Volume 36 (2), pp. 73–88
http://orion.journals.ac.za

*ORiON*
ISSN 0259–191X (print)
ISSN 2224–0004 (online)
©2020

# A machine learning approach for automated strip packing algorithm selection

RG Rakotonirainy*

*Received: 17 August 2020; Revised: 2 December 2020; Accepted: 2 December 2020*

**Abstract**

This paper deals with strip packing metaheuristic algorithm selection using data mining techniques. Given a set of solved strip packing problem instances, the relationship between the instance characteristics and algorithm performance is learned and is used to predict the best algorithms to solve a new set of unseen problem instances. A framework capable of modelling this relationship for an automated packing algorithm selection is proposed. The effectiveness of the proposed framework is evaluated in the context of a large set of strip packing problem instances and the state-of-the-art strip packing algorithms. The results suggest a 91% accuracy in correctly identifying the best algorithm for a given instance.

**Key words:** Packing problems, machine learning, metaheuristics.

## 1 Introduction

The two-dimensional strip packing problem (2D-SPP) consists of packing a set of rectangu- lar items into a single object of fixed width in a non-overlapping manner, with the objective of minimising its height. This problem has a wide range of applications, and is typically encountered in the wood, glass and paper industries [27]. For many decades, researchers have been developing evermore sophisticated algorithms for solving this problem. Various experimental studies have been reported in the literature demonstrating the effectiveness of newly developed algorithms, usually based on publicly available collections of benchmark data instances [39, 53].

It has, however, been documented that the conclusions drawn from such comparative algorithmic studies are often not insightful, limited by the scale of the studies which typically restrict either the type or quantity of benchmark problem instances used, or consider only a small number of algorithms [4, 39]. Moreover, a description of the conditions under which an algorithm can be expected to succeed or fail is rarely included in the study [39]. No adequate method for selecting the most appropriate algorithm to solve a particular instance is also found in the strip packing literature.

Besides, as raised by Smith-Miles & Lopes [47], the true value of a comparative algorith- mic study lies in its ability to answer the following two questions: "Which algorithm in a (broad) portfolio

*Department of Statistical Sciences, University of Cape Town, Rondebosch 7701, South Africa, email: rosephine.rakotonirainy@uct.ac.za

http://dx.doi.org/10.5784/36-2-686

is likely to be best for a relevant set of problem instances", and "For which types of problem instances can we expect a given algorithm in a portfolio to perform well, and why"? Answers to the latter question lead to understanding of the conditions under which a particular algorithm can be expected to succeed or fail with respect to the features of the benchmark instances, and help in developing improved algorithm design. Answers to the first question, on the other hand, hold the key for uncovering relationships between characteristics of problem instances and algorithm performance, and have implications for effective algorithm selection model capable of predicting the algorithm from a given portfolio that is likely to be best for a given (unseen) instance.

These questions are certainly relevant in the field of packing problems and need to be addressed for a more comprehensive tool to be developed. Useful are studies where diverse packing algorithms are compared across enough instances, making statistical conclusions valid, with the types of instances matched to the interests of the study. Experimental studies are also suggested to be conducted to uncover relationships between features of instances and algorithm performance. The outcome can be an automated packing algorithm selection model.

This paper aims to provide a starting point for addressing the aforementioned research gap in the 2D-SPP literature. A methodology for characterising packing algorithm performance based on critical features of packing problem instances is introduced. The framework is utilised to predict algorithm performance on previously unseen instances. More precisely, the proposed methodology consists of two phases: *training* and *prediction* phases. The training phase constitutes the kernel of the selection process. In this phase, starting from a set of training packing problem instances solved with a representative sample of packing algorithms, machine learning techniques are applied, in particular clustering and classification, to learn the relationship among the algorithm performance and problem instances characteristics. In the prediction phase, the relationship learned during the training phase is applied to select the best performing algorithm for a new given instance.

The effectiveness of the methodology is evaluated in the context of a large set of strip packing problem instances and the state-of-the-art strip packing algorithms. Large scale computational studies involving the assessment of the relative performance of a variety of strip packing algorithms across a collection of diverse classes of benchmark instances are investigated. This case study demonstrates how the characteristics of the test instances can be used to predict algorithm performance on previously unseen instances with high accuracy.

The remainder of this paper is as follows. Section 2 is devoted to a brief literature review. The main components of the proposed framework are presented in §3. The detailed steps of the methodology when applied to the 2D-SPP are then described in §4. Discussion of the results obtained follows in §5, and a conclusion along with future research directions are provided in §6.

