



Kent Academic Repository

Ribeiro, G.A.S., Nascimento, C.G.M., de Souza, B.F., Cortes, O.A.C., Canuto, A.M.P., Xavier-Junior, Joao C., Freitas, Alex A. and de Oliveira, A.C.M. (2026) *Meta-Learning for selecting Meta-heuristics based on Iterated Local Search to solve Permutation Flow-shop Problem*. SN Computer Science, 7 . pp. 1-18.

Downloaded from

<https://kar.kent.ac.uk/116017/> The University of Kent's Academic Repository KAR

The version of record is available from

<https://doi.org/10.1007/s42979-026-05312-x>

This document version

Publisher pdf

DOI for this version

Licence for this version

CC BY (Attribution)

Additional information

Versions of research works

Versions of Record

If this version is the version of record, it is the same as the published version available on the publisher's web site. Cite as the published version.

Author Accepted Manuscripts

If this document is identified as the Author Accepted Manuscript it is the version after peer review but before type setting, copy editing or publisher branding. Cite as Surname, Initial. (Year) 'Title of article'. To be published in **Title of Journal** , Volume and issue numbers [peer-reviewed accepted version]. Available at: DOI or URL (Accessed: date).

Enquiries

If you have questions about this document contact ResearchSupport@kent.ac.uk. Please include the URL of the record in KAR. If you believe that your, or a third party's rights have been compromised through this document please see our [Take Down policy](https://www.kent.ac.uk/guides/kar-the-kent-academic-repository#policies) (available from <https://www.kent.ac.uk/guides/kar-the-kent-academic-repository#policies>).



Meta-Learning for Selecting Meta-heuristics Based on Iterated Local Search to Solve Permutation Flow-Shop Problem

Guilherme Alberto Sousa Ribeiro¹ · Chrystian Gustavo Martins Nascimento² · Bruno Feres de Souza² · Omar Andrés Carmona Cortés³ · Anne Magaly de Paula Canuto⁴ · João Carlos Xavier-Junior⁴ · Alex A. Freitas⁵ · Alexandre César Muniz de Oliveira²

Received: 25 June 2025 / Accepted: 17 August 2026
© The Author(s) 2026

Abstract

In the realm of intelligent systems, addressing the NP-hard Permutation Flow-Shop (PFS) problem presents a notable challenge, especially with burgeoning complexities in manufacturing systems. Conventional wisdom relies on metaheuristic techniques to tackle such intricacies, yet the best metaheuristic remains elusive. This paper introduces a new meta-learning framework that leverages pre-trained metaheuristics rooted in Iterated Local Search (ILS) to forecast the most effective metaheuristic for each PFS instance. Notably, our approach pioneers the extraction of novel meta-features from graph representations, discerning the best metaheuristics in 99.7% of cases using a small set of meta-features and baseline classifiers, such as decision trees and support vector machines. Moreover, the interpretability of our meta-model unveils insights into feature impact, showcasing the efficacy of abstract graph computations, such as centrality and connectivity, over bespoke meta-features related to problem size.

Keywords Meta-learning · Permutation flow-shop (PFS) problem · Iterated local search (ILS)

Introduction

The Permutation Flow-Shop Problem (PFSP) is a combinatorial optimisation problem that models scheduling jobs on a set of machines, where jobs are processed in the same order on all machines and each job spends a given processing time on each machine. Usually, its objective is to discover the

sequence of jobs that minimises the total time required to process all jobs (the makespan) [1–3].

Various optimisation methods have been proposed in the literature to solve PFSP. The state-of-the-art methods include exact and approximate algorithms. However, as in many combinatorial problems, metaheuristics have gained prominence for obtaining near-optimal results in shorter

✉ Guilherme Alberto Sousa Ribeiro
guilherme.asr@einstein.br
Chrystian Gustavo Martins Nascimento
chrystian.gmn@ufma.br
Bruno Feres de Souza
bruno.feres@ufma.br
Omar Andrés Carmona Cortés
omar@ifma.edu.br
Anne Magaly de Paula Canuto
anne.canuto@ufrn.br
João Carlos Xavier-Junior
jcxavier@imd.ufrn.br
Alex A. Freitas
a.a.freitas@kent.ac.uk

Alexandre César Muniz de Oliveira
alexandre.cesar@ufma.br

- ¹ Hospital Israelita Albert Einstein, Morumbi, Sao Paulo 05652-900, Brazil
- ² Federal University of Maranhao (UFMA), Sao Luis 65080-805, Brazil
- ³ Computing Department (DComp), Federal Institute of Maranhao (IFMA), Sao Luis 65095-460, Brazil
- ⁴ Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte (UFRN), Natal, RN, Brazil
- ⁵ University of Kent, Canterbury, UK

computation times than exact methods based on integer programming and other enumeration methods, especially for large problem instances [3–5].

The literature on metaheuristics for the PFSP is vast, and selecting the most suitable algorithm for a given instance remains a challenging problem. According to the No-Free Lunch theorem, no single algorithm can be expected to perform best across all problem instances [6]. Therefore, the central challenge is not only to design effective metaheuristics, but also to determine which metaheuristic is most appropriate for each PFSP instance. In this context, meta-learning [7] has emerged as a promising strategy for learning the relationship between instance characteristics and solver performance.

More specifically, a meta-learning-based selection system should be able to predict the most suitable metaheuristic for solving a particular problem instance. Feature extraction and selection play a crucial role in meta-learning and have been extensively studied for several kinds of problems, such as the Travelling Salesman Problem [8], the Permutation Flowshop Problem [9], the Maximum Satisfiability Problem [10], and Constraint Satisfaction Problems [11]. The next works reflect a broad spectrum of innovative approaches and methodologies to enhance learning efficiency, accuracy, and the effectiveness of optimisation algorithms.

In [12], an advanced framework for Safe Bayesian Optimisation (BO) has been proposed, leveraging meta-learning to formulate priors that improve the accuracy of uncertainty quantification. The primary technique, named Fully Probabilistic Design of Combinatorial Hyperparameter and Algorithm Configuration Optimisation with Hamiltonian Monte Carlo (F-PACOH), facilitates the empirical selection of kernel hyperparameters and safety modifications based on uncertainty metrics. This methodology is applied to both benchmarks and precise motion systems, demonstrating that priors refined via meta-learning significantly expedite the convergence of Bayesian optimisation (BO) methods without compromising operational safety.

An application of meta-learning to the Travelling Salesman Problem (TSP) has been explored for large-scale instances, with an emphasis on low-complexity methods. Various meta-learning models were developed to predict the performance of fast and greedy heuristics based on instance features. The study utilised neural networks, decision trees, and k-nearest neighbours, introduced new meta-features, and evaluated the feasibility of using artificial data for model training. Results indicated that the models effectively captured TSP instance patterns and outperformed simple selection techniques. However, the approach still relies on problem-specific features and does not explore alternative representations, such as graph-based structures, which

may limit its generalisability across different optimisation problems [13].

In [14], a novel approach to the algorithm selection problem using neural networks with an attention mechanism as meta-learners is presented. The study highlighted the superiority of this method in efficiently mapping problems to specific algorithms, outperforming conventional models such as Multi-Layer Perceptron (MLP), k-nearest neighbour (k-NN), *softmax* regression, support vector machines (SVM), and decision trees. Using a dataset generated from the vehicle routing problem with time windows, the attention-based model achieved significantly better results, showcasing the potential of advanced neural network architectures in enhancing algorithm selection. Despite its strong predictive performance, the model operates as a largely black-box approach, offering limited interpretability, and depends heavily on large datasets, which may hinder its applicability in scenarios with limited data availability.

In [15], an innovative data-driven meta-heuristic recommendation model, named Meta-heuristic Recommendation Model (MRM), is proposed to tackle the multi-mode resource-constrained project scheduling problem (MRCPSP). By learning the associations between problem meta-attributes and algorithm performance, MRM identifies the most suitable algorithm for different MRCPSP instances. Utilising a Multi-Class Support Vector Machine (MCSVM) to train classifiers, the model evaluated its performance through accuracy, sensitivity, and a comprehensive evaluation index. Experiments covering four scenarios and two strategies demonstrated that MRM achieves optimisation and prediction accuracy above 90% without increasing computational complexity, excelling across various MRCPSP instances. Nevertheless, the model relies on predefined meta-attributes and does not consider alternative representations that capture deeper structural properties of problem instances. In addition, the evaluation focuses primarily on predictive performance and does not discuss the cost of constructing the performance space.

A hybrid meta-heuristic algorithm combining the Greedy Randomised Adaptive Search Procedure (GRASP) and Genetic Algorithm (GA), integrated with a meta-learning tuning method, has been proposed to optimise supply chain management by minimising transportation costs and extra stock production. Extensive evaluation with realistic instances provided by an industrial partner showed that their hybrid approach outperforms known complete and incomplete solvers, especially in large-scale instances, delivering significant weekly economic benefits for the company [16].

Although meta-learning has shown promising results in optimisation problems, important limitations remain in the PFSP algorithm-selection context. First, many existing approaches rely heavily on handcrafted or problem-specific

descriptors, which often require expert knowledge and may fail to capture deeper structural relationships among jobs, machines, and processing times. As a consequence, the resulting models may present limited generalisation capabilities when applied to instances with characteristics different from those observed during training. Second, although graph-based representations have demonstrated their effectiveness in other combinatorial optimisation domains, they remain largely underexplored as the primary source of meta-features for PFSP algorithm selection. This represents a missed opportunity, since graph structures can naturally encode both local and global interactions among the elements that define a scheduling instance. Third, the construction of performance spaces in many algorithm-selection studies relies on extensive experimental campaigns that involve parameter tuning and repeated executions of candidate solvers, resulting in substantial computational cost and limited reproducibility.

