Approximate Scalarizations with Application to Multi-Objective Path Planning

Nils Wilde[1] and Dylan A. Shell[2]

[1]Dalhousie University [2]Texas A&M University

Abstract. Useful robot behavior must inevitably strike a balance between a variety of criteria. In multi-objective optimization, the Pareto front is used to represent the set of reasonable choices, with every element being preferred for some prioritization of the criteria. Unfortunately, the widely used method of linear scalarization misses points whenever the Pareto front is non-convex. Although this deficiency can be addressed via Pareto-complete methods like the Chebyshev scalarization, doing so can exact a substantial computational penalty: indeed, some robotics-relevant problems lose the practical efficiency they exhibit under linear scalarization. We propose a straightforward and intuitive idea, namely, the use of p-norm–based scalarizations, whose completeness improves as $p \to \infty$. This paper explores the basis for the method's attractiveness. A combination of theoretical and empirical results help explain the p-norm's surprising effectiveness especially with moderately small values of p. The key theoretical insight involves considering a notion of regret and showing that the worst-case regret is not, as one might presume, at the points of "deepest" non-convexity. Our experiments consider path search problems with Dijkstra-like methods (i.e., where dynamic programming aggregates across stages). For such methods, though one might need, in the worst-case, to keep exponentially many candidates for potential extension, when using moderate p values one can do well with far fewer.

Keywords: Multi-objective optimization · Approximately Pareto-complete scalarization · Path search.

1 Introduction

Though we wish many things from our robots (e.g., energy efficiency, smoothness of motion, safety, longevity), practical and physical constraints mean that such objectives are often in conflict. We thus expect that, when a robot plans its actions, some compromise must be struck. The framework of multi-objective optimization provides the notion of the Pareto front in order to express the set of sensible compromises. Each element of the set is an appropriate choice for some relative prioritization or weighting of criteria. After gleaning insights about the possible trade-offs involved—as is concisely expressed by the Pareto front itself—designers, or users may then specify the compromise they wish. While the Pareto front corresponds to a specific problem instance, the human input is most useful when it is some stable preference capable of generalizing over instances. Hence, we take the specified compromise to be given via a weighting

of criteria, rather than a mere point on the Pareto front, so that satisfactory solutions to other problems instances may be found using that same weighting.

Probing the Pareto front is, accordingly, an important problem. The most widely used method is to combine objectives into a single number via a weighted summation (WS). This linear scalarization has the advantage of being easy to understand and it can be efficient to compute in practice as it reduces the problem to a standard cumulative scalar cost. Also, all points reported by the method are elements of the Pareto front. The converse does not hold: even after exhaustively trying all weights, the method will fail to report Pareto points whenever the front is non-convex. To regain these *blind spots*, some methods employ a different scalarization, for instance that of weighted maximization (WM) also known as Chebyshev weighting. A multi-objective planning extension to RRT* using this scalarization was presented in [13]. Recent work by one of the authors [12] shows very directly how much more expressive the outputs found by WM are over those of WS, but also that WM becomes NP-hard when planning on graphs.

In both WS and WM, for a given candidate (e.g., a path found by a planner) the costs associated to each criterion are multiplied by weights, and then a single scalar result is obtained via summation or a 'max' operation, respectively. One can also consider the continuum using the p-norm, $\|\mathbf{x}\|_p \coloneqq (\sum_{i=1}^{n} |x_i|^p)^{1/p}$, as with $p = 1$ we obtain plain summation, while as p approaches ∞, the p-norm approaches the maximum operation. Intuitively, increasing p could buy better coverage of the Pareto front at the price of more computation. The aim of this paper is to explore this idea, showing that it not only holds true, but that fortune smiles because multiple aspects align quite favourably. Specifically, we will examine the marginal increase in coverage, formulating this question in terms of a notion of worst-case regret. It becomes clear that a small increase in p above 1 can yield a large reduction in regret; this result holds generally.

All the same, fast and exact solvers for cases where $p \neq 1$ are rare. We turn attention to path search problems in particular. In a timely result, Carlson et al. [5] proved there are no constant-factor approximations for the p-norm for shortest (cumulative lengthed) paths. Whenever using techniques that employ dynamic programming to aggregate across stages, as in Dijkstra-like methods, intractability may be seen as the demand to maintain an outsize collection of candidate sub-paths for potential extension. Imposing a polynomial-sized bound on the working memory must sacrifice optimality on some instances. The experimental data we present in Section 5 suggest that increasing p tends to increase memory demands, so being able to get away with small p values would appear to reduce the likelihood sub-optimal results, assuming a fixed budget of space.

Related Work A growing body of multi-objective techniques have been proposed for planning and optimization problems in robotics. These include papers which address the problem of constructing or sampling the *entire* Pareto front. For instance [6], employ Markov Chain Monte Carlo techniques to obtain good diversity of samples on the front. The method presented in [4] iteratively select samples to reduce inaccuracy using a notion of regret so that samples spread to improve coverage where it matters most; their idea of regret-based formulation

prefigures our adoption of the concept, though we do not make use of it for selection of sample locations in the same way. The authors of [10] tackle the problem of finding complete representations of Pareto fronts in the context of path search on graph.

Given both the use of weights to express general preferences and WM's completeness mentioned above, the present paper is focused on approximate surrogates for WM designed to interrogate part of the Pareto front corresponding to some pre-provided weighting. Other work in the literature focusing on returning a single specific solution to a multi-objective problem includes the modification of RRT* by [9], who provide a neutral compromise solution rather than one based on expressing a preference across objectives.

Finally, the works of [2] and [8] are highly relevant. The former reports on a Dijkstra-like algorithm to find approximate p-norm shortest paths. And the paper of Li et al. [8, Appendix A], written thereafter, constructs an example showing that for $p = O(\log n)$, that the solution is only $O(n)$-approximate. The negative result of [5] on the absence of constant-factor approximations, shown subsequently, was mentioned above.

Contributions Before turning to the detailed problem formulation, we give a brief inventory of contributions: (i) This paper characterizes worst-case Pareto fronts for WS approaches and examines examples. (ii) It gives tight approximation factors when using WS or, more generally, a p-norm scalarization for $p \geq 1$ as a surrogate for WM optimization. (iii) The paper provides an algorithm for approximating WM with a WS solver or a p-norm heuristic, and presents numerical results based on path planning problems showing their potential.

2 Problem Formulation

2.1 Preliminaries

Basics of multi-objective optimization: Let $n \in \mathbb{N}$ with $n \geq 2$. Then a multi-objective optimization problem (MOOP) involving n objectives takes the form

$$\min_{s \in \mathcal{S}} \big(f_1(s), f_2(s), \ldots, f_n(s)\big). \tag{1}$$

One wishes to simultaneously minimize all the objectives $f_i(s)$, $i \in \{1, \ldots, n\}$, subject to whatever additional constraints are specified. We shall write $\mathcal{S}$ for the set of feasible solutions satisfying the given constraints. Solutions to MOOPs are generally not unique, so a set of Pareto-optimal solutions is sought; Pareto-optimality is defined in terms of *dominated solutions*:

Definition 1 (Dominated solution). *Let $\boldsymbol{x}$,$\boldsymbol{x}'$ be two solutions to a MOOP. Vector $\boldsymbol{x}$* dominates *$\boldsymbol{x}'$ when $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}')$ holds for all $i \in \{1, \ldots, n\}$ and there exists some $j \in \{1, \ldots, n\}$ where $f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}')$. This is denoted $\boldsymbol{x} \prec \boldsymbol{x}'$.*

The set of Pareto-optimal solutions form the *Pareto front*, defined as:

Definition 2 (Pareto front). *Given a MOOP, the set of* Pareto-optimal solutions *is the subset of all solutions that are* not dominated *by another solution. This set is referred to as the* Pareto front.

