How Many Subproblems? A Controlled Study of Decomposition Size in Multi-Policy Multi-Objective Reinforcement Learning

1st Neele Kemper
Organic Computing Group
University of Augsburg
Augsburg, Germany
0000-0003-2110-1006

2nd Jonathan Wurth
Organic Computing Group
University of Augsburg
Augsburg, Germany
0000-0002-5799-024X

3rd Michael Heider
Organic Computing Group
University of Augsburg
Augsburg, Germany
0000-0003-3140-1993

4th Jörg Hähner
Organic Computing Group
University of Augsburg
Augsburg, Germany
0000-0003-0107-264X

***Abstract*—Decomposition-based multi-objective reinforcement learning reduces a multi-objective problem to a set of scalar subproblems, each defined by a weight vector on the preference simplex. While this approach has proven effective, the number of weight vectors is typically chosen heuristically without systematic justification. This paper presents a controlled empirical study examining how the number of weight vectors affects Pareto front quality under fixed training budgets. A multi-policy decomposition framework is evaluated with Soft Actor-Critic and Proximal Policy Optimization, training one independent policy per weight vector via linear scalarization across configurations from two to twenty-one weight vectors and six MuJoCo tasks with two and three objectives. A Bayesian temporal ranking model quantifies uncertainty about which configuration is optimal throughout training, and the best-found configurations are compared against state-of-the-art baselines. The results show that the optimal decomposition size fundamentally depends on the algorithmic structure. Off-policy learning favors small decompositions, with four weight vectors serving as a robust default. A shared replay buffer enables information exchange between subproblems and ensures high sample efficiency. On-policy learning operates without replay, requiring larger decompositions that scale with the training budget and produce less pronounced optima. The common default of six weight vectors is suboptimal for both algorithms, and minimal corner weight configurations lead to significant performance degradation. With properly tuned configurations, the proposed implementations outperform all baselines.**

***Index Terms*—Multi-Objective Reinforcement Learning, Decomposition-Based Optimization, Multi-Policy Learning, Weight Vector Selection**

I. Introduction

Many sequential decision problems involve multiple conflicting objectives. For example, an autonomous robot must balance speed and energy consumption. Multi-objective reinforcement learning (MORL) addresses this by learning a set of policies representing different trade-offs, analogous to approximating a Pareto front in multi-objective optimization (MOO). A common approach is decomposition. MOO algorithms such as MOEA/D [1] decompose a multi-objective problem into multiple scalar subproblems, each defined by a weight vector λ on the preference simplex and a scalarization function $g : \mathbb{R}^m \to \mathbb{R}$. This principle can be directly transferred to multi-policy MORL [2], where one policy is trained per weight vector and snapshots are periodically added to a Pareto archive retaining only non-dominated candidates. The number of subproblems k controls a fundamental tension: larger k improves coverage of the preference simplex, but leaves less new data for each subproblem under a fixed computational budget. In classical decomposition-based MOO, do not require sequential trajectory collection and are typically cheaper to add than in MORL. In MORL, however, policy improvement depends on expensive environment interaction, making this coverage–compute trade-off particularly pronounced. Existing methods often use small, fixed values (typically $k \approx 6$) without systematic justification [3], [4], [5]. How k interacts with learning dynamics, sampling efficiency, and objective dimensionality has not yet been characterized.

This paper presents a controlled empirical study on the role of k in decomposition-based multi-policy MORL, systematically varying k while fixing scalarization (weighted sum), base learner (Soft Actor-Critic and Proximal Policy Optimization), and weight distribution. Experiments span six MuJoCo locomotion tasks with $m \in \{2, 3\}$ objectives, evaluated using Hypervolume, Expected Utility, Cardinality, and Sparsity. A Bayesian temporal ranking model quantifies uncertainty about the optimal k throughout training, and the best-found configurations are compared against established MORL baselines.

Our main contributions are: (1) A controlled empirical study isolating the effect of decomposition size k on Pareto-front quality and coverage, using a Bayesian temporal ranking model to quantify configuration uncertainty throughout training; (2) Evidence that optimal k is algorithm-dependent: off-policy learning favors small, stable decompositions, while on-policy learning requires larger, budget-dependent configurations–with both corner weights and the common default ($k = 6$) proving suboptimal. (3) Comparisons show that optimized decompositions outperform the MORL baselines in all environments, alongside an ablation study that quantifies the influence of the evaluation interval and critic formulation. Code is available at https://github.com/NeeleKemper/morl-d-subproblems

II. Literature Review

Decomposition-based MORL reduces a multi-objective Markov decision process (MOMDP) to k scalar subproblems, each defined by a weight vector and scalarization. MORL/D [2] frames this as an outer-loop wrapper with design choices of scalarization, weight handling, and cooperation mechanisms such as shared replay buffers. There are two architectural paradigms: multi-policy approaches train a separate network per weight vector [6], [2], [3], [4], [5], while general-policy approaches train a single preference-conditioned network over the weight simplex [7], [8], at the cost of gradient interference between conflicting preferences [9]. We focus on the multi-policy setting with k independently trained agents. Unless otherwise stated, MORL refers to this setting.

In MOO, reference directions discretize the preference simplex and thereby control both the approximation resolution and the computational effort. NSGA-III [10] uses a predefined reference set to preserve diversity, while MOEA/D uses reference directions to define scalar subproblems [1]. Weight generation methods range from structured lattices like Das-Dennis [11] to flexible-cardinality approaches such as Riesz-s-energy optimization [12]. Prior work on MORL has focused on *how* a selected weight set can be utilized through scalarization design, experience reuse, or adaptive refinement, rather than examining the effect of k itself. Van Moffaert and Nowé [13] show that weighted sums dominate due to Bellman compatibility, although Chebyshev scalarizations can better cover non-convex fronts. Subsequent methods differ mainly in how they allocate training effort: informative weight selection for linear utility functions [14], envelope-style value reuse across preferences [15], prediction-guided and active preference selection [5], [7], evolutionary refinement [6], [4], archive densification via Pareto-ascent gradients [16] or constrained policy extension [3], and divide-and-conquer with achievement scalarizing functions [17].

Despite these advances, k is typically chosen heuristically, with values around $k = 6$ being used without systematic justification. Previous studies have not isolated the effect of k under controlled conditions, which motivates our empirical investigation.