These limitations reveal a clear research gap. To the best of our knowledge, there is still no sufficiently explored meta-learning framework for PFSP algorithm selection that simultaneously: (i) represents instances through graph-based structures specifically designed to capture scheduling relationships, (ii) relies exclusively on graph-derived meta-features instead of handcrafted descriptors, and (iii) reuses previously available performance data to reduce the cost of constructing the meta-learning environment. Addressing these three challenges simultaneously is important because practical algorithm-selection systems must not only achieve high predictive performance but also remain scalable, reproducible, and sufficiently generic to support future extensions. In this context, the paper's contributions are:

1. A graph-based representation strategy for PFSP instances: We propose the use of two graph representations, namely the workflow graph and the job-machine graph, to characterise PFSP instances through graph-derived meta-features.
2. A meta-learning framework based exclusively on graph-derived meta-features: Unlike previous PFSP studies that rely on handcrafted or problem-specific descriptors, the proposed approach uses only graph-based meta-features to support algorithm selection.
3. A cost-reducing strategy for constructing the performance space: The framework reuses available solver performance data, opening the possibility of salvaging previously executed experimental campaigns.
4. An empirical assessment of predictive and optimisation performance: Experiments on 360 PFSP instances from Taillard and VRF datasets show that the proposed approach can accurately recommend competitive tuned

metaheuristics, with decision trees achieving the best results among the evaluated meta-learners.

5. An interpretable recommendation model: The induced decision tree provides explicit decision rules, allowing the identification of graph-based structural patterns associated with the most suitable metaheuristics for different PFSP instances.

While building a comprehensive performance space from scratch remains computationally expensive, an immense volume of high-quality algorithmic performance data already exists across literature. Pioneering a data-reusability paradigm, this work demonstrates that previously executed experimental campaigns can be systematically salvaged. By transforming raw historical performance matrices into abstract graph-based metadata, we prove that the initial computational investment of classic benchmarks can be continuously amortized to train highly accurate meta-learners without repeating exhaustive simulations. As meta-learning setups require extensive experimentation during training, reusing prior experience increases the system's flexibility and facilitates the development of further extensions. Here, we have considered the performance of automatically designed heuristics for PFSP instances, as reported in [4]. However, new solver algorithms can be included as a new performance table is obtained for the same instance dataset.

The remainder of this paper is organised as follows: Section 2 presents some studies related to meta-learning and the Permutation Flow-Shop Problem. Section 3 presents our proposed meta-learning framework, including creating the meta-dataset. In Section 4, the computational results are discussed, while the final section, Section 5, highlights the main findings and future research directions.

Related Work

Permutation Flowshop Scheduling is a special case of FSPs in which the processing order of the jobs is the same on every machine, and there is no overlap. The objective is to find the best sequence of jobs to be processed by machines to minimise a given performance measure. This problem arises in manufacturing facilities where material handling devices transfer jobs from one machine to another. The total flow time and the makespan are important performance measures that lead to rapid job turnaround and the minimisation of in-process inventory [17–19].

The Permutation Flow-Shop Problem has been proven strongly NP-complete, with the number of possible job schedules growing factorially. As a result, it has mainly been addressed using heuristic and metaheuristic approaches [5, 20–23].

Different types of integration between metaheuristics and machine learning approaches have been proposed. According to the taxonomy presented in [24], this integration can be done by (1) Algorithm selection employing prediction techniques; (2) Search space modelling by approximating computationally expensive fitness functions; (3) Evolution encompassing from initialisation up to operator selection and neighbour generation; (4) Parameter setting and adaptive controlling; and (5) Cooperation among metaheuristics to solve optimisation problems in parallel or sequentially by smartly adjusting behaviour during the search process.

In Algorithm selection, a new research frontier seeks to construct a comprehensive dataset of different optimisation problem instances using automatically generated features, while maintaining a strong relationship with the structuring elements of the original problems, thereby enabling an explainable data model to be learned. This section presents a brief literature review of approaches based on approximation algorithms and meta-learning [24].

Metaheuristics for Permutation Flow-Shop Problem

In [23], the Iterated Greedy (IG) algorithm was performed to generate a sequence of jobs obtained by applying a destruction operator (randomly job elimination) and construction (job reinsertion) greedy heuristics based on the *Nawaz-Enscore-Ham* (NEH) heuristic [25]. Once completed, the next step was to apply an acceptance criterion to determine whether the new solution should replace the current solution. This simple algorithm effectively solves Taillard's instances [26] of the permutation flow-shop scheduling problem. However, such approaches are designed for direct optimisation and do not address the problem of selecting the most suitable algorithm for different instance characteristics.

The scheduling mechanisms of different variants of flow-shop problems, such as permutation, no-wait, and no-idle flow-shop problems, have been assessed to measure the impact of constraints and other correlation criteria across these variants [27]. Additionally, graphs were presented to illustrate concepts essential to understanding the analysis.

A new benchmark of hard instances for the permutation flow-shop scheduling problem has been proposed, consisting, among others, of 240 large instances with up to 800 jobs and 60 machines [28]. Extensive generation and computational experiments demonstrate that the proposed benchmark is more challenging and has greater discriminative power than the widely used Taillard benchmark [26].

In [21], a new priority rule was proposed by employing the skewness concept to improve the well-known *Nawaz-Enscore-Ham* (NEH) heuristic [25]. NEH is a widely used polynomial-time heuristic for minimising makespan in static-deterministic PFSP [29], which generates an initial

job order using a greedy rule. Then, it iteratively inserts jobs into a partial sequence according to the previous initial order. Variants of the new heuristic were successfully applied to PFSP benchmarks, yielding better results than the original versions of the same polynomial heuristic. Despite these improvements, the method remains dependent on handcrafted heuristics and does not incorporate learning mechanisms to adapt algorithm selection across different instances.

In [20], several possibilities of refinement of the Iterated Greedy (IG) algorithm were evaluated, following the proposal of a technique to optimise partial solutions by well-known local search operators. The results obtained by the proposed Iterated Greedy with Local Search on Partial solutions were significantly better than those reported in the literature [20]. However, the focus remains on improving a single algorithmic strategy, without considering adaptive selection mechanisms based on instance characteristics.

A recently proposed template for automatic algorithm configuration (AAC) has been employed to obtain efficient metaheuristics and has reached the best lower bounds for permutational and non-permutational flow-shop problems [30]. The approach explores combinations of algorithmic components within a framework based on the Iterated Greedy (IG) algorithm, utilising the Itrace package for tuning. This approach, however, does not incorporate learning and performance prediction to guide solver selection for unseen instances, which limits its applicability in adaptive optimisation scenarios.

In [31], a Tutorial on Iterated Greedy (IG) Algorithms for Flow-Shop Scheduling Problems is delivered, including a problem taxonomy according to their scheduling scenarios, objective functions, and constraints. The authors summarise the main strategies employed in encoding and decoding, solution initialisation, local search, destruction-construction operators, and acceptance-termination criteria.

Recently, a hybrid strategy has been proposed, based on a cooperative composition among the Fruitfly Optimisation Algorithm, which explores multiple neighbourhood structures selected by the ϵ -greedy Q-learning mechanism, and a simulated annealing acceptance criterion to avoid premature convergence. The results are claimed to be the best across a wide range of settings, including those of Taillard, adapted to the distributional version of the PFSP [32].

Meta-Learning Integrated with Metaheuristics

Machine learning techniques integrated with metaheuristics have presented a meaningful growth in popularity [24, 33]. Among the integration types, the Algorithm Selection Problem has been solved offline, in which a meta-model is constructed from a particular set of training instances. Its main

goal is to predict the problem-algorithm mapping for new problem instances [8, 10, 34].

In [34], a very representative experiment concerning the Flow-Shop Problem has been conducted. Four meta-heuristics, including Hill Climbing, Tabu Search, Simulated Annealing, and Iterated Local Search, have been proposed as candidate algorithms for solving 540 PFSP instances, ranging from no-wait FSP to no-idle FSP. The *bespoke* low-level meta-features (number of jobs and machines, processing time distribution, problem variant, objective, and evaluations budget), as well as landmarks, have been used to build a meaningful dataset containing 27,000 meta-features [34]. While this approach provides valuable insights, it relies heavily on handcrafted descriptors, which may limit its ability to capture more complex structural patterns. Moreover, constructing such a performance space requires extensive experimentation, which incurs high computational costs.

The Irace package tuner has been integrated with a knowledge discovery approach based on Classification and Regression Trees and Meta-learning to uncover insights into the relationships among a set of bespoke meta-features (FSP variant, instance size, and objective), metaheuristics, and performance parameters [35]. The constructed meta-data, encompassing meta-features and tuning parameters, was interpretable and could contribute to understanding the target problem [35].

Algorithm selection has also been applied to recommend search operators and algorithm performance parameters. In [9], meta-learning relies on problem-specific features and fitness landscape metrics, using multi-label recommendation models trained on over 6,000 instances. The quality of the recommendations could surpass that of state-of-the-art solver algorithms, such as the Iterated Greedy (IG) algorithm proposed in [23]. However, the approach depends on problem-specific and landscape-based features, which may restrict generalisation and do not explicitly capture structural relationships within problem instances.

Feature extraction from an intermediate problem representation, such as a graph, is a promising approach compared with direct characterisation based on the problem's taxonomy. In [10], for instance, new graph representations have been proposed by encoding more information about the structure of the MaxSAT problem. Representations commonly found in the literature, such as variable and clause graphs, have been enhanced by incorporating edge weights, thereby expanding the feature set. Despite these promising results, the framework does not address the tuning of candidate algorithms, and the construction of the performance space still depends on computationally intensive experiments. Furthermore, graph-based features are combined with other descriptors rather than being used as the primary representation.

More recently, the authors explored the feature-extraction issue in [36] to facilitate more effective algorithmic recommendations. They dealt with real-world capacitated vehicle routing problems (CVRPs), proposing three sets of meta-features extracted from 1) the problem itself (e.g., distribution of customer demand, the relationship between customer demand and vehicle capacity), 2) feasible solutions generated by local search and 3) images associated with CVRPs instances. The authors reported promising results with a (k -NN)-based meta-learner. However, the use of multiple heterogeneous feature sources increases model complexity and may limit scalability and generalisability across different optimisation problems.

Graphs can be obtained from different methods, but not always simple ones [37]. However, they have the advantage of capturing better element relationships and dynamics [10]. Meta-learning approaches that use graph-derived meta-features typically combine them with bespoke meta-features, resulting in a dataset that remains unstructured and problem-specific [34, 35].

This work differs from [13, 34, 37] in that it exclusively employs graph-related meta-features to capture the distinct dynamics of the problem instances. Our approach provides a consistent alternative to bespoke meta-features, yielding characterisation spaces of similar quality. Works commonly found in the literature utilize features extracted from problem-specific heuristic solutions cite Pihera2014, images [36], and computationally exhaustive experiments [10, 34, 35].

