Meta-Learning-Based Algorithm Selection for Multi-Objective Ridesharing-Inspired Optimization

Tarcisio A. P. Brito[1], **Juvenal B. A. Silva**[1], **Islame F. C. Fernandes**[1]

[1]Institute of Computing, Federal University of Bahia, Salvador, Bahia, Brazil

{tarcisio.brito, juvenal.bruno, islame.felipe}@ufba.br

Abstract. *Algorithm selection aims to identify the most suitable optimizer for a given instance, but remains challenging in multi-objective combinatorial optimization because performance depends on Pareto set characteristics. This paper investigates meta-learning for algorithm selection in a three-objective ridesharing-inspired problem integrating route planning, passenger assignment, and reward collection. We use this problem as a structured testbed derived from the Traveling Salesman Problem. We compare static, Pareto-based landscape, and combined meta-feature representations using 252 synthetic instances and three multi-objective evolutionary algorithms: MOEA/D, SPEA2, and IBEA. A multi-output Random Forest model is trained to predict algorithm performance under HV and IGD indicators. Results show that static features are highly informative, while Pareto-based landscape features alone provide limited predictive power. Combined features consistently outperform the single-algorithm baseline, but the marginal gain over static-only meta-features is limited and indicator-dependent.*

1. Introduction

Multi-objective evolutionary algorithms often exhibit complementary behavior across problem instances. An algorithm that performs well for one instance may be less effective for another due to differences in size, objective function correlations, feasibility constraints, or search-space structure. This variability motivates meta-learning for algorithm selection, whose goal is to predict, from descriptive information about an instance, which algorithm is expected to produce the best approximation set [Rice 1976, Brazdil et al. 2022]. Although this idea has been widely explored in single-objective optimization [Cenikj et al. 2026], its application to multi-objective combinatorial optimization remains challenging because performance depends simultaneously on convergence, diversity, and the geometry of the Pareto front.

A central issue in meta-learning is how to represent problem instances. Static meta-features are computed directly from the instance definition and capture global properties, such as size or graph structure. Pareto-based landscape meta-features, in turn, are obtained through exploratory sampling and describe local search-space properties using dominance relations. Although previous studies show that landscape analysis can explain performance differences among multi-objective algorithms [Daolio et al. 2017, Liefooghe et al. 2019, Cosson et al. 2022], it remains unclear whether these meta-features provide useful information beyond simple static features in structured multi-objective combinatorial problems. The novelty of this work is to empirically assess this

complementarity in a ridesharing-inspired three-objective problem, rather than in standard benchmark landscapes.

The ridesharing problem studied in this work is an extension of the Traveling Salesman Problem (TSP) that includes passengers and reward collection [Marques et al. 2019, Silva et al. 2020, Silva et al. 2025]. In this problem, a driver plans a route while sharing the vehicle with persons or goods, referred to as *passengers*, and performing services at selected locations. These services provide non-monetary *rewards* to the driver, such as reputation scores or platform credits. Passengers are associated with origin-destination pairs and individual constraints, including maximum fare and maximum ride time. Thus, the driver must jointly decide which locations to visit, which rewards to collect, and which passenger requests to accept.

These decisions create conflicting objectives. Shorter routes may reduce travel time but limit reward collection, whereas visiting additional locations may increase service time and make some passenger requests infeasible. Accepting passengers can reduce the driver's travel cost through cost sharing, but may impose capacity and ride-time restrictions on the route. The formulation considers three objective functions for the driver: minimizing route cost, minimizing route time, and maximizing collected rewards. Although abstracted from full operational ridesharing systems, the problem preserves relevant algorithmic components, namely routing, passenger feasibility, cost sharing, and reward collection. This makes it a controlled testbed for studying meta-feature contributions for algorithm selection.

This study first evaluates MOEA/D [Zhang and Li 2007], IBEA [Li et al. 2017], and SPEA2 [Zitzler et al. 2001] on 252 instances, showing that their performance varies according to instance characteristics. This variability justifies the use of meta-learning for algorithm selection. We then compare three meta-feature scenarios: static-only, Pareto-based landscape-only, and combined representations. Static meta-features include instance size, graph type, and generation class, while landscape meta-features are extracted via random and adaptive walks that exploit dominance relations within sampled neighborhoods. The combined scenario concatenates static and landscape meta-features. From these configurations, we constructed 122 datasets considering HV and IGD. A multi-output Random Forest model is trained to predict algorithm performance and evaluated through predictive accuracy, merit, sampling cost, and relationships between meta-features and performance. All datasets, source codes, and results are available at `https://github.com/motspppeniac/ridesharingeniac2026`.