III. Background

The multi-objective sequential decision problem is modeled as an MOMDP [18], where each transition yields a vector reward $\vec{r}_t \in \mathbb{R}^m$. Each policy π induces a vector-valued discounted return

$$\vec{v}^{\pi} = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \vec{r}_t\right] \in \mathbb{R}^m, \tag{1}$$

where v_j^{π} denotes the expected discounted return for objective $j \in 1, \ldots, m$. Because objectives may conflict with each other, policies are compared using Pareto dominance:

$$\pi \succ \pi' \iff \left(\forall j : v_j^{\pi} \geq v_j^{\pi'}\right) \wedge \left(\exists j : v_j^{\pi} > v_j^{\pi'}\right). \tag{2}$$

A policy is Pareto-optimal if it is not dominated by any other policy. The corresponding Pareto set $\mathcal{P}^{\star}$ and Pareto front $\mathcal{F}^{\star}$ are

$$\mathcal{P}^{\star} = \pi : \nexists \pi' \text{ s.t. } \pi' \succ \pi, \qquad \mathcal{F}^{\star} = \vec{v}^{\pi} : \pi \in \mathcal{P}^{\star}. \tag{3}$$

To approximate $\mathcal{F}^{\star}$, the vector-valued problem is decomposed into scalar subproblems via scalarization. Under the Scalarized Expected Return (SER), a scalarization function g and weight vector $\lambda \in \Delta^{m-1}$ define a scalar utility $u_{\lambda}(\pi) = g(\vec{v}^{\pi}; \lambda)$, which for linear g reduces to:

$$u_{\lambda}(\pi) = \lambda^{\top} \vec{v}^{\pi} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \lambda^{\top} \vec{r}_t\right]. \tag{4}$$

IV. Methodology

To isolate the effect of k on Pareto front quality, we implement a controlled decomposition-based MORL framework with fixed computational budgets and systematically vary only the number of subproblems.

A. Weight Vector Generation

The multi-objective problem is decomposed into k scalar subproblems, each defined by a weight vector $\lambda_i \in \Delta^{m-1} = \{\lambda \in \mathbb{R}_+^m \mid \sum_{j=1}^m \lambda_j = 1\}$, where m is the number of objectives. The weights are fixed throughout training (no weight adaptation) and are generated deterministically using nearly uniform simplex constructions. For $m = 2$, we use the Das–Dennis simplex lattice design [11]. In this case the construction produces exactly k evenly spaced points on the line segment $\lambda_1 + \lambda_2 = 1$ by setting the number of partitions to $k - 1$. For $m = 3$, Das–Dennis restricts k to triangular numbers, thus we use the layered Riesz-s-energy construction [12] for $m = 3$, which offers finer-grained k selection while maintaining uniform coverage. It optimizes a multilayer simplex scaling to obtain a well-distributed set of reference directions for a given number of points. Figure 1 illustrates the resulting distributions and the corresponding layer configurations.

B. Decomposition-based MORL Framework

A decomposition-based MORL approach is implemented as an outer-loop wrapper [2]. At initialization, k fixed weight vectors $\{\lambda_i\}_{i=1}^k \subset \Delta^{m-1}$ are sampled and one independent actor-critic agent is trained per weight vector (no weight adaptation and no parameter sharing). Each agent optimizes the SER. To ensure fair comparison across k, all policies are evaluated at fixed timestep intervals and their return vectors are stored alongside policy snapshots in a Pareto archive $\mathcal{A}$.

a) Multi-Objective Soft Actor-Critic (MO-SAC): MO-SAC agents are off-policy actor-critic learners based on SAC [19]. Each agent i receives vector rewards $\vec{r} \in \mathbb{R}^m$ and learns two vector-valued critics $\vec{Q}_{\phi_{i,1}}, \vec{Q}_{\phi_{i,2}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^m$ together with an actor π_{θ_i} and target critics $\vec{Q}_{\bar{\phi}_{i,1}}, \vec{Q}_{\bar{\phi}_{i,2}}$ updated by Polyak averaging. All agents interact with the environment in parallel and write transitions $(s, a, \vec{r}, s', d)$ to a shared replay buffer $\mathcal{D}$. For each update, a single minibatch

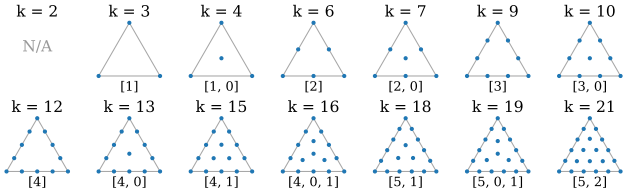

Fig. 1. Layer-wise Riesz s-energy reference directions for $m = 3$. The points lie on the 2-simplex; k denotes the number of weight vectors. The labels below each simplex indicate the corresponding layer configuration.

$(s, a, \vec{r}, s', d) \sim \mathcal{D}$ is shared across all agents. Each agent computes scalarized target scores

$$u_{i,j}(s', a') = g\Big(\vec{Q}_{\bar{\phi}_{i,j}}(s', a'); \lambda_i\Big), \quad j \in \{1, 2\}, \tag{5}$$

selects $j^\star \in \arg\min_j u_{i,j}(s', a')$, and forms the vector Temporal Difference (TD) target

$$\vec{y}_i = \vec{r} + (1 - d)\gamma\, \vec{Q}_{\bar{\phi}_{i,j^\star}}(s', a'). \tag{6}$$

The critic loss minimizes per-objective MSE between both predictions and $\vec{y}_i$:

$$\mathcal{L}^{(i)}_{\text{crit}} = \mathbb{E}\left[\left\|\vec{Q}_{\phi_{i,1}}(s, a) - \vec{y}_i\right\|^2\right] + \mathbb{E}\left[\left\|\vec{Q}_{\phi_{i,2}}(s, a) - \vec{y}_i\right\|^2\right]. \tag{7}$$