Large datasets comprising graph-related meta-features enable studies in a joint discussion domain, allowing for more robust insights into algorithm recommendations by clustering and correlating different optimisation problems within a single multidimensional space. Furthermore, this work includes little-explored aspects, such as algorithm tuning, which considers performance parameters as part of the performance-prediction labels.

Related Works Discussion

Overall, recent studies have demonstrated that meta-learning can significantly improve algorithm selection and recommendation in combinatorial optimisation problems. However, a closer examination of the literature reveals that existing approaches still exhibit important methodological limitations that restrict their applicability, interpretability, and generalizability.

The first limitation concerns the representation of problem instances. Most studies rely on handcrafted descriptors, problem-specific characteristics, landscape features, or combinations of heterogeneous feature sources. While such representations may be effective within a particular

domain, they typically require substantial expert knowledge and often fail to capture structural relationships that emerge from the interaction among problem components. Consequently, transferring these representations across different optimisation problems becomes difficult.

The second limitation concerns the limited use of graph-based representations as the primary means of characterising instances. Although graphs have demonstrated their ability to model structural dependencies in several optimisation domains, most existing meta-learning frameworks either ignore graph representations entirely or combine graph-derived descriptors with handcrafted features. As a result, the actual contribution of graph-based structural information remains poorly understood, particularly in the context of PFSP algorithm selection.

The third limitation concerns the construction of the performance space. Many algorithm-selection studies require extensive experimental campaigns involving repeated executions, parameter tuning procedures, and large computational budgets to generate performance labels. While such strategies can produce high-quality datasets, they significantly increase the cost of building and maintaining recommendation systems, limiting reproducibility and practical deployment.

A fourth limitation concerns interpretability. Several recent approaches employ complex machine learning models that achieve strong predictive performance but often at the cost of reduced transparency. In algorithm-selection scenarios, understanding why a particular algorithm is recommended can be as important as the recommendation itself, particularly in industrial environments where decision support systems must be explainable and trustworthy.

Taken together, these limitations indicate that the PFSP algorithm-selection problem remains far from fully explored. There is still a need for approaches that combine structural instance representations, reuse of performance space, competitive predictive performance, and interpretable recommendation models within a unified framework.

Table 1 highlights the main differences between recent studies and the proposed approach. Most prior work relies on bespoke or hybrid feature sets and does not exploit prior performance data. Moreover, interpretability is often limited

or not considered at all. In contrast, the proposed framework combines graph-based representations, performance data reuse, and interpretable models, providing a more comprehensive solution to the algorithm selection problem in PFSP.

In contrast to the work most recent and closely related to our study [9], our proposed approach differs in three main aspects: a) *Feature Representation*: While their method relies on handcrafted and landscape-based features, we propose a graph-based meta-learning framework where graph-derived representations serve as the primary source of instance characterisation; b) *Interpretability*: Our framework emphasizes interpretability by adopting models that provide explicit decision rules, enabling the identification of structural patterns associated with algorithmic performance; c) *Agnostic Perspective*: Our approach shifts characterisation to a structurally agnostic perspective. By unifying instance characteristics under a generalised graph framework, meta-learning performs independently of the explicit problem rules. This opens the door for true cross-domain knowledge transfer, where a model trained on scheduling graphs might recognise similar structural patterns in routing or assignment graphs.

The Proposed Framework

In this work, meta-learning is applied to model the relationship between the characteristics (meta-features) of PFSP instances and the performances of a pool of previously defined metaheuristics. Figure 1 illustrates the general view of the proposed meta-learning framework.

The selection process works as follows. First, the data representation of the problem at hand is defined. Here, PFSP instances comprise the so-called *Problem space*. Next, each instance in that space undergoes two steps: data characterisation and algorithm evaluation. The former aims to extract meta-features from the *Problem space*, yielding a set of meta-feature vectors that describe the problem instances and build the *Feature space*. The latter aims to assess the performance of the tuned algorithms in constructing the *Algorithm space* and the *Performance space*, and to define

Table 1 Comparison between recent related studies and the proposed work

Study	Problem	Feature type	Graph-based	Uses prior performance data	Interpretable model
Delgado et al. (2018)	PFSP/FSP variants	Bespoke + landmarks	No	No	Limited
Alex et al. (2020)	PFSP	Bespoke + landscape features	No	No	No
Miranda et al. (2018)	MaxSAT	Graph-derived features	Yes	No	Limited
Jiang and Hao (2021)	CVRP	Problem + solution + image features	No	No	No
Bossenbroek et al. (2023)	TSP	Instance features	No	No	No
This work	PFSP	Exclusively graph-derived features	Yes	Yes	Yes

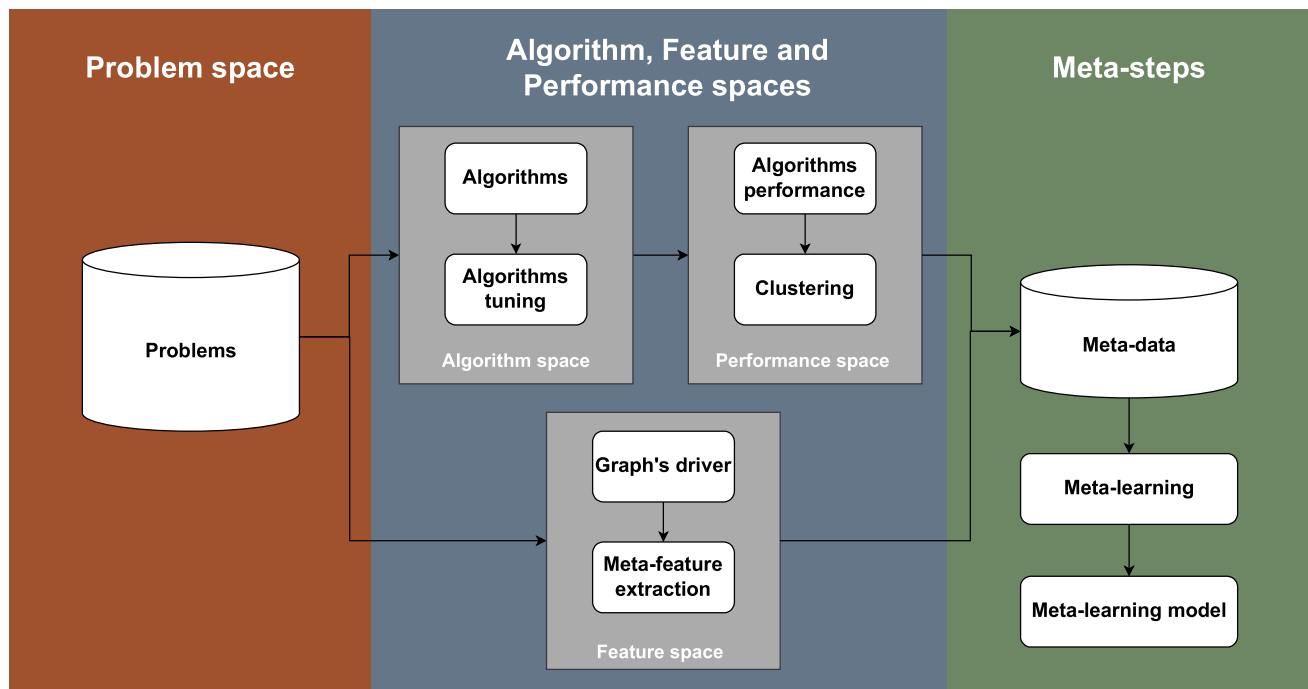


Fig. 1 Framework proposed based on algorithm selection (Adapted from [38])

the target meta-class, i.e., the algorithm that best solves each instance.

The combination of meta-features and a target meta-class composes a meta-instance. When all meta-instances are available in a metadata set, a meta-learning algorithm is applied to induce a mapping from meta-features to meta-classes. The meta-knowledge generated by this meta-learner is then used to recommend or select metaheuristics for previously unseen PFSP instances. Key aspects of this meta-learning process are detailed in the following subsections.

This design follows the algorithm selection paradigm proposed by Rice [38], which provides a well-established theoretical foundation for mapping problem instances to suitable algorithms. In this context, the methodological choices aim to balance predictive performance, interpretability, and computational feasibility, all of which are critical for practical deployment in optimisation scenarios.

Definition of the Meta-Feature Space

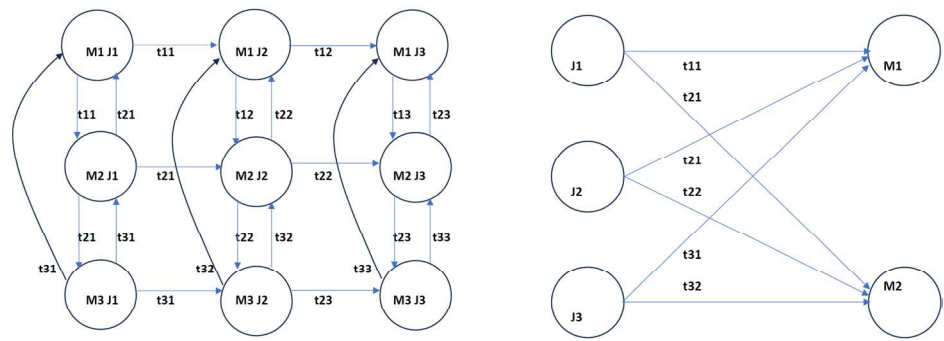
Graph representations are particularly interesting as we can use the vast machinery of Graph Theory to extract meta-features. Selecting a *powerful* graph representation is also important: different ways of representing PFSP instances as graphs can generate meta-features with different predictive power. We distinguish between domain-dependent data content (e.g., processing times acting as edge weights) and problem-agnostic structural syntax. While the graph's values are derived from scheduling, the meta-features themselves (i.e.

degree entropy or centrality variance) are strict topological descriptors. If this framework is mapped to an entirely different optimisation problem, the feature extraction pipeline remains identical, preserving an abstract, transferable taxonomy. Previous applications of meta-learning to the Max-SAT problem have shown that graph-based meta-features are as good as bespoke meta-features [10].

Graph-Based Representations

This work uses metrics to extract meta-features from graphs that represent each instance of the optimisation problem. These metrics are expected to capture structural information about the analysed problem. The graph-based representation can encode both local neighbourhood relations and the global characteristics of the spatiality and density of the structural elements in the problem instance [39].

