion of this path to a goal state. Paths are taken from the queue, following a priority rule, expanded by appending possible successor transitions, and moved to a closed list. This is analogous to classical single-objective A*.

If the cost $\vec{g}_s$ of a new path found to state s is dominated by the cost of a path to s that has already been expanded (i.e., that is in the closed list), then successors of the new path do not need to be explored, and it is not added to the open queue. Mandow and Pérez-de-la-Cruz (2010) showed that the NAMOA* algorithm, like classical single-objective A*, has the property that if the MO heuristic is consistent, then the algorithm will not find a dominated path before the path that dominates it, and therefore no path once closed will be reopened. Consistency of an MO heuristic was defined by Stewart and White (1991), as follows: H is consistent iff for all states s, t and every non-dominated s–t-path P, $\forall \vec{v} \in H(t)\, \exists \vec{u} \in H(s)\, (\vec{u} \preceq \vec{c}(P) + \vec{v})$ holds. They also showed that consistency holds iff it holds for all paths of length 1, i.e., state transitions.

A path $(s, \vec{g}_s, F_s)$ to a non-goal state s is pruned from the open queue when a set of plans Π has been found such that for each $\vec{f}_s \in F_s$ there exists a plan $\pi \in \Pi$ such that $\vec{c}(\pi) \preceq \vec{f}_s$. (This pruning is applied when the path is generated as well as to paths in the queue when a new non-dominated plan is found.) It follows that if $H(s) \subset H'(s)$, using heuristic H' in place of H can not result in a path through s being pruned earlier (assuming paths are explored in the same order), or, in other words, a heuristic set that is a strict superset can not be more informed. That is not the same as saying a smaller heuristic set is to be preferred; indeed, as shown by Geißer et al. (2022) the so-called ideal point heuristic, which consists of a single cost vector of independent admissible estimates for each objective, is often outperformed by MO heuristics that estimate the Pareto coverage set more accurately with a set of cost vectors. But simply adding more cost vectors to an already admissible heuristic set cannot improve it.

4. MO Maximum Operators

4.1 Component-Wise Maximum (comax)

Let $\vec{v}$ and $\vec{u}$ be cost vectors of equal dimension. We define $\max(\vec{v}, \vec{u}) = \langle \max(\vec{v}^1, \vec{u}^1), \max(\vec{v}^2, \vec{u}^2), \ldots, \max(\vec{v}^k, \vec{u}^k)\rangle$. The component-wise maximum is defined by Geißer et al. (2022) as follows:

Definition 7. *Let V_1 and V_2 be sets of cost vectors of dimension k. The* component-wise maximum *of V_1 and V_2 is* $\mathrm{comax}(V_1, V_2) = \mathsf{ND}(\{\max(\vec{v}_1, \vec{v}_2) \mid \vec{v}_1 \in V_1, \vec{v}_2 \in V_2\})$.

The following proposition summarises properties of comax that were stated by Geißer et al. (2022):

Proposition 8. *Let H_1 and H_2 be MO heuristics.*

(i) If H_1 and H_2 are admissible, so is $H(s) = \mathrm{comax}(H_1(s), H_2(s))$.

(ii) If H_1 and H_2 are consistent, so is $H(s) = \mathrm{comax}(H_1(s), H_2(s))$.

(iii) comax *is commutatitve and associative.*

(iv) $V_1 \preceq \mathrm{comax}(V_1, V_2)$ and $V_2 \preceq \mathrm{comax}(V_1, V_2)$.

Proof. Proofs of (i) and (ii) were given by Geißer et al. (2022). Note that (i) also follows from Proposition 9 below, since admissibility implies $\vec{c}(\mathsf{PCS}(s))$ is an upper bound on $H_1(s)$ and $H_2(s)$.
(iii) Follows from commutativity and associativity of $\max$.
(iv) Consider $\vec{u} \in \mathrm{comax}(V_1, V_2)$: $\vec{u}$ is the result of combining, by coordinate-wise maximum, vectors $\vec{v}_1 \in V_1$ and $\vec{v}_2 \in V_2$. Since $\vec{u}^i = \max(\vec{v}_1^i, \vec{v}_2^i)$ we have $\vec{v}_1^i \leq \vec{u}^i$ and $\vec{v}_2^i \leq \vec{u}^i$, for $i = 1, \ldots, k$. Hence $\vec{v}_1 \preceq \vec{u}$ and $\vec{v}_2 \preceq \vec{u}$. □

4.1.1 comax IS THE GREATEST ADMISSIBLE MAXIMUM

Recall from Corollary 5 that $\mathcal{P}_k^{\mathsf{ND}}$ is the set of all sets of k-dimensional cost vectors that contain no dominated cost vector, and that this set is partially ordered by weak set dominance.

Proposition 9. comax *is the least upper bound of any pair of elements in* $\mathcal{P}_k^{\mathsf{ND}}$.

Proof. Note that comax by definition excludes dominated cost vectors; thus, $\mathrm{comax}(V, U) \in \mathcal{P}_k^{\mathsf{ND}}$ for any $V, U \in \mathcal{P}_k^{\mathsf{ND}}$. That comax is an upper bound was shown by Proposition 8(iv). Let $W \in \mathcal{P}_k^{\mathsf{ND}}$ be any upper bound on V and U, i.e., such that $V \preceq W$ and $U \preceq W$, and let $\vec{w} \in W$. Since $V \preceq W$ and $U \preceq W$ there exists $\vec{v} \in V$ and $\vec{u} \in U$ such that $\vec{v} \preceq \vec{w}$ and $\vec{u} \preceq \vec{w}$. Consider the ith objective: Since $\vec{v} \preceq \vec{w}$ and $\vec{u} \preceq \vec{w}$, we have $\vec{v}^i \leq \vec{w}^i$ and $\vec{u}^i \leq \vec{w}^i$, and therefore $\max(\vec{v}^i, \vec{u}^i) \leq \vec{w}^i$. Since this holds for all k objectives, $\max(\vec{v}, \vec{u}) \preceq \vec{w}$. Either $\max(\vec{v}, \vec{u}) \in \mathrm{comax}(V, U)$ or there is another $\vec{x} \in \mathrm{comax}(V, U)$ such that $\vec{x} \preceq \max(\vec{v}, \vec{u})$; in either case, there is a cost vector in $\mathrm{comax}(V, U)$ that dominates or equals $\vec{w}$. Hence $\mathrm{comax}(V, U) \preceq W$. □

If H_1 and H_2 are admissible MO heuristics, then for any state s $H_1(s) \preceq \vec{c}(\mathsf{PCS}(s))$ and $H_2(s) \preceq \vec{c}(\mathsf{PCS}(s))$, i.e., $\vec{c}(\mathsf{PCS}(s))$ is an upper bound on $H_1(s)$ and $H_2(s)$. Given only this information, it is possible that $\vec{c}(\mathsf{PCS}(s))$ is also equal to the least upper bound on $H_1(s)$ and $H_2(s)$, i.e., equal to $\mathrm{comax}(H_1(s), H_2(s))$. Therefore, no MO maximum operator that is strictly greater than comax can be guaranteed to preserve admissibility.

Corollary 10. *Let H_1 and H_2 be admissible MO heuristics, and* $\mathrm{momax}' : \mathcal{P}_k \times \mathcal{P}_k \to \mathcal{P}_k$ *any binary operator on cost vector sets. If* $\mathrm{comax}(H_1(s), H_2(s)) \prec \mathrm{momax}'(H_1(s), H_2(s))$*, then* $\mathrm{momax}'(H_1(s), H_2(s))$ *cannot be guaranteed to be admissible.*

Given the superiority of comax, why search for alternative MO maximum operators? The potential disadvantage of comax is its computational cost: $|\mathrm{comax}(V_1, V_2)|$ can be as large as the product of $|V_1|$ and $|V_2|$. A simple example in $k = 4$ dimensions is $V_1 = \{\langle i, n - i, 0, 0\rangle \mid i = 1, \ldots, n - 1\}$ and $V_2 = \{\langle 0, 0, i, n - i\rangle \mid i = 1, \ldots, n - 1\}$. All cost vectors are mutually non-dominating in both sets, and the maximum contains all n^2 cost vectors of the form $\langle i, n - i, j, n - j\rangle$, with $1 \leq i, j < n$, none of which dominate each other. For the special *bi-objective* case, we can show the following:

Proposition 11. *If $k = 2$, then* $|\mathrm{comax}(V_1, V_2)| \leq |V_1| + |V_2|$.

The proof of this proposition is in Appendix A. What is the worst case size bound when $k = 3$ is an open question.

4.1.2 COMPUTING comax EFFICIENTLY

A naive implementation of comax first constructs the set of all pair-wise vector maximums and then filters out dominated cost vectors. This has a complexity of $O((|V_1||V_2|)^2)$, regardless of the size of the resulting set. An implementation that constructs $\mathrm{comax}(V_1, V_2)$ incrementally, adding only cost vectors not dominated by any already in the set, and removing any dominated by each newly added cost vector, will be more efficient on average if the non-dominated set is small, in particular if non-dominated cost vectors are added early, but has the same complexity in the worst case. Ren, Zhan, Rathinam, Likhachev, and Choset (2022) show that a set of 3-dimensional cost vectors that is generated incrementally can be stored as a balanced binary search tree, such that the complexity of testing whether a cost vector is dominated by or equal to any in the set is logarithmic in the size of the set. This representation could also be used to compute the comax set. However, the complexity of updating the set representation when a new cost vector that dominates some already in the set is added remains linear in the size of the set, so the worst case total complexity of the comax computation does not change.