Scalarization is not applied inside the Bellman backup (unlike approaches that scale reward vectors directly [2]). Instead, vector-valued critics are maintained throughout, with scaling applied only for target network selection and policy optimization. Entropy is omitted from the critic backup so that $\vec{Q}$ represents discounted return vectors; entropy regularization is retained only in the actor loss:

$$\mathcal{L}^{(i)}_{\text{actor}} = \mathbb{E}\left[\alpha \log \pi_{\theta_i}(a|s) - \min_j g\Big(\vec{Q}_{\phi_{i,j}}(s, a); \lambda_i\Big)\right]. \tag{8}$$

b) Multi-Objective Proximal Policy Optimization (MO-PPO): MO-PPO agents are on-policy learners based on PPO [20]. Each agent i maintains a policy $\pi_{\theta_i}(a|s)$ and a vector-valued critic $\vec{V}_{\phi_i} : \mathcal{S} \to \mathbb{R}^m$ predicting expected returns per objective. During training, rollouts of length T are collected, yielding vector rewards $\vec{r}_t \in \mathbb{R}^m$ at each timestep. Per-objective returns $\vec{R}_t$ are computed using Generalized Advantage Estimation. The scalarization function g is applied to both vector returns and value predictions to obtain a scalar learning signal:

$$A^{\lambda_i}_t = g(\vec{R}_t; \lambda_i) - g(\vec{V}_{\phi_i}(s_t); \lambda_i). \tag{9}$$

The policy is updated using the standard clipped PPO surrogate objective:

$$\mathcal{L}^{(i)}_{\text{policy}} = -\mathbb{E}_t\Big[\min\Big(r_t(\theta_i)A^{\lambda_i}_t,\ \text{clip}(r_t(\theta_i), 1-\epsilon, 1+\epsilon)A^{\lambda_i}_t\Big)\Big], \tag{10}$$

where $r_t(\theta_i) = \pi_{\theta_i}(a_t|s_t)/\pi^{\text{old}}_{\theta_i}(a_t|s_t)$ denotes the probability ratio. The value function is trained on vector returns using per-objective MSE:

$$\mathcal{L}^{(i)}_{\text{value}} = \frac{1}{2}\mathbb{E}_t\Big[\|\vec{V}_{\phi_i}(s_t) - \vec{R}_t\|^2\Big], \tag{11}$$

with optional value clipping as in standard PPO. This preserves the full multi-objective structure in the critic, enabling accurate per-objective value estimates regardless of the scalarization function used for policy updates. The total loss combines policy loss, value loss, and entropy regularization:

$$\mathcal{L}^{(i)} = \mathcal{L}^{(i)}_{\text{policy}} + c_v\mathcal{L}^{(i)}_{\text{value}} - c_e H(\pi_{\theta_i}), \tag{12}$$

where c_v and c_e are coefficients for value loss and entropy bonus, respectively. Both MO-SAC and MO-PPO use Pareto archives with no capacity limit.

C. Experimental Setup

Six multi-objective MuJoCo-v5 locomotion tasks from MO-Gymnasium [21] are evaluated: four with $m = 2$ objectives (HalfCheetah, Humanoid, Swimmer, Walker2d) and two with $m = 3$ objectives (Ant, Hopper). For each environment, 30 independent seeds are run across $k \in \{2, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18, 19, 21\}$, excluding $k = 2$ for $m = 3$ (since $k \geq m$ is required). The upper bound of $k = 21$ is chosen such that each agent receives at least 500K environment steps under the fixed total budget of 10.5M interactions, balancing sufficient per-agent compute with a tractable experimental scale while capturing the dominant trends in how k affects Pareto-front quality. To ensure that differences are solely due to the distribution of experience across subproblems, the ratio of updates to data is kept constant across all k. Linear weighted-sum scalarization is used throughout; all other hyperparameters remain constant and follow standard values [3], [5], [2].

At each evaluation checkpoint t, four metrics are computed on the Pareto archive $\mathcal{A}_t$: Hypervolume, measuring convergence and coverage; Expected Utility under uniform preferences, measuring average performance across weightings; Cardinality $|\mathcal{A}_t|$, counting non-dominated solutions; and Sparsity, measuring evenness of front coverage (lower is better; higher is better for all others). All Pareto fronts are normalized per environment using shared objective-wise min–max bounds across all runs, seeds, and methods, thereby removing reward-scale differences between objectives while preserving Pareto dominance. The Hypervolume is computed using the reference point $(-0.1, \ldots, -0.1)^m$, placing it 10% below the empirical minimum in each objective.

Results are compared against four MORL baselines at the 3M-step checkpoint: C-MORL [3], PG-MORL [5], and MORL/D [2] as multi-policy approaches, and CAPQL [8] as the sole general-policy baseline. All baselines use linear scalarization and are evaluated using their original reported hyperparameters.

a) Bayesian Ranking Model: To obtain probabilistic statements about which subproblem count $k \in \mathcal{K}$ performs best at each checkpoint despite noisy metric estimates, a Bayesian temporal Plackett–Luce (PL) ranking model [22] is employed. For each evaluation time t and seed, all $|\mathcal{K}|$ candidates are ranked by their metric values (descending for higher-is-better metrics and ascending for lower-is-better

metrics), yielding a permutation π_t. Rankings are modelled via

$$p(\pi_t \mid \mathbf{u}_t) = \prod_{i=1}^{K-1} \frac{\exp(u_{t,\pi_{t,i}})}{\sum_{j=i}^{K} \exp(u_{t,\pi_{t,j}})}, \tag{13}$$

where $u_{t,k}$ is the latent utility of candidate k at time t. To capture gradual changes during training, an independent Gaussian process prior $u_{\cdot,k} \sim \mathcal{GP}(0, \kappa)$ with a Matérn-3/2 kernel is placed on each candidate's utility, coupling inference across nearby checkpoints. From posterior samples obtained via Markov chain Monte Carlo, two quantities are computed: (1) ratings $r_{t,k} = \mathrm{softmax}(\mathbf{u}_t)_k$, the normalized win probability of k at time t, and (2) $P(\text{best} = k \mid t)$, the posterior probability that k has the highest win rate among all configurations, estimated as the proportion of posterior samples where $k = \arg\max_{k'} u_{t,k'}$.

V. Results and Discussion

The Hypervolume is reported as the primary quality metric and Cardinality as a measure of archive coverage. Expected Utility and Sparsity show analogous trends (see Appendix[1]). Environment-specific and metric-specific posterior rankings of the Bayesian temporal model are also included in the Appendix.

A. Decomposition Size and Algorithmic Structure

The effect of decomposition size is strongly algorithm-dependent. Given a fixed total interaction budget, increasing k improves coverage of the preference space but reduces the amount of new interaction data available to each scalarized subproblem to roughly $1/k$. The practical implications of this trade-off depend on whether experience can be reused across subproblems. In MO-SAC, the shared replay buffer allows transitions collected under one preference to support many subsequent updates across subproblems, substantially reducing the effective cost of larger k. In MO-PPO, rollout batches are reused only within the PPO epoch loop and then discarded, so performance depends much more directly on the on-policy data budget per subproblem. Overall, replay makes MO-SAC less sensitive to decomposition size. In MO-PPO, the preferred k shifts upward over training: small k dominates early when per-subproblem data scarcity is the binding constraint, while larger k becomes favorable later when coverage of the preference space gains importance. More generally, the same k corresponds to much finer coverage in 2-objective problems than in 3-objective problems. Therefore decomposition sizes should always be interpreted relative to the dimensionality of the preference simplex.