Two graphs for PFSP are proposed in this work: (1) the Workflow graph and (2) the Job-machine graph. The workflow graph simplifies the permutation graph detailed in [27]. The workflow graph (Fig. 2a) is a directed graph in which each node represents an operation and contains at least three edges: a) edge to the node representing the next job on the machine; b) edge to the node representing the previous job on the machine; and c) edge to the next machine that executes the current job if it exists. The weights on each output edge correspond to the job execution time on the machine. The job-machine graph (Fig. 2b) is a bipartite directed graph that associates jobs with machines, with nodes divided into

Fig. 2 Graphs employed for the meta-features generation

two groups. Jobs form the first group, and machines form the second one. The edge weights represent the job execution time on a given machine.

The adoption of graph-based representations was motivated by the need to capture structural relationships that are difficult to represent through traditional handcrafted descriptors. In PFSP instances, the interactions among jobs, machines, processing orders, and execution times naturally form dependency structures that can be represented as networks. Graphs provide a flexible abstraction that can encode both local interactions and global organisational patterns, enabling the extraction of topological properties independent of problem-specific assumptions. This characteristic is particularly attractive for meta-learning because it enables the construction of more generic feature spaces that may facilitate knowledge transfer across different classes of optimisation problems.

Furthermore, graph representations have demonstrated promising results in related optimisation domains, such as MaxSAT and routing problems, where structural information has been shown to be highly informative for algorithm recommendation tasks. Motivated by these findings, we investigate whether graph-derived representations can provide a similarly effective characterisation of PFSP instances while reducing the dependence on manually designed problem-specific descriptors.

Extracted Meta-Features

The measures used in this paper are briefly described below. Some of them are self-describing, and the others are briefly explained:

- *Number of nodes* n ;
- *Number of edges* m ;
- *Dependency density*: the ratio between the edge and node numbers, given by $dd = \frac{m}{n(n-1)}$;
- *Degree centrality*: the number of incident connections of a vertex, so that the degree can be interpreted as the

probability of the vertex receiving some information from the network [40, 41];

- *Connectivity*: average degree of neighbour vertices [42];
- *Degree*: the number of edges connected to a node;
- *Weight of edges*: sum of the weights of the edges that arrive at a node.

Meta-features such as centrality, connectivity, degree, and weighted edge counts are calculated for each node. The following statistical measures are taken as global characteristics of a graph: mean, standard deviation, coefficient of variation, minimum, maximum, and entropy [43], reaching a total of $4 \times 6 = 24$ meta-features in addition to the first 3 meta-features (number of nodes, number of edges, and dependency density). Considering the two representing graphs (Fig. 2), each meta-instance is described by $2 \times 27 = 54$ input meta-features. The data, characterised in this way, constitute the Feature Space of the meta-learning method.

The selected meta-features were chosen because they capture complementary aspects of graph structure that may influence algorithm performance. Degree-related measures quantify the distribution of interactions among graph elements and provide information about local connectivity patterns. Centrality measures identify the relative importance of nodes within the graph and may reflect the presence of dominant processing relationships. Connectivity measures characterise how information or dependencies propagate throughout the network, while weighted-edge statistics incorporate processing-time information directly into the representation. Finally, entropy-based measures quantify structural heterogeneity and complexity, providing an estimate of how uniformly the scheduling relationships are distributed across the instance.

Rather than relying on domain-specific descriptors, the proposed feature set aims to characterise PFSP instances using general structural properties computable directly from graph representations. This strategy reduces dependence on expert-designed features and aligns to build a more generic and transferable meta-learning framework.

Statistical aggregation measures such as mean, standard deviation, coefficient of variation, minimum, maximum, and entropy were adopted to summarise node-level properties into compact graph-level descriptors. This aggregation strategy preserves relevant structural information while controlling feature dimensionality, contributing to both model interpretability and robustness.

Definition of the Recommended Algorithm and Performance Spaces

Once the candidate metaheuristics have been defined, they must be tuned and assessed to build the Performance Space depicted in Fig. 1. The assessment is performed using a *tuning package*, applied to each instance of the *training set* individually, to obtain the parameters that yield the best performance within predefined *budget limits*. Based on the performance achieved by each candidate algorithm, a temporary ranking is constructed, after which only the best algorithm and its respective set of parameters are used as a *meta-class* (algorithm + parameters). If a winning algorithm appears in more than one meta-label with different parameterisations, a *profiling algorithm* must be applied to reduce the number of similar meta-classes by grouping performance parameters around more representative tuned profiles. Algorithm profiling is equivalent to clustering parameters using a non-parametric clustering method, for which the number of clusters is unknown *a priori*.

The *Affinity Propagation Algorithm* (APA [44]) is well-suited to this role, choosing the median configuration as the group centre (median parameters). The Performance Space can be constructed from the performance of metaheuristics under those archetypal parameter settings. In the following, we present an example Irace configuration file and the tuning outcomes before and after clustering.

```
IRACE CONFIGURATION FILE
# Elitist race
# Elitist new instances: 1
# Elitist limit: 2
# nbIterations: 5
# minNbSurvival: 5
# nbParameters: 11
# seed: 1406598565
# confidence level: 0.95
# budget: 1000
# mu: 5
# deterministic: FALSE
before APA
.ID. algorithm localssearch alpha beta
rho ants nnls q0
1 1 as 0 1.0000 1.0000 0.9500 10 NA NA
2 2 eas 0 3.8873 7.2545 0.4842 17 NA NA
```

```
3 3 eas 2 2.8022 8.5451 0.9700 16 21 NA
4 4 as 1 3.1363 3.0791 0.0884 61 50 NA
5 5 ras 2 1.2820 8.3284 0.3265 8 40 NA
6 6 acs 3 2.2422 7.8826 0.1091 14 40
0.0174
after APA
.ID. algorithm localssearch alpha beta
rho ants nnls q0
1 1 as 0.5 2.06815 2.03955 0.5192 35.5
NA NA
2 2 eas 1 3.34475 7.8998 0.7271 16.5
NA NA
3 5 ras 2 1.2820 8.3284 0.3265 8 40 NA
4 6 acs 3 2.2422 7.8826 0.1091 14 40
0.0174
```

This example shows that records 2 and 3 (“eas” algorithm before APA) become record (“eas” algorithm after APA) by averaging the numerical performance parameters. The same applies to records 1 and 4 related to the “as”. We emphasise that the decision to group similar algorithm profiles is based on similarity or representativeness. Different parameter profiles for the same algorithm may eventually yield different labels: the same algorithm but with distinct parameter sets.

Despite the possibility of building the performance space from scratch, our framework can also operate on the Feasibility of Data Reuse Hypothesis: historical performance landscapes retain predictive validity when mapped to generalised structural graph properties. We explicitly acknowledge that a baseline performance space must initially be constructed via traditional computational campaigns (either by the practitioner or sourced from historical literature). However, once this baseline is established, our framework serves as an abstraction layer that unlocks its latent predictive power, eliminating the need for future end users to replicate these costly configurations.

Generation of the Meta-Learning Model

Once the complete set of meta-instances in the metadata is created, the meta-learning process can be applied. To do this, supervised learning algorithms can induce a suitable meta-model that will hopefully classify previously unseen PFSP instances (meta-instances) correctly. In this context, a meta-instance $\vec{x}_i$ is a multidimensional object represented by m meta-features $x_{i,j}$ with an associated meta-class y_i . The job is thus to learn a discriminant function $f(\vec{x})$ from a training set $S = \{(\vec{x}_i, y_i), \dots, (\vec{x}_n, y_n)\}$, that can relate the n $\vec{x}_i$ with the corresponding y_i and later to exploit this relation to classify previously unseen meta-instances.

The Empirical Analysis

An empirical analysis has been conducted to validate the proposed meta-learning approach for a threefold purpose. First, the performance of the recommendation system is evaluated at the meta-level, i.e., by assessing the meta-learner's predictive accuracy. In other words, we assess the extent to which the meta-learner selects the best metaheuristic for each PFSP instance. Second, the selected metaheuristic is evaluated at the base level. This means that once a meta-model has recommended a given metaheuristic for a specific PFSP instance, we apply that metaheuristic to that instance and assess the quality of the solution it produces. This base-level evaluation is, in fact, the most critical aspect for a user, as they deal directly with the solution returned by a metaheuristic. Third, the interpretability of the induced meta-learning model is considered, as analysing that model may provide valuable insights into which meta-features have greater predictive power for selecting the best metaheuristic for each PFSP problem.

The evaluation was conducted at two complementary levels. At the meta-level, predictive performance was assessed using classification accuracy, which measures the proportion of PFSP instances that the meta-learner correctly labels with the best-performing metaheuristic or compound label. Accuracy was adopted because the task is formulated as a supervised algorithm-selection problem, in which each instance is associated with a target meta-class.

At the base level, optimisation performance was evaluated using Error and Gap values computed with respect to the best-known solution. Error measures the absolute deviation between the solution obtained by the selected metaheuristic and the best-known reference value, whereas Gap expresses this deviation as a percentage. These measures allow us to assess not only whether the meta-learner predicts the correct class, but also the practical optimisation cost of an incorrect recommendation.

Experimental Setup

Three well-known classification algorithms were used as meta-learners [45]: Support Vector Machines (SVM), Decision Trees (DT), and k -Nearest Neighbours (k -NN). These classifiers were selected because they represent distinct inductive biases and complementary learning paradigms, allowing a broader assessment of the proposed feature space. Decision Trees provide highly interpretable models and explicit decision rules, facilitating the analysis of the relationship between meta-features and algorithm performance. SVMs offer strong generalisation capabilities in high-dimensional spaces and are effective at modelling complex decision boundaries, whereas k -NN captures local similarity

patterns among instances through an instance-based learning strategy. This diversity enables a more comprehensive evaluation of the predictive information contained in the proposed graph-derived meta-features. Furthermore, these techniques are powerful and computationally efficient methods that have been extensively validated in both the scientific literature and practical applications [46].