The main contributions of this paper are: (i) an empirical study of meta-learning for algorithm selection in a ridesharing-inspired multi-objective problem; (ii) a comparison of static, Pareto-based landscape, and combined meta-feature scenarios; (iii) evidence that static meta-features are highly informative; and (iv) an analysis showing that Pareto-based landscape meta-features are insufficient alone, but may provide complementary information when combined with static meta-features.

2. Related work

Algorithm selection aims to identify the most suitable optimizer for a given problem instance. This idea was formalized by [Rice 1976] and later motivated meta-learning approaches that use instance meta-features to predict algorithm performance

[Brazdil et al. 2022]. Although such methods have achieved promising results in single-objective optimization [Cenikj et al. 2026], extending them to multi-objective optimization remains challenging, as performance depends not only on convergence but also on the structure of the Pareto front.

In this context, multi-objective fitness landscape analysis has been used to understand how problem structure affects search behavior. Previous studies showed that objective correlations influence the number and distribution of Pareto local optima [Verel et al. 2011], while landscape features can explain variations in algorithm performance [Daolio et al. 2017] and characterize funnel structures [Ochoa et al. 2024]. More recently, landscape analysis has been combined with algorithm selection using dominance-, indicator-, and decomposition-based features [Liefooghe et al. 2019, Liefooghe et al. 2021, Cosson et al. 2021, Cosson et al. 2022].

Despite these advances, most studies focus on benchmark problems. This work addresses this gap by investigating static, landscape, and combined meta-feature representations for predicting algorithm performance in multi-objective ridesharing optimization.

3. Ridesharing-inspired multi-objective problem

Let $G = (V, E)$ be a graph representing the transportation network, where $V = \{1, \ldots, n\}$ denotes the set of vertices and $E = \{(i, j) \mid i, j \in V, i \neq j\}$ denotes the set of edges. Each edge $(i, j) \in E$ is associated with a travel cost c_{ij} and a travel time y_{ij}. Each vertex $i \in V$ is associated with a non-negative reward b_i and a collection time g_i. If the driver decides to execute the service at location i, the corresponding reward b_i is obtained, while an additional time g_i is added to the route duration.

The driver operates a vehicle with capacity for R passengers (excluding the driver). Let Ξ denote the set of passenger ride requests. Each request $\xi \in \Xi$ is defined by an origin vertex o_ξ, a destination vertex d_ξ, a maximum acceptable fare w_ξ, and a maximum admissible ride duration t_ξ. A request can be served only if the driver's route contains a feasible subpath connecting o_ξ to d_ξ while respecting both the vehicle capacity and the passenger-specific constraints.

A feasible solution γ corresponds to a route starting and ending at a depot vertex $s \in V$. The route visits a subset of locations $V' \subseteq V \setminus \{s\}$, each at most once, defining a sequence of edges $E' \subseteq E$ that forms a cycle $\Phi(V' \cup \{s\}, E')$. Along this route, the driver decides which rewards to collect and which passenger requests to accept. Travel costs are shared among all vehicle occupants. The cost of traversing an edge (i, j) is equally divided among the driver and all passengers currently in the vehicle. The fare charged to a passenger ξ corresponds to the sum of their share of the edge costs along the subpath from o_ξ to d_ξ. The ride duration experienced by passenger ξ equals the total travel time along this subpath, including reward collection times at intermediate locations.

The problem involves three objective functions. The first objective, defined in Equation (1), minimizes the total travel cost incurred by the driver, where x_{ij} indicates whether edge (i, j) is traversed by the driver and h_{ij}^{ξ} indicates whether passenger ξ is transported along that edge. The second objective, given by Equation (2), minimizes the sum of times of the selected edges and the time required to collect rewards at visited vertices, where p_i determine whether the driver collects the reward at vertex i. The third

objective, defined in Equation (3), maximizes the total collected rewards. The detailed mathematical formulation of the problem can be found in [Silva et al. 2025].

$$\min f_1(\gamma) = \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} \frac{c_{ij}\, x_{ij}}{1 + \sum_{\xi \in \Xi} h_{ij}^{\xi}} \tag{1}$$

$$\min f_2(\gamma) = \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} y_{ij}\, x_{ij} + \sum_{i \in V} g_i\, p_i \tag{2}$$

$$\max f_3(\gamma) = \sum_{i \in V} b_i\, p_i \tag{3}$$

The problem is related to several variants of the TSP, such as those involving prize collection and quota [Silva et al. 2020], and passenger transportation [Marques et al. 2019, Silva et al. 2025]. These variants motivate the ridesharing formulation considered in this paper. Since the ridesharing-inspired problem generalizes the TSPs, it is NP-hard.