There is also the potential that the additional structure of the comax set can be exploited to compute it more efficiently. In Appendix A we propose a linear-time algorithm for the special bi-objective case ($k = 2$). Here we make only two observations that hold in the general case.

Lemma 12. *If $\vec{v} \prec \vec{v}'$, then $\max(\vec{v}, \vec{u}) \preceq \max(\vec{v}', \vec{u})$; symmetrically, if $\vec{u} \prec \vec{u}'$, then $\max(\vec{v}, \vec{u}) \preceq \max(\vec{v}, \vec{u}')$;*

Proof. $\max(\vec{v}, \vec{u})^i$ equals either $\vec{v}^i$ or $\vec{u}^i$. If $\max(\vec{v}, \vec{u})^i = \vec{v}^i$, then $\vec{v}^i \geq \vec{u}^i$; since $\vec{v} \prec \vec{v}'$, $\vec{v}'^i \geq \vec{v}^i$, and thus $\max(\vec{v}', \vec{u})^i = \vec{v}'^i \geq \max(\vec{v}, \vec{u})^i$. If $\max(\vec{v}, \vec{u})^i = \vec{u}^i$, then $\vec{u}^i \geq \vec{v}^i$; since $\max(\vec{v}', \vec{u})^i \geq \vec{u}^i$, we also have $\max(\vec{v}', \vec{u})^i \geq \max(\vec{v}, \vec{u})^i$. The symmetric claim follows from same argument, since the maximum is commutative. □

Corollary 13. $\mathrm{comax}(V_1, V_2) = \mathrm{comax}(\mathsf{ND}(V_1), \mathsf{ND}(V_2))$.

That is, $\mathrm{comax}(V_1, V_2)$ can be computed from the subsets of non-dominated cost vectors in V_1 and V_2 respectively. This is significant because the complexity of finding the non-dominated cost vectors in these two sets is $O(|V_1|^2 + |V_2|^2)$, while the complexity of finding the non-dominated cost vectors in the set of all pair-wise maximums is $O((|V_1||V_2|)^2)$. Unfortunately, the converse of Lemma 12 does not hold: combining cost vectors that are non-dominated in V_1 and in V_2, respectively, can still yield dominated cost vectors. For a simple example, consider $V_1 = V_2 = \{\langle i, n - i\rangle \mid i = 0, \ldots, n\}$. As in the earlier example, all cost vectors in this set are mutually non-dominating. From Lemma 14 below, each of these cost vectors is (non-dominated) in $\mathrm{comax}(V_1, V_2)$. However, every combination $\max(\langle i, n - i\rangle, \langle j, n - j\rangle)$, where $i \neq j$, results in a cost vector $\langle \max(i, j), n - \min(i, j)\rangle$ which is strictly dominated by both $\langle i, n - i\rangle$ and $\langle j, n - j\rangle$.

Lemma 14. *If $\vec{v}_1 \in \mathsf{ND}(V_1)$, $\vec{v}_2 \in \mathsf{ND}(V_2)$ and $\vec{v}_1 \preceq \vec{v}_2$, then $\vec{v}_2 \in \mathrm{comax}(V_1, V_2)$.*

Proof. $\vec{v}_1 \preceq \vec{v}_2$ means that $\vec{v}_1^i \leq \vec{v}_2^i$ for all dimensions i; thus $\max(\vec{v}_1, \vec{v}_2) = \vec{v}_2$, which means $\vec{v}_2 \in \mathrm{comax}(V_1, V_2)$ unless it is dominated by another pair-wise maximum. Suppose there exists $\vec{w} \in \mathrm{comax}(V_1, V_2)$, such that $\vec{w} \prec \vec{v}_2$. $\vec{w} = \max(\vec{v}, \vec{u})$ for some $\vec{v} \in V_1$ and $\vec{u} \in V_2$, and hence $\vec{u} \preceq \vec{w}$. But $\vec{u} \preceq \vec{w}$ and $\vec{w} \prec \vec{v}_2$ implies $\vec{u} \prec \vec{v}_2$, contradicting that $\vec{v}_2 \in \mathsf{ND}(V_2)$. □

As a special case, Lemma 14 implies that any $\vec{v} \in \mathsf{ND}(V_1) \cap \mathsf{ND}(V_2)$ is in the comax set.

4.2 Anti-Dominance Maximum (admax)

The anti-dominance maximum is defined by Geißer et al. (2022) as follows:

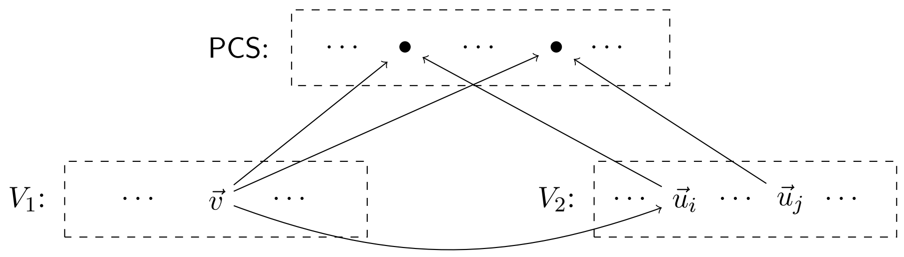


Figure 1: Example of dominance between cost vectors in V_1, V_2 (both admissible) and PCS.

Definition 15. *Let V_1 and V_2 be sets of k-dimensional cost vectors. The* anti-dominance maximum *of V_1 and V_2 is* $\mathrm{admax}(V_1, V_2) = \{\vec{v_1} \in \mathsf{ND}(V_1) \mid \forall \vec{v_2} \in \mathsf{ND}(V_2) : \vec{v_1} \not\prec \vec{v_2}\} \cup \{\vec{v_2} \in \mathsf{ND}(V_2) \mid \forall \vec{v_1} \in \mathsf{ND}(V_1) : \vec{v_2} \not\prec \vec{v_1}\}$.

The intuition behind admax is that if $\vec{v}_1 \in V_1$ strictly dominates a non-dominated cost vector $\vec{v}_2 \in V_2$, then including $\vec{v}_1$ in the union unnecessarily weakens it; admissibility of V_2 should ensure that any cost vector $\vec{u} \in \vec{c}(\mathsf{PCS}(s))$ is still dominated by or equal to some vector in V_2. The flaw in Definition 15 is that this is done to both sets independently. The following counterexample shows how this can lead to inadmissibility.

Example 16. *Let* $V_1 = \{\langle 1, 2\rangle, \langle 3, 1\rangle\}$, $V_2 = \{\langle 2, 1\rangle, \langle 1, 3\rangle\}$. *Then* $\mathrm{admax}(V_1, V_2) = \{\langle 3, 1\rangle, \langle 1, 3\rangle\}$.

If $\vec{c}(\mathsf{PCS}(s)) = \{\langle 2, 2\rangle, \langle 3, 1\rangle, \langle 1, 3\rangle\}$ *then V_1 and V_2 are both admissible heuristic sets for state s, but* $\mathrm{admax}(V_1, V_2)$ *is not, since no cost vector in it dominates or equals* $\langle 2, 2\rangle$.

When we remove a cost vector $\vec{v} \in V_1$ from the union of V_1 and V_2, then given only the knowledge that V_1 is an admissible heuristic set, for each cost vector $\vec{w} \in \vec{c}(\mathsf{PCS}(s))$ such that $\vec{v} \preceq \vec{w}$, we cannot know if any cost vector other than $\vec{v}$ in V_1 also dominates or equals $\vec{w}$. Therefore, to ensure admissibility, we have to keep at least one vector $\vec{u}_j \in V_2$ such that $\vec{u}_j \preceq \vec{w}$. Figure 1 illustrates the situation. If $\vec{v} \prec \vec{u}_i$, then $\vec{v} \prec \vec{w}$ for any $\vec{w}$ such that $\vec{u}_i \preceq \vec{w}$, but the reverse does not hold. Thus, we have to keep from V_2 also any cost vector $\vec{u}$ that is incomparable with $\vec{v}$. This means if we remove $\vec{v} \in V_1$ from the union, we can only remove from V_2 any vector $\vec{u}'$ such that $\vec{u}' \preceq \vec{v}$. But since the reason for removing $\vec{v}$ is that $\vec{v} \prec \vec{u}_i \in \mathsf{ND}(V_2)$, there can be no $\vec{u}' \in V_2$ such that $\vec{u}' \prec \vec{v}$, since that would imply $\vec{u}' \prec \vec{u}_i$ and hence $\vec{u}_i \notin \mathsf{ND}(V_2)$. Consequently, any admissible union of subsets of V_1 and V_2 must include at least one of V_1 or V_2 in full. This, together with the observation that enlarging an admissible heuristic set, while still admissible, cannot make it a more informed heuristic, motivates the new MO maximum operator, which we define in the next section.

4.3 Select-One Maximum (somax)

The idea of the select-one maximum is that if V_1 and V_2 are both admissible heuristic sets for a state s, then selecting either one of them to be their maximum is also admissible. The reason why we can still expect this maximum of two admissible heuristics to be more informed than using just one of them is that the selection is done per state, and that if one of the two sets is, in some sense, "strictly better" than the other, we select the better one; only if the two sets are incomparable is the choice arbitrary.