They were implemented using the Python libraries *Scikit-learn*, *NumPy*, and *Pandas*. The default parameters provided by Scikit-learn were adopted for all classifiers. This choice was intentional and intended to evaluate the predictive power of the proposed graph-derived meta-feature space independently of extensive classifier-specific tuning. By avoiding additional hyperparameter optimisation, the experimental design provides a more conservative assessment of the framework. It reduces the possibility that performance gains are primarily attributable to classifier fine-tuning rather than to the proposed instance representation itself. Furthermore, the use of default settings improves reproducibility and facilitates comparisons with future studies. Although additional performance gains might be achieved through dedicated hyperparameter optimisation, such an investigation falls outside the scope of this work, whose primary objective is to assess the effectiveness of graph-derived meta-features for selecting PFSP algorithms. Moreover, we have used the *Python* language version 3.7 and ran the experiments on a computer with a Core i7 processor with six real cores, frequency of 2.6 GHz, 32 GB of RAM, and running the operating system *Windows 64-bit*, version 11.

The implementation of the proposed framework involves three main stages: graph construction, meta-feature extraction, and meta-model induction. The graph generation and feature extraction procedures were implemented using Python-based data processing libraries (i.e., *pandas*), while the classification models were implemented using *Scikit-learn*. Since the framework relies on previously available performance data, no additional computationally expensive tuning was required during the experiments, significantly reducing implementation cost compared to approaches that require exhaustive algorithm configuration. In practical terms, the most computationally demanding stage of the framework is generating the original performance space, a step that was reused from previous work. Once the performance data is available, the proposed meta-learning pipeline can be trained and evaluated with relatively low computational cost.

At the base level, the performances of all candidate metaheuristics are evaluated to define the target meta-class representing the metaheuristic that best solves the current PFSP instance. However, the performance of each candidate metaheuristic depends on how its parameters are adjusted.

Therefore, the meta-classes are defined after the tuning process is complete. This paper uses a procedure similar to that in [4], in which the tuning process involves constructing a grammar to combine the algorithmic components within the iterated local search (ILS) framework. The ILS is a conceptually simple, flexible, and easy-to-implement metaheuristic with a modular architecture, making it suitable for algorithm engineering [47].

In [4], the *Irace* package explored the algorithmic search space to create ILS variations from previously coded and adjustable combinations of categorised, ordinal, and numeric parameters. The coded algorithm space encompasses constructive heuristics (*NEH*, *FRB5*, and Insertion Local Search), tiebreakers (non-increasing and non-decreasing order of total processing time, and orders *KK1* and *KK2*), perturbation and local search functions, and acceptance criterion. The top 10 algorithms, labelled A0, A1, A2, A3, A4, A5, A6, A7, A8, and A9, form the base level for this work.

Two datasets containing PFSP instances for minimising makespan have been employed: the Taillard [26] and the VRF datasets [28]. The former contains 12 groups of 10 PFSP instances, ranging from 20 jobs and 5 machines to 500 jobs and 20 machines. The latter consists of 24 groups of 10 PFSP instances varying from 100 jobs and 20 machines to 800 jobs and 60 machines.

Meta-Level Performance

In this section, we describe the methodology and results for the meta-level problem, in which classifiers are employed to generate a decision model for the base-level problem. The meta-level accuracy evaluates the recommendation capability of the model, while Error and Gap evaluate the practical impact of the recommendation on the optimisation task.

Multi-label treatment Since both datasets share the same meta-features, we have created a single dataset combining all instances from the original Taillard and VRF datasets. Before producing this combined Taillard+VRF dataset, the Taillard dataset, originally a multi-label dataset due to ties in the candidate algorithms' performance, was converted into a single-label dataset using the Label Power set (LP) method. LP transforms a multi-label dataset into a single-target multi-class dataset by using the unique label sets as separate classes. Hence, any regular multi-class classifier can handle the transformed dataset as a single-label dataset [48].

The following is a short code explaining the multi-label convention. Since numbers represent candidate algorithms '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', we use letters meaning composite labels 'A', 'B', 'C', 'D', i.e., algorithms that tie in performance for a given instance. Additionally, we use the meta-class y (as defined in Section 3.3). For multi-label treatment,

y is the set of candidate algorithms (metaheuristics) tied for the winner on a given dataset. As can be seen, there are only four cases in which multiple candidate algorithms were tied as winners, i.e., they have found the best solution (probably easy-to-solve instances).

```
label 'A' if y= '0','1','2','3','4','6',
              '7','8','9'
label 'B' if y= '1','3','8'
label 'C' if y= '0','1','2','3','4','5',
              '6','7','8','9'
label      'D' if y=
'0','2','3','4','5','6','8','9'
```

Training and test Subsequently, the well-known k -fold cross-validation procedure is used, dividing the data instances into $k = 10$ folds. At each execution, $k-1$ folds are used to train the model, and the remaining fold is used for testing. This process is repeated until all 10 folds have been used as the testing fold, and the results are averaged over the 10 testing folds.

Feature selection The Recursive Feature Elimination (RFE) method [49] is used as a wrapper method for meta-feature selection during a pre-processing phase on the combined Taillard+VRF dataset. More precisely, each training set (within the cross-validation procedure) was divided into a *reduced training subset* and a *validation subset* (66% and 33% respectively). RFE is performed 5-fold on the reduced training set, returning a ranked list of the meta-features that appeared most frequently in the best training results. This work employs the *F1-score* [50] as the ranking function, yielding a ranked list of meta-features.

A simplified procedure was used to identify the meta-features with the greatest predictive power by progressively reducing the ranked list to at most 5 features. For each S -sized meta-feature list, the training was performed on the filtered, reduced training subset, and the model was evaluated on the validation subset.

Based on the average across all folds on the validation set, the best size for the ranked meta-feature list is $S = 5$. The choice of a reduced feature subset ($S = 5$) aims to avoid overfitting and improve model interpretability while maintaining competitive predictive performance. This trade-off is particularly important in meta-learning scenarios, where high-dimensional feature spaces may lead to unstable models.

The choice of $S = 5$ also reflects the objective of obtaining a compact and interpretable representation of PFSP instances. Since the proposed framework emphasises the interpretability of the recommendation process, retaining a small subset of highly informative meta-features facilitates the analysis of the induced models while reducing the risk of incorporating redundant information. This is particularly important in meta-learning settings, where the number of

available instances is often limited and unnecessarily large feature spaces may undermine model robustness and generalisation. Furthermore, decision trees provide an intrinsic ranking of feature importance and yield highly interpretable models, making them particularly well-suited to identifying the most informative meta-features in the proposed framework.

For this procedure, only the decision tree (DT) was used as a meta-classifier, given the importance of interpreting the meta-model in this work. Thus, we leverage this $S = 5$ result for the other two meta-classifiers described next.

To leverage a 5-sized meta-feature list for the other two meta-classifiers, SVM and K-NN were trained using the five best meta-features selected by the wrapper feature selection

procedure. Again, this involved applying the two meta-classifiers to the original (full) training set and measuring their predictive accuracy on the test set.

Figure 3 describes the meta-feature selection frequency for each meta-classifier after the 10 iterations of the cross-validation procedure. As can be observed, the workflow graph tends to capture more relevant information since the top 3 meta-features for all meta-learners are obtained from graph 1: `std_dcg1` (centrality degree standard deviation), `mean_dcg1` (centrality degree mean), `ent_degreeg1` (degree entropy), and `ent_connectivityg1` (connectivity entropy). Some details of these meta-features and others selected by the algorithms are shown in Table 2.

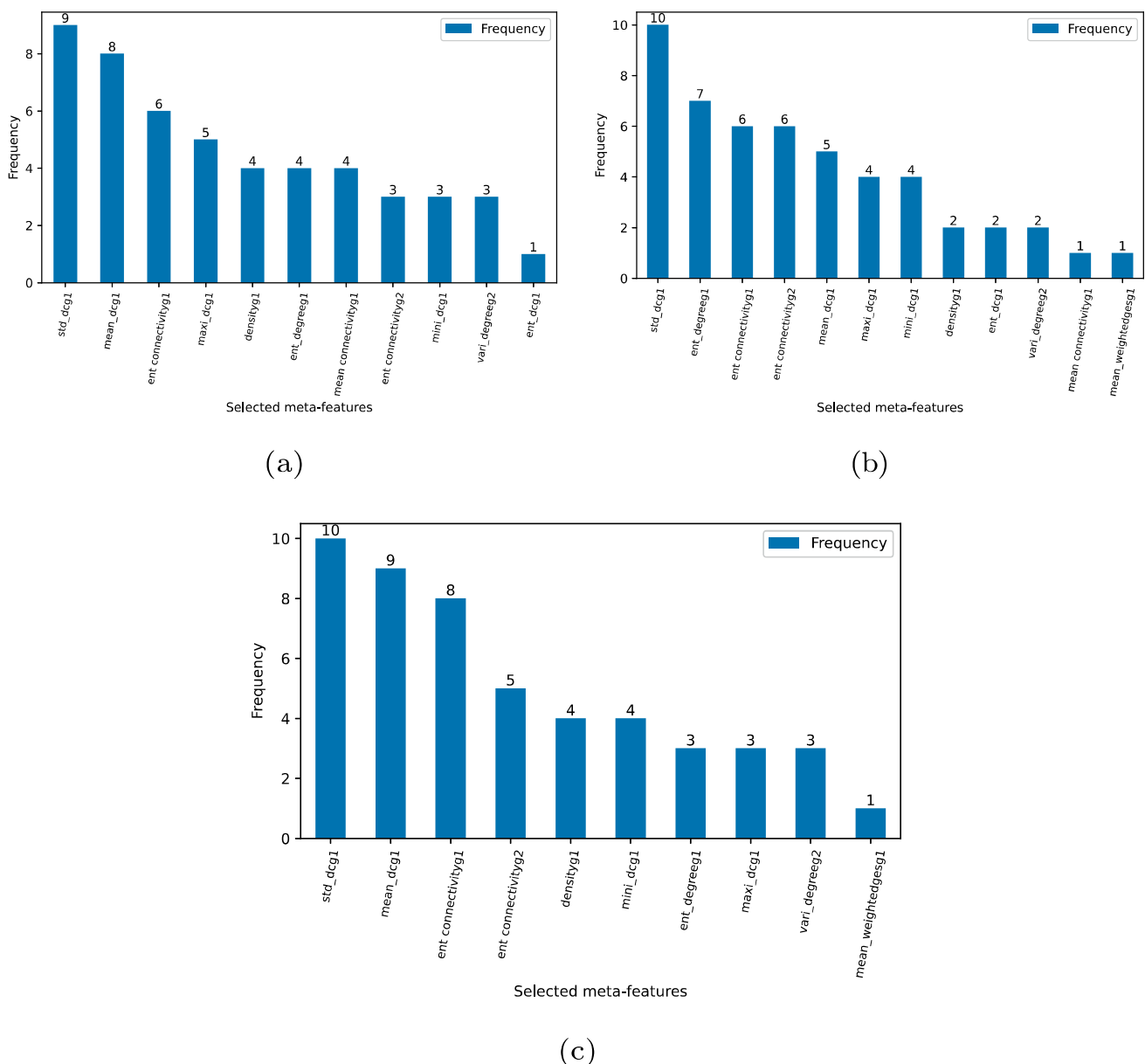


Fig. 3 Meta-feature frequency for a) SVM, b) K-NN, and c) DT classifiers after 10 executions