In MO-SAC, the replay ensures that small decompositions are highly effective for front quality (Figures 2–4). Globally, the Hypervolume is maximized by $k = 4$ from 0.5M to 7.5M steps with near-certain posterior support ($P(\text{best}) \geq 0.998$) and remains the best choice at 10.5 million ($P(\text{best}) = 0.776$). In 2-objective environments, preferred decompositions are consistently small. Swimmer prefers $k = 2$, while HalfCheetah, Humanoid, and Walker2d favor $k = 3$ with generally strong support. The main exceptions are the 3-objective tasks, where covering a Pareto surface requires more weight vectors than covering a Pareto curve. Toward the end of training, Ant shifts strongly toward larger k values and ends at $k = 15$, while Hopper increases more gradually and ends at $k = 19$. Cardinality and Sparsity generally favor large decompositions ($k = 19$–21), because denser preference coverage yields broader coverage of the approximated front. Humanoid is the only exception and prefers a medium value ($k = 7$, $P(\text{best}) = 0.59$), which is consistent with its small Pareto archive of approximately 30 solutions. Walker2d shows no clear Cardinality optimum, with posterior mass spread across multiple k values throughout training (Figure 3).

MO-PPO shows a qualitatively different pattern (Figures 5–7). Without replay, large decompositions are disadvantaged early as many subproblems receive too little on-policy data to make meaningful progress. As the budget increases, this per-subproblem data scarcity weakens and the advantage of finer preference-space coverage becomes increasingly important. Accordingly, the globally preferred Hypervolume decomposition shifts upward over training: $k = 4$ at 0.5M, $k = 7$ at 1.5M, $k = 12$ at 3M, and $k = 21$ at 10.5M. Unlike MO-SAC, this k-shifting persists even in 2-objective tasks (e.g., Walker2d: $2 \to 6 \to 9 \to 21$). Swimmer is the only stable exception that prefers $k = 6$, which appears to be due more to lower variance than to a large mean advantage over neighboring decompositions. MO-PPO is also characterized by much flatter posterior rankings than MO-SAC, with $P(\text{best})$ often in the range 0.5–0.8, which is consistent with smaller performance differences and higher seed sensitivity across neighboring k values. For diversity, MO-PPO generally prefers medium decompositions ($k \approx 12$–16) rather than the largest values, although Ant and Humanoid retain substantial posterior mass on multiple competitors (Figure 6). Smaller k also converges faster in MO-PPO but at the cost of final quality. In MO-SAC this effect is weaker, consistent with replay reducing sensitivity to decomposition size.

The corner-weight setting $k = m$ performs poorly in both algorithms because it allocates one subproblem to each simplex vertex, training only extreme preferences and collapsing the approximation set toward extreme trade-offs. This reduces Hypervolume, drastically lowers Cardinality, and worsens Sparsity. For 3-objective problems this is particularly restrictive, as three corner weights cover only the vertices of a Pareto surface. The effect is mild only in the simplest 2-objective environment. In Swimmer, MO-SAC tolerates $k = 2$ with essentially no Hypervolume loss and otherwise incurs only small deficits. For 3-objective problems, the degradation is substantial relative to the best non-corner decomposition at the same budget. MO-SAC loses approximately 15–33% Hypervolume (Hopper $\approx 15\%$, Ant $\approx 33\%$), while MO-PPO loses approximately 22–64% (Hopper $\approx 22\%$, Ant $\approx 64\%$). Cardinality drops by 57–85% relative to the best non-corner configuration, recovering only a small fraction of

[1] https://zenodo.org/records/19479525

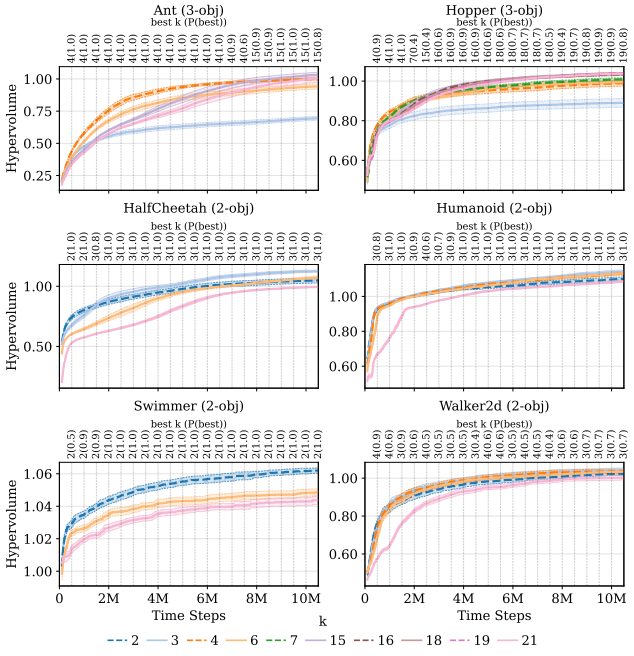

Fig. 2. Hypervolume learning curves for **MO-SAC** across all environments under a fixed budget of 10.5M steps. Top axis: best k with posterior probability $P(\text{best})$ at each 500K checkpoint. Only k values achieving optimality at any point are shown, alongside $k \in \{m, 6, 21\}$. Lines: means over 30 seeds; shaded regions: 95% bootstrap CI.