Scalarization of multi-objective optimization problems: In literature dealing with robot planning, control, and learning it is extremely common to see the conversion of optimization problems involving multiple objectives into a single objective optimization using scalarization.

Definition 3 (Parametrized scalarization). *For objectives $f_1, f_2, \ldots, f_n$, a* parametrized scalarization *is a function $h : \mathcal{S} \to \mathbb{R}$ with m tunable parameters $\boldsymbol{w} = [w_1, w_2, \ldots, w_m]$, satisfying the following: Fixing $\boldsymbol{w}$, for any two solutions s and s' with $f_i(s) \leq f_i(s')$, $\forall i \in \{1, \ldots, n\}$, we have $h(s, \boldsymbol{w}) \leq h(s', \boldsymbol{w})$.*

We study the interrelation of three classes of parametrized scalarization. The first is the commonly used *weighted summation* (WS), or *linear scalarization*:

$$h^{\text{sum}}(s, \boldsymbol{w}) \coloneqq \sum_i f_i(s) \cdot w_i = \boldsymbol{f}(s) \cdot \boldsymbol{w}. \tag{2}$$

Secondly, we consider *weighted maximization* (WM), also referred to as Augmented Chebyshev scalarization

$$h^{\text{max}}(s, \boldsymbol{w}) \coloneqq \max_i f_i(s) \cdot w_i + \xi \cdot \sum_i f_i(s), \quad \text{with } 0 < \xi \ll 1. \tag{3}$$

As ξ is a small constant, the second term serves to break ties that lead to Pareto-optimal solutions. Thirdly, we consider a p-norm (PN) scalarization

$$h^{\text{p}}(s, \boldsymbol{w}) \coloneqq \sum_i f_i(s)^p \cdot w_i^p, \tag{4}$$

where $p \in [1, \infty]$ is a tunable parameter. Note that for $p = 1$, (4) is identical to (2). Also, increasing p causes the i corresponding to the largest term in the sum to have outsized influence on the overall value such that one gets closer to a max operator, with the convention being taking it to be exactly that when $p = \infty$.

We are interested in comparing the quality of solutions obtained under the different scalarizations just given, so next we define two evaluation criteria. Li et al. [7] give a comprehensive survey of indicators for assessing Pareto front approximations, all being quality metrics for differences in the objective space. The same holds true for the following:

Definition 4 (Pareto front distance). *Given two solutions $s', s^* \in \mathcal{S}$ the* Pareto front distance *is*

$$d^{PF}(s'|s^*) \coloneqq \left\| \boldsymbol{f}(s^*) - \boldsymbol{f}(s') \right\|_2. \tag{5}$$

The difference between solutions is expressed as a (Euclidean) distance between the objective values realized by those solutions. This distance may also be interpreted as applying the widely used inverted generational distance (IGD) performance indicator [1, 7], to a pair of points. The authors of [4] introduced a notion of *regret* under a specific scalarization.

Definition 5 (Regret [4]). *For a fixed scalarization h, given weight $\bar{\boldsymbol{w}}$ and some solution s', the* regret *is defined as*

$$r(s'|\bar{\boldsymbol{w}}) \coloneqq h(s', \bar{\boldsymbol{w}}) - \min_{s \in \mathcal{S}} h(s, \bar{\boldsymbol{w}}). \tag{6}$$

The regret is non-negative and, using h, evaluates the difference in $\bar{\boldsymbol{w}}$-weighted scalar cost of candidate solution s' versus the optimum solution within $\mathcal{S}$.

2.2 Problem Statement

We consider a general MOOP where we seek to find the optimal solution for a WM scalarization with fixed weight $\boldsymbol{w}$, which we formally state as

Problem 1 (WM Problem (WMP)). Given a MOOP and a fixed weight $\boldsymbol{w}$, find some solution $s^* = \arg\min_{s \in \mathcal{S}} h^{\max}(s, \boldsymbol{w})$.

3 Worst-case errors of Pareto front approximations

As introduced above, WM is complete in the sense that every any Pareto-optimal solution can be produced with some weight. But the NP-hardness WMP for shortest path problems was established by [12]. And worst-case intractably aside, WM also suffers practically since solvers for WM is hard to come by. In contrast, WS formulations are widely used [3] and solutions can be fast to obtain using available solvers. Thus, we are, interested in characterizing the gap arising from using WS or PN as a proxy for WM scalarization.

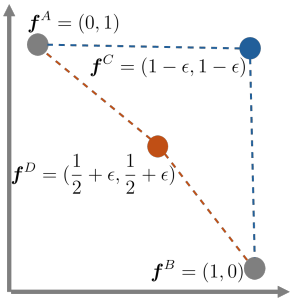


Fig. 1: Two example Pareto fronts with different worst-case approximations.

3.1 Elbows up: A didactic example

Figure 1 provides a example with minimal complexity for $n = 2$, exhibiting the boundary cases for an elbow on a Pareto front.

Sharp Elbow: Consider the blue Pareto front in Figure 1. Suppose, a user expresses that they prefer solution C. Using WM scalarization, this is attained for the preference weight $\boldsymbol{w}_C = [1/2, 1/2]$. However, if WS scalarization is employed, it is unable to return C since solution C is not on the convex hull. For $\boldsymbol{w}_C = [1/2, 1/2]$, WS will return either s^A or s^B. We evaluate those with respect to s^C:

$$d^{PF}(s^A|s^C) = \left\| [1-\epsilon, -\epsilon] \right\|_2 = \sqrt{2\epsilon^2 - 2\epsilon^2 + 1} = \left\| [-\epsilon, 1-\epsilon] \right\|_2 = d^{PF}(s^B|s^C),$$

which, as the elbow becomes sharper, approaches 1 with $\epsilon \to 0$. The regret evaluated with $h^{\max}$ is

$$\begin{aligned} r(s^A|\boldsymbol{w}_C) &= \max\{0, 1/2 \cdot 1\} + \xi - \max\{1/2(1-\epsilon), 1/2(1-\epsilon)\} - \xi(1-\epsilon) \\ &= (1/2 + \xi)\epsilon \quad = r(s^B|\boldsymbol{w}_C). \end{aligned}$$

So the regret *diminishes* as the elbow becomes sharper (i.e., for decreasing ϵ). And $\lim_{\epsilon \to 0}(r(s^A|\boldsymbol{w}_C)) = 0$.