Table 2 Description of the most selected meta-features used in the proposed meta-learning framework

Feature	Description
std_dcg1	Centrality degree standard deviation graph 1
mean_dcg1	Centrality degree mean graph 1
ent_degreeg1	Degree entropy graph 1
ent_connectivityg1	Connectivity entropy of graph 1
maxi_dcg1	Maximum value of centrality degree of graph 1
densityg1	Density of graph 1
ent_degreeg1	Entropy degree of graph 1
mean_connectivityg1	Connectivity average of graph 1
ent_connectivityg2	Connectivity entropy of graph 2
mini_dcg1	Minimum value of centrality degree of graph 1
vari_degreeg2	Degree variance of graph 2
ent_dcg1	Centrality degree entropy of graph 1
mean_weightedgesg1	Weight edges mean of graph 1
NumNodesg1	Number of nodes on graph 1

Table 3 Accuracy on each test fold and mean accuracy obtained in the 10-fold cross-validation procedure by SVM, k -NN, and DT for Taillard + VRF dataset using the five best-ranked meta-features

Execution	SVM	k -NN	DT
#1	0.305	1.000	1.000
#2	0.722	0.972	1.000
#3	0.805	1.000	1.000
#4	0.555	1.000	1.000
#5	0.555	1.000	1.000
#6	0.583	1.000	1.000
#7	0.555	1.000	1.000
#8	0.555	1.000	0.972
#9	0.527	1.000	1.000
#10	0.833	0.777	1.000
Average	0.600	0.975	0.997

Table 3 presents the accuracy results on the 10 test folds of the cross-validation procedure for the experiments with the mixed data set (*VRF + Taillard*), using SVM, k -NN and DT as the classification algorithms to learn the meta-model. Note that, for all the 10 executions of the classification algorithms in the 10-fold cross-validation procedure, the decision tree and the k -NN algorithms gave the best results, with the decision tree algorithm performing slightly better overall.

This result suggests that the relationships between meta-features and algorithm performance are likely non-linear but still sufficiently structured to be captured by interpretable models such as decision trees. The strong performance of k -NN also indicates that instances with similar structural characteristics tend to share similar optimal algorithms, reinforcing the relevance of the proposed feature space. However, the extremely high predictive accuracy observed raises concerns about potential overfitting or limited diversity

among the benchmark instances, particularly given the relatively small number considered. This reinforces the need for further validation on larger and more diverse datasets.

The practical significance of these results becomes evident when considering the algorithm-selection objective. High predictive accuracy implies that the recommendation model can reliably identify the best-performing metaheuristic for a given PFSP instance, reducing the need for exhaustive experimentation across multiple candidate solvers. Consequently, the benefits of the proposed approach extend beyond classification performance, as accurate recommendations can translate directly into improved optimisation outcomes, reduced computational effort, and more effective use of tuned metaheuristics in real-world scheduling scenarios.

The near-perfect predictive accuracy (99.7%) achieved by the Decision Tree across the 10-fold cross-validation procedure demonstrates exceptional generalisation capabilities across the problem space. Because this accuracy is strictly measured on unseen test folds during cross-validation, the risk of traditional model overfitting is minimised. Instead, this high performance indicates that the proposed graph-derived meta-features capture highly structured, distinct topological boundaries that map cleanly to solver performance.

Base Level Performance

The performance space, as provided in [4], is used as a basis to evaluate the gain in optimisation accuracy associated with selecting the best solver using the meta-learning approaches (SVM, DT, and k -NN). Using the instances' deviation values (Error and Gap), calculated over the best-known solution [20], the meta-heuristic error for each instance is conjugated with classification error to provide a cumulative measure regarding classifier performance $\times$ solver performance. A class correctly predicted by a meta-model means choosing the best solver for a given PFSP problem instance, while a class wrongly predicted can lead to a solver that performs poorly. This experiment has used the whole benchmark (Taillard+VRF dataset), including compound labels (multi-label classes A, B, C, and D). The reuse of previously available performance data reduces computational cost and improves reproducibility. However, this design choice also implies a trade-off, as the diversity and representativeness of the performance space depend on the original study.

The candidate algorithms used to construct the performance space were validated through the Wilcoxon signed-rank test with Bonferroni correction at a significance level of 0.05, demonstrating statistically significant differences between several AAC-generated algorithms and the state-of-the-art IG_{lsp} algorithm. Significant improvements

were reported for algorithms A0, A2, A3, A4, A5, and A9 ($p < 1.3 \times 10^{-8}$), as well as A7 ($p < 0.002$) [4].

The results are shown in Table 4, in which our approach is compared with algorithms A0, A1,..., A9, generated by the Automatic Algorithm Configuration (AAC), proposed in [4]. Besides, we also provide results for the state-of-the-art IG_{lsp} (Iterated Greedy with Local Search on Partial Solutions) algorithm to broaden our comparisons. Recalling error and gap are calculated over the best-known solution [20].

As can be seen in Table 4, most AAC approaches have superior performance (mainly A2, A3, and A4, reaching 0.4466, 0.4455, and 0.4157 of error, respectively) when compared to IG_{lsp} (which has an average error of 0.6424 and Gap of 67.88% concerning the optimal choice). It is essential to emphasise the improvements achieved by the meta-learners over state-of-the-art models. For instance, when applied to large VRF instances, the A4 algorithm has achieved the best overall performance among the 10 AAC algorithms (Error: 0.4157 and Gap: 8.65%). Selecting A4 is equivalent to the trivial choice of the best-performing algorithm applied to all instances.

However, when considering the meta-learning approaches, the errors (gaps) obtained by SVM, DT, and k -NN were smaller: 0.406 (6.07%), 0.384 (0.28%), and 0.393 (2.77%), respectively. As one can see, DT has achieved the best results, i.e., 0.28% above the ground truth (optimal choice). Even SVM (the worst-performing of the three meta-models in our experiments) performs better than selecting only the well-tuned metaheuristic A4.

These results indicate that the proposed meta-learning framework effectively adapts the algorithm selection process to instance-specific characteristics, outperforming the

static selection of a single best algorithm. This highlights the importance of instance-aware optimisation strategies.

From a practical perspective, the results demonstrate that adaptive algorithm selection can be more effective than relying on a single globally competitive solver. Although algorithm A4 achieved the best overall performance among the AAC-generated algorithms, the meta-learning approaches consistently obtained lower Error and Gap values by tailoring the recommendation to the characteristics of each PFSP instance. This finding reinforces the central premise of algorithm selection: no single metaheuristic is expected to perform optimally across all instances. Therefore, the observed improvements are not merely associated with predictive accuracy, but correspond to measurable gains in optimisation quality.

Interpreting the Best Meta-Model

In this section, we interpret the best meta-model, a decision tree, to gain insights into the interrelationship between solver algorithms and problem instances. As we can observe, it is possible to gain a deeper understanding of the problem's structure from the data model and the algorithms that best solve each instance.

Figure 4 shows the decision tree meta-model learned from the entire dataset. In terms of individual meta-features, the most important meta-feature is located at the root node, which is used to classify all meta-instances: `vari_dcg1`, short for *centrality degree variance* calculated over graph 1 (workflow graph exemplified in Fig. 2a).

For a more detailed interpretation of this decision tree, however, it is useful to focus on full paths (from the root to a leaf node) where the leaf node represents meta-class predictions with both very high accuracy and high coverage (classifying a relatively large number of meta-instances). The two strongest paths in this decision tree, given these characteristics, are:

First, the path corresponding to the rule:

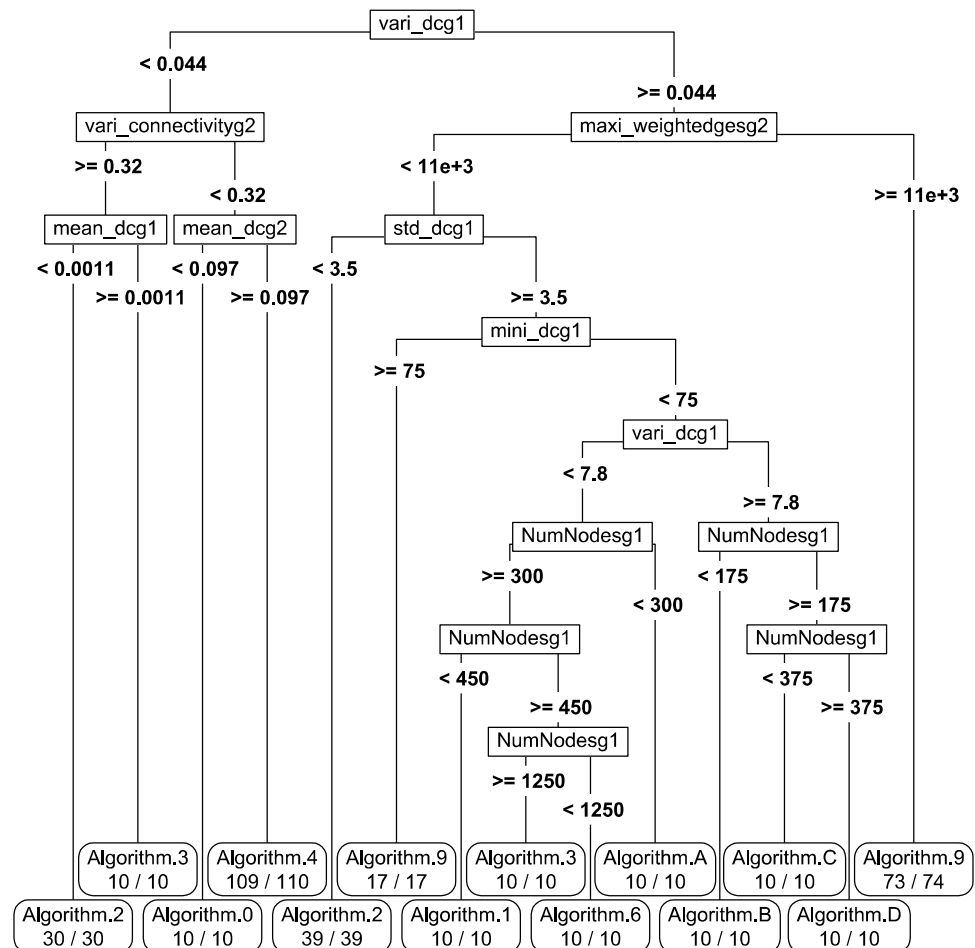
```
IF (vari_dcg1 < 0.044) and (vari_connectivityg2 < 0.32)
  and (mean_dcg2 >= 0.097)
  THEN predict Algorithm.4
```