4. Meta-learning

Meta-learning is a data-driven paradigm that aims at selecting, for a given problem instance, the algorithm expected to yield the best performance by exploiting knowledge extracted from previously solved instances [Brazdil et al. 2022]. In this work, meta-learning is employed to estimate the performance of multi-objective metaheuristics applied to the problem defined in section 3, enabling the automatic selection of the most suitable algorithm for unseen instances.

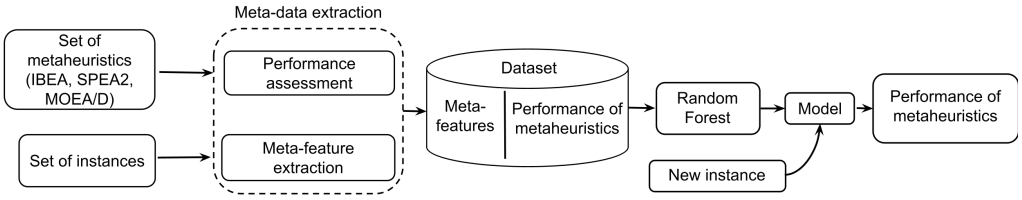


Figure 1. Meta-learning pipeline

Figure 1 illustrates the proposed meta-learning pipeline. The process starts with the *meta-data extraction*, in which a set of representative instances is solved using multiple multi-objective metaheuristics, and descriptive *meta-features* are computed for each instance. These meta-features aim to characterize the structure of the search space. Algorithm performance is assessed using an indicator that quantifies the quality of the approximation set produced by each metaheuristic. The meta-features and the corresponding performance values are then combined to construct a supervised learning dataset. In the learning phase, a multi-output regression Random Forest model is trained to predict the expected performance of each candidate metaheuristic for new problem instances. Given an unseen instance, its meta-features are computed and fed to the trained model, which estimates the performance of each metaheuristic using the chosen quality indicator. The algorithm with the best predicted performance is subsequently selected and recommended for solving that instance. The following sections detail the meta-features and metaheuristics considered in this study.

4.1. Meta-features

This work employs two complementary classes of meta-features: static and landscape meta-features. Static meta-features describe physical properties of instances and are computed without solving the optimization problem. Landscape meta-features aim to capture the structural characteristics of the search space by exploiting information from exploratory sampling procedures. The following static meta-features are employed:

1. the instance `size` defines the number of vertices in the graph;
2. the instance `type` indicates whether the graph is symmetric or asymmetric. In symmetric instances, the graph is undirected and satisfies $c_{ij} = c_{ji}$ and $y_{ij} = y_{ji}$ for all edges $(i, j) \in E$. In contrast, asymmetric instances are directed graphs in which $c_{ij} \neq c_{ji}$ and $y_{ij} \neq y_{ji}$ may hold.
3. the instance `class` characterizes the correlation between c_{ij} and y_{ij}, taking one of three values: random, correlated, or anticorrelated. The random class consists of instances in which c_{ij} and y_{ij} are independently generated from a uniform distribution. In the correlated class, c_{ij} and y_{ij} are positively correlated, while in the anticorrelated class they are negatively correlated.

Landscape meta-features in this work are Pareto dominance measures derived from local relations in the search space [Liefooghe et al. 2019, Cosson et al. 2022]. A solution γ dominates γ' if γ is no worse than γ' in all objectives and strictly better in at least one. Solutions γ and γ' are incomparable if neither γ dominates γ' nor γ' dominates γ. Let $\mathcal{N} : \Gamma \rightarrow 2^{\Gamma}$ be a neighborhood relation over the solution space Γ, where $\mathcal{N}(\gamma)$ is the set of neighbors of γ. Four neighborhood structures are considered to capture different aspects of the search space: (i) vertex swap, which exchanges two vertices in the route; (ii) vertex insertion, which adds a non-visited vertex at a random position in the route; (iii) vertex removal, which eliminates a randomly selected vertex from the route; and (iv) reward inversion, which flips the reward collection decision at a chosen vertex.