Shallow Elbow: Turning now to the red Pareto front in Figure 1, assume the user prefers D. This, again, is attained for $\boldsymbol{w}_D = [1/2, 1/2]$. For $\epsilon > 0$, the WS is unable to return solution s^D, and will instead produce either s^A or s^B. Here, the Pareto front distance is

$$d^{PF}(s^A|s^D) = \left\| \left[1/2 + \epsilon, -1/2 + \epsilon\right] \right\|_2 = \sqrt{2\epsilon^2 + 1/2} = d^{PF}(s^B|s^D),$$

and now, as the elbow straightens with $\epsilon \to 0$, the distance approaches $\frac{1}{\sqrt{2}}$. While the regret is

$$\begin{aligned} r(s^A|\boldsymbol{w}_D) &= \max\{0, 1/2 \cdot 1\} + \xi - \max\{1/2(1/2 + \epsilon), 1/2(1/2 + \epsilon)\} - \xi(1 - \epsilon) \\ &= 1/4 - (1/2 + \xi)\epsilon \quad = r(s^B|\boldsymbol{w}_D). \end{aligned}$$

And the regret in this instance approaches $1/4 > 0$ as $\epsilon \to 0$.

Ramifications: Since WS is blind to all but convex points, a fairly intuitive expectation is that the more non-convex points are, the worse WS's solutions will be at approximating them. That view is reinforced both by thinking in terms of nearness, and also through the use of quality criteria, like d^{PF}, which are oblivious to how costs change quantitatively in the objective space. As the specific examples above show, that intuition is misguided. The extreme case of a sharply concave Pareto front, the blue elbow in Figure 1, leaves the user who is provided with WS's solution as a surrogate much happier than in the red case of Figure 1. Definition 5 captures directly a sense of how far the provided solution is deficient for someone expecting WM's solution.

3.2 Regret bound and approximation factor

Moving from WS to PN scalarization, we will now give a generalized version the previous example. Assume that the objectives have been normalized to lie within $[0, 1]$, so that regret is some interpretable units.

Proposition 1 (Bi-objective regret bound). *Given a bi-objective MOOP, the regret $r^{\mathsf{max}}(p)$ of a PN scalarization with fixed p is bounded as follows*

$$r^{\mathsf{max}}(p) < \frac{1}{2}(1 - 2^{-1/p}) + 2\xi. \tag{7}$$

Proof. Owing to symmetry, let $\bar{\boldsymbol{w}} = \left[1/2, 1/2\right]$, and suppose that we have $\boldsymbol{f}^A = \left[a_1, a_2\right]$ obtained under a search for the PN scalarization, and WM would have produced $\boldsymbol{f}^C = \left[c_1, c_2\right]$. The regret is

$$\begin{aligned} r(s^A|\bar{\boldsymbol{w}}) &= \max\{1/2 \cdot a_1, 1/2 \cdot a_2\} + \xi \cdot (a_1 + a_2) \\ &\qquad - \max\{1/2 \cdot c_1, 1/2 \cdot c_2\} - \xi \cdot (c_1 + c_2) \\ &= \max\{1/2 \cdot a_1, 1/2 \cdot a_2\} - \max\{1/2 \cdot c_1, 1/2 \cdot c_2\} + \Delta \\ &\approx \max\{1/2 \cdot a_1, 1/2 \cdot a_2\} - \max\{1/2 \cdot c_1, 1/2 \cdot c_2\}, \end{aligned}$$

where $-2\xi \leq \Delta \leq 2\xi$, and we write $\approx$ for equality $\pm 2\xi$.

For worst case regret, $0 \leq c_1, c_2 \leq 1$ should be chosen to minimize the amount subtracted. But they are constrained from below because $\boldsymbol{f}^A$ is obtained under PN, so $h^{\mathsf{p}}(\left[c_1, c_2\right], \bar{\boldsymbol{w}}) > h^{\mathsf{p}}(\left[a_1, a_2\right], \bar{\boldsymbol{w}})$, that is

$$(c_1)^p + (c_2)^p > (a_1)^p + (a_2)^p.$$

On the left, the constraint reflects an equal exchange of c_1 for c_2, so then $\max\{1/2 \cdot c_1, 1/2 \cdot c_2\}$ has its minimum attained with $c_1 = c_2$.

Without loss of generality owing to symmetry, suppose $a_1 \geq a_2$. Hence,

$$2(c_1)^p > (a_1)^p + (a_2)^p. \tag{8}$$

Then an optimum is obtained with $a_2 = 0$ as objective $r(s^A|\bar{\boldsymbol{w}}) \approx 1/2(a_1 - c_1)$ does not include a_2, and diminishing a_2 only loosens the constraint (8). To minimize c_1 let the constraint be tight up to equality $c_1 = (1/2)^{1/p}(a_1)$, hence: $r(s^A|\bar{\boldsymbol{w}}) \approx 1/2\big(a_1(1 - (1/2)^{1/p})\big)$. Thus, the maximum is attained when $a_1 = 1$, and that maximum is $1/2 \cdot \big(1 - 2^{-1/p}\big)$. Adding the maximal correction for the ignored tie-breaking terms, yields the result. □

Another useful figure of merit is a relative error in the costs obtained, expressed as a ratio. (This is similar to the relative regret proposed in [11]).

$$R(p, \bar{\boldsymbol{w}}) = \frac{\max_{s'} h^{\mathsf{max}}(s', \bar{\boldsymbol{w}})}{h^{\mathsf{max}}(\arg\max_s h^p(s, \bar{\boldsymbol{w}}), \bar{\boldsymbol{w}})}. \tag{9}$$

Proposition 2 (Bi-objective approximation factor). *Consider a bi-objective MOOP. The approximation factor of a* PN *scalarization with fixed p is given by*

$$2^{1/p} \cdot (1 + 4\xi).$$

Proof. We follow the same line of reasoning and manipulations above, but with the following relative error is

$$R(p|\bar{\boldsymbol{w}}) = \frac{\max\{1/2 \cdot a_1, 1/2 \cdot a_2\} + \xi \cdot (a_1 + a_2)}{\max\{1/2 \cdot c_1, 1/2 \cdot c_2\} + \xi \cdot (c_1 + c_2)} \leq \frac{\max\{1/2 \cdot a_1, 1/2 \cdot a_2\} + 2\xi}{\max\{1/2 \cdot c_1, 1/2 \cdot c_2\}}.$$

The result follows the same justifications, the denominator is minimized with $c_1 = c_2 = (1/2)^{1/p}$, and the numerator maximized with $a_1 = 1$, $a_2 = 0$. □

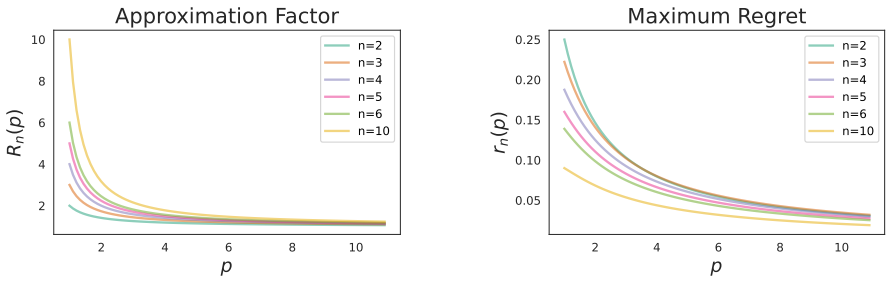
We also give analogous expressions for n-dimensional problems.