Definition 17. *The* select-one maximum *of V_1 and V_2 is*

$$\mathrm{somax}(V_1, V_2) = \begin{cases} V_1 & \textit{(i) if } V_2 \prec V_1 \\ V_2 & \textit{(ii) if } V_1 \prec V_2 \\ V_i & \textit{(iii) where } i \in \{1,2\} \textit{ chosen by some tie-breaking condition otherwise} \end{cases}$$

The choice of vector set dominance as "strictly better" is natural, but not the only choice. All that is really needed is that the condition is asymetric, so that at most one of cases (*i*) or (*ii*) hold. The tie-breaking condition can be arbitrary, including always choosing one of the two arguments. The following proposition shows admissibility.

Proposition 18. *Let $H(s) = \mathrm{somax}(H_1(s), H_2(s))$. If H_1 and H_2 are admissible, then so is H.*

Proof. (a) Let $\vec{v} \in \vec{c}(\mathsf{PCS}(s))$. By definition, $H(s) = H_i(s)$, for some $i \in \{1, 2\}$. Since H_1 and H_2 are admissible, there exist $\vec{u}_1 \in H_1(s)$ such that $\vec{u}_1 \preceq \vec{v}$ and $\vec{u}_2 \in H_2(s)$ such that $\vec{u}_2 \preceq \vec{v}$. At least one of $\vec{u}_1$ and $\vec{u}_2$ is in $H(s)$. Hence, H is admissible. □

Note that Proposition 18 is independent of the tie-breaking condition in Definition 17; in fact, it is independent of cases (*i*) and (*ii*) as well. It follows simply because when applied to admissible heuristics, somax selects, for each state, one of two admissible heuristics to use in that state.

Compared to comax, somax has very weak theoretical guarantees. In general, we cannot guarantee that $V_i \preceq \mathrm{somax}(V_1, V_2)$ for both $i = 1, 2$, since the non-selected set may be incomparable. If the tie-breaking condition depends only on the content (not index) of the two vector sets, then somax is commutative. Without further assumptions on the tie-breaking order, however, somax is *not* associative: Let A, B and C be vector sets, and suppose $A \prec C$ but A and C are both incomparable to B. Furthermore, suppose the tie-breaking condition prefers A to B and B to C. Then we have $\mathrm{somax}(A, B) = A$, $\mathrm{somax}(B, C) = B$ and $\mathrm{somax}(A, C) = C$, and therefore $\mathrm{somax}(A, \mathrm{somax}(B, C)) = A \neq C = \mathrm{somax}(\mathrm{somax}(A, B), C)$.

somax is *not* guaranteed to preserve MO heuristic consistency. In fact, there is no tie-breaking rule under which such a guarantee can be obtained, as the following counterexample shows.

Example 19. *We have three states, s_1, s_2 and t, with transitions from s_1 to t and from s_2 to t, both with cost $\langle 1, 1\rangle$. Heuristics H_1 and H_2 assign values as follows:*

	s_1	s_2	t
H_1	$\{\langle 2,3\rangle\}$	$\{\langle 1,1\rangle\}$	$\{\langle 1,2\rangle\}$
H_2	$\{\langle 1,1\rangle\}$	$\{\langle 3,2\rangle\}$	$\{\langle 2,1\rangle\}$
$\mathrm{somax}(H_1, H_2)$	$\{\langle 2,3\rangle\}$	$\{\langle 3,2\rangle\}$	*?*

Note that both heuristics assign singleton cost vector sets to all three states.

H_1 is consistent w.r.t. the s_1–t transition because $\langle 2, 3\rangle \preceq \langle 1, 1\rangle + \langle 1, 2\rangle$ and w.r.t. the s_2–t transition because $\langle 1, 1\rangle \preceq \langle 1, 1\rangle + \langle 1, 2\rangle$.

H_2 is consistent w.r.t. the s_1–t transition because $\langle 1, 1\rangle \preceq \langle 1, 1\rangle + \langle 2, 1\rangle$ and w.r.t. the s_2–t transition because $\langle 3, 2\rangle \preceq \langle 1, 1\rangle + \langle 2, 1\rangle$.

$\mathrm{somax}(H_1(s_1), H_2(s_1)) = H_1(s_1) = \{\langle 2, 3\rangle\}$*, because there exists a cost vector in $H_2(s_1)$, namely $\langle 1, 1\rangle$, which dominates all in $H_1(s_1)$. Likewise,* $\mathrm{somax}(H_1(s_2), H_2(s_2)) = H_2(s_2) = \{\langle 3, 2\rangle\}$.

However, $\text{somax}(H_1(t), H_2(t))$ *can be either* $H_1(t) = \{\langle 1, 2\rangle\}$ *or* $H_2(t) = \{\langle 2, 1\rangle\}$*, since neither vector set dominates the other.*

Suppose $\text{somax}(H_1(t), H_2(t)) = H_1(t) = \{\langle 1, 2\rangle\}$*. Then the maximum of the two heuristics is inconsistent w.r.t. the* s_2*–*t *transition, since* $\langle 3, 2\rangle \not\preceq \langle 2, 3\rangle = \langle 1, 1\rangle + \langle 1, 2\rangle$*. If, on the other hand,* $\text{somax}(H_1(t), H_2(t)) = H_2(t) = \{\langle 2, 1\rangle\}$*, then the maximum is inconsistent w.r.t. the* s_1*–*t *transition, since* $\langle 2, 3\rangle \not\preceq \langle 3, 2\rangle = \langle 1, 1\rangle + \langle 2, 1\rangle$*.*

Hence, there is no choice of tie-breaking rule that will ensure somax *preserves consistency.*

5. Experimental Comparison

We compare the use of comax and somax in the canonical MO PDB, the MO $H^{\max}$ and MO H^2 heuristics defined by Geißer et al. (2022) on their collection of 437 benchmark MO planning problems. We use their MO planner implementation, modifying only the MO maximum operators, and like them we use all non-redundant patterns of 2 or 3 variables for the PDB heuristic.

Comparing somax **and** comax**.** The number of problems solved is summarised in Table 1. Figure 2(a–c) shows a comparison of node expansions and runtime between the two operators in each heuristic on pairwise commonly solved problems. Comparisons are shown as cumulative distributions of the $\log_2$ of the ratio "alternative/baseline", where the baseline is (a naive implementation of) comax, and the alternative is somax. For example, a value of -1 on the x-axis means the value plotted (expansions or runtime) achieved by the alternative is $\frac{1}{2}$ that of the baseline; a corresponding y-value of p means that the alternative achieves this, or better, in p percent of the problem instances.

Replacing comax with somax reduces the informativeness, as measured by node expansions, of all the compared MO heuristics to varying degrees. In the canonical MO PDB heuristic, where node expansions often do not increase, using somax does frequently reduce total runtime.

Improved comax **implementations.** We also compare improved implementations of comax. These have no effect on expansions, since the maximum, and therefore the heuristic, computed is always the same, but have an effect on runtime. Figure 2(d) shows the result. Runtime ratio distributions for PDB(2), PDB(3) and $H^{\max}$ using somax – i.e., the curves from Figure 2(c) – are shown in gray to make comparison easier.

The baseline (naive) implementation computes all pair-wise vector maximums first, and then filters out dominated ones. (This is the implementation that was used in the experiments reported by Geißer et al.) The first improved version ("v.1") filters dominated cost vectors on-the-fly, as each pair-wise maximum is added to the result set. The second improved version ("v.2") does the the same, but also filters dominated cost vectors from both input sets before computing the maximum. In $H^{\max}$ and H^2, where each input to the maximum is a minimum, this is already the case, since the MO minimum of two cost vector sets V_1 and V_2 is defined as $\mathsf{ND}(V_1 \cup V_2)$. Hence, in $H^{\max}$ and H^2 versions 1 and 2 are the same (we label them both "v.2"). The baseline version of comax used in $H^{\max}$ and H^2 does not perform filtering of dominated cost vectors at all, since the maximums are recursively input to the minimum operator. In the canonical PDB heuristic, however, inputs to the maximum are sums: these are dominance-filtered in version 2. The third improved version ("v.3") checks for cost vectors common to V_1 and V_2 first, and adds these directly to the result set (by Lemma 14). This can be done in time $O(|V_1| + |V_2|)$ because the set implementation sorts set elements in lexicographic order. It then computes the remaining cost vectors in $\text{comax}(V_1, V_2)$, with on-the-fly dominance filtering. It is also applied to dominance-filtered input sets.

In the MO $H^{\max}$ heuristic, introducing any dominance filtering of the MO maximum has a runtime overhead and little or no benefit. In the canonical MO PDB heuristic, however, incremental dominance filtering (version 1) often reduces runtime, and in combination with dominance filtering of the input vector sets (version 2) even more so, with a median reduction around 12% for PDBs of size 3. The special handling of common cost vectors (version 3) does not improve on this, however. Note, though, that even the best comax implementation does not outperform the heuristic using somax, which solves 82% and 60% of problems in less time when used with PDBs of size 2 and 3, respectively, although the heuristic using comax (v.2) solves more problems overall.

Tie-breaking in somax**.** When neither $V_1 \prec V_2$ nor $V_2 \prec V_1$, somax falls back on a tie-breaking condition to choose one of the two sets. Lastly, we examine how frequently this tie-breaking occurs, and evaluate the effect of some alternative tie-breaking rules.