## 2 Literature review

In 1976, Rice [41] proposed a framework for the algorithm selection problem, which seeks to predict the best performing algorithm from a given portfolio on a collection of problem instances of various complexities. Rice applied this approach to predict the performance of partial differential equation solvers. The framework was since then readily generalised to other domains such as in artifical intelligence, machine learning, and operations research areas. The use of supervised learning or regression models have been the focus in these research fields to predict the performance ranking of a set of algorithms with respect to a set of features of the problem instances under investigation. Interested readers are referred to the survey paper by Smith-Miles [49] for a review.

Limited amount of work has, however, focused on packing algorithm prediction or selection. The use of data mining approaches in cutting and packing problems are mostly related to the process of converting problem information into measurable factors in order to reflect the main problem characteristics and compare algorithm performance with different types of problem instances [32]. In a rare attempt at predicting bin packing algorithm performance predictors, Perez *et*

*al.* [38] proposed a methodology that model the relation- ship between algorithm performance and characteristics of bin packing problem instances using machine learning techniques. In [47], Smith-Miles & Lopes proposed a methodology for adequately characterising the features of a problem instance and showed how such features can be defined and measured for various optimisation problems including the bin packing problems. They suggested that the methodology could be applied to the task of algorithm selection.

A methodology capable of identifying the strengths and weaknesses of algorithms as well as their relative power with respect to instance space was proposed by Smith-Miles *et al.* [46]. Based on a set of problem instances with various properties, they applied data mining methods to measure algorithm footprint — the boundary in instance space where an algorithm can be expected to perform well, and relate this boundary to properties of instances to infer the relative performance of algorithms across all instances. In the same vein, Smith-Miles *et al.* [48] explored the ideas of footprints of algorithms in the context of graph coloring and demonstrated the use of data mining to reveal the performance of algorithms, including their strengths and weaknesses, with respect to the search space.

In [44], Santoyo *et al.* conducted a cluster analysis to characterise the difficulty of bench- mark instances for the bin packing problem. They applied a linear correlation analysis to reduce a total of 27 features to five metrics to describe the problem instances and to compare the performance of six heuristic solutions. For the 0-1 knapsack problem, Hall & Posner [23] developed a methodology based on a set of computed problem characteristics (problem size, the characteristics of rectangle value and size, the relationship between rectangle value and rectangle size, knapsack capacity and characteristics of the linear relaxation solution) to predict the best solution procedure from existing ones (branch- and search or dynamic programming algorithm).

# 3 Proposed framework

The methodology proposed in this paper consists of two phases as shown in Figure 1. Given a set of packing benchmark instances, solved using a selection of packing algorithms, the relationship between the algorithm performance and the problem characteristics is learned during the *training phase*. The relationship learned in the previous phase is then used to predict the best algorithm for a new instance during the *prediction phase*. Details of these phases are provided in this section.

## 3.1 Training phase

The training phase involves six steps as depicted by the dashed arrows in Figure 1. In step 1, *instance generation*, suitable packing instances are selected. This is achieved by collecting benchmark problem instances documented in the literature, and also by generating new instances from existing benchmark generators. The choice of instances, and their diversity, play an important role in learning the boundaries of algorithm performance, and also in determining the limits and behaviour of the problem. It is vital to take steps to ensure that the generated instances are as broad as possible and well separated. The key is to include a variety of characteristics of the problem and to avoid biased information. For packing problems, a large collection of instances exist (*e.g.*, in the repository of ESICUP [19]).

Step 2, *feature selection*, consists of selecting the most appropriate features in respect of which to cluster the benchmark data and to measure the influence of problem characteristics on the algorithm performance. Features must be chosen so that the varying complexities of the problem instances are exposed, any known structural properties of the problems are captured, and any known advantages and limitations of the different algorithms are related to the features. Generally, feature selection consists of two-steps: First, the determination of all metrics which likely measure