Proposition 3 (Regret bound). *Given a MOOP with n-objective functions, the regret $r^{\mathsf{max}}(p)$ of a* PN *scalarization with fixed p is bounded as follows*

$$r^{\mathsf{max}}(p) < \frac{1}{n}\big(1 - n^{-1/p}\big) + n\xi.$$

Proposition 4 (Approximation factor). *Given an MOOP with n objective functions, the approximation factor of a* PN *scalarization with fixed p is given by*

$$n^{1/p} \cdot (1 + n^2\xi).$$



(a) Approximation factor. **(b)** Regret.

Fig. 2: Worst case regret and approximation factor of PN scalarization (Proposition 3 and 4) for increasing p and number of objectives n. WS is represented by $p = 1$.

We omit their proofs because the follow the previous arguments, mutatis mutandis, but starting from $\bar{\boldsymbol{w}} = \left[1/n, \ldots, 1/n\right]$. Plots of the regret and approximation factors are provided in Figure 2.

4 Approximate Solutions for Weighted Maximization

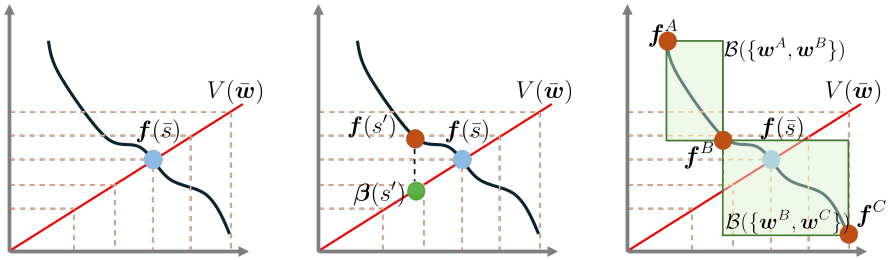
For many applications in robot planning, Problem 1 is computationally intractable [12]. Motivated by the worst-case approximation results just provided, we now propose a heuristic solution to Problem 1 using WS or PN scalarization.

There are two key design parameters. The first one is the value of p. As observed in Section 3, PN approaches WM as p is taken to ∞. Yet, the intractability of WM extends to PN for $p > 1$. For instance, path search on graphs, is NP-hard under both, WM and PN [5,8]. A key challenge for WM is that it is *blind* to all but the largest cost term. Yet, heuristic solvers can be effective for small values of p, since they are less prone to this short sightedness. Section 5 will provide empirical support for this statement. Together with the quickly diminishing approximation factor established in Proposition 4, this motivates approximate algorithms with small p. However, note that as p decreases, the best approximation might not be obtained using the given $\boldsymbol{w}$. This can be observed in a numerical example with two solutions s^A and s^B where $\boldsymbol{f}^A = (1, 0)$ and $\boldsymbol{f}^B = (.5, .3)$. Using weight $\boldsymbol{w} = (.3, .7)$, the optimal WM solution is s^B since $h^{\max}(s^A, \boldsymbol{w}) = .3$ and $h^{\max}(s^B, \boldsymbol{w}) = .21$. Yet, we have $h^{\text{sum}}(s^A, \boldsymbol{w}) = .3$ and $h^{\text{sum}}(s^B, \boldsymbol{w}) = .36$. The solution s^B can also be obtained with WS, but requires a weight where $w_1 > 3/7$. This suggests our second design parameter: the weight $\boldsymbol{w}$. To obtain the best approximation of WS or PN with small p we need to explore *surrogate weights*.

4.1 Virtual solutions

To approximate a WM solution for a fixed $\boldsymbol{w}$, we introduce *virtual solutions* as a gadget to bound the regret. We define the *virtual line* of $\boldsymbol{w}$ as

$$V(\boldsymbol{w}) = \left\{ \boldsymbol{f} \in \mathbb{R}_{\geq 0} \;\middle|\; f_i w_i = f_{i+1} w_{i+1} \text{ for all } i \in \{1, \ldots, n-1\} \right\}. \tag{10}$$



(a) Solution and iso-cost lines. **(b)** Regret bound for solution s'. **(c)** Solutions and Bounding-Box.

Fig. 3: Virtual line and regret bounds for WM.

We illustrate this in Figure 3a: Dotted orange lines illustrate *iso-cost* lines, i.e., lines of feature vectors $\boldsymbol{f}$ with the same cost under $h^{\max}$ such that $f_i w_i = f_{i+1} w_{i+1} = c$ for varying constants c, the red line is the virtual line $V(\boldsymbol{w})$. From the definition of the WM cost we have

$$\begin{aligned} &h^{\max}(s, \boldsymbol{w}) \geq f_i(s) w_i \text{ for all } i \in \{1, \dots, n\}, \text{ and} \\ &h^{\max}(s, \boldsymbol{w}) = f_j(s) w_j \text{ for at least one } j \in \{1, \dots, n\}. \end{aligned} \tag{11}$$

We distinguish two cases: In case (i) the non-zero–weighted cost terms are equal, i.e., $h^{\max}(s, \boldsymbol{w}) = f_i(s) w_i$ for all $w_i > 0$. In this case, the point $\boldsymbol{f}(s)$ lies on the virtual line $V(\boldsymbol{w})$ (as in Figure 3a). In case (ii) at least one cost does not attain the maximum such that $h^{\max}(s, \boldsymbol{w}) > f_i(s) w_i$ for some i where $w_i > 0$. This is attained when the Pareto front is discontinuous and the virtual line does not intersect the Pareto front. Without solving the WM problem it is not known which case holds. However, we observe that for any solution s that falls under case (ii), there exist a *hypothetical* cost vectors that satisfy case (i), i.e., is on $V(\boldsymbol{w})$, and tightly approximates the cost of s. To that end, we introduce the notion of an *virtual solution*:

Definition 6 (Virtual solution). *Given a weight $\boldsymbol{w}$ and some solution s, the virtual solution $\sigma(s)$ has a cost vector $\boldsymbol{f}(\sigma)$ where $f_i(\sigma(s)) = {}^{h^{\max}(s,\boldsymbol{w})}/_{w_i}$ for all non-zero w_i, and 0 otherwise.*

Virtual solutions may not correspond to actual solutions to the planning problem, but are useful to derive bounds when approximating WM. We observe that $\max_i f_i(\sigma(s)) w_i = \max_i f_i(s) w_i$. Thus, the WM cost of s and its virtual solution $\sigma(s)$ are equal besides a difference in the tie-breaking term in (3). Since the tie-breaking term is negligibly small, it can be bounded by some arbitrarily small non-zero constant ϵ. The cost difference then is

$$h^{\max}(\sigma(s), \boldsymbol{w}) \leq h^{\max}(s, \boldsymbol{w}) + \epsilon. \tag{12}$$

Now, let $r_\sigma^{\max}$ be the WM regret when using a virtual solution, i.e., $r_\sigma^{\max}(s', \bar{\boldsymbol{w}}) = h^{\max}(s', \bar{\boldsymbol{w}}) - \min_{s \in \mathcal{S}} h^{\max}(\sigma(s), \bar{\boldsymbol{w}})$. This leads to the following result:

Lemma 1 (Virtual Regret). *Given* WM *weight* $\bar{\boldsymbol{w}}$ *and some solution* s'*, then*

$$r^{\max}(s', \bar{\boldsymbol{w}}) \leq r^{\max}_{\sigma}(s', \bar{\boldsymbol{w}}) + \epsilon. \tag{13}$$

The lemma follows directly from (12) and the definition of regret. Hence, given some solution s' and a weight $\boldsymbol{w}$, we can tightly upper-bound the WM-regret $r^{\max}(s', \bar{\boldsymbol{w}})$ by only considering solutions on the virtual line.