Our default tie-breaking rule (used in the comparison with comax above) selects the first argument: under this rule, $\mathrm{somax}(V_1, V_2)$ equals V_2 if $V_1 \prec V_2$ and V_1 otherwise. This has the advantage that only one set dominance check is required. However, to measure how often tie-breaking occurs, we modified the somax implementation to perform both dominance checks, and then explicitly apply a tie-breaking rule when neither $V_1 \prec V_2$ nor $V_2 \prec V_1$. In addition to the default, "choose first" rule, we trialled two other tie-breaking rules: 1) "choose small", which selects the smaller (by cardinality) of V_1 and V_2; and 2) "choose big", which selects the larger of V_1 and V_2. When V_1 and V_2 have the same size, both rules fall back to choose first. The rationale for preferring larger heuristic sets is that these may be a more accurate representation of the Pareto cover set of cost vectors, and therefore more informative, while the reason for preferring smaller sets is that the size of the heuristic set impacts the time required to perform many operations, including dominance checking and subsequent maximisations.

Figure 3(a) shows the fraction of somax calls that were decided by the tie-breaking rule, i.e., for which neither $V_1 \prec V_2$ nor $V_2 \prec V_1$, under different MO heuristics. We call this the tie-breaking ratio. (Note, though, that instances in which V_1 and V_2 are equal also count as "decided by tie-breaking" here, even though the rule used does not matter in such cases.) The ratio shown is using the "choose first" rule; using one of the other tie-breaking rules makes very small changes to the tie-breaking ratio. Figures 3(b–c) show the impact of the variant tie-breaking rules on heuristic informativeness (as measured by number of expansions) and total runtime, in comparison with the baseline "choose first" rule. Note that the implementation of the baseline rule used here performs both dominance checks, rather than just one. We omit H^2 from the comparison in Figures 3(b–c) due to the small number of instances solved with it.

The MO $H^{\max}$ and MO H^2 heuristics have a much higher tie-breaking ratio, compared to the canonical MO PDB heuristic. Note also that MO $H^{\max}$ performs, at median, 7.25 times more somax calls per heuristic evaluation compared to the canonical MO PDB heuristic with patterns of size 3, and 36 times more than the PDB heuristic with patterns of size 2. Consistent with this, $H^{\max}$ shows the greatest amount of variation in informativeness and runtime when we vary the tie-breaking rule, while the MO PDB heuristic with patterns of size 2 shows the least. Somewhat surprisingly, preferring smaller heuristic sets yields more informed heuristics more frequently than tie-breaking in favour of larger sets with the MO $H^{\max}$ and canonical MO PDB heuristic with patterns of size 3. Neither rule, however, yields a consistent improvement over the default, choose first, rule.

	comax (naive)	comax (v.2)	somax
PDB(2)	275	275	272
PDB(3)	308	312	295
$H^{\max}$	223	216	195
H^2	43		7

Table 1: Number of problems solved with each heuristic using different MO maximum operators. The baseline (naive) and version 2 of comax differ in how $\mathrm{comax}(V_1, V_2)$ is implemented; this affects the time to compute the operator, but not its result.

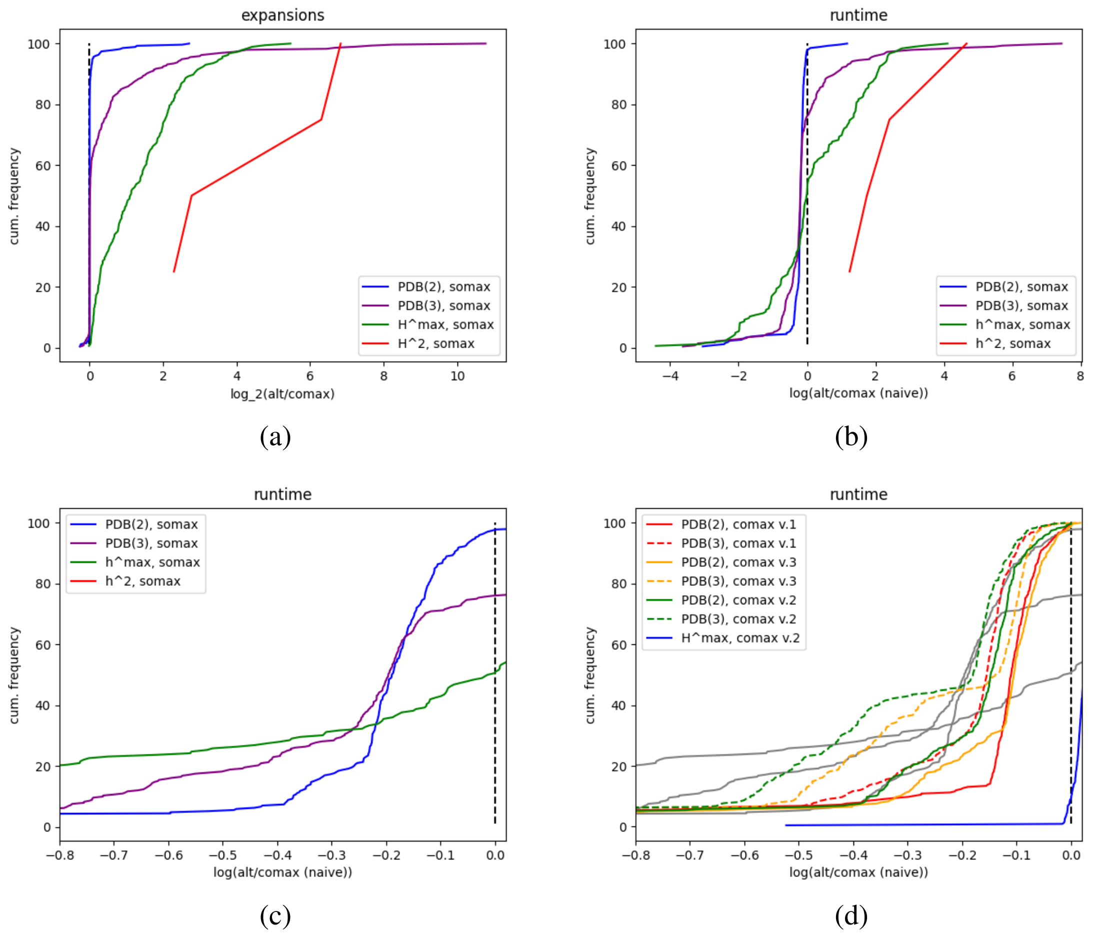
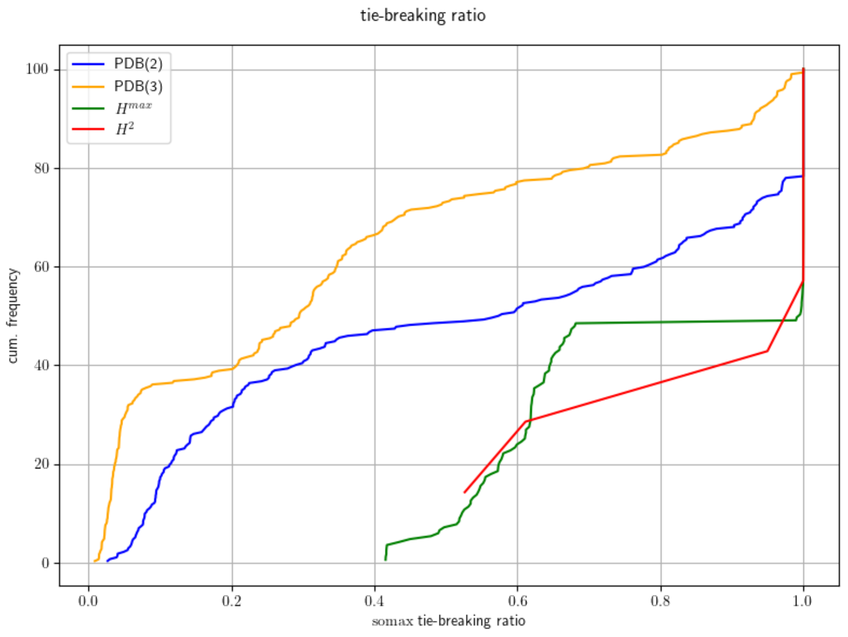
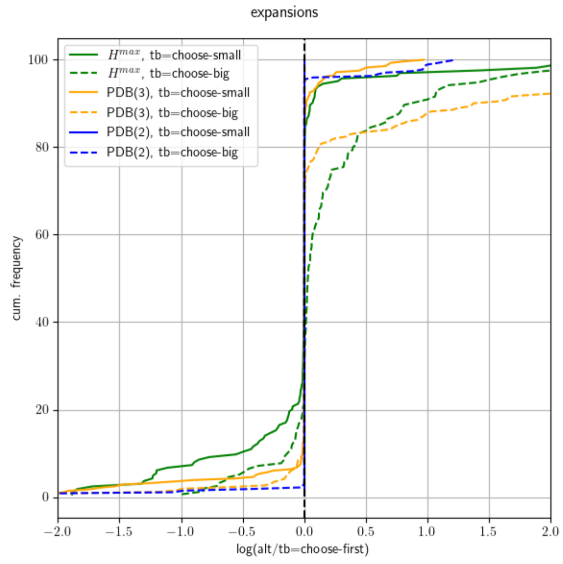


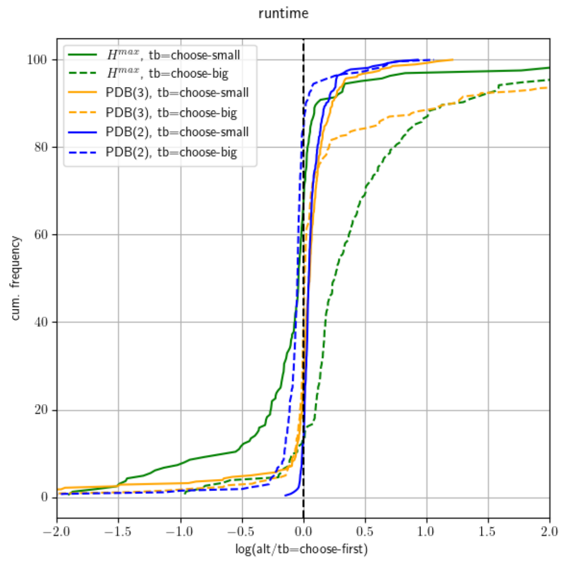
Figure 2: Comparison of MO maximum operators in the context of MO PDB, $H^{\max}$ and H^2 heuristics. The x-axis is $\log_2$ of the ratio "alternative/baseline", where the baseline is the naive implementation of comax. The y-axis is cumulative percentage, of instances solved by both. (a) Node expansions. (b) Runtime; (c) closeup of the range of ratios -0.8–0 in (b). (d) Runtime comparison of different comax implementations in canonical MO PDB heuristics (see text for descriptions).