A walk is an ordered sequence of solutions $\mathcal{W}_L = (\gamma^0, \gamma^1, \ldots, \gamma^L)$ such that $\gamma^{t+1} \in \mathcal{N}(\gamma^t)$ for all $t = 0, \ldots, L-1$. For each solution γ^t visited during the walk, four Pareto-based measures are computed based on dominance relations within its neighborhood: the number of neighbors dominated by γ^t (`inf`); the number of neighbors dominating γ^t (`sup`); the number of neighbors incomparable to γ^t (`inc`); and the number of locally non-dominated neighbors (`ind`). These measures are aggregated along the walk using five statistical descriptors: average (`avg`), standard deviation (`sd`), first autocorrelation coefficient (`r1`), kurtosis (`kr`), and skewness (`sk`). Combining the four dominance-based measures with the five statistics yields a total of 20 fitness landscape meta-features. For example, `inf.avg` is the mean number of dominated neighbors, whereas `sup.sd` is the variability of dominating neighbors.

Random walk and *adaptive walk*, two neighborhood-based exploration procedures, extracted the Pareto-based meta-features [Cosson et al. 2022]. Both strategies start from an initial solution γ^0 and iteratively move to a neighboring solution, differing in the rule used to select the next solution. Random walks select from the neighborhood at random, whereas adaptive walks restrict transitions to Pareto-improving moves. As a consequence, random walks require a user-defined parameter L to define the walk length, while adaptive walks terminate upon reaching a Pareto local optimum.

To control computational effort, random and adaptive walks rely on a truncated neighborhood. Given a sampling rate $r \in (0, 1]$, the method constructs a subset $\mathcal{N}_r(\gamma^t)$ of $r \times |\mathcal{N}(\gamma^t)|$ solutions sampled uniformly at random from $\mathcal{N}(\gamma^t)$, for all $t = 0, ..., L - 1$. Setting $r = 1$ enables full neighborhood exploration, whereas smaller r values limit the number of evaluated neighbors.

4.2. Metaheuristics

MOEA/D [Zhang and Li 2007], *SPEA2* [Zitzler et al. 2001], and *IBEA* [Li et al. 2017] are the metaheuristics adopted in this paper. These algorithms were selected because they provide complementary search mechanisms for balancing convergence toward the Pareto front and diversity among non-dominated solutions. *MOEA/D* decomposes the multi-objective problem into scalar subproblems associated with weight vectors and optimized using a Tchebycheff aggregation function. *SPEA2* adopts a dominance-based strategy that combines fitness assignment and density estimation, whereas *IBEA* guides the search through a quality indicator that measures each solution's relative contribution to the population.

All three metaheuristics employ the same solution encoding and genetic operators to ensure a fair comparison. Each solution is a triple (Π, B, Ψ), where Π denotes an ordered sequence of visited vertices defining the driver's route, B is a binary vector indicating whether the reward is collected at each visited vertex, and Ψ represents the set of passengers assigned to the route. The population initialization generates feasible solutions randomly by sampling a route length, constructing a permutation of vertices, and assigning reward collection decisions uniformly at random. It then allocates passengers using the *Ride-Matching Heuristic* (*RMH*) [Silva et al. 2020], which processes requests ordered by w_ξ and sequentially inserts passengers along the route, accepting a request only if capacity, maximum ride time, and fare constraints are satisfied.

The crossover operator generates an offspring by inheriting a prefix of the route and corresponding reward decisions from one parent, and completing it with the remaining non-duplicated vertices from the second parent while preserving relative order. Passenger assignments are then recomputed using *RMH*. Mutation applies one of four operators with equal probability: vertex swap (exchange positions), insertion (add a vertex), removal (delete a vertex), and reward inversion (flip B entries). After each mutation, *RMH* assigns passengers.

5. Experimental methodology

5.1. Multi-objective test and assessment

Experiments used 252 instances, each represented by a weighted complete graph whose edges have cost and travel time values, and whose vertices have reward and collection time values. The instances vary in `size`, with 5, 8, 10, 20, 50, 100, or 200 vertices, and six distinct graphs are generated for each `size`. The dataset includes two instance `types`, symmetric and asymmetric, each comprising 126 instances. For both `types`, edge weights follow three correlation `classes`: random, correlated, and anticorrelated.