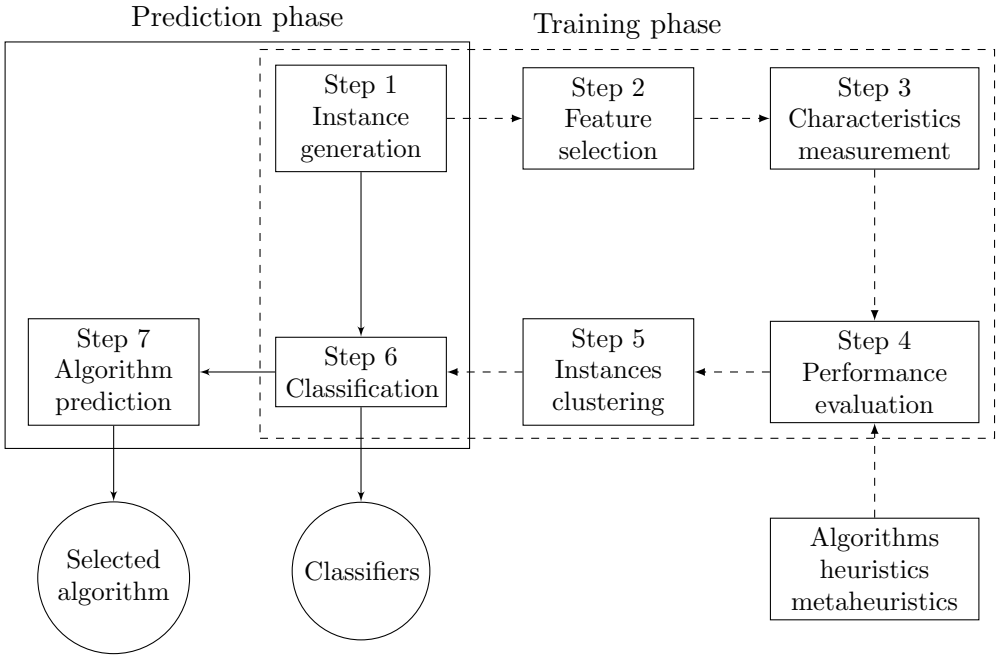


**Figure 1:** *A framework for automated packing algorithm selection.*

the goodness of the instance space and then utilisation of a search strategy to find the subset that maximise the goodness metrics. A wide range of feature selection methods have been proposed in the literature, including supervised feature selection approaches and principal component analysis (PCA) for dimensionality reduction (see [22] for a comprehensive review). Any of these methods could be used to select appropriate features in the context of packing problems.

In the next step, *characteristics measurement*, the characteristic values of each instance generated in step 1 are calculated based on the selected features of step 2. In step 4, *performance evaluation*, feasible solutions of the problem instances are calculated by means of packing solution techniques. Various approaches have been proposed in the literature for solving cutting and packing problems. These approaches may be classified into the classes of exact methods, heuristic approaches, and metaheuristic techniques [39]. Exact methods are typically based on a mathematical programming modelling approach and find a best packing solution, but are slow and may hence only be used to solve small problem instances. Heuristic and metaheuritsic techniques, on the other hand, are approximate solution approaches that attempt to provide near-optimal solutions in minimal time. They are more practical and provide solutions to large problem instances within reasonable time frames.

In this paper, the problem instances are solved using a representative sample of metaheuris- tic algorithms from the literature. The effectiveness of these algorithms are evaluated by means of a standard performance measure utilised in the packing field. There exists a variety of performance evaluation methods in the literature but the most commonly used computational measure for evaluating the performance of a packing algorithm is the relative difference between the packing height returned by the algorithm and the height of an optimal solution of the problem [27]. Such measure is often expressed in terms of a percentage gap or ratio. Alternative performance measure is the packing time efficiency or the time required by the algorithm to find a packing solution for the problem [39]. Computation time may be measured by time tracking during the execution of the algorithm.

The step *instances clustering* entails a cluster analysis whereby the benchmark instances of step 1 are grouped into different classes of test problems based on their underlying features calculated from step 3. Each group comprises instances with similar characteristics, and for which an algorithm had better performance than the others according to the performance evaluation of step 4. Typical clustering analysis involves clustering algorithm design and clustering output assessment. Clustering algorithm design encompasses the selection of a proximity or distance measure, and the choice of an appropriate clustering algorithm for subsequent use. An abundance of clustering algorithms has been proposed in the literature for solving different types of clustering problems in a variety of different fields [8, 55]. There is, however, no clustering algorithm that is generally applicable to all types of clustering problems. It is, therefore, important to investigate the problem at hand properly in order to select an appropriate clustering method.

Clustering output assessment refers to the process of evaluating the clustering results derived from the selected algorithms for validation purposes. Usually, different clustering techniques result in different clusters, and even for the same algorithm, different input parameters typically lead to different cluster results [30]. Effective evaluation or testing criteria are, therefore, required for the assessment of the performance of the algorithms considered.

In the last step, *classification*, the identified grouping in step 5 is learned into formal classifiers, which are predictors that model the relationship between problem characteristics and algorithm performance. Standard machine learning methodologies that use a subset of the instances of step 1 (the training set) to learn the relationship between the instance features and the label assigned to each algorithm (thereof the corresponding cluster) of each instance can be employed during this task. Machine learning classification methods such as decision tree classifiers, Naive Bayes classifiers or support vector machines, can hence be used to perform the task of this step.

### 3.2 Prediction phase

The relationship learned during the training phase of §3.1 is used to predict the best algorithm to solve a new given instance during this prediction phase. The steps of this phase are depicted by the solid arrows in Figure 1. For a new problem instance, which can be generated using step 1 of the training phase, its critical characteristic values are calculated using the characteristic measurement step of §3.1. Based on these characteristics, the learned classifiers from step 6 are employed to determine the cluster into which the instance belongs to in step 7, the *algorithm prediction*. The algorithm associated to this cluster is the expected best algorithm for the instance.