(a)



(b)



(c)

Figure 3: (a) somax reliance on the tie-breaking condition in the context of different MO heuristics. The "tie-breaking ratio" (x-axis) is the fraction of somax calls that were decided by the tie-breaking rule. (b–c) Distribution of the number of expansions and runtime over varying MO heuristics and tie-breaking rules. The x-axis is the $\log_2$ of the ratio "alternative/baseline", where the baseline is the same heuristic with the "choose first" rule. The y-axis, in all three plots, is the cumulative frequency (in %) of instances solved with the respective heuristic/tie-breaking rule.

6. Conclusion

Multi-objective heuristics, because dominance imposes only a partial order on heuristic values, have more options for admissible combination than their single-objective counterparts. Although we have shown that the component-wise maximum (comax) is the greatest possible admissibility-preserving MO maximum operator, its potential computational overhead, particularly when heuristic sets are large and high-dimensional, motivates continued investigation into its efficient implementation as well as of alternative MO maximum operators.

The experimental evaluation conducted by Geißer et al. (2022) suggested that heuristics using the comax MO maximum operator were, in aggregate across the benchmark set, superior to those using the admax operator, with exceptions for certain combinations of heuristic and domain. Our experimental results are consistent with this. While the admax MO maximum operator turns out to be not admissibility-preserving, the somax operator we have proposed achieves a similar trade-off in canonical MO PDB heuristics, sacrificing some informativeness for, in a majority of cases, lower runtime, also when compared to an improved implementation of comax.

Appendix A. comax in the Bi-Objective Case

As is often the case in multi-objective search and optimisation, the special case of $k = 2$ objectives exhibits some particular properties. Specifically, in this section, we prove for this case a characterisation of the cost vectors that are in $\mathrm{comax}(V_1, V_2)$ (Proposition 21). A consequence of this is that when $k = 2$, the size of $\mathrm{comax}(V_1, V_2)$ is bounded by $|V_1| + |V_2|$, i.e., the worst case size is the sum of the input set sizes rather than the product, as it is in the general case (Proposition 11). We also propose a linear-time algorithm for computing comax in this case. Zhang, Salzman, Felner, Kumar, Skyler, Ulloa, and Koenig (2023) also propose a polynomial-time algorithm for computing comax in the bi-objective case, based on a different approach.

A.1 Characterisation of the comax Set

Assume $k = 2$, V_1 and V_2 contain only non-dominated, within each set, cost vectors, and furthermore that V_1 and V_2 are sorted lexicographically, in increasing order, on dimension 0 first. Let $<_{\text{lex}}$ denote this lexicographic order. Then, for any two mutually non-dominating cost vectors $\langle x, y\rangle$ and $\langle x', y'\rangle$ such that $\langle x, y\rangle <_{\text{lex}} \langle x', y'\rangle$, we have $x < x'$ and $y > y'$. This is necessary since both cost vectors must have one component that is strictly smaller than the corresponding component of the other, and the lexicographic order implies $x \leq x'$. Hence, $V_1 = \{\langle x_1, y_1\rangle, \ldots, \langle x_n, y_n\rangle\}$, such that $x_i < x_j$ and $y_i > y_j$ for $i < j$, and likewise for V_2. Another important property that holds in the bi-objective case is that $\vec{v} <_{\text{lex}} \vec{u}$ implies $\vec{u} \not\prec \vec{v}$, i.e., a cost vector that is lexicographically greater cannot strictly dominate one that is lexicographically smaller.

Lemma 20. *Let $\vec{v} = \langle x, y\rangle \in V_1$, $\vec{v}' = \langle x', y'\rangle \in V_1$ $\vec{u} = \langle a, b\rangle \in V_2$, and $\vec{u}' = \langle a', b'\rangle \in V_2$ such that $\vec{v}' <_{lex} \vec{v} <_{lex} \vec{u} <_{lex} \vec{u}'$, and $\vec{v} \not\prec \vec{u}$ (also $\vec{v}' \not\prec \vec{v}$ and $\vec{u} \not\prec \vec{u}'$). Then (i) $\max(\vec{v}, \vec{u}) \prec \max(\vec{v}, \vec{u}')$, and (ii) $\max(\vec{v}, \vec{u}) \prec \max(\vec{v}', \vec{u})$.*

Proof. Due to the lexicographic order and non-dominance, we have $y' > y > b > b'$ and $x' < x < a < a'$. Hence, (*i*) $\max(\vec{v}, \vec{u}) = \langle a, y\rangle$ and $\max(\vec{v}, \vec{u}') = \langle a', y\rangle$, and $\langle a, y\rangle \prec \langle a', y\rangle$; and (*ii*) $\max(\vec{v}', \vec{u}) = \langle a, y'\rangle$ and $\max(\vec{v}, \vec{u}) = \langle a, y\rangle$, and $\langle a, y\rangle \prec \langle a, y'\rangle$. □

Proposition 21. *Assume $k = 2$, V_1 and V_2 contain only non-dominated, within each set, cost vectors, and let $\vec{v}_1, \ldots, \vec{v}_m$ be $V_1 \cup V_2$ sorted by $<_{lex}$. The non-dominated cost vectors in* $\mathrm{comax}(V_1, V_2)$ *are exactly*

(i) $\vec{v}_i$ such that $\vec{v}_i \in V_1 \cap V_2$;

(ii) $\vec{v}_i$ such that there exists $\vec{v}_j$ such that $\vec{v}_j \prec \vec{v}_i$ ($\vec{v}_i$ and $\vec{v}_j$ must be in opposite sets);

(iii) $\max(\vec{v}_i, \vec{v}_{i+1})$ *such that $\vec{v}_i$ and $\vec{v}_{i+1}$ are in opposite sets and neither $\vec{v}_i$ nor $\vec{v}_{i+1}$ dominates nor is dominated by any cost vector in the opposite set;*

(iv) $\max(\vec{v}_i, \vec{v}_{i+1})$ *such that $\vec{v}_i$ and $\vec{v}_{i+1}$ are in opposite sets, $\vec{v}_{i+1} \prec \vec{v}_p$ for some $p > i + 1$ ($\vec{v}_p$ must be in the opposite set to $\vec{v}_{i+1}$, and therefore in the same set as $\vec{v}_i$), $\vec{v}^0_{i+1} < \vec{v}^0_p$, and $\vec{v}_i$ does not dominate and is not dominated by any cost vector in the opposite set;*

(v) $\max(\vec{v}_i, \vec{v}_p)$ *such that $\vec{v}_i$ and $\vec{v}_p$ are in opposite sets, $i + 1 < p$, $\vec{v}_i \prec \vec{v}_l$ for all $i < l < p$ ($\vec{v}_l$ must be in the opposite set to $\vec{v}_i$, and therefore in the same set as $\vec{v}_p$), $\vec{v}^1_i < \vec{v}^1_{p-1}$, $\vec{v}_p$ is not dominated by any cost vector in the opposite set, and either $\vec{v}_p$ does not dominate any cost vector in the opposite set, or $\vec{v}^0_p < \vec{v}^0_{p+1}$.*

Proof. That cost vectors satisfying condition (i) or (ii) are in $\mathrm{comax}(V_1, V_2)$ was shown in Lemma 14.

Condition (iii): Consider a cost vector $\vec{w} = \max(\vec{v}_i, \vec{v}_{i+1})$, where $\vec{v}_i$, $\vec{v}_{i+1}$ are in opposite sets (i.e., one in V_1 and the other in V_2, and neither is in both V_1 and V_2), and both $\vec{v}_i$ and $\vec{v}_{i+1}$ do not dominate and are not dominated by any vector in their respective opposite sets. Suppose $\vec{w} \notin \mathrm{comax}(V_1, V_2)$, i.e., that there is a cost vector $\vec{w}' \in \mathrm{comax}(V_1, V_2)$ such that $\vec{w}' \prec \vec{w}$. $\vec{w}' = \max(\vec{v}_j, \vec{v}_{j'})$, where $\vec{v}_j \in V_1$ and $\vec{v}_{j'} \in V_2$; note that here, $\vec{v}_j$ may equal $\vec{v}_{j'}$, and either of them may equal $\vec{v}_i$ or $\vec{v}_{i+1}$. Since $<_{\text{lex}}$ is a total order, all four cost vectors are ordered in some way. We examine the possibilities:

(1) $\vec{v}_j <_{\text{lex}} \vec{v}_i$ and $\vec{v}_{j'} <_{\text{lex}} \vec{v}_i$ (i.e., $j, j' < i$). Note that this includes also the possibility that $\vec{v}_j = \vec{v}_{j'}$. In this case, $\vec{v}^1_j > \vec{v}^1_i$ and $\vec{v}^1_{j'} > \vec{v}^1_i$, and, since $\vec{v}^1_i > \vec{v}^1_{i+1}$, consequently $\vec{w}'^1 > \vec{w}^1$: $\vec{w}'$ cannot dominate $\vec{w}$.