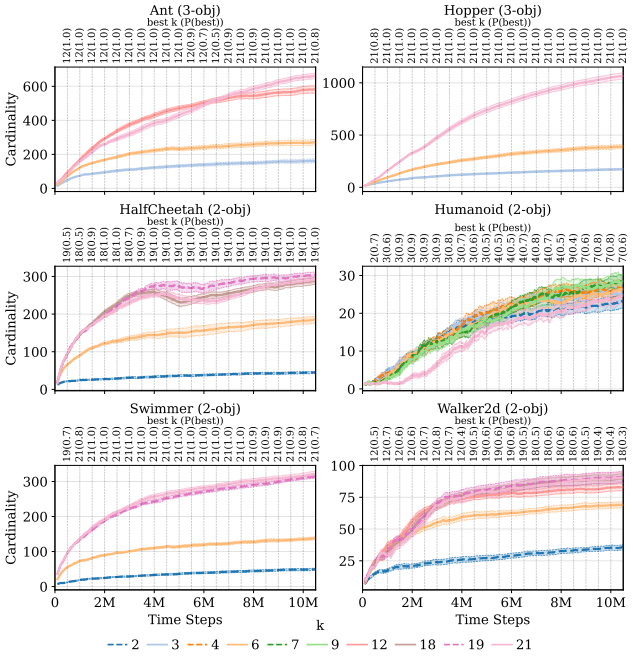

Fig. 3. Cardinality learning curves for **MO-SAC**. Conventions as in Figure 2.

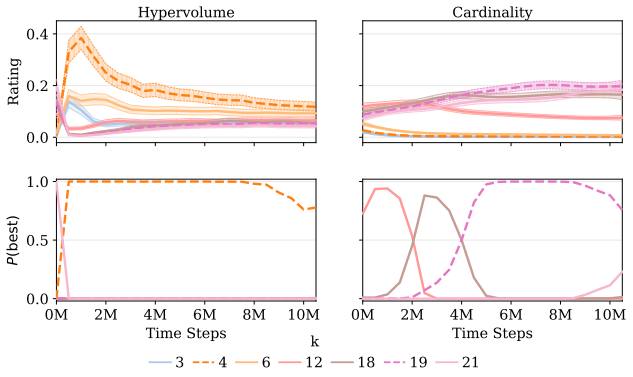

Fig. 4. Global Bayesian ranking of k for **MO-SAC** aggregated across all environments. Top row: posterior mean ratings from a Plackett-Luce Gaussian Process model with 95% credible intervals. Bottom row: posterior probability $P(\text{best})$ that each k has the highest rating. Left: Hypervolume; Right: Cardinality. Only k values achieving optimality at any timestep are shown alongside $k \in \{m, 6, 21\}$.

non-dominated solutions. The larger degradation in MO-PPO is consistent with its higher sensitivity to data scarcity in individual subproblems. Corner weights should generally be avoided in both algorithms.

These results yield clear algorithm-specific recommendations. The common default $k = 6$ is unreliable. For MO-SAC it never exceeds $P(\text{best}) = 0.25$ (below 0.05 in all environments except Walker2d), and for MO-PPO it is effective only in Swimmer and early training, but remains ineffective for 3-objective tasks ($P(\text{best}) < 0.32$). For MO-SAC, $k = 4$ is a robust default value for front quality; $k \in \{2, 3\}$ is often sufficient for 2-objective tasks, and $k = 15$–19 only becomes useful for 3-objective problems and larger budgets; for diversity, $k = 19$–21 is preferable, except for Humanoid ($k = 7$–9). For MO-PPO, no single decomposition is best throughout training: $k = 6$–7 is advantageous early, $k \approx 12$ is a reasonable mid-range compromise, and $k = 19$–21 becomes competitive only at larger budgets. Corner weights should generally be avoided in both algorithms.

B. Comparison Against Benchmarks

Having established how decomposition size shapes performance, we next compare representative MO-SAC and MO-PPO configurations against established MORL baselines. Table I reports Hypervolume after 3M environment steps for MO-SAC, MO-PPO, and four baselines: C-MORL, PG-MORL, MORL/D, and CAPQL. All multi-policy baselines are evaluated at $k = 6$, consistent with their original publications and the prevailing community standard. Reported values may deviate from original publications due to differences in MuJoCo environment versions. Note that tuning k for the baselines would likely improve their performance, which we leave as future work. For our methods, three variants are reported: *best* (oracle per-environment k) and *global* (single k across all environments), both selected by Hypervolume at 3M steps, and $k = 6$ for direct comparison at equal decomposition size. We therefore interpret the $k = 6$ results as the fairest direct comparison to the published multi-policy baselines, while the *best* and *global* variants quantify the gains available from decomposition-size selection. Additional metrics and Bayesian rankings are provided in the Appendix.

MO-SAC achieves the highest Hypervolume in 4/6 environ-

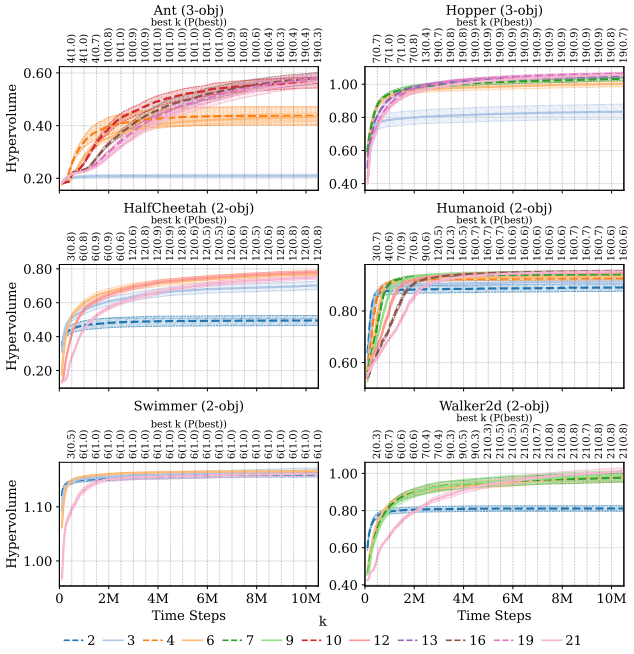

Fig. 5. Hypervolume learning curves for **MO-PPO**. Conventions as in Figure 2.