4.2 Virtual WM Regret Bounds

Using virtual regret allows us to tightly approximate the worst-case regret of some solution s'. Let $j = \arg\min_i f_i(s')\bar{w}_i$, i.e., the index of minimal cost of s', weighted by $\bar{\boldsymbol{w}}$. We define $\boldsymbol{\beta}(s')$ as the point on the virtual line $V(\bar{\boldsymbol{w}})$ where $\beta_j(s') = f_j(s')$. We observe that $\beta_i(s') \leq f_i(s')$ holds for all i: By definition of the virtual line, $\beta(s')_i\bar{w}_i = \beta(s')_j\bar{w}_j$. Since j is the index of the minimum weighted cost, we have $\beta_i(s') \leq f_i(s')$ for all i. Figure 3b illustrates $\boldsymbol{f}(s')$, $V(\bar{\boldsymbol{w}})$, and $\boldsymbol{\beta}(s')$. The point $\boldsymbol{\beta}(s')$ provides a tight approximation of regret:

Proposition 5 (Max. WM regret). *Given a weight* $\bar{\boldsymbol{w}}$ *and a Pareto-optimal solution* s'*, the* WM *regret of* s' *with respect to* $\bar{\boldsymbol{w}}$ *is tightly bounded at* $\boldsymbol{\beta}(s')$ *by*

$$r^{\max}(s', \bar{\boldsymbol{w}}) \leq \max_i f(s')_i\bar{w}_i - \beta(s')_i\bar{w}_i + \epsilon. \tag{14}$$

Proof. Lemma 1 established that the regret with respect to $\bar{s}$ is approximated to within ϵ by using virtual solutions; thus we only consider points on $V(\bar{\boldsymbol{w}})$.

We now show that the bound is correct. Assume that there exists a point $\boldsymbol{x}$ in $V(\bar{\boldsymbol{w}})$ with higher regret than $\boldsymbol{\beta}$, implying that $x_\ell < \beta_\ell(s')$ where ℓ is the index attaining the maximum in (14). By the definition of $V(\bar{\boldsymbol{w}})$, this implies that $x_i < \beta_i(s')$ for all indices i. Since $\beta_i(s') \leq f_i(s')$, it follows that $\boldsymbol{x}$ *dominates* $\boldsymbol{f}(s')$, contradicting that s' was Pareto-optimal.

Lastly, we show that the bound is tight. Suppose there is some $\boldsymbol{x}$ in $V(\bar{\boldsymbol{w}})$ where $x_\ell > \beta_\ell(s')$. This increases the substracted term in (14), and thus can only decrease the maximum regret $r^{\max}$. Hence, $\boldsymbol{\beta}(s')$ provides a tight bound. □

Proposition 5 implies that if a Pareto-optimal solution s' is in the set $V(\bar{\boldsymbol{w}})$, it is an optimal solution to Problem 1. Thus, Problem 1 reduces to finding a Pareto-optimal solution s' that is *closest* to $V(\bar{\boldsymbol{w}})$, where closeness is measured by the WM regret, i.e., the maximum weighted difference between $f_i(s')$ and $f_i(\bar{s})$ along all cardinal directions $i \in \{1, \ldots, n\}$.

This motivates a search-based approach that seeks to find Pareto-optimal solutions as close to $V(\bar{\boldsymbol{w}})$ as possible. Borrowing from [4] we define a *neighbourhood* N as a collection of n linearly independent scalarization weights, and $\mathcal{C}(N)$ as its convex hull. Further, given a solver for a scalarization h, let $\Omega^h(N) = \{s_1, \ldots, s_n\}$ be the set of solutions returned by the solver for the weights in N. Let $\boldsymbol{\ell}(N)$ be the vector containing the minimum value for each cost attained by solutions $\Omega^h(N)$, i.e., $\ell_i(N) = \min_{s \in \Omega^h(N)} f_i(s)$, and let $\boldsymbol{u}(N)$ be the vector of respective maximum values. This defines a bounding box

$$\mathcal{B}(N) = \left\{ \boldsymbol{f} \in \mathbb{R}_{\geq 0} \,\middle|\, \ell_i(N) \leq f_i \leq u_i(N) \text{ for all } i \in \{1, \ldots n\} \right\}. \tag{15}$$

An example is provided in Figure 3c. To design a search algorithm, we make an assumption on the quality of a solver, for a given scalarization.

Assumption 1 (Intermediate solution)**.** Given a solver for a scalarization h and a neighbourhood N, with solutions $\Omega^h(N) = \{s_1, \dots, s_n\}$, for any weight $\boldsymbol{w}'$ in $\mathcal{C}(N)$ the solver returns a solution s' where $\boldsymbol{f}(s')$ lies in the bounding box $\mathcal{B}(N)$.

This is a weaker assumption than having an optimal solver. Instead, it requires that when optimizing for an intermediate weight, the resulting solution is not dominating the neighbourhood solutions, nor is it dominated by them.

Under Assumption 1 we can compute a lower bound on the WM regret of a solution s' for some weight $\boldsymbol{w}'$ inside $\mathcal{C}(N)$, i.e., a point inside $\mathcal{B}(N)$ that minimizes (14). We refer to this as the *min-max regret of a neighbourhood* $\eta(N, \bar{\boldsymbol{w}})$, which can be written as an optimization problem $\eta(N, \bar{\boldsymbol{w}}) = \min_{\boldsymbol{x} \in \mathcal{B}(N), \boldsymbol{p} \in V(\bar{\boldsymbol{w}})} \max_i f(s')_i \bar{w}_i - p_i \bar{w}_i$. Since the max is taken over n discrete indices, this can be reformulated as a linear program:

$$
\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{p}} \ \alpha \\
&s.t.\ (x_i - p_i)\bar{w}_i \le \alpha && \text{//minimize maximum difference} \\
&\quad p_j \bar{w}_j = p_{j+1} \bar{w}_{j+1} && \text{//ensuring } \boldsymbol{p} \in V(\bar{\boldsymbol{w}}) \\
&\quad \ell_i(N) \le x_i \le u_i(N), && \text{//ensuring } \boldsymbol{x} \in \mathcal{B}(N) \\
&\quad i = 1, \dots, n,\ j = 1, \dots, n-1.
\end{aligned}
\tag{16}
$$

Thus, for any neighbourhood, we can evaluate the *optimistic* potential outcome for sampling another weight in it. In the example of Figure 3c, we can identify the solutions to (16): For the neighbourhood $\{\boldsymbol{w}^A, \boldsymbol{w}^B\}$ the minimum regret point is $\boldsymbol{f}(s^B)$, while in $\{\boldsymbol{w}^B, \boldsymbol{w}^C\}$ the minimum regret point is any point on $V(\bar{\boldsymbol{w}})$. Thus, $\eta(\{\boldsymbol{w}^A, \boldsymbol{w}^B\}, \bar{\boldsymbol{w}})$ is given by (14), while $\eta(\{\boldsymbol{w}^B, \boldsymbol{w}^C\}, \bar{\boldsymbol{w}}) = 0$.