A case study on the 2D-SPP is considered in the following section to illustrate how this framework can be applied to achieve effective packing algorithm selection.

## 4 Case study: The 2D-SPP

The methodology described in the previous section is demonstrated here when applied to the 2D-SPP. The process of instance generation is presented in §4.1, followed by the task of feature selection and characteristics measurement in §4.2 and §4.3, respectively. Performance evaluation using seven state-of-the-art strip packing algorithms is described in §4.4. This is then followed by the clustering analysis step in §4.5, and the classification process in §4.6. The task of algorithm prediction is finally presented in §4.7.

### 4.1 Instance generation

The use of data mining techniques to predict the packing algorithm requires large problem instance data sets, mainly to accurately represent the limits and behaviour of the problem and the variability

that the instances encompass, and also to consider the influence of aspects and characteristics that affect the algorithm performance.

A total of 1 718 benchmark 2D-SPP instances were identified in the literature which are grouped in two classes. The first class consists of zero-waste problem instances for which the respective optimal solutions are known and do not contain any wasted regions (regions of the strip not occupied by items). This class of benchmark instances comprises nine data sets. The second class consists of non-zero-waste instances for which optimal solutions are not known in some cases. Those with known optimal solutions involve some wasted regions. This second class of problem instances comprises eleven data sets. The main characteristics of these data sets organised by name, number of problem instances, minimum and maximum number of items, organisation and source, are described in Table 1.

## 4.2 Feature selection

The greatest challenge is the derivation of suitable metrics as features to characterise the data sets. Relevant features of the problem parameters need to be identified, and expressions to measure the values of identified critical characteristics must be derived. A methodology based on linear correlations and PCA has been employed by Júnior *et al.* [32] to identify the most significant characteristics for the 2D-SPP benchmark instances. They considered 56 descriptive variables, based on parameters found in the most used packing problem generators, and conducted an exploratory analysis to determine the most relevant characteristics with respect to a set of frequently used benchmark data sets. Their analysis suggested that the problem can be reduced to 19 characteristics, retaining most of the total variance.

Recently, Rakotonirainy [39] has investigated a cluster analysis in an attempt to group 2D-SPP benchmark instances into different categories based on four descriptive variables. These variables were selected based on the parameters and characteristics produced by the most popular problem generators. The computational results reported in [39] demonstrated that the chosen four features are practically sufficient for discovering the properties of 2D-SPP problem instances.

The same four descriptive variables were considered in this work to characterise the data sets of § 4.1. These four features are the *maximum aspect ratio* of all items of an instance, the *maximum area ratio* of all pairs of items of an instance, the *heterogeneity ratio*, and the *width ratio*.

- The *aspect ratio* of an item $i$ is defined as

$$\rho(i) = \frac{d_{max}(i)}{d_{min}(i)},$$

  where $d_{min}(i)$ and $d_{max}(i)$ denote its length along the smaller side dimension and its length along the larger side dimension, respectively. The *maximum aspect ratio* of all items of an instance is determined by $\rho_{max} = \max\{\rho(1), \ldots, \rho(n)\}$. The parameter $n$ represents the total number of items involved in the given instance.
- The *area ratio* of a pair of items $i, j$ is given by

$$\gamma(i,j) = \frac{a(i)}{a(j)},$$

  where $a(i)$ denotes the area of item $i$. The *maximum area ratio* of all pairs of items of an instance is defined as $\gamma_{max} = \max\{\gamma(i,j) \mid i,j = 1,\ldots,n;\ i \neq j\}$.
- The *heterogeneity ratio* is given by $\nu = n_t/n$, where $n_t$ denotes the number of distinct types of items in an instance. Two items are of the same type if they have identical smaller and larger side dimensions.