Each metaheuristic was run 20 times per instance, using 8×10^4 objective function evaluations as the stopping criterion. The *irace* framework [López-Ibáñez et al. 2016] tuned the parameters. All metaheuristics operated with a population size of 100. The

MOEA/D used a neighborhood size of 25, crossover and mutation rates of 0.58 and 0.34, respectively. IBEA used crossover and mutation rates of 0.81 and 0.42, respectively, whereas SPEA2 used 0.92 and 0.64.

The approximation sets Υ' were evaluated using *HV* and *IGD* after normalizing all objective values to the interval $[1, 2]$. The *HV* indicator measures the volume of the objective space dominated by Υ', using $(2.1, 2.1, 0.9)$ as the reference point. The *IGD* indicator computes the average distance from each point in the reference set to its closest solution in Υ'. Both indicators were computed with respect to a reference set formed by the non-dominated union of all solutions produced by the metaheuristics. Thus, lower values indicate better performance for both indicators. Friedman test and Nemenyi post-hoc test, at a significance level of 0.05, assessed statistical differences among the results.

5.2. Dataset construction and landscape analysis

Datasets associate meta-features with the performance of each metaheuristic on each instance, where rows represent instances and columns store meta-features and average quality indicator values (HV or IGD). In total, 122 datasets are constructed (61 per indicator), enabling a controlled evaluation of meta-features. We consider three scenarios: static-only, landscape-only, and combined. Landscape meta-features are extracted via random and adaptive walks, using 25 random-walk configurations defined by $L \in \{5, 10, 25, 50, 100\}$ and $r \in \{0.05, 0.1,\ 0.25, 0.5, 1.0\}$, and 5 adaptive-walk configurations with $r \in \{0.05, 0.1, 0.25, 0.5, 1.0\}$, yielding 30 datasets per indicator in the landscape-only scenario. Static-only uses instance descriptors (`size`, `types`, `classes`), yielding one dataset per indicator. The combined scenario concatenates static and landscape meta-features, preserving each sampling configuration, thus producing 30 datasets per indicator.

We analyze the number of objective function evaluations required to compute the Pareto-based meta-features. For each (L, r) configuration, we evaluate the Spearman correlation between the meta-features and the instance `size` [Cosson et al. 2022]. We also assess the variability of each meta-feature using the *GAP* defined in equation (4). For each Pareto-based meta-feature φ, the *GAP* is the difference between the value of φ under a given (L, r) configuration, denoted by $V_{L,r}(\varphi)$, and the value of φ under the most expensive configuration $V_{\max}(\varphi)$. For random walks, $V_{\max}(\varphi)$ uses $L = 100$ and $r = 1.0$; for adaptive walks, $V_{\max}(\varphi)$ uses $r = 1.0$.

$$GAP = \frac{|V_{\max}(\varphi) - V_{L,r}(\varphi)|}{V_{\max}(\varphi)} \tag{4}$$

5.3. Meta-learning test and assessment

The meta-learning evaluation follows the protocol of [Cosson et al. 2022]. The instances are split into a training set $\mathcal{T}$ and a validation set $\mathcal{V}$. Set $\mathcal{T}$ was used to train the multi-output Random Forest regression model. A repeated random hold-out strategy with a 90/10% split between $\mathcal{T}$ and $\mathcal{V}$ was employed. Given the meta-features of an unseen instance in $\mathcal{V}$, the model predicts HV and IGD for each metaheuristic, and the best one for each indicator is chosen as output. Reported values correspond to the mean across 100 independent folds. Random Search tuned the Random Forest hyperparameters for

each indicator: $n_estimators = 1000$, $max_depth = 5$, $min_samples_split = 2$, and $min_samples_leaf = 3$.

The assessment uses MAPE and R^2 regression metrics, as well as the merit m^*. The merit m^* measures the distance between the algorithm predicted by the meta-learner and two baselines: (i) the *Single Best Solver* (*SBS*), i.e., the algorithm with the best average performance over $\mathcal{T}$, and (ii) the *Virtual Best Solver* (*VBS*), which selects the best algorithm per instance. Let $\mathcal{A}$ be the set of metaheuristics and $\overline{ri}(\alpha, \mathcal{J})$ the average performance of $\alpha \in \mathcal{A}$ over a subset of instances $\mathcal{J}$ regarding the indicator $\iota \in \{HV, IGD\}$. The *SBS* and m^* are defined in equations (5) and (6), respectively, where α_s is the algorithm predicted by Random Forest. Values $m^* = 0$ indicate a perfect oracle, whereas $m^* < 1$ ($m^* > 1$) indicates performance better (worse) than the *SBS*.