4.3 Sampling Algorithm

We now design a sampling-based algorithm to solve Problem 1. The principle is similar to the regret-based Pareto sampling in [3, 4]: We begin with the single objective solutions, forming an initial neighbourhood, and then recursively identify the best candidate weight to minimize the WM regret bound from (14), refining the neighbourhood.

We refer to our proposed method the MINIMUM-REGRET APPROXIMATION (MRA) and outline it in Algorithm 1. We begin by computing the n-single objective solutions, which forms an initial neighbourhood (lines 1–2). For all current neigbourhoods $\mathcal{N}$, we select the neighbourhood N with minimal min-max regret $\eta(N, \bar{\boldsymbol{w}})$, which we obtain from the LP in (16). We then select a new candidate weight $\boldsymbol{w}'$ as the geometric midpoint of N, compute the corresponding solution using the given solver, and add it to a collection of sampled solutions (lines 4–7). The selected neighbourhood is then split into n new neighbourhoods by substituting $\boldsymbol{w}'$ for each of the weights of N (lines 8–10). When the sampling budget T is exhausted, we return the solution $\boldsymbol{w}'$ that attain the best WM cost among all sampled weights (line 11).

Algorithm 1: Minimum-Regret Approximation (MRA)

Input: WM objective weight $\bar{\boldsymbol{w}}$, WS or PN solver, planning budget T
Output: Solution s' approximating $\min_s h^{\max}(s, \boldsymbol{w})$

Compute basis solutions $s(\boldsymbol{b}^1), \dots, s(\boldsymbol{b}^n)$ with single objective solver
$\mathcal{N} \leftarrow \{\big(s(\boldsymbol{b}^1), \dots, s(\boldsymbol{b}^n)\big)\}$, $W \leftarrow \{\boldsymbol{b}^1, \dots, \boldsymbol{b}^n\}$
for $t = n$ *to* $T - n$ **do**
  Select N from $\mathcal{N}$ with smallest min-max regret $\eta(N, \bar{\boldsymbol{w}})$ // LP in (16)
  Get mid-point weight $\boldsymbol{w}'$ of the neighbourhood N
  Obtain $s(\boldsymbol{w}', \boldsymbol{f}(s(\boldsymbol{w}')$ from WS or PN solver
  $W \leftarrow W \cup \{\boldsymbol{w}'\}$
  for $\boldsymbol{w}^i$ *in* N **do**
    $N^i \leftarrow N \setminus \{\boldsymbol{w}^i\} \cup \{\boldsymbol{w}'\}$ // Substitute weight
    $\mathcal{N} = \mathcal{N} \cup N^i$
return $\boldsymbol{w}' = \arg\min_{\boldsymbol{w} \in W} h^{\max}(s(\boldsymbol{w}), \bar{\boldsymbol{w}})$

5 Numerical Results

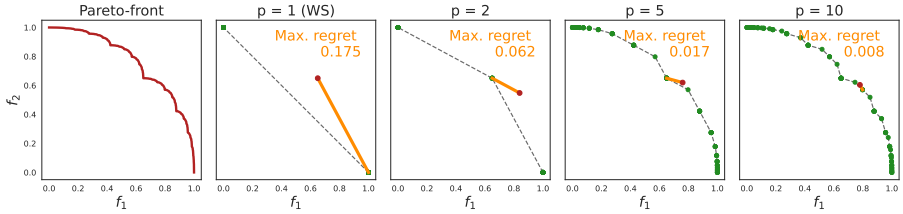
We present results on a set of simulation experiments to showcase the approximation bounds of WS and PN from Section 3 as well as the efficacy the proposed sampling-based algorithm from Section 4.

5.1 Explicatory Instances with Synthetic Pareto fronts

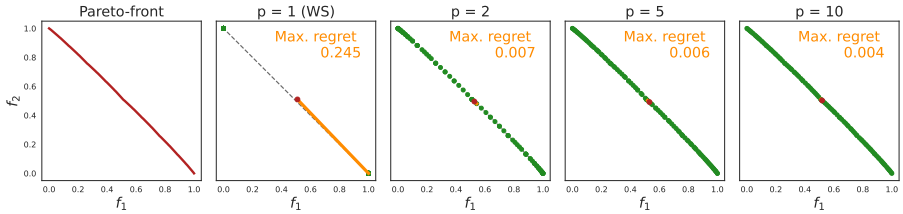
First, we consider a synthetic example for a non-convex Pareto front to highlight the effect of different p-values in PN under ideal conditions. Instead of solving a planning problem, we explicitly construct Pareto fronts with a fractal shape. Examples appear in Figure 4. They are constructed by starting with two extremal points and then recursively placing additional points on their separating hyperplane; a tunable parameter $\alpha \in [0, 1]$ can move the added point between the most convex and most concave configuration. Given a weight, optimal solutions under any scalarization can be found by iterating over samples of the fractal.

For the *sharp* Pareto front in Figure 4a, we observe that $p = 1$ (i.e., WS) is only able to attain the endpoints, but increasing p allows a greater proportion of the Pareto front to be recovered. Moreover, the maximum regret reduces very quickly, with $p = 5$ already providing a high quality approximation. The plot in Figure 4b shows a *shallow* Pareto front. Here WS is still only able to attain the endpoints, but even $p = 2$ closely approximates the entire Pareto front. Noticeably, the maximum regret for WS is drastically increased compared to the *sharper* case in 4a, confirming the worst-case characterized in Section 3.

In Section 4, we observed that approximation of WM for a fixed $\bar{\boldsymbol{w}}$ can require searching over suitable surrogate weights $\boldsymbol{w}'$ as well as p. We highlight this interconnection in Figure 5, using a modification of the fractal algorithm with randomized α value. Plot 5a shows that with increasing p and using $\boldsymbol{w}' = \bar{\boldsymbol{w}}$ we are able to obtain more accurate approximations of WM, with $p = 10$ being close to optimal. However, plots 5b and 5c reveal that equally good approximations were also attainable for any $p \geq 2.6$ by adjusting the surrogate weight



(a) Sharp Pareto-front, $\alpha = .65$.



(b) Shallow Pareto-front, $\alpha = .51$.