| Set | Authors & Year | Description | Comment |
|---|---|---|---|
| **Zero-waste problem instances** | | | |
| J | Jakobs (1996) [31] | 2 problems instances | Both optimal solutions known |
| SCP | Hifi (1998) [24] | 25 problem instances | All optimal solutions known |
| babu | Babu & Babu (1999) [2] | 1 problem instance | Optimal solution know |
| C | Hopper & Turton (2001) [27] | 21 problem instances | All optimal solutions known and 14 guillotineable |
| NT(n), NT(t) | Hopper & Turton (2002) [26] | 70 problem instances | All optimal solutions known and 35 guillotineable |
| N | Burke *et al.* (2004) [11] | 12 problem instances | All optimal solutions known and guillotineable |
| CX | Pinto & Oliveira (2005) [21] | 7 problem instances | All optimal solutions known |
| IY | Imahori & Yagiura (2010) [28] | 170 problem instances | All optimal solutions known |
| **Non-zero-waste problem instances** | | | |
| cgcut | Christofides & Whitlock (1977) [15] | 3 problem instances | 1 optimal solution known |
| beng | Bengtsson (1982) [6] | 10 problem instances | 6 optimal solutions known |
| gcut, ngcut | Beasley (1895) [5] | 25 problem instances | 13 guillotineable (2 of which with optimal solutions known) and 12 non-guillotineable (11 of which with optimal solutions known) |
| bwmv | Berkey & Wang (1987) [7] | 300 problem instances | All optimal solutions unknown |
| bwmv | Martello & Vigo (1998) [35] | 200 problem instances | All optimal solutions unknown |
| DP | Dagli & Poshyanonda (1997) [16] | 1 problem instance | Optimal solution unknown |
| BK | Burke & Kendall (1999) [10] | 1 problem instance | Optimal solution known and guillotineable |
| SCPL | Hifi (1998) [25] | 9 problem instances | Optimal solutions unknown |
| Nice, Path | Wang & Valenzuela (2001) [50] | 480 problem instances | Floating-point data sets and all optimal solutions known |
| AH | Bortfeldt & Gehring (2006) [9] | 360 problem instances | Optimal solutions unknown |
| Zdf | Leung *et al.* (2011) [33] | 16 problem instances | 5 optimal solutions known |

**Table 1:** *The 1 718 benchmark problem instances employed in this study.*

- The *width ratio* is determined by $\delta = W/d_{mean}$, where $W$ denotes the width of the strip, and $d_{mean}$ represents the mean value of all (smaller and larger) items dimensions.

The aspect ratio provides information on the shapes of the items in an instance: Relatively small values of the maximum aspect ratio feature indicate that the respective instance is heavily populated by approximately square-shaped items. The variety in the sizes of the items in an instance, on the other hand, may be deducted from the maximum area ratio: Large value of this feature implies that the corresponding instance is dominated by items of widely varying sizes. Furthermore, the miscellany of items in an instance may be gauged from the heterogeneity ratio: An instance with a value of the heterogeneity ratio feature close to unity indicates that the dimensions of the items involved in that instance are all different (*i.e.*, heterogeneous). Finally, the width ratio characterises the mean item width relative to that of the strip: That is, an instance with a relatively large value of the width ratio feature contains a large number of wide items.

## 4.3 Characteristics measurement

The critical characteristic values of each instance of the 1 718 data sets obtained in §4.1 were calculated using the selected features of §4.2. A two-dimensional PCA scatter plot of the data sets with respect to the four features is given in Figure 2. The axes in this figure represent a projection that best spreads the data.

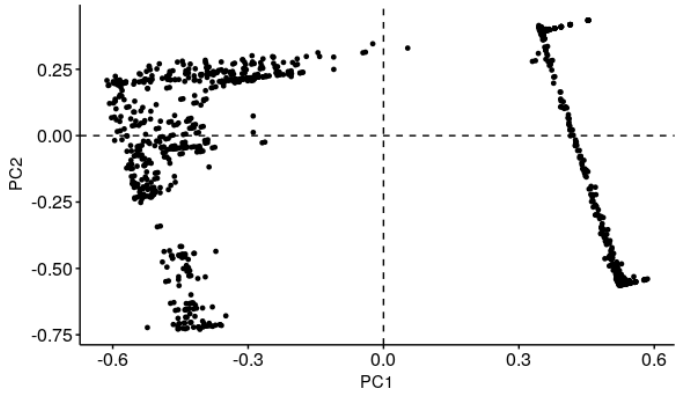


**Figure 2:** *A PCA of the data — the axes, labelled PC1 and PC2, indicate respectively the first and second principal components and represent a projection that best spreads the data.*

## 4.4 Performance evaluation

The problem instances were solved using seven state-of-the-art strip packing metaheuristics from the literature. The first algorithm is the two-stage intelligent search algorithm (ISA) of Leung *et al.* [34], which combines a local search algorithm with the method of simulated annealing (SA) in an attempt to find feasible packing solutions. The second algorithm is a hybrid technique, where a genetic algorithm is executed in conjunction with the constructive heuristic of Leung *et al.* [34]. The simple randomised algorithm (SRA) of Yang *et al.* [56] and the efficient intelligent search algorithm (IA) of Wei *et al.* [54] are also considered, which are both improvements of the ISA algorithm.