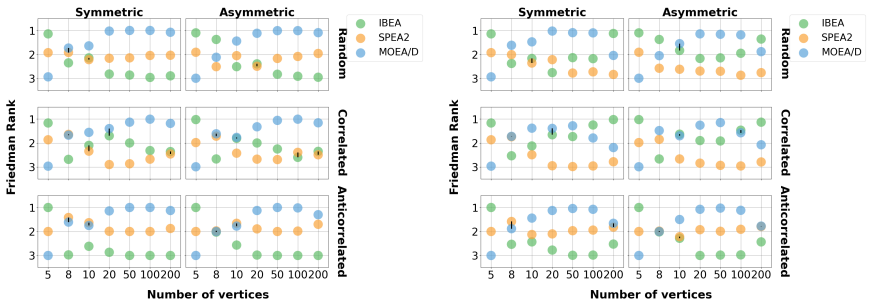
$$SBS = \arg\min_{\alpha \in \mathcal{A}} \overline{ri}(\alpha, \mathcal{T}) \tag{5}$$

$$m^* = \frac{\overline{ri}(\alpha_s, \mathcal{V}) - \overline{ri}(VBS, \mathcal{V})}{\overline{ri}(SBS, \mathcal{V}) - \overline{ri}(VBS, \mathcal{V})} \tag{6}$$

6. Results and discussion

6.1. Metaheuristics results

The Friedman test revealed statistical differences among the metaheuristics for HV and IGD. Figure 2 summarizes the Friedman and Nemenyi post-hoc results for each combination of instance `type` and `class`. The horizontal axis shows instance size, while the vertical axis reports the Friedman rank, with rank 1 indicating the best algorithm and rank 3 the worst. Bold lines connecting two circles indicate configurations in which the Nemenyi test detected no statistical differences between algorithms.



(a) HV (b) IGD

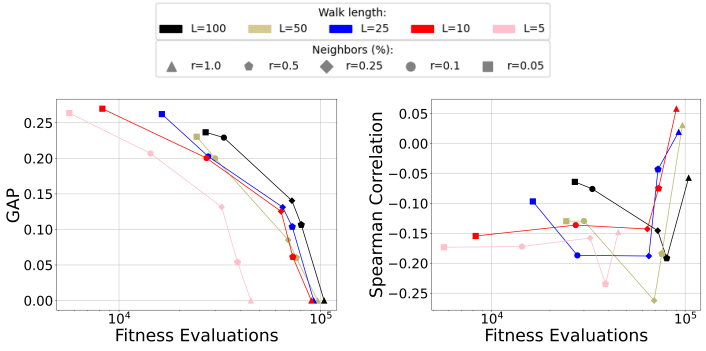
Figure 2. Summary of Friedman and Nemenyi post-hoc test results

For HV, the best algorithm varies mainly on small instances: IBEA usually performs best for 5 vertices, whereas for 8 vertices the best rank depends on the instance `type` and `class`. From 20 vertices onward, MOEA/D consistently produced the best HV. This behavior can be explained by the structured trade-offs induced by the objective functions. Since cost, route time, and reward collection favor different routing and passenger-assignment patterns, MOEA/D can benefit from decomposing the problem into scalar subproblems associated with distinct trade-off directions, maintaining search pressure across multiple regions of the Pareto front.

For IGD, the pattern is also instance-dependent. IBEA performs well for 5-vertex instances and achieves the best ranks for 200-vertex correlated and random instances of both types, as well as for 100-vertex correlated symmetric instances. Conversely, MOEA/D obtains the best IGD ranks for random and anticorrelated instances with 20, 50, and 100 vertices. This variability reinforces the need for meta-learning instead of relying on a single algorithm.

6.2. Feature landscape analysis

The sampling configuration strongly influences the landscape meta-features, and their Spearman correlation with instance `size` is feature-dependent. In particular, increasing the sampling effort tends to reduce the GAP values, indicating more stable estimates. Figure 3 illustrates these effects for `inf.avg` under random walks. Similar trends were observed for the remaining meta-features. Figure 3(a) plots the GAP values (y-axis) against the number of objective function evaluations (x-axis) for all (L, r) configurations. Each line color corresponds to a value of L, while geometric markers denote the corresponding r values. The number of evaluations increases with both L and r, since larger values expand the portion of the search space sampled. This behavior suggests that, although `inf.avg` is sensitive to the sampling configuration, its estimate stabilizes as the number of sampled neighbors increases. Figure 3(b) complements this analysis by showing the Spearman correlation between `inf.avg` and instance size under the same configurations. For low-cost configurations, the correlation values are more variable. As the sampling effort increases, the correlation becomes more consistent and gradually reveals a positive relationship with instance size.