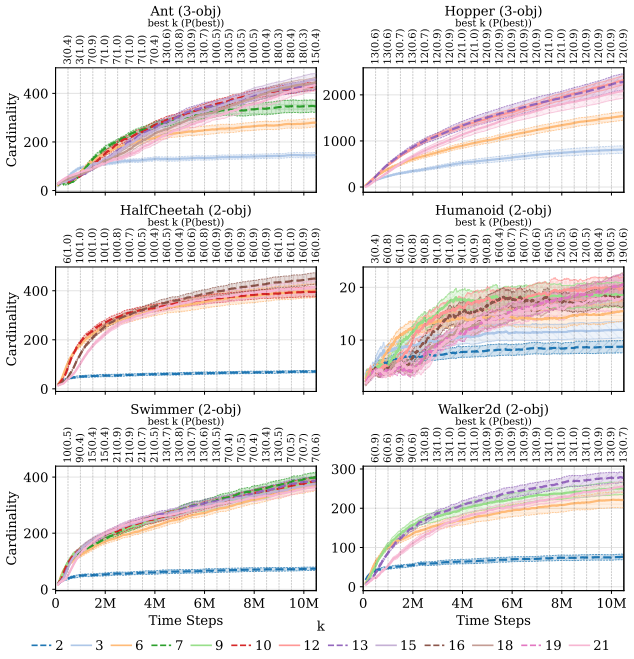

Fig. 6. Cardinality learning curves for **MO-PPO**. Conventions as in Figure 2.

ments (Ant, HalfCheetah, Humanoid, Walker2d), while MO-PPO leads in Hopper and Swimmer. The Bayesian Plackett–Luce model confirms clear performance gaps: in each environment, either MO-SAC or MO-PPO receives essentially all posterior mass ($P(\text{best}) \approx 1$), while the baselines receive negligible posterior support. The largest gain occurs in Ant, where MO-SAC achieves a Hypervolume of 0.857 compared to 0.449 for MORL/D (+91%). CAPQL, the only general-

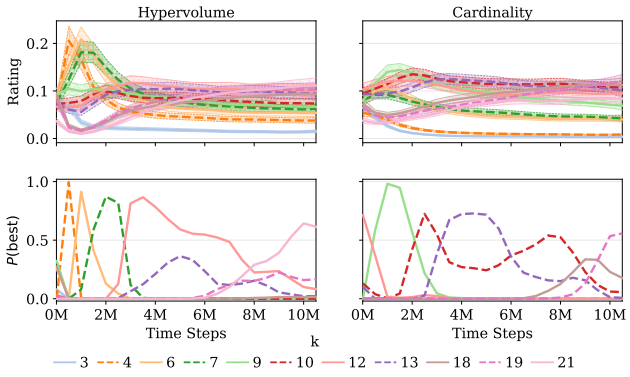

Fig. 7. Global Bayesian ranking of k for **MO-PPO**. Conventions as in Figure 4.

policy baseline, ranks last in 5/6 environments and shows substantially higher variance, consistent with an advantage of multi-policy approaches in these tasks: each policy receives an unambiguous learning signal for a single preference, while Pareto archives preserve previously discovered non-dominated solutions.

Even at $k = 6$, MO-SAC matches or exceeds the best baseline in 4/6 environments (most notably Ant: Hypervolume of 0.765 vs. 0.449), confirming that the observed gains are not solely attributable to k-tuning. Comparing *best* with *global* shows that a single global k remains viable, though sensitivity varies (e.g., HalfCheetah drops from 0.954 to 0.888 in Hypervolume).

The Hypervolume and Cardinality results together reveal a consistent trade-off between coverage and quality (Tables I–II). MO-PPO achieves the highest Cardinality in 5/6 environments, while MO-SAC attains higher Hypervolume in 4/6 environments. Since both methods use unbounded archives, this contrast is unlikely to be an artifact of archive truncation; rather, it points to a genuine algorithmic difference in how coverage and solution quality are achieved. The observed results are consistent with MO-PPO's independent on-policy rollouts producing broader but lower-quality approximation sets, while MO-SAC's shared replay enables more targeted optimization yielding fewer but higher-quality solutions. A similar but more confounded pattern emerges among the baselines: the PPO-based C-MORL and PG-MORL often find more non-dominated solutions than the SAC-based MORL/D, though this comparison should be interpreted cautiously given further differences in algorithmic approach and archiving practices. Solution count alone is therefore not a reliable proxy for Pareto-front quality.

C. Ablation Study of MO-SAC

To determine whether MO-SAC's advantage over baselines stems from algorithmic improvements or experimental design choices, we conduct two ablation studies.

1) Evaluation Frequency.: Policies are evaluated and archived every $\Delta t = 10$K steps, whereas prior work uses coarser intervals (≈30–40K steps) [2], [3], [5]. To assess

TABLE I
HYPERVOLUME COMPARISON AT 3M TIME STEPS (30 SEEDS, MEAN ± 95% CI). BOLD INDICATES BEST MEAN PERFORMANCE PER ENVIRONMENT. SUBSCRIPTS DENOTE THE k VALUE USED.

Algorithm	Ant (3-obj)	Hopper (3-obj)	HalfCheetah (2-obj)	Humanoid (2-obj)	Swimmer (2-obj)	Walker2d (2-obj)
C-MORL	0.291 ± 0.008	0.888 ± 0.012	0.585 ± 0.018	0.942 ± 0.006	1.152 ± 0.004	0.791 ± 0.016
PG-MORL	0.339 ± 0.020	0.918 ± 0.016	0.641 ± 0.022	0.939 ± 0.008	1.154 ± 0.004	0.872 ± 0.022
MORL/D	0.449 ± 0.008	0.838 ± 0.011	0.883 ± 0.008	0.945 ± 0.006	0.971 ± 0.002	0.811 ± 0.016
CAPQL	0.188 ± 0.041	0.534 ± 0.061	0.652 ± 0.089	0.804 ± 0.087	0.893 ± 0.070	0.552 ± 0.045
MO-SAC (best)	$\mathbf{0.857 \pm 0.022}_{k=4}$	$0.946 \pm 0.006_{k=16}$	$\mathbf{0.954 \pm 0.024}_{k=3}$	$\mathbf{1.036 \pm 0.005}_{k=3}$	$1.049 \pm 0.002_{k=2}$	$\mathbf{0.968 \pm 0.014}_{k=3}$
MO-SAC (k=4)	$\mathbf{0.857 \pm 0.022}$	0.929 ± 0.015	0.888 ± 0.033	1.032 ± 0.004	1.044 ± 0.001	0.964 ± 0.011
MO-SAC (k=6)	0.765 ± 0.023	0.920 ± 0.010	0.820 ± 0.029	1.024 ± 0.006	1.038 ± 0.002	0.959 ± 0.014
MO-PPO (best)	$0.453 \pm 0.019_{k=10}$	$\mathbf{1.008 \pm 0.009}_{k=19}$	$0.691 \pm 0.015_{k=12}$	$0.937 \pm 0.004_{k=12}$	$\mathbf{1.163 \pm 0.002}_{k=6}$	$0.930 \pm 0.023_{k=7}$
MO-PPO (k=12)	0.415 ± 0.012	1.004 ± 0.011	0.691 ± 0.015	0.937 ± 0.004	1.151 ± 0.005	0.898 ± 0.014
MO-PPO (k=6)	0.384 ± 0.015	0.972 ± 0.017	0.696 ± 0.019	0.928 ± 0.006	$\mathbf{1.163 \pm 0.002}$	0.920 ± 0.019

TABLE II
CARDINALITY COMPARISON AT 3M TIME STEPS. CONVENTIONS AS IN TABLE I.