(2) $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_j$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_{j'}$ (i.e., $i+1 < j, j'$). This also includes the possibility that $\vec{v}_j = \vec{v}_{j'}$. In this case, $\vec{v}^0_j > \vec{v}^0_{i+1}$ and $\vec{v}^0_{j'} > \vec{v}^0_{i+1}$, and, since $\vec{v}^0_{i+1} > \vec{v}^0_i$, consequently $\vec{w}'^0 > \vec{w}^0$: $\vec{w}'$ cannot dominate $\vec{w}$.

(3) $\vec{v}_j <_{\text{lex}} \vec{v}_i$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_{j'}$. In this case, $\vec{v}^0_{j'} > \vec{v}^0_{i+1} > \vec{v}^0_i$ and $\vec{v}^1_j > \vec{v}_i > \vec{v}^1_{i+1}$; consequently, $\vec{w}'^0 > \vec{w}^0$ and $\vec{w}'^1 > \vec{w}^1$: $\vec{w}'$ cannot dominate $\vec{w}$.

(4) $\vec{v}_{j'} <_{\text{lex}} \vec{v}_i$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_j$. This case is symmetric with (3).

(5) $\vec{v}_j = \vec{v}_i$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_{j'}$. Since $\vec{v}_i \notin V_1 \cap V_2$, this implies $\vec{v}_i \in V_1$, $\vec{v}_{i+1} \in V_2$ and $\vec{v}_{j'} \in V_2$. By Lemma 20, $\vec{w} \prec \vec{w}'$. The case when $\vec{v}_{j'} = \vec{v}_i$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_j$ is symmetric, with V_1 and V_2 exchanged.

(6) $\vec{v}_j = \vec{v}_i$ and $\vec{v}_{j'} <_{\text{lex}} \vec{v}_i$. We have $\vec{v}^1_{j'} > \vec{v}^1_i > \vec{v}^1_{i+1}$, and thus $\vec{w}'^1 > \vec{w}^1$: $\vec{w}'$ cannot dominate $\vec{w}$. Again, the case with $\vec{v}_j$ and $\vec{v}_{j'}$ changing places is symmetric.

(7) $\vec{v}_j = \vec{v}_{i+1}$ and $\vec{v}_{j'} <_{\text{lex}} \vec{v}_i$. Since $\vec{v}_{i+1} \notin V_1 \cap V_2$, this implies $\vec{v}_{i+1} \in V_1$, $\vec{v}_i \in V_2$ and $\vec{v}_{j'} \in V_2$. By Lemma 20, $\vec{w} \prec \vec{w}'$. The case when $\vec{v}_{j'} = \vec{v}_{i+1}$ and $\vec{v}_j <_{\text{lex}} \vec{v}_i$ is symmetric, with V_1 and V_2 exchanged.

(8) $\vec{v}_j = \vec{v}_{i+1}$ and $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_{j'}$. We have $\vec{v}^0_{j'} > \vec{v}^0_{i+1} > \vec{v}^0_i$, and thus $\vec{w}'^0 > \vec{w}^0$: $\vec{w}'$ cannot dominate $\vec{w}$. The case with $\vec{v}_j$ and $\vec{v}_{j'}$ changing places is symmetric.

This shows that all cost vectors satisfying condition (iii) are in $\mathrm{comax}(V_1, V_2)$.

Condition (iv): Let $\vec{w} = \max(\vec{v}_i, \vec{v}_{i+1})$ such that $\vec{v}_i$ and $\vec{v}_{i+1}$ are in opposite sets, $\vec{v}_{i+1} \prec \vec{v}_p$ for some $p > i+1$, $\vec{v}^0_{i+1} < \vec{v}^0_p$, and $\vec{v}_i$ does not dominate and is not dominated by any cost vector in the opposite set. Suppose there exists $\vec{w}' = \max(\vec{v}_j, \vec{v}_{j'}) \in \mathrm{comax}(V_1, V_2)$ such that $\vec{w}' \prec \vec{w}$. The only difference to condition (iii) is in case (8), with $\vec{v}_{i+1}$ taking the role of $\vec{v}_j$ and $\vec{v}_p$ taking the role of $\vec{v}_{j'}$: since $\vec{v}_{i+1} \prec \vec{v}_p$, the ordering $\vec{v}_{i+1} <_{\text{lex}} \vec{v}_p$ only implies $\vec{v}^0_{i+1} \leq \vec{v}^0_p$, but the additional condition ensures this inequality is strict.

Condition (v): Let $\vec{w} = \max(\vec{v}_i, \vec{v}_p)$ such that $\vec{v}_i$ and $\vec{v}_p$ are in opposite sets, $i+1 < p$, $\vec{v}_i \prec \vec{v}_l$ for all $i < l < p$ ($\vec{v}_l$ must be in the opposite set to $\vec{v}_i$, and therefore in the same set as $\vec{v}_p$), $\vec{v}^1_i < \vec{v}^1_{p-1}$, and $\vec{v}_p$ does not dominate and is not dominated by any cost vector in the opposite set. Suppose there exists $\vec{w}' = \max(\vec{v}_j, \vec{v}_{j'}) \in \mathit{comax}(V_1, V_2)$ such that $\vec{w}' \prec \vec{w}$. The difference to condition (iii) here arises in two cases: (1) we can have $\vec{v}_j = \vec{v}_i$ and $\vec{v}_i <_{\text{lex}} \vec{v}_{j'} <_{\text{lex}} \vec{v}_p$ i.e., $\vec{w}'$ is formed combining one of the vectors $\vec{v}_{i+1}, \ldots, \vec{v}_{p-1}$ with $\vec{v}_i$. Lemma 20 does not apply, since it requires $\vec{v}_i \not\prec \vec{v}_{j'}$. Because $\vec{v}_i$ does not dominate $\vec{v}_p$, and $\vec{v}_{j'}$ does not dominate $\vec{v}_p$ (they are in the same set), the lexicographic ordering implies $\vec{v}^0_i \leq \vec{v}^0_{j'} < \vec{v}^0_p$, and $\vec{v}^1_{j'} \geq \vec{v}^1_{p-1} > \vec{v}^1_p$ ($\vec{v}^1_{j'} \geq \vec{v}^1_{p-1}$ because j' may equal $p-1$), so $\vec{w} = \langle \vec{v}^0_p, \vec{v}^1_i \rangle$ and $\vec{w}' = \vec{v}_{j'}$. The additional condition ensures $\vec{w}'^1 = \vec{v}^1_{j'} \geq \vec{v}^1_{p-1} > \vec{v}^1_i$ so $\vec{w}'$ cannot dominate $\vec{w}$. (2) $\vec{v}_p$ can dominate some other cost vector $\vec{v}_q$. Note that $\vec{v}_q$ must be in the opposite set to $\vec{v}_p$ (and therefore in the same set as $\vec{v}_i$) and that $\vec{v}_p <_{\text{lex}} \vec{v}_q$, i.e., $q > p$. Again, Lemma 20 does not apply, but the additional condition $\vec{v}^0_p < \vec{v}^0_{p+1}$ ensures $\vec{v}^0_p < \vec{v}^0_q$ and therefore that $\max(\vec{v}_i, \vec{v}_p) \preceq \max(\vec{v}_i, \vec{v}_q)$.

It remains to show that no other combination $\max(\vec{v}_i, \vec{v}_j)$, where $\vec{v}_i$ and $\vec{v}_j$ are from opposite sets, not satisfying any of the conditions above, is in $\mathrm{comax}(V_1, V_2)$, i.e., that any such cost vector is dominated by or equal to some other cost vector in $\mathrm{comax}(V_1, V_2)$. Since the component-wise maximum of two vectors is commutative, we can assume w.l.o.g. that $i < j$.

(a) Suppose neither v_i nor v_j dominates or is dominated by any cost vector in their respective opposite sets. If $j \neq i+1$, then there exists some $\vec{v}_l$ such that $\vec{v}_i <_{\text{lex}} \vec{v}_l <_{\text{lex}} \vec{v}_j$. However, $\vec{v}_l$ either belongs the same set as $\vec{v}_i$ or $\vec{v}_j$: in both cases, Lemma 20 shows that $\max(\vec{v}_i, \vec{v}_l) \prec \max(\vec{v}_i, \vec{v}_j)$ or $\max(\vec{v}_l, \vec{v}_j) \prec \max(\vec{v}_i, \vec{v}_j)$, respectively.

(b) If $\vec{v}_i$ is dominated by some cost vector in the opposite set, then $\vec{v}_i$ is itself in $\mathrm{comax}(V_1, V_2)$, and $\vec{v}_i \preceq \max(\vec{v}_i, \vec{v}_j)$. Same if $\vec{v}_j$ is dominated by some cost vector in the opposite set.