The last three algoritms are among the recently proposed strip packing techniques: The improved skyline-based heuristic algorithm (ISH) of Wei *et al.* [53], which may be considered as an improved version of the constructive heuristic embedded in the IA algorithm, the CIBA algorithm of Chen & Chen [14], and the modified intelligent search algorithm (IAm) of Rakotonirainy & Van Vuuren

[40]. These algorithms have been selected for consideration as they were among the most recently proposed algorithms, and were reported as best algorithms, for solving instances of the 2D-SPP [40].

The relative effectiveness of these algorithms were evaluated according to a performance measure — the ratio between the packing height returned by an algorithm and the height of an optimal solution. A set of best algorithms, which is defined in such a way that the performance ratios of any pair of algorithms in the set are equal to almost 1%, was extracted for each sample instance. An example of 5 data instances, each associated with the corresponding values of the four factors and the best performing algorithms, is shown in Table 2.

The seven packing algorithms were coded in Python using Spyder Version 2.7.6. They were run in the same environment on an Intel Core i7-4790 CPU running at 3.60 GHz with 8 GB RAM in the Windows 10 operating system. Further details on the algorithms implementations, together with the appropriate parameter fine-tuning, can be found in [39, 40].

| Problem Instances | Aspect Ratio | Area Ratio | Width Ratio | Het Ratio | Best Algorithms |
|---|---|---|---|---|---|
| 1985BeasleyJORS11.csv | 2.684211 | 4.443422 | 2.130001 | 1.0 | Hybrid GA |
| 2000HopperT3b.csv | 6.913043 | 53.263158 | 5.017301 | 1.0 | SRA, Hybrid GA |
| 2000HopperN4a.csv | 6.77777 | 131.6875 | 7.12209 | 0.04 | CIBA, ISA |
| 2001WangNice10.25.csv | 3.79454 | 6.662807 | 4.81687 | 1.0 | ISH, CIBA, IA |
| 2001WangPath2.50.csv | 93.389605 | 68.843953 | 5.78435 | 1.0 | IAm |

**Table 2:** *Example of instances with their respective characteristics and the corresponding best algorithms.*

## 4.5 Instances clustering

The cluster analysis performed in this step consists of grouping a training data, a sample of 1 100 instances randomly extracted from the entire problem instances of §4.1, into different classes of test problems based on their characteristics as calculated in §4.3. Rakotonirainy [39] has conducted such analysis in an attempt to classify the overall 1 178 2D-SPP problem instances. In the latter study, the clustering process consisted of three steps: The first step involved preparation of the data sets based on their selected features. This entailed feature scaling by applying normalisation. The next step consisted of estimating the optimal number of clusters in which to partition the benchmark data by means of a variety of indices. The final step involved evaluation of the performance of different clustering algorithms with respect to a set of validation measures so as to choose the best performing one.

The same clustering process was adopted in this study in order to generate a sound data clustering output result. The `R` package, *NbClust*, of Charrad *et al.* [13] was employed to estimate an appropriate number of clusters that best partitions the normalised benchmark instances. With a single function call, it computes thirty indices and determines the relevant number of clusters accordingly. A histogram of the distribution of the output is shown in Figure 3. The majority of the indices suggested four as the best number of clusters, and this was utilised in the performance evaluation of clustering techniques in the following step.

Four popular clustering techniques, the $k$-means algorithm [30, 55], the agglomerative method [29, 30, 36, 55], the DBSCAN algorithm [20], and the spectral clustering algorithm [1, 3, 52, 57], were evaluated in this study. The solution quality achieved by these algorithms was assessed by means of the following four validation measures: The Silhouette coefficient [43], the *Calinski-Harabasz* (CHa) index [12], the Dunn index [18], and the *Davies-Bouldin* (DB) index [17]. A larger Silhouette

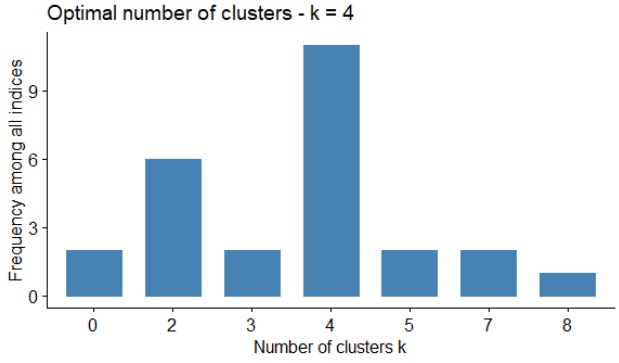


**Figure 3:** *Histogram of the distribution of the results obtained when estimating the number of clusters in which to partition the training data instances of §4.1 by means of the NbClust function in* `R` *of Charrad* et al. *[13]. According to the majority rule, the preferred number of clusters is clearly 4.*

coefficient value corresponds to a well-defined cluster structure. Similarly, CHa and Dunn indices are large when clusters are dense and well separated. In contrast, a small DB index corresponds to better defined clusters.