(a) *GAP* (b) Spearman correlation

Figure 3. *GAP* and Spearman correlation for `inf.avg` computed by random walk

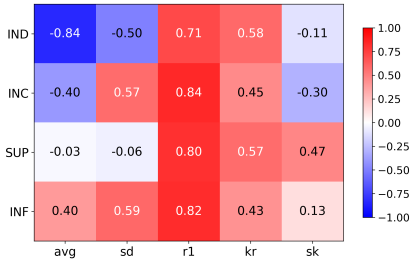


Figure 4. Average Spearman correlations for adaptive walk

Figure 4 presents the Spearman correlations between instance `size` and the meta-features extracted through adaptive walks. The `r1` and `kr` descriptors are consistently positively correlated with instance size, indicating that larger instances tend to produce landscapes with stronger autocorrelation and more pronounced higher-order structural patterns. In contrast, the `avg`, `sd`, and `sk` statistics exhibit mixed behaviors depending on the dominance measure, suggesting that the influence of instance `size` on dominance relations is not uniform.

6.3. Meta-learning performance

The static-only scenario achieved $m^* = 0.64$ for HV and $m^* = 0.62$ for IGD, outperforming the SBS. Table 1 reports m^* for the combined and landscape-only scenarios, where RW and AW denote Random Walk and Adaptive Walk, respectively, and grey cells indicate $m^* < 1$. All combined configurations achieved $m^* < 1$, with stable results across sampling strategies, parameter settings (L, r), and quality indicators. In contrast, landscape-only features rarely outperformed the SBS: only 5 configurations for HV and 3 for IGD obtained $m^* < 1$. These cases are scattered, with no monotonic trend regarding L or r, indicating that increasing sampling effort does not necessarily improve meta-learning performance.

Table 1. Average merit m^* for combined and landscape-only scenarios

	Combined						Landscape-only					
	RW					AW	RW					AW
r	L=100	L=50	L=25	L=10	L=5		L=100	L=50	L=25	L=10	L=5	
HV												
1.00	0.67	0.71	0.59	0.72	0.62	0.69	1.17	1.32	1.22	1.10	1.10	1.09
0.50	0.71	0.76	0.64	0.73	0.75	0.71	1.07	1.06	1.04	1.01	1.10	1.30
0.25	0.75	0.72	0.70	0.72	0.76	0.68	0.98	1.12	1.10	0.97	1.17	1.23
0.10	0.75	0.72	0.70	0.71	0.71	0.75	1.23	1.24	1.15	1.37	1.04	0.86
0.05	0.72	0.67	0.68	0.68	0.58	0.58	1.36	1.18	1.37	0.88	0.98	1.52
IGD												
1.00	0.78	0.72	0.74	0.79	0.68	0.85	1.15	1.13	1.10	1.16	1.18	1.14
0.50	0.80	0.84	0.76	0.77	0.69	0.73	1.30	1.19	1.18	1.25	1.35	0.81
0.25	0.75	0.80	0.72	0.77	0.70	0.98	1.13	1.30	1.16	1.11	1.14	0.92
0.10	0.81	0.78	0.73	0.76	0.71	0.85	1.00	1.22	1.26	1.17	1.05	1.18
0.05	0.81	0.74	0.73	0.75	0.73	0.98	1.13	0.99	1.25	1.00	1.13	1.08

These results suggest that static meta-features capture key global information, especially because `size` strongly affects difficulty, while `type` and `class` influence solution structure. Pareto-based landscape features, extracted from local sampling, are insufficient alone but provide complementary information when combined with static descriptors.

Regression metrics indicate that HV is predicted more accurately than IGD in the combined scenario ($R^2 \in [0.996, 0.997]$, MAPE $\in [0.318, 0.409]$ vs. $R^2 \in [0.889, 0.929]$, MAPE $\in [7.647, 12.112]$), with sparse sampling degrading IGD predictions. Feature importance analysis reveals that `size` is the dominant meta-feature in static-only and combined scenarios, while `inc` and `ind` are the most informative among landscape meta-features, reinforcing their complementary role.