Fig. 4: Fractal Pareto fronts and approximations with p-norms. Orange lines connect the point on the Pareto-front (red) with highest regret and its approximate solution.

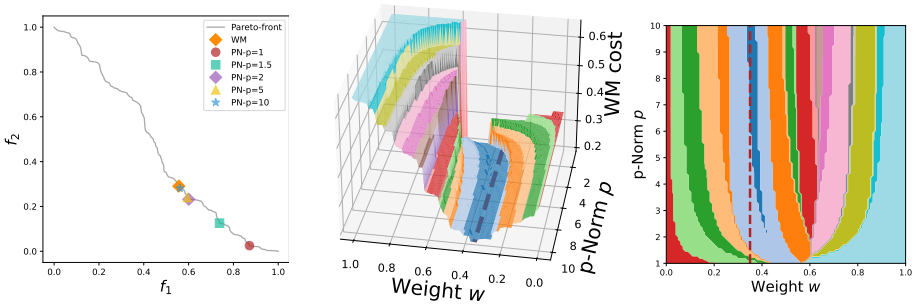
$\boldsymbol{w}'$ accordingly. Furthermore, the same solution as originally found for $p = 5$ was actually attainable for WS. This demonstrates the benefits of searching for surrogate weights when finding an approximate solution to WM.

5.2 Path Search

To evaluate the proposed MRA algorithm, we tackle Problem 1 for a multi-objective shortest path problem, similar to the experiment in [12]. We consider a concrete multi-objective shortest path search on a PRM graph with 500 vertices and connections to up to 6 neighbours per vertex. We design two problem variants, Setup 1 and Setup 2, shown in Figure 6. There are two cost functions, path length and closeness to obstacles; both are summed over all edges in a path.

To solve WM and PN we use the Dijkstra-style algorithm presented in [12]. To avoid exponential expansion, the method uses a memory budget K for the number of stored predecessor paths for every vertex. We compare the following methods: The WM approach with $K = 1$ and $K = 5$ (WM-K1 and WM-K5), the WS approach using the given weight $\bar{\boldsymbol{w}}$ (WS-FIX), and our proposed MRA algorithm with weight budget $T = 10$, using scalarizations WS and PN with $p = 2$ and $K = 1$ (MRA-WS and MRA-PN, respectively). Experiments are implemented in Python and executed on a Intel I9.

Tractability of PN: Before comparing the solution quality of different WM approximations, we demonstrate that optimizing PN for small values of p has computational benefits compared to WM. We sample 100 weights $\bar{\boldsymbol{w}}$ and compute a stand-in for the optimal solution for WM and PN with $p \in \{1.5, 2, 3, 4\}$ using a large memory budget $K = 100$. We then solve the problem for smaller values of K. Figure 7 reports the ratio in cost, compared to $K = 100$. In Setup 1,

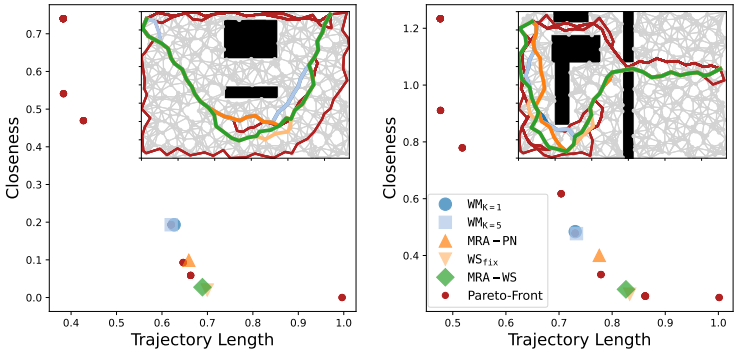


(a) Example approximations with fixed $\boldsymbol{w}$ and varying p.

(b) WM cost over $\boldsymbol{w}$ and p.

(c) WM cost over $\boldsymbol{w}$ and p (projection).

Fig. 5: Example approximation of a WM solution for fixed weight $\bar{\boldsymbol{w}} = [.35, .65]$ with different weights $\boldsymbol{w}'$ and p-norms. In plots (b) and (c), the dotted line shows the $\bar{\boldsymbol{w}}$.



(a) Setup 1.

(b) Setup 2.

Fig. 6: Experiment setup for path search.

all methods perform comparably for small K. Yet, PN substantially outperforms WM for $K \geq 7$ and is close to optimal for $K = 15$, while WM makes little progress for $K \geq 5$. In Setup 2, the difference between WM and PN is even more prevalent. For most K, PN performs substantially stronger, especially so for small p. Only at larger memory budgets of $K = 30$, WM becomes comparable to PN in terms of closeness to optimality. We observe that smaller values of p consistently perform strongest. This can be understood by considering the bottleneck in Setup 2. Since WM disregards the second highest weighted cost term, a low memory budget may lead to choosing a locally optimal path for the left half of the problem, which, unfortunately, turns out to be a poor subpath for the full problem.

Solution Quality: We now consider the how well different methods solve Problem 1 with respect to solution quality and runtime, again using 100 sampled weights $\bar{\boldsymbol{w}}$. Quality is captured in the cost ratio, compared to WM with a large memory budget of $K = 100$. The runtime ratio compares to a single-objective execution of Dijkstra's algorithm, minimizing path length.

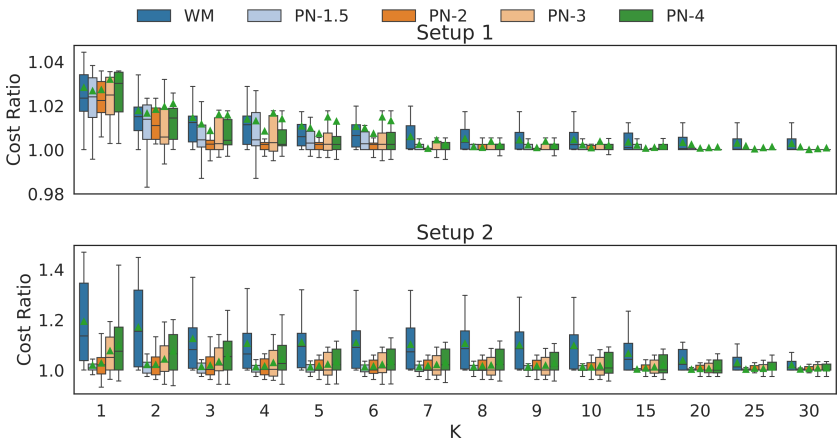


Fig. 7: Effect of small memory budgets K on the heuristic search for PN and WM.

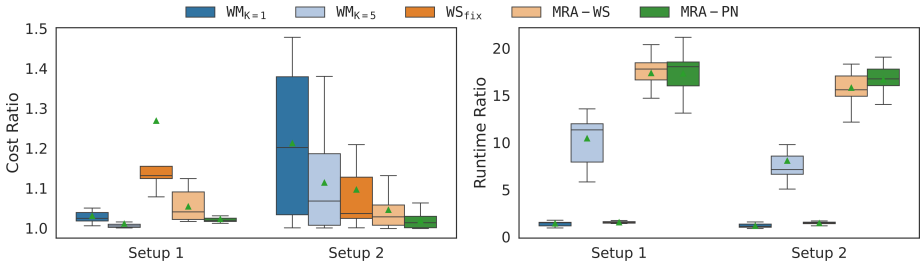


Fig. 8: Numerical results for path search across compared methods.