where `vari_connectivityg2` is the *connectivity variance* calculated over graph 2 (job-machine graph exemplified in Fig. 2b), `mean_dcg2` is the *centrality degree mean*, and `Algorithm.4` (label A4) is the algorithm with the most hits (wins) for all instances in the two datasets used. *Algorithm 4* is initiated by the *Nawaz-Enscore-Ham* (NEH) heuristic [25] with a tie-breaker based on *Kalczynski-Kamburowski* procedure [51] and alternating between two distinct local search procedures [4]. This rule achieves 99.1%

Table 4 Performance comparison concerning error and gap for various algorithms considering Automatic Algorithm Configuration(AAC) and Meta Learning Approaches (MLA)

Algorithm	Error	Gap
IG_{lsp}	0.6424	67.88%
Ground truth	0.3826	0.00%
A0 (AAC)	0.4846	26.64%
A1 (AAC)	0.6322	65.23%
A2 (AAC)	0.4466	16.72%
A3 (AAC)	0.4455	16.43%
A4 (AAC)	0.4157	8.65%
A5 (AAC)	0.5122	33.87%
A6 (AAC)	0.6859	79.26%
A7 (AAC)	0.6798	77.67%
A8 (AAC)	0.7501	96.03%
A9 (AAC)	0.4632	21.06%
SVM (MLA)	0.406	6.07%
DT (MLA)	0.384	0.28%
k -NN (MLA)	0.393	2.77%

Fig. 4 Decision tree learned using the whole dataset



accuracy (109/110) and covers 110 meta-instances in this meta-dataset.

Rather than forcing human experts to interpret complex, problem-specific landscape metrics, our model offers structural interpretability. For example, the rule predicting Algorithm 4 relies on centrality variance and connectivity boundaries, demonstrating that the meta-learner is capturing universal structural layout configurations rather than just bespoke problem sizes, allowing us to interpret solver suitability through the lens of network topology.

The second strongest path in this decision tree meta-model corresponds to the rule:

IF (vari_dcg1 ≥ 0.044) and (maxi_weightedgesg2 $\geq 11e+3$)
THEN predict Algorithm.9

where maxi_weightedgesg2 denotes the maximum edge weight taken from the job-machine graph. This rule also achieves 98.6% (73/74) accuracy and covers 74 meta-instances. *Algorithm 9* differs from *Algorithm 4* by using NEH in a loop, recovering the best solution initialised, with

a tie-breaker alternating *Kalczynski-Kamburowski* procedure and random choice, and only one local search strategy [4].

These decision rules reveal that structural properties captured by graph-based meta-features are strongly associated with algorithm performance. In particular, features related to centrality and edge weights appear to play a critical role, suggesting that both structural variability and the distribution of processing times influence solver effectiveness.

Interestingly, algorithm designers are more likely to use bespoke meta-features (e.g., the number of machines and jobs, critical path length, and timing) to select the best solver algorithm. In our approach, edge weights, which represent processing times, end up being relevant as attributes taken from the job-machine graph; however, nodes and edges (specifically from the workflow graph), which are misrepresented as graphsm, are poorly discriminating, appearing associated with just 70 instances, spread across seven different algorithms, including those encompassed by compound labels (multi-label classes A, B, C, and D). Recall that label C, for example, covers all candidate algorithms. Research efforts are needed to better understand the relationship between the main meta-features and the flow-shop taxonomy.

Conclusion

In the realm of optimisation problems, selecting the most suitable algorithm for a given instance presents a formidable challenge. This work introduces a graph-based meta-learning framework for algorithm selection in the Permutation Flow-Shop Problem (PFSP). The main contributions of the proposed approach are threefold: (i) the representation of PFSP instances through graph structures specifically designed to capture scheduling relationships, (ii) the use of exclusively graph-derived meta-features to characterise problem instances, and (iii) the development of an interpretable recommendation model capable of selecting the most appropriate tuned metaheuristic for a given instance.

A core structural advantage of the proposed framework is its architectural independence from the performance space. The graph feature extraction operates entirely upstream of algorithm evaluation. Consequently, the framework is not restricted to a single configuration repository; rather, it is designed to dynamically scale with the community's expanding open-data benchmarks, providing a highly reproducible path for automated algorithm selection.

The meta-level performance results indicate that the Decision Tree (DT) achieved the best predictive performance among the evaluated classifiers, consistently outperforming the others in accuracy when using the top five meta-features. This suggests that the selected meta-features effectively capture the essential characteristics required for accurate predictions, thereby contributing to the meta-learning literature. In addition, the results demonstrate that the proposed framework remains robust across different classifiers and can effectively address challenges in parameter tuning and performance space.

The findings demonstrate that meta-learning approaches, especially DT, can significantly improve optimisation accuracy compared to traditional algorithms such as IG_{lsp} and even well-configured AAC approaches. Interpreting the decision tree meta-model reveals that certain meta-features, such as centrality degree variance and connectivity variance, play crucial roles in determining the best solver algorithm. The decision rules derived from the tree provide valuable insights into the relationship between problem characteristics and solver effectiveness, which can guide future algorithm design and selection processes.

In contrast to recent studies that rely on handcrafted descriptors, mixed feature spaces, or problem-specific representations, the proposed framework shows that graph-derived meta-features alone can support competitive and interpretable algorithm selection for PFSP. Moreover, by reusing existing tuned solver performance data, the framework reduces the cost of constructing the meta-learning

environment, which is an important practical advantage over studies that depend on exhaustive experimentation.

Our results demonstrate that meta-learning can effectively identify predictive structural characteristics and support accurate algorithm selection for PFSP instances. The decision tree and k-Nearest Neighbours classifiers achieved results above 97.5% accuracy in 10-fold cross-validation tests, significantly outperforming the default application of the best overall algorithm. The experiment can be replicated for any new problem for which one or more graphs capture aspects of the problem under study, thereby producing a feature space. Moreover, it emphasises the importance of meta-feature selection and the interpretability of the meta-model, thereby contributing to a deeper understanding of the underlying factors influencing algorithm performance and making our findings directly applicable to real-world scenarios.

A limitation of the proposed method is the requirement to represent the problem as a graph, which may not always be straightforward depending on the problem structure. Future work should address these limitations by investigating alternative representations, such as extracting meta-features directly from mathematical models or syntactic trees [52], and by expanding the dataset with additional PFSP instances and solver configurations. Another important direction is to evaluate the impact of incorporating new tuned algorithms into the performance space and analysing how this affects the stability and adaptability of the meta-model. Finally, extending the proposed framework to a multi-problem setting, where knowledge can be transferred across different optimisation problems, represents a promising avenue for improving both generalisation and practical applicability.

While the high predictive accuracy reflects the strength and discriminative power of graph-based representation, it also indicates a high degree of structural clarity within the combined Taillard and VRF benchmarks. Rather than a symptom of overfitting, which is guarded against by the cross-validation protocol, this outcome suggests that these classic benchmarks have reached saturation with this meta-learning framework. Future work will therefore focus on testing the model against synthetic, highly noisy, or dynamically generated industrial instances to evaluate the boundaries of this structural clarity.

Author Contributions Data curation: Guilherme Alberto Sousa Ribeiro, Chrystian Gustavo Martins Nascimento, Bruno Feres de Souza, Alexandre César Muniz de Oliveira; Methodology: Guilherme Alberto Sousa Ribeiro, Chrystian Gustavo Martins Nascimento, Omar Andrés Carmona Cortés, Alexandre César Muniz de Oliveira; Formal analysis and investigation: Guilherme Alberto Sousa Ribeiro, Chrystian Gustavo Martins Nascimento; Writing - original draft preparation: Guilherme Alberto Sousa Ribeiro, Chrystian Gustavo Martins Nascimento, Bruno Feres de Souza, Omar Andrés Carmona Cortés, Anne Magaly de Paula Canuto, João Carlos Xavier-Junior, Alex A. Freitas, Alexan-

dre César Muniz de Oliveira; Writing - review and editing: Guilherme Alberto Sousa Ribeiro, Omar Andrés Carmona Cortés, Alex A. Freitas, Alexandre César Muniz de Oliveira; Supervision: Alexandre César Muniz de Oliveira.

Funding The Article Processing Charge (APC) for the publication of this research was funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (ROR identifier: 00x0ma614). The authors acknowledge the financial support given by CAPES (88881.119363/2016-01).

Data Availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of Interest The authors declare that the research was conducted without any commercial or financial relationships that could potentially create a conflict of interest.