7. Conclusion

This paper investigated static and Pareto-based landscape meta-features for algorithm selection in a three-objective ridesharing-inspired optimization problem. The results show that static meta-features are highly informative for predicting the relative performance of MOEA/D, SPEA2, and IBEA. This finding is consistent with the TSP-like nature of the problem and indicates that basic structural descriptors already capture major sources of variation among the generated instances. Landscape features alone achieved limited predictive performance and did not consistently outperform the single best solver. When combined with static descriptors, they yielded better predictions than SBS. However, the contribution of the Pareto-based meta-features is limited and insufficient to replace static ones.

Future work includes designing specific meta-features for the multi-objective ridesharing problem, exploring alternative learning models, and extending the analysis to other real-world multi-objective problems.

Acknowledgment

This work was supported by Kunumi Institute, and the authors thank the institution for its financial support and commitment to advancing scientific research. This work was also supported by the Bahia Research Foundation (FAPESB) through the INCITE II program under TO PIE0007/2026 and a doctoral scholarship; Federal University of Bahia (UFBA); the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES); and the National Council for Scientific and Technological Development (CNPq), under grant 402842/2023-5.

References

Brazdil, P., Van Rijn, J. N., Soares, C., and Vanschoren, J. (2022). *Metalearning: Applications to automated machine learning and data mining*. Springer Nature.

Cenikj, G., Nikolikj, A., Petelin, G., Van Stein, N., Doerr, C., and Eftimov, T. (2026). A survey of features used for representing black-box single-objective continuous optimization. *Swarm and Evolutionary Computation*, 101:102288.

Cosson, R., Derbel, B., Liefooghe, A., Aguirre, H., Tanaka, K., and Zhang, Q. (2021). Decomposition-based multi-objective landscape features and automated algorithm selection. In *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*, pages 34–50. Springer.

Cosson, R., Derbel, B., Liefooghe, A., Verel, S., Aguirre, H., Zhang, Q., and Tanaka, K. (2022). Cost-vs-accuracy of sampling in multi-objective combinatorial exploratory landscape analysis. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 493–501.

Daolio, F., Liefooghe, A., Verel, S., Aguirre, H., and Tanaka, K. (2017). Problem features versus algorithm performance on rugged multiobjective combinatorial fitness landscapes. *Evolutionary Computation*, 25(4):555–585.

Li, W., Özcan, E., John, R., Drake, J. H., Neumann, A., and Wagner, M. (2017). A modified indicator-based evolutionary algorithm (mIBEA). In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1047–1054. IEEE.

Liefooghe, A., Daolio, F., Verel, S., Derbel, B., Aguirre, H., and Tanaka, K. (2019). Landscape-aware performance prediction for evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 24(6):1063–1077.

Liefooghe, A., Vérel, S., Lacroix, B., Zavoianu, A.-C., and McCall, J. (2021). Landscape features and automated algorithm selection for multi-objective continuous optimisation problems. In *GECCO*, pages 421–429.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Marques, T. S., Cezario, S. F., Goldbarg, E. F. G., Goldbarg, M. C., and Maia, S. M. D. M. (2019). Quota traveling salesman with passengers and collection time. In *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 299–304. IEEE.

Ochoa, G., Liefooghe, A., and Verel, S. (2024). Funnels in multi-objective fitness landscapes. In *International Conference on Parallel Problem Solving from Nature (PPSN)*, pages 343–359. Springer.

Rice, J. R. (1976). The algorithm selection problem. *Advances in Computers*, 15:65–118.

Silva, B. C., Fernandes, I. F., Goldbarg, M. C., and Goldbarg, E. F. (2020). Quota travelling salesman problem with passengers, incomplete ride and collection time optimization by ant-based algorithms. *Computers & Operations Research*, 120:104950.

Silva, J. B. A., Brito, T. A. P., Rios, R. A., Rios, T. N., and Fernandes, I. F. C. (2025). Multi-objective traveling salesman problem with profits and passengers: Exact, heuristic and evolutionary approaches. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 419–422.

Verel, S., Liefooghe, A., Jourdan, L., and Dhaenens, C. (2011). Pareto local optima of multiobjective NK-landscapes with correlated objectives. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 226–237. Springer.

Zhang, Q. and Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). SPEA2: Improving the strength pareto evolutionary algorithm. *TIK Report*, 103.