Figure 8 shows the main results. In Setup 1 WM-K1 and WM-K5 find close to optimal solutions, clearly outperforming both WS methods. Since WS only explores convex parts of the Pareto front it sometimes can only find a poor approximation for large subsets of WM weights. MRA-WS performs substantially better than WS-FIX, highlighting that exploring different weights can lead to better approximations. MRA-PN performs comparably to WM-K1 and WM-K5 with respect to the cost ratio, yet the exploration of up to $T = 10$ weights takes it toll on the runtime. WM-K1 and WS-FIX are comparable to simple Dijkstra's, while both variants of MRA increase the runtime by almost a factor of 20. Yet, it is noteworthy that this still results in < 0.5 s on the tested system.

The contrast shown between Setup 1 and Setup 2 is stark. Both WM approaches show a high cost ratio, owing to the challenge of the bottlenecked environment: this causes all paths to go through the same vertex, and the limited memory budget at that vertex restricts the diversity of available subpaths for WM as it minimizes the largest cost, disregarding other cost terms during exploration. Indeed, even WS-FIX performs better as it always finds Pareto-optimal solutions, while the solutions returned by WM-K1 and WM-K5 can be dominated. As before, MRA-WS is able to further improve the performance of WS by finding suitable surrogate weights. MRA-PN performs strongest, finding close to optimal solutions and showing smaller deviations than MRA-WS. This

illustrates that in addition to tuning the weights, having $p > 1$ enables exploration of non-convex parts of the Pareto front and thus better approximation of WM. The runtimes closely resemble the result of Setup 1.

In summary, we observe that MRA-PN has the virtue of being consistent in finding high quality solutions. In contrast, WS is challenged by non-convex Pareto fronts. While WM heuristics may be strong in some settings, their performance can also be quite poor due, ultimately, to their tractability issues, as already observed in Figure 7. Accordingly, the increased runtime of MRA-PN is justified by the consistent performance providing strong approximations of WM solutions.

6 Discussion

Summary: We studied approximate scalarization methods for multi-objective planning and provided the surprising result that *shallow* Pareto fronts are posing the hardest challenge for the commonly used WS. Further, we characterized the worst-case regret and approximation factor for PN as a function of p and the number of objectives n. Finally, we derived a heuristic algorithm to approximate WM using WS and PN, and showcased its potential in simulations.
Dirty Laundry: While our theoretical findings provide a strong motivation for PN formulations in planning, our heuristic approach itself does not come with an approximation factor due to the intractability of PN. Further, while the heuristic graph search is less computationally demanding for PN than WM it still increases substantially over simple path search which may become prohibitive for large or high-dimensional problems. Specialized planning algorithms for PN could close the gap and guarantee worst-case bounds. Our experiments focused only on a synthetic example and graph based planning; yet, the PN approach has potential to be applied to a range of other planning and control problems in robotics. Lastly, our work focused on only approximating the WM solution for a single, given weight and does not recover the entire Pareto front. Since the proposed heuristic is already exploring different weights, this could be a natural extension.
Conclusion: An obvious motivation for employing the p-norm scalarization is that its ability to uncover parts of the Pareto front improves as p increases; however, our investigation shows that the approach is particularly effective for *small* values of p. This is a tale of two approximations with effects which are felicitous, not only individually but also in their interaction.

First, small values of p suffice to have a marked reduction in the significance of the neglected regions on the Pareto front. So, while it is guaranteed to never miss solutions, it is rather that those which may be overlooked are less important. This is directly observable in terms of regret, an indicator that appears to be considerably less misleading to intuition than some others. Secondly, because obtaining exact solutions under the p-norm is expensive, it is natural to settle for sub-optimal solvers. In the cases we studied that, even for challenging problem instances, small p values make better use of limited memory budgets. An improvement of the approximation quality is possible by increasing the value of p, but only at substantial increases in memory requirements. In fact, running a larger p to improve the p-norm's approximation of the $\max(\cdot)$, without also meeting the demand for increased memory, leads to worse overall results.

References

1. Bezerra, L.C., López-Ibánez, M., Stützle, T.: An Empirical Assessment of the Properties of Inverted Generational Distance on Multi- and Many-Objective Optimization. In: International Conference on Evolutionary Multi-Criterion Optimization. vol. 10173, pp. 31–45. Springer (2017)
2. Bilò, V., Caragiannis, I., Fanelli, A., Flammini, M., Monaco, G.: Simple greedy algorithms for fundamental multidimensional graph problems. Dagstuhl Reports **80**(125), 1–13 (2017)
3. Botros, A., Sadeghi, A., Wilde, N., Alonso-Mora, J., Smith, S.L.: Error-Bounded Approximation of Pareto Fronts in Robot Planning Problems. In: International Workshop on the Algorithmic Foundations of Robotics. pp. 506–522. Springer (2022)
4. Botros, A., Wilde, N., Sadeghi, A., Alonso-Mora, J., Smith, S.L.: Regret-based sampling of pareto fronts for multi-objective robot planning problems. IEEE Trans. Robotics **40**, 3778–3794 (2024)
5. Carlson, C., Makarychev, Y., Mosenzon, R.: Hardness of approximation for shortest path with vector costs. In: Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). pp. 3905–3935. SIAM (2026)
6. Lee, J., Yi, D., Srinivasa, S.S.: Sampling of pareto-optimal trajectories using progressive objective evaluation in multi-objective motion planning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1–9 (2018)
7. Li, M., Yang, S., Liu, X.: Diversity Comparison of Pareto Front Approximations in Many-Objective Optimization. IEEE Transactions on Cybernetics **44**(12), 2568–2584 (2014)
8. Li, S., Xu, C., Zhang, R.: Polylogarithmic approximation for robust st path. arXiv preprint arXiv:2305.16439 (2023)
9. Rahman, M.M., Bobadilla, L., Rapp, B.: Sampling-based planning algorithms for multi-objective missions. In: IEEE International Conference on Automation Science and Engineering (CASE). pp. 709–714 (2016)
10. Ren, Z., Hernández, C., Likhachev, M., Felner, A., Koenig, S., Salzman, O., Rathinam, S., Choset, H.: EMOA*: A framework for search-based multi-objective path planning. Artificial Intelligence **339**, 104260 (2025)
11. Wilde, N., Kulić, D., Smith, S.L.: Active preference learning using maximum regret. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 10952–10959. IEEE (2020)
12. Wilde, N., Smith, S.L., Alonso-Mora, J.: Scalarizing multi-objective robot planning problems using weighted maximization. IEEE RA-L **9**(3), 2503–2510 (2024)
13. Yi, D., Goodrich, M.A., Seppi, K.D.: MORRF*: Sampling-Based Multi-Objective Motion Planning. In: IJCAI. pp. 1733–1741 (2015)