The corresponding evaluation results are reported in Table 3. From this table, it is clear that the $k$-means algorithm yields larger CHa and Silhouette values than the other methods. The corresponding Dunn index is comparatively large, and the value of the DB index is also comparatively small. This indicates that the $k$-means clustering algorithm produces a better clustering output than the other techniques. The $k$-means algorithm was therefore selected as the clustering method adopted in this study to cluster the instances into groups, whereby the similarity among members of each group was determined through the characteristics indicators of the instances and the number assigned to the best perform- ing algorithms obtained in the performance evaluation of §4.4.

| Algorithm | CHa index | Dunn index | DB index | Silhouette |
|---|---|---|---|---|
| $k$-means | 530.0126 | 0.3967 | 1.2849 | 0.6271 |
| Hierarchical | 445.231 | 0.2591 | 1.2641 | 0.4376 |
| Spectral | 445.6919 | 0.418 | 1.478 | 0.413 |
| DBSCAN | 238.1507 | 0.1048 | 1.1885 | 0.3699 |

**Table 3:** *Performance evaluation of the different clustering algorithms in respect of the training data, based on the CHa, Dunn, DB, and Silhouette indices.*

The $k$-means algorithm was performed repetitively until it generates the best clustering results. Each resulting cluster was associated a label with respect to the aggregate characteristics values of its instances members and a best performing algorithm for it. A summary of best performing algorithm for each cluster is given in Table 4. It is noted that the ISA, SRA, and IA algorithms were outperformed by the three algorithms reported in this table (or do not differ significantly from their performances for certain instances), and that CIBA and ISH algorithms performed relatively similarly with respect to the fourth benchmark cluster, so the CIBA algorithm was selected as the best algorithm for that cluster. This dominance result applies only to the benchmark instances explored in this work.

| Cluster | Best Algorithm |
|---|---|
| Cluster 1 | IAm |
| Cluster 2 | Hybrid GA |
| Cluster 3 | IAm |
| Cluster 4 | CIBA |

**Table 4:** *Summary of the clustering output obtained when performing step 5 of the proposed framework. The first cluster contains 170 instances, the second cluster 351 instances, and the third and fourth clusters contain 391 and 198 instances, respectively.*

## 4.6 Classification

The identified grouping in §4.5 is learned into formal classifiers, which are predictors that model the relationship between problem instances characteristics and algorithm perfor- mance. Decision trees can be very powerful tools for modelling this relationship and for elucidating rules that can be used to predict the best performing algorithm for new instances. The decision tree algorithm [42] was, therefore, employed as a machine learning technique to generate classification rules for this purpose. The decision tree algorithm builds a decision tree from the training data instances, which is then converted to a set of classification rules using the cluster labels as target variables. The rules are ordered by accuracy and are applied in sequence to classify instances in the corresponding group.

To obtain the classification rules, the four indicators of §4.2 were used as independent variables and the best algorithms associated to each cluster as class variables. The percentage of new correctly classified observations is an indicator of the effectiveness of the classification rules. If these rules are effective on the training datasets, it is expected that they will perform well on new observations with unknown group. The classification analysis was conducted using the Decision Tree Classification package available in Python Scikit-learn and the classifier was trained on the 1 1000 training data sets.

In order to optimise the performance of the decision tree classifier, two decision tree methods with two different attribute selection criteria were compared. The first decision tree method employs the "information gain"[1] as a selection criterion, while the second decision tree approach uses the "gini index"[2] criterion. The accuracy of these two methods when applied to the training datasets is shown in Table 5. A pre-pruning was also conducted. This was achieved by controlling the values of parameters and variables defining the classifier. The parameter "maximum depth of the tree", which defines the height or the number of nodes in the tree, was varied in this experimental work. The corresponding accuracy results for the training data sets are also shown in Table 5.

| Decision tree methods with | Gini index | Information gain |
|---|---|---|
| Maximum depth of the tree = 5 | 70% | 69.6% |
| Maximum depth of the tree = 8 | 80% | 75.4% |
| Maximum depth of the tree = 10 | 91% | 90% |

**Table 5:** *Classification accuracy of the various decision tree methods when applied to the training data sets.*

The accuracies ranged from 91%, for the decision tree classifier using gini index as a selection

[1]Information gain is a statistical property that measures how well a given attribute separates the training data according to their target variables [37].

[2]Gini index is a cost function used to evaluate splits in the datasets. It is calculated by subtracting the squared probabilities of each class from one [37].

criterion and a value equal to 10 for the 'maximum depth of the tree' parameter, down to 69.6% accuracy for the decision tree classifier using information gain method as a selection criterion and a value equal to 5 for the 'maximum depth of the tree' parameter. The method which exhibits the highest accuracy was employed to predict algorithm performance in this work.