Ethical Approval Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Nagano MS, Ruiz R, Lorena LAN. A constructive genetic algorithm for permutation flowshop scheduling. *Comput Ind Eng*. 2008;55(1):195–207. <https://doi.org/10.1016/j.cie.2007.11.018>.
- Demsar J. Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res*. 2006;7:1–30.
- Fonseca THL, Oliveira ACM. Tuning of clustering search based metaheuristic by cross-validated racing approach. In: *International Work-Conference on Artificial Neural Networks*. Springer; 2017. p. 62–72.
- Brum A, Ritt M. Automatic design of heuristics for minimizing the makespan in permutation flow shops. In: *2018 IEEE Congress on Evolutionary Computation (CEC)*, p. 1–8 2018. <https://doi.org/10.1109/CEC.2018.8477787>
- Fernandez-Viagas V, Framinan JM. Neh-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness. *Comput Operations Res*. 2015;60:27–36. <https://doi.org/10.1016/j.cor.2015.02.002>.
- Wolpert DH, Macready WG. No free lunch theorems for optimization. *Evol Comput IEEE Transac*. 1997;1(1):67–82.
- Brazdil P, Giraud-Carrier C, Soares C, Vilalta R. *Metalearning: applications to data mining*. 1st ed. Berlin Heidelberg: Springer; 2008.
- Kanda J, Carvalho A, Hruschka E, Soares C, Brazdil P. Meta-learning to select the best meta-heuristic for the traveling salesman problem: a comparison of meta-features. *Neurocomputing*. 2016;205:393–406.
- Pavelski LM, Delgado M, Kessaci ME, Freitas AA. Stochastic local search and parameters recommendation: a case study on flowshop problems. *Int Trans Oper Res*. 2021. <https://doi.org/10.1111/itor.12922>.
- Miranda ES, Fabris F, Nascimento CGM, Freitas AA, Oliveira ACM. Meta-learning for recommending metaheuristics for the maxsat problem. In: *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, p. 169–174 2018. <https://doi.org/10.1109/BRACIS.2018.00037>
- Amadini R, Biselli F, Gabbrielli M, Liu T, Mauro J. Feature selection for sunny: A study on the algorithm selection library. In: *Tools with Artificial Intelligence (ICTAI)*, 2015 IEEE 27th International Conference On, p. 25–32 2015. IEEE
- Rothfuss J, Koenig C, Rupenyan A, Krause A. Meta-learning priors for safe bayesian optimization. In: *Conference on Robot Learning*. 2023. p. 237–65 (PMLR).
- Bossenbroek MB. Meta-learning using neural networks in large-scale traveling salesman problems. Master's thesis, Erasmus University Rotterdam; 2023.
- León-Hicks E, Conant-Pablos SE, Ortiz-Bayliss JC, Terashima-Marin H. Addressing the algorithm selection problem through an attention-based meta-learner approach. *Appl Sci*. 2023;13:4601.
- Chu X, Li S, Gao F, Cui C, Pfeiffer F, Cui J. A data-driven meta-learning recommendation model for multi-mode resource constrained project scheduling problem. *Comput Operations Res*. 2023;157:106290.
- Pérez C, Climent L, Nicolás G, Arbelaez A, Salido MA. A hybrid metaheuristic with learning for a real supply chain scheduling problem. *Eng Appl Artif Intell*. 2023;126:107188.
- Johnson SM. Optimal two-and three-stage production schedules with setup times included. *Naval Res Logistics Quarterly*. 1954;1(1):61–8.
- Framinan JM, Gupta JN, Leisten R. A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *J Operat Res Soc*. 2004;55:1243–55.
- Kumar S, Jadon P. A novel hybrid algorithm for permutation flow shop scheduling. 2014. arXiv preprint [arXiv:1407.5931](https://arxiv.org/abs/1407.5931).
- Dubois-Lacoste J, Pagnozzi F, Stützle T. An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem. *Comput Operat Res*. 2017;81:160–6. <https://doi.org/10.1016/j.cor.2016.12.021>.
- Liu W, Jin Y, Price M. A new improved NEH heuristic for permutation flowshop scheduling problems. *Int J Prod Econ*. 2017;193:21–30. <https://doi.org/10.1016/j.ijpe.2017.06.026>.
- Li X, Yin M. A discrete artificial bee colony algorithm with composite mutation strategies for permutation flow shop scheduling problem. *Scientia Iranica*. 2012;19(6):1921–35. <https://doi.org/10.1016/j.scient.2012.10.034>.
- Ruiz R, Stützle T. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *Eur J Oper Res*. 2007;177(3):2033–49. <https://doi.org/10.1016/j.ejor.2005.12.009>.
- Karimi-Mamaghan M, Mohammadi M, Meyer P, Karimi-Mamaghan AM, Talbi E-G. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. *Eur J Oper Res*. 2022;296(2):393–422. <https://doi.org/10.1016/j.ejor.2021.04.032>.
- Nawaz M, Enscore EE, Ham I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*. 1983;11(1):91–5. [https://doi.org/10.1016/05-0483\(83\)90088-9](https://doi.org/10.1016/05-0483(83)90088-9).
- Taillard E. Benchmarks for basic scheduling problems. *Eur J Oper Res*. 1993;64(2):278–85.

27. Makuchowski M. Permutation, no-wait, no-idle flow shop problems. *Archi Control Sci.* 2015;25:189–99. <https://doi.org/10.1515/acsc-2015-0012>.
28. Vallada E, Ruiz R, Framinan JM. New hard benchmark for flow-shop scheduling problems minimising makespan. *Eur J Oper Res.* 2015;240(3):666–77. <https://doi.org/10.1016/j.ejor.2014.07.033>.
29. Framinan JM, Leisten R, Rajendran C. Different initial sequences for the heuristic of nawaz, enscore and ham to minimize makespan, idle time or flowtime in the static permutation flowshop sequencing problem. *Int J Prod Res.* 2003;41(1):121–48. <https://doi.org/10.1080/00207540210161650>.
30. Brum A, Ruiz R, Ritt M. Automatic generation of iterated greedy algorithms for the non-permutation flow shop scheduling problem with total completion time minimization. *Comput Ind Eng.* 2022;163:107843. <https://doi.org/10.1016/j.cie.2021.107843>.
31. Zhao Z, Zhou M, Liu S. Iterated greedy algorithms for flow-shop scheduling problems: A tutorial. *IEEE Trans Autom Sci Eng.* 2022;19(3):1941–59. <https://doi.org/10.1109/TASE.2021.3062994>.
32. Zhao C, Wu L, Zuo C, Zhang H. An improved fruit fly optimization algorithm with q-learning for solving distributed permutation flow shop scheduling problems. *Complex Intell Syst.* 2024;1–24.
33. Talbi E-G. Machine learning into metaheuristics: a survey and taxonomy. *ACM Comput Surv.* 2021. <https://doi.org/10.1145/3459664>.
34. Pavelski LM, Kessaci M, Delgado MR. Recommending metaheuristics and configurations for the flowshop problem via meta-learning: Analysis and design. In: 2018 7th Brazilian Conference on Intelligent Systems (BRACIS), p. 163–168 2018. <https://doi.org/10.1109/BRACIS.2018.00036>.
35. Pavelski L, Delgado M, Kessaci M-E. Meta-learning for optimization: A case study on the flowshop problem using decision trees. In: 2018 IEEE Congress on Evolutionary Computation (CEC), p. 1–8 2018. <https://doi.org/10.1109/CEC.2018.8477664>.
36. Jiang H, Wang Y, Tian Y, Zhang X, Xiao J. Feature construction for meta-heuristic algorithm recommendation of capacitated vehicle routing problems. *ACM Trans Evol Learn Optim.* 2021. <https://doi.org/10.1145/3447540>.
37. Pihera J, Musliu N. Application of machine learning to algorithm selection for TSP. In: IEEE 26th Int. Conf. on Tools with AI. 2014. p. 47–54.
38. Rice JR. The algorithm selection problem. *Adv Comput.* 1976;15:65–118.
39. Morais G, Prati RC. Complex network measures for data set characterization. In: Brazilian Conf. on Intel. Systems (BRACIS). IEEE; 2013. p. 12–8.
40. Metcalf L, Casey W. Chapter 5 - graph theory. In: Metcalf, L., Casey, W. (eds.) *Cybersecurity and Applied Mathematics*, pp. 67–94. Syngress, Boston 2016. <https://doi.org/10.1016/B978-0-12-804452-0.00005-1>. <https://www.sciencedirect.com/science/article/pii/B9780128044520000051>
41. Golbeck J. Chapter 21 - analyzing networks. In: Golbeck, J. (ed.) *Introduction to Social Media Investigation*, p. 221–235 2015. Syngress, Boston. <https://doi.org/10.1016/B978-0-12-801656-5.00021-4>. <https://www.sciencedirect.com/science/article/pii/B9780128016565000214>
42. Kim J, O S. Average connectivity and average edge-connectivity in graphs. *Discret Math.* 2013;313(20):2232–8. <https://doi.org/10.1016/j.disc.2013.05.024>.
43. Nudelman E, Leyton-Brown K, Hoos HH, Devkar A, Shoham Y. Understanding random sat: Beyond the clauses-to-variables ratio. In: *International Conference on Principles and Practice of Constraint Programming*. Springer; 2004. p. 438–52.
44. Frey BJ, Dueck D. Clustering by passing messages between data points. *Science.* 2007;315(5814):972–6.
45. Bishop CM. *Pattern recognition and machine learning* (Information Science and Statistics). Berlin, Heidelberg: Springer; 2006.
46. Wiyono S, Abidin T, Wibowo D, Hidayatullah M, Dairoh D. Comparative study of machine learning KNN, SVM, and decision tree algorithm to predict students performance. *Int J Res-Granthaalayah.* 2019;7(1):190–6.
47. Lourenço HR, Martin OC, Stützle T. Iterated local search: Framework and applications. In: Gendreau, M., Potvin, J.-Y. (eds.) *Handbook of Metaheuristics*, p. 363–397. Springer, Boston, MA 2010. https://doi.org/10.1007/978-1-4419-1665-5_12
48. Bogatinovski J, Todorovski L, Džeroski S, Koccev D. Comprehensive comparative study of multi-label classification methods. *Expert Syst Appl.* 2022;203:117215. <https://doi.org/10.1016/j.eswa.2022.117215>.
49. Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. *Mach Learn.* 2002;46:389–422. <https://doi.org/10.1023/A:1012487302797>.
50. Yacouby R, Axman D. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In: *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pp. 79–91. Association for Computational Linguistics, Online 2020. <https://doi.org/10.18653/v1/2020.eval4nlp-1.9>. <https://aclanthology.org/2020.eval4nlp-1.9>
51. Kalczyński PJ, Kamburowski J. An empirical analysis of the optimality rate of flow shop heuristics. *Eur J Oper Res.* 2009;198(1):93–101. <https://doi.org/10.1016/j.ejor.2008.08.021>.
52. Matsumoto D, Nakai T. Syntactic theory of mathematical expressions. *Cogn Psychol.* 2023;146:101606. <https://doi.org/10.1016/j.cogpsych.2023.101606>.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.