(c) Suppose $\vec{v}_i$ dominates some cost vector in the opposite set, and that condition (v) does not apply (i.e., $\vec{v}_j$ is not the $\vec{v}_p$ required by the condition). There are four cases: (1) $\vec{v}_j$ is dominated, by $\vec{v}_i$ or some other cost vector in the same set as $\vec{v}_i$. This is covered by the case above. (2) There exists a $\vec{v}_p$ satisfying condition (v) and $\vec{v}_p <_{\text{lex}} \vec{v}_j$. Note that $\vec{v}_p$ and $\vec{v}_j$ are in the same set, so are mutually non-dominating. In this case, $\max(\vec{v}_i, \vec{v}_p) \prec \max(\vec{v}_i, \vec{v}_j)$, by Lemma 20. (3) The additional condition $\vec{v}^1_i < \vec{v}^1_{p-1}$ of condition (v) is not satisfied, i.e., the last, in lexicographic order, of the cost vectors dominated by $\vec{v}_i$ equals $\vec{v}_i$ in dimension 1, and $\vec{v}_{p-1} <_{\text{lex}} \vec{v}_j$. In this case, $\max(\vec{v}_i, \vec{v}_{p-1})^1 = \max(\vec{v}_i, \vec{v}_j)^1$ and $\max(\vec{v}_i, \vec{v}_{p-1})^0 < \max(\vec{v}_i, \vec{v}_j)^0$, so $\max(\vec{v}_i, \vec{v}_j)$ is dominated. (4) $\vec{v}^0_p = \vec{v}^0_{p+1}$: this means $\vec{v}_p \prec \vec{v}_{p+1}$, so $\vec{v}_p$ and $\vec{v}_{p+1}$ must be in opposite sets (and therefore $\vec{v}_{p+1}$ in the same set as $\vec{v}_i$). We have $\max(\vec{v}_i, \vec{v}_p) = \langle \vec{v}^0_p, \vec{v}^1_i \rangle$, and $\max(\vec{v}_p, \vec{v}_{p+1}) = \vec{v}_{p+1}$; due to the lexicographic ordering and non-dominance, $\vec{v}^1_i > \vec{v}^1_{p+1}$, so $\vec{v}_{p+1} \prec \max(\vec{v}_i, \vec{v}_p)$.

(d) Suppose $\vec{v}_j$ dominates some cost vector in the opposite set (and $\vec{v}_i$ does not) and that condition (iv) does not apply. There are three cases: (1) $\vec{v}_i$ is dominated, by $\vec{v}_j$ or some other cost vector in the same set as $\vec{v}_j$. This is covered by case (b) above. (2) $\vec{v}_{j-1}$ satisfies condition (iv) and $\vec{v}_i <_{\text{lex}} \vec{v}_{j-1}$. Again, $\vec{v}_i$ and $\vec{v}_{j-1}$ are in the same set, so are mutually non-dominating, and $\max(\vec{v}_{j-1}, \vec{v}_j) \prec \max(\vec{v}_i, \vec{v}_j)$ by Lemma 20. (3) The additional condition $\vec{v}^0_j < \vec{v}^0_p$ of condition (iv) is not satisfied. Since $\vec{v}_j \prec \vec{v}_p$, this means $\vec{v}_j^0 = \vec{v}_p^0$. $\vec{v}_i$ and $\vec{v}_p$ are mutually non-dominating (they are in the same set) and $\vec{v}_i <_{\text{lex}} \vec{v}_p$, which means $\max(\vec{v}_i, \vec{v}_j)^1 > \vec{v}_p^1$, so $\vec{v}_p \prec \max(\vec{v}_i, \vec{v}_j)$. □

We can now state the proof of Proposition 11:

Proposition 11. *Assume* $k = 2$*:* $|\operatorname{comax}(V_1, V_2)| \leq |V_1| + |V_2|$.

Proof. We can assume V_1 and V_2 contain only non-dominated, within each set, cost vectors. Let $D = \{\vec{v} \in V_1 \mid \exists \vec{u} \in V_2 : \vec{u} \prec \vec{v}\} \cup \{\vec{v} \in V_2 \mid \exists \vec{u} \in V_1 : \vec{u} \prec \vec{v}\}$, i.e., the subset of cost vectors in $V_1 \cup V_2$ that are dominated by a cost vector from the opposite set. For each $\vec{v}_i \in (V_1 \setminus D)$, there is at most one $\vec{v}_j \in (V_2 \setminus D)$ with $i < j$, that can satisfy at most one of conditions (iii)-(v), giving rise to a non-dominated cost vector in $\operatorname{comax}(V_1, V_2)$, and likewise for each $\vec{v}_i \in (V_2 \setminus D)$. Hence, $\operatorname{comax}(V_1, V_2)$ contains at most $|D| + |V_1 \setminus D| + |V_2 \setminus D|$ cost vectors. □

A.2 A Geometric Interpretation

The set of cost vectors dominated by $\vec{v}$ is $\mathcal{D}(\vec{v}) = \{\vec{u} \mid \vec{v}^i \leq \vec{u}^i, 0 \leq i < k\} - \{\vec{v}\}$. The set of cost vectors dominated by a set V is the union of the sets dominated by each cost vector in V, i.e., $\mathcal{D}(V) = \bigcup_{\vec{v} \in V} \mathcal{D}(\vec{v})$. From Proposition 9, we have that the set of cost vectors dominated by $\operatorname{comax}(V_1, V_2)$ is the intersection of the sets dominated by V_1 and V_2, i.e., $\mathcal{D}(\operatorname{comax}(V_1, V_2)) = \mathcal{D}(V_1) \cap \mathcal{D}(V_2)$.

In the special case of $k = 2$, $\vec{v} = \langle v^x, v^y \rangle$, $\mathcal{D}(\vec{v})$ can be viewed in the plane as the quadrant above and to the right of $\vec{v}$, including the line segments $\{\langle v^x, y \rangle \mid y > v^y\}$ and $\{\langle x, v^y \rangle \mid x > v^x\}$. Figure 4 illustrates conditions (ii)–(v) of Proposition 21.

A.3 Algorithm

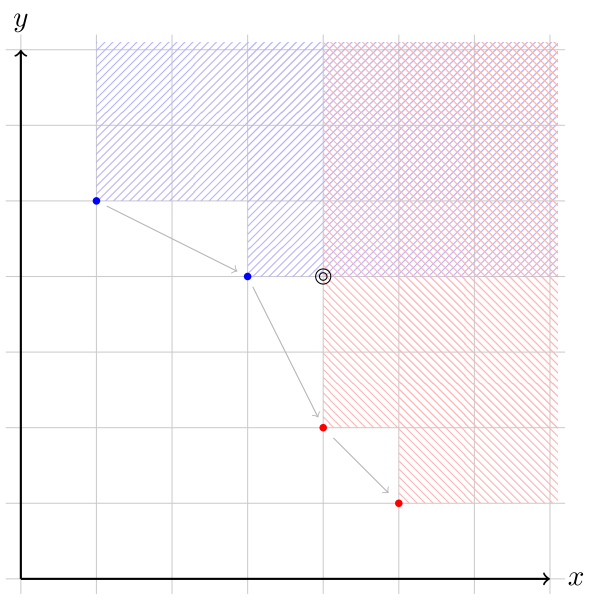
We can now use the above result to create a specialised algorithm for computing comax in the bi-objective case. This algorithm examines only the relevant pairs of cost vectors from the two input sets, and generates only the non-dominated cost vectors in the resulting set (i.e., dominance filtering of the result set is not needed). Thus, it runs in linear time. First, we need to show one more fact about the relation between dominance and lexicographic ordering:

Lemma 22. *If* $\vec{v}_1 <_{lex} \vec{v}_2 <_{lex} \vec{v}_3$ *for distinct cost vectors* $\vec{v}_1, \vec{v}_2, \vec{v}_3$*, and* $\vec{v}_1 \prec \vec{v}_3$*, then* $\vec{v}_1 \prec \vec{v}_2$ *or* $\vec{v}_2 \prec \vec{v}_3$.

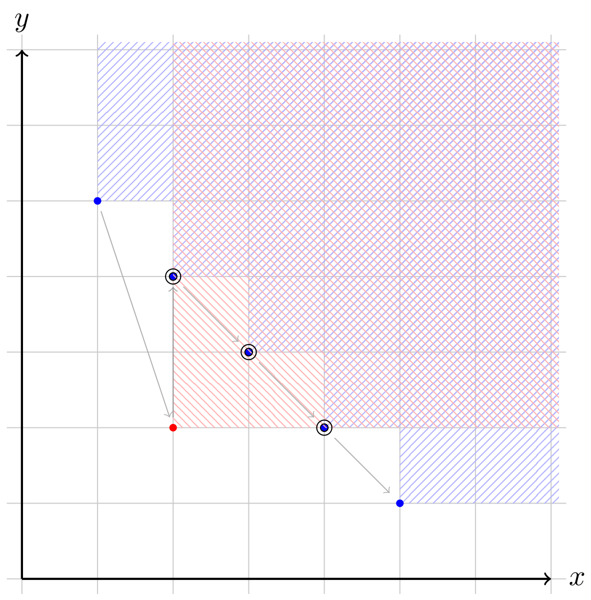
Proof. Let $\vec{v}_i = \langle x_i, y_i \rangle$, $i = 1, 2, 3$. If $y_2 < y_1$, then $x_2 \leq x_3$ (due to $\vec{v}_2 <_{\text{lex}} \vec{v}_3$) and $y_2 < y_1 \leq y_3$ (because $\vec{v}_1 \prec \vec{v}_3$ implies $y_1 \leq y_3$), and thus $\vec{v}_2 \prec \vec{v}_3$. If, on the other hand, $y_2 \geq y_1$, we also have $x_1 \leq x_2$ (due to $\vec{v}_1 <_{\text{lex}} \vec{v}_2$), with at least one of the inequalities strict since the cost vectors are distinct; this means means $\vec{v}_1 \prec \vec{v}_2$. □

Lemma 22 implies that if $\vec{v} \in V_1$ dominates some cost vector $\vec{u} \in V_2$, and V_1 and V_2 contain no dominated, within the same set, cost vectors, then all cost vectors in V_2 dominated by $\vec{v}$ are consecutive in the lexicographic order (and ordered after $\vec{v}$). Suppose $\vec{v} \prec \vec{u}_j$ but $\vec{v} \not\prec \vec{u}_i$, with $\vec{v} <_{\text{lex}} \vec{u}_i <_{\text{lex}} \vec{u}_j$: by Lemma 22, $\vec{v} \prec \vec{u}_i$ or $\vec{u}_i \prec \vec{u}_j$, but the latter is ruled out since $\vec{u}_i$ and $\vec{u}_j$ are both in V_2.