Algorithm	Ant (3-obj)	Hopper (3-obj)	HalfCheetah (2-obj)	Humanoid (2-obj)	Swimmer (2-obj)	Walker2d (2-obj)
C-MORL	187.9 ± 10.0	200.0 ± 0.0	191.4 ± 7.0	7.2 ± 1.4	171.4 ± 18.4	162.4 ± 19.8
PG-MORL	63.1 ± 6.4	174.9 ± 9.1	72.3 ± 4.9	12.1 ± 1.9	57.9 ± 2.8	53.1 ± 3.6
MORL/D	98.2 ± 3.8	127.9 ± 4.7	56.9 ± 3.1	10.0 ± 0.8	34.4 ± 1.3	31.4 ± 2.4
CAPQL	29.7 ± 7.1	32.3 ± 5.6	41.5 ± 5.4	$\mathbf{20.5 \pm 10.1}$	56.0 ± 10.7	20.2 ± 6.6
MO-SAC (best)	$92.5 \pm 6.9_{k=4}$	$434.4 \pm 13.5_{k=19}$	$79.0 \pm 4.2_{k=3}$	$12.9 \pm 1.1_{k=7}$	$28.9 \pm 1.8_{k=2}$	$37.6 \pm 2.3_{k=3}$
MO-SAC (k=4)	92.5 ± 6.9	153.1 ± 6.7	117.1 ± 4.8	12.8 ± 1.2	67.3 ± 2.4	42.9 ± 1.9
MO-SAC (k=6)	203.0 ± 7.9	220.4 ± 10.2	135.9 ± 6.3	12.8 ± 1.2	100.7 ± 2.0	55.0 ± 2.8
MO-PPO (best)	$\mathbf{218.3 \pm 14.1}_{k=10}$	$956.0 \pm 59.9_{k=19}$	$287.5 \pm 18.0_{k=12}$	$14.5 \pm 1.8_{k=12}$	$198.2 \pm 10.3_{k=6}$	$164.7 \pm 13.2_{k=7}$
MO-PPO (k=12)	186.1 ± 13.5	$\mathbf{1126.3 \pm 42.0}$	287.5 ± 18.0	14.5 ± 1.8	$\mathbf{226.9 \pm 11.7}$	$\mathbf{174.3 \pm 10.1}$
MO-PPO (k=6)	193.3 ± 10.9	754.5 ± 47.3	$\mathbf{293.5 \pm 19.4}$	13.6 ± 1.4	198.2 ± 10.3	156.3 ± 9.9

whether frequent evaluation artificially inflates archive-based metrics, MO-SAC is retrained with $\Delta t \in \{50\text{K}, 100\text{K}, 250\text{K}\}$. Hypervolume degrades modestly: even at $\Delta t = 250\text{K}$ ($25\times$ fewer evaluations), MO-SAC outperforms MORL/D in 5/6 environments (Table III). Cardinality, however, collapses mechanically with fewer archiving opportunities (e.g., Ant drops from 92.5 to 18.9; Table IV). This confirms that MO-SAC's quality advantage is algorithmic, while coverage comparisons require matched Δt values for fair comparability.

2) Vector vs. Scalar Critic.: The vector-valued critic is compared against a scalar formulation in which scalarization is applied before computing the Bellman error, following the MORL/D-style update rule:

$$y_i = g(\vec{r}; \lambda_i) + (1-d)\gamma \left(\min_j g(\vec{Q}_{\bar{\phi}_{i,j}}; \lambda_i) - \alpha \log \pi \right). \quad (14)$$

In the scalar critic, the loss is computed on the scalarized TD target rather than on individual objectives. As a result, gradients flow only along the preference direction λ_i, and TD errors in objective directions orthogonal to λ_i produce no learning signal. This is particularly restrictive in 3-objective tasks. In contrast, the vector critic preserves objective-wise TD information: trained with vector Bellman targets and per-objective TD errors, each objective contributes a learning signal throughout training. The scalar critic's early aggregation of objective information is consistent with more diverse but less precisely optimized trade-offs across subproblems, which could explain its higher Cardinality in 5/6 environments despite lower Pareto quality. The vector critic achieves higher Hypervolume in 4/6 environments (Table III), with clear gains on Ant (+0.17) and Walker2d (+0.10). Notably, the scalar critic with tuned k outperforms MORL/D, confirming that decomposition size matters independently of critic architecture.

VI. CONCLUSION

This paper presents a systematic empirical analysis of decomposition-based multi-policy MORL, yielding several key findings. The optimal number of subproblems k differs fundamentally between off-policy and on-policy learning. MO-SAC's shared replay buffer enables small decompositions ($k = 4$) to achieve near-optimal Pareto front quality with high certainty, while MO-PPO requires progressively larger k as the training budget increases, with no single value optimal throughout training. The widely used default $k = 6$ is suboptimal in most settings: for MO-SAC it is never optimal, and for MO-PPO it is mainly effective in early training. Corner weights ($k = m$) should generally be avoided: they degrade quality markedly in 3-objective tasks (with substantially larger failures for MO-PPO) and collapse coverage, recovering only a small fraction of non-dominated solutions while worsening sparsity due to concentration on extreme trade-offs. Tuned decompositions make the proposed multi-policy MORL approach competitive against established baselines. MO-SAC attains the best Hypervolume in 4/6 environments (with particularly large gains on Ant), while MO-PPO wins 2/6. The ablation study on MO-SAC shows that frequent evaluation

TABLE III
MO-SAC ABLATION ON HYPERVOLUME. EFFECT OF EVALUATION INTERVAL (Δt) AND CRITIC FORMULATION ($k = 4$). RESULTS SHOW MEAN $\pm$ 95% CI. BOLD INDICATES BEST MEAN PERFORMANCE PER ENVIRONMENT.