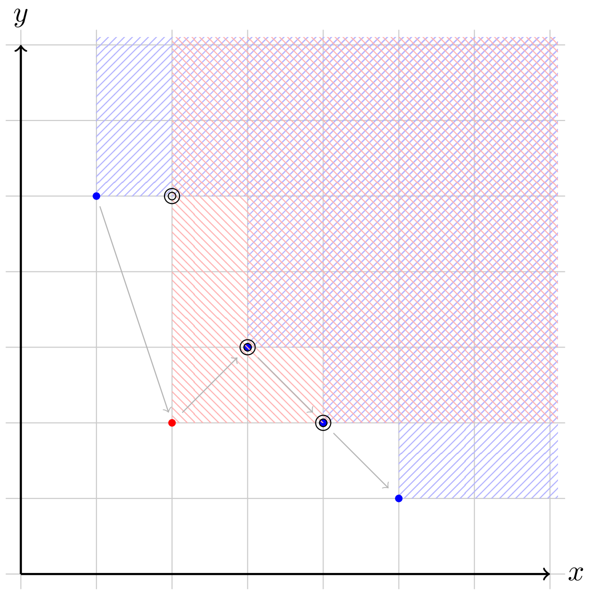
This, together with the characterisation above, means that all relevant pairs of cost vectors in V_1 and V_2 can be found by examining them in sequence according to the merged lexicographic order,



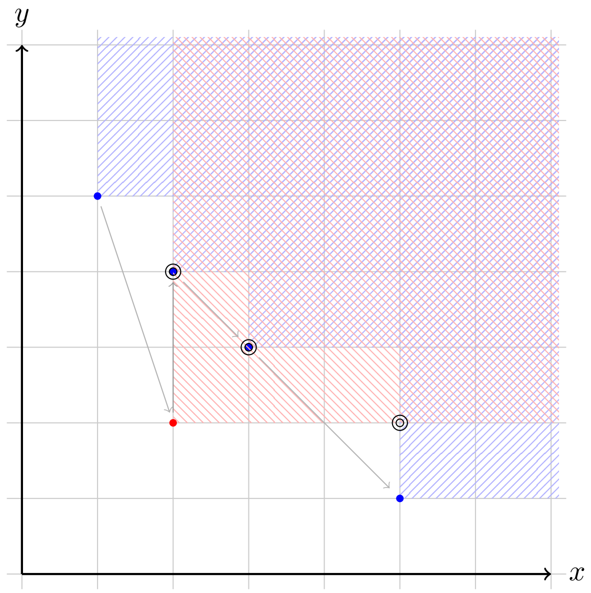
Condition (iii): Non-dominated/dominating cost vectors adjancent in the merged lexicographic order combine to form non-dominated elements in comax.



Condition (ii): cost vectors dominated by a cost vector in the opposite set are non-dominated in comax.



Condition (iv): $\vec{v}_i$ can combine with $\vec{v}_{i+1}$ to form a non-dominated element of comax also when $\vec{v}_{i+1} \prec \vec{v}_{i+2}$, provided $\vec{v}^x_{i+1} < \vec{v}^x_{i+2}$.



Condition (v): $\vec{v}_i$ can combine with $\vec{v}_j$ to form a non-dominated element of comax also when $\vec{v}_i \prec \vec{v}_l$, $i < l < j$, provided $\vec{v}^y_i < \vec{v}^y_{j-1}$.

Figure 4: Illustation of conditions (ii)–(v) of Proposition 21. Gray arrows show the lexicographic order, from smaller to larger.

and by looking only at two or three positions in that sequence at a time. We assume V_1 and V_2 are individually sorted in lexicographic order; rather than merge and sort the two sets, we can find the merged order by maintaining the index of the "next" cost vector in each set (analogous to the implementation of the merging step in the merge sort algorithm). The algorithm is then as follows:

```
(1) Assume $V_1 = v_1, v_2, \ldots, v_{|V_1|}$ and $V_2 = u_1, u_2, \ldots, u_{|V_2|}$ are individually sorted in lexicographic order. Initialise $p = q = 1$ ($p$ and $q$ are indices into $V_1$ and $V_2$, respectively).
(2) While $p \leq |V_1|$ and $q \leq |V_2|$:
    (2.1) If $v_p = u_q$, add it to *Result* (by condition (i)) and increment both $p$ and $q$ by 1.
    (2.2) Else if $v_p <_{\text{lex}} u_q$:
        (2.2.1) If $v_p \prec u_q$: Add $u_q$ to *Result* (by condition (ii)) and increment $q$ by 1.
            (2.2.1.1) While $v_p \prec u_q$: Add $u_q$ to *Result* (by condition (ii)) and increment $q$ by 1.
            (2.2.1.2) (Check condition (v).) If $q \leq |V_2|$,
                (2.2.1.2.1) If $p = |V_1|$ and $v_p^1 < u_{q-1}^1$, add $\max(v_p, u_q)$ to *Result*.
                (2.2.1.2.2) Else if $p < |V_1|$, $y_q <_{\text{lex}} v_{p+1}$, $v_p^1 < u_{q-1}^1$, and $u_q \not\prec v_{p+1}$ or $u_q^0 < v_{p+1}^0$, add $\max(v_p, u_q)$ to *Result*.
                (2.2.1.2.3) Increment $p$ by 1.
        (2.2.2) Else if $p = |V_1|$ or $u_q <_{\text{lex}} v_{p+1}$
            (2.2.2.1) If $p < |V_1|$ and $u_q \prec v_{p+1}$
                (2.2.2.1.1) (Check condition (iv).) If $u_q^0 < v_{p+1}^0$, add $\max(v_p, u_q)$ to *Result*.
            (2.2.2.2) Else, add $\max(v_p, u_q)$ to *Result* (by condition (iii), $v_p$ is $\vec{v}_i$ and $u_q$ is $\vec{v}_{i+1}$).
            (2.2.2.3) Increment $p$ by 1.
        (2.2.3) Else, increment $p$ by 1.
    (2.3) Else ($u_q <_{\text{lex}} v_p$): Symmetric to 2.2, with $V_1/v_p/p$ and $V_2/u_q/q$ exchanged.
(3) Return *Result*.
```

To see that it runs in linear time, note that at least one of p and q is incremented in each iteration of every loop, and all loops end when one of them reaches the end of the respective input sequence. Since $k = 2$, all comparions ($<_{\text{lex}}$, $\prec$ and $=$) can be done in constant time. A more careful ordering of the cases in the algorithm should permit us to ensure that the vectors of the comax set are generated in lexicographic order, and thereby linear time computation of nested comax expressions.

Lemma 22 also implies that we can find the non-dominated cost vectors within a lexicographically ordered set in linear time, since the set of cost vectors dominated by some $\vec{v}_i \in V$, and not by any other cost vector in V, must follow consecutively after v_i. Hence, with a small change, the algorithm above can be applied to input sets of cost vectors that are not filtered for dominance within each set, without increasing its complexity: at each point where p is incremented, instead of advancing it by 1 we can simply advance it to the index of the next cost vector in V_1 not dominated by v_p (and likewise for $q/V_2/u_q$). This maintains the invariant that v_p is non-dominated within V_1 (resp. u_q within V_2).

References

Geißer, F., Haslum, P., Thiébaux, S., & Trevizan, F. (2022). Admissible heuristics for multi-objective planning. In *Proc. 32nd International Conference on Automated Planning and Scheduling (ICAPS)*, pp. 100–109.

Haslum, P., Helmert, M., Bonet, B., Botea, A., & Koenig, S. (2007). Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proc. AAAI'07*, pp. 1007 – 1012.

Holte, R., Felner, A., Newton, J., Meshulam, R., & Furcy, D. (2006). Maximizing over multiple pattern databases speeds up heuristic search. *Artificial Intelligence*, *170*, 1123–1136.

Mandow, L., & Pérez-de-la-Cruz, J. (2010). Multiobjective A* search with consistent heuristics. *Journal of the ACM*, *57*(5), 27:1–27:25.

Ren, Z., Zhan, R., Rathinam, S., Likhachev, M., & Choset, H. (2022). Enhanced multi-objective A* using balanced binary search trees. arXiv:2022.08992v3 [cs.AI].

Roijers, D. M., Vamplew, P., Whiteson, S., & Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *Journal of AI Research*, *48*, 67–113.

Seipp, J., Keller, T., & Helmert, M. (2020). Saturated cost partitioning for optimal classical planning. *Journal of AI Research*, *67*, 129–167.

Stewart, B. S., & White, III, C. C. (1991). Multiobjective A*. *Journal of the ACM*, *38*(4), 775–814.

Zhang, H., Salzman, O., Felner, A., Kumar, T. K. S., Skyler, S., Ulloa, C. H., & Koenig, S. (2023). Towards effective multi-valued heuristics for bi-objective shortest-path algorithms via differential heuristics. In *Proc. 16th International Symposium on Combinatorial Search (SoCS)*, pp. 101–109.