MO-SAC (k=4)	Ant (3-obj)	Hopper (3-obj)	HalfCheetah (2-obj)	Humanoid (2-obj)	Swimmer (2-obj)	Walker2d (2-obj)
Vector Critic (Δt=10K)	$\mathbf{0.857 \pm 0.022}$	$\mathbf{0.929 \pm 0.015}$	0.888 ± 0.033	$\mathbf{1.032 \pm 0.004}$	1.044 ± 0.001	$\mathbf{0.964 \pm 0.011}$
Vector Critic (Δt=50K)	0.831 ± 0.021	0.907 ± 0.016	0.880 ± 0.033	1.025 ± 0.004	1.037 ± 0.002	0.946 ± 0.014
Vector Critic (Δt=100K)	0.816 ± 0.021	0.896 ± 0.015	0.876 ± 0.033	1.021 ± 0.004	1.034 ± 0.002	0.936 ± 0.014
Vector Critic (Δt=250K)	0.798 ± 0.020	0.877 ± 0.015	0.870 ± 0.032	1.016 ± 0.004	1.031 ± 0.002	0.920 ± 0.017
Scalar Critic (Δt=10K)	0.686 ± 0.016	0.906 ± 0.013	$\mathbf{0.987 \pm 0.004}$	0.986 ± 0.004	$\mathbf{1.066 \pm 0.001}$	0.867 ± 0.021

TABLE IV
MO-SAC ABLATION ON CARDINALITY. CONVENTIONS AS IN TABLE III.

MO-SAC (k=4)	Ant (3-obj)	Hopper (3-obj)	HalfCheetah (2-obj)	Humanoid (2-obj)	Swimmer (2-obj)	Walker2d (2-obj)
Vector Critic (Δt=10K)	92.5 ± 6.9	153.1 ± 6.7	117.1 ± 4.8	$\mathbf{12.8 \pm 1.2}$	67.3 ± 2.4	42.9 ± 1.9
Vector Critic (Δt=50K)	46.7 ± 3.5	62.0 ± 2.5	56.2 ± 2.8	7.8 ± 0.8	38.6 ± 1.6	25.2 ± 1.5
Vector Critic (Δt=100K)	31.3 ± 2.5	41.3 ± 1.7	37.5 ± 2.0	6.3 ± 0.8	28.9 ± 1.1	19.3 ± 1.6
Vector Critic (Δt=250K)	18.9 ± 1.3	22.4 ± 1.2	21.2 ± 1.2	4.6 ± 0.5	18.7 ± 1.1	13.1 ± 0.7
Scalar Critic (Δt=10K)	$\mathbf{108.6 \pm 4.4}$	$\mathbf{213.9 \pm 14.0}$	$\mathbf{122.4 \pm 3.6}$	12.0 ± 1.3	$\mathbf{63.2 \pm 2.1}$	$\mathbf{72.7 \pm 3.5}$

mainly inflates coverage metrics mechanically by adding more policy snapshots to the Pareto archive, while Hypervolume degrades only moderately. It also shows that the vector-critic update contributes to MO-SAC's advantage by preserving per-objective information, especially in the 3-objective setting.

The experiments are limited to MO-MuJoCo locomotion with two or three objectives and linear scalarization with uniform simplex sampling. The observed budget-dependent shifts in MO-PPO suggest potential benefits from dynamically adjusting k during training. Future work could explore non-linear scalarizations, adaptive weight distributions, and alternative knowledge sharing mechanisms such as shared network layers or policy distillation.

REFERENCES

[1] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, 2007.

[2] F. Felten, E.-G. Talbi, and G. Danoy, "Multi-objective reinforcement learning based on decomposition: A taxonomy and framework," *Journal of Artificial Intelligence Research*, vol. 79, 2024.

[3] R. Liu, Y. Pan, L. Xu *et al.*, "Efficient discovery of pareto front for multi-objective reinforcement learning," in *The Thirteenth International Conference on Learning Representations*, 2025.

[4] H.-L. Tran, L. Doan, N. H. Luong *et al.*, "A two-stage multi-objective evolutionary reinforcement learning framework for continuous robot control," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2023.

[5] J. Xu, Y. Tian, P. Ma *et al.*, "Prediction-guided multi-objective reinforcement learning for continuous robot control," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[6] D. Chen, Y. Wang, and W. Gao, "Combining a gradient-based method and an evolution strategy for multi-objective reinforcement learning," *Applied Intelligence*, vol. 50, no. 10, 2020.

[7] L. N. Alegre, A. L. C. Bazzan, D. M. Roijers *et al.*, "Sample-efficient multi-objective learning via generalized policy improvement prioritization," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2023.

[8] H. Lu, D. Herman, and Y. Yu, "Multi-objective reinforcement learning: Convexity, stationarity and pareto optimality," in *International Conference on Learning Representations (ICLR)*, 2023.

[9] P. Li, H. Tang, Y. Yuan *et al.*, "Cola: Towards efficient multi-objective reinforcement learning with conflict objective regularization in latent space," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[10] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, 2014.

[11] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, 1998.

[12] J. Blank, K. Deb, Y. Dhebar *et al.*, "Generating well-spaced points on a unit simplex for evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, 2021.

[13] K. Van Moffaert, M. M. Drugan, and A. Nowé, "Scalarized multi-objective reinforcement learning: Novel design techniques," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 2013.

[14] H. Mossalam, Y. M. Assael, D. M. Roijers *et al.*, "Multi-objective deep reinforcement learning," 2016.

[15] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[16] T. Hu and B. Luo, "Pa2d-morl: Pareto ascent directional decomposition based multi-objective reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[17] W. Röpke, M. Reymond, P. Mannion *et al.*, "Divide and conquer: Provably unveiling the pareto front with multi-objective reinforcement learning," in *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, 2025.

[18] C. F. Hayes, R. Rădulescu, E. Bargiacchi *et al.*, "A practical guide to multi-objective reinforcement learning and planning," *Autonomous Agents and Multi-Agent Systems*, 2022.

[19] T. Haarnoja, A. Zhou, P. Abbeel *et al.*, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[20] J. Schulman, F. Wolski, P. Dhariwal *et al.*, "Proximal policy optimization algorithms," 2017.

[21] F. Felten, L. N. Alegre, A. Nowé *et al.*, "A toolkit for reliable benchmarking and research in multi-objective reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[22] J. Wurth, H. Stegherr, N. Kemper *et al.*, "Pareto-optimal anytime algorithms via bayesian racing," 2026. [Online]. Available: https://arxiv.org/abs/2603.08493