### 4.7 Algorithm prediction

The learned classifier of §4.6 was applied to predict the best performing algorithm for each one of the remaining 618 test instances of §4.1. The decision tree method using gini index as a selection criterion and a value equal to 10 for the 'maximum depth of the tree' parameter was employed for this purpose, as it provides the highest accuracy results (see Table 5). The output results are given in Table 6. Column 2 of this table contains the real best algorithms for each instance while column 3 contains the predicted algorithms. If the predicted algorithm is one of the real best algorithms, the match is counted (equal to 1), as shown in column 4 of Table 6. A 'Match' value equal to 0, for a particular instance, indicates that the classifier fails to predict the correct algorithm for that instance. As shown in this table, the classifier predicted the correct best performing algorithm with accuracy of 91%.

| **Instance** | **Real best algorithms** | **Predicted best algorithm** | **Match** |
|---|---|---|---|
| 1985BeasleyJORS8.csv | Hybrid GA | IAm | 0 |
| 1987BerkeyWangClass6.41.csv | IAm | IAm | 1 |
| 1987BerkeyWangClass6.44.csv | IAm | Hybrid GA | 0 |
| 2000BerkeyKendallWhitwellN4.csv | SRA, Hybrid GA | Hybrid GA | 1 |
| ⋮ | ⋮ | ⋮ | ⋮ |
| 1999Hifi2.csv | IAm | IAm | 1 |
| 1998MartelloVigoClass4.47.csv | Hybrid GA, SRA | Hybrid GA | 1 |
| 2001WangValenzuelaPath11.100.csv | ISH, CIBA, IA | CIBA | 1 |
| | | Accuracy | 91% |

**Table 6:** *Classification results for 618 problem instances. The column 'Match' indicates if the predicted algorithm is correct (its value equal to 1) or not (its value equal to 0).*

## 5 Discussion

The result presented in the previous section suggests that the methodology adopted in this paper achieves a 91% accuracy in correctly predicting the best performing algorithm for a new strip packing instance. This could be considered as a highly accurate prediction of algorithm performance on the basis of packing instances grouping according to their underlying features. The systematic selection of the features that characterise the problem instances was crucial for obtaining this result accuracy. Four different characteristics, extracted from various parameters and factors used in popular packing generators, were considered to group the benchmark instances into four categories in this work. These chosen features have proven to be sufficient for discovering the properties of the benchmark instances.

It should be noted that the decision to include a clustering process before the classification step has meant that the meaningful characteristics that best describe the data instances are identified and that the relationship between the identified features and algorithm performance is accurately explored. Each cluster was assigned one best performing algo- rithm and that it is learned into classifiers for predicting the best algorithm for a new given instance. Obviously the result depends

on the clustered instances output. One could expect a different result by directly classifying the instances based on their characteristics and best performing algorithms. This research direction is the focus in an ongoing project of the author.

Besides, the majority of the instances incorrectly classified are tied results. In fact, it was assumed a definition of a best algorithm as an algorithm with performance ratio relatively 1% higher than others and that ties are solved by randomly choosing one of the algorithms. In reality, however, the assigned algorithm may be best for some proportion (possibly, not for all) of problem instances in a cluster and that it may perform well for a specific instance of other clusters. An example is the classification result of the 1987BerkeyWangClass6.44.csv instance shown in Table 6, whereby the real best algorithm of the instance is the IAm algorithm, while the classification model predicts the Hybrid GA algorithm as its best algorithm. A more robust condition is probably needed to avoid such biased prediction and to improve on the performance accuracy.

While this paper has focused on application of the methodology to strip packing problem, the proposed framework can be applied to any type of packing problem and could also be easily adopted for automated algorithm selection in other combinatorial optimisation problems. The adaptation of the methodology to other problems presents no difficulties aside from the challenge of selecting and deriving relevant features to characterise the problem instances, which might require careful check.

# 6 Conclusion

A methodology capable of modelling the relationship between packing instance characteris- tics and algorithm performance for an automated packing algorithm selection was proposed in this paper. The model was applied to predict best performing algorithms for unseen packing instances with high accuracy.

It has been shown, through a case study of strip packing problems, that data mining techniques like clustering analysis and decision trees can be employed to explore the high-dimensional feature space of the problem instances, to cluster the test problems into different classes of instances according to their features, and to learn the clusters into classifiers for an effective automated packing algorithm selection. The result obtained suggested a 91% accuracy in predicting the best performing algorithms on a new set of packing problem instances based on their characteristics.

While the chosen features have proven to be sufficient for discovering the properties of instances, it is suggested that further features be added to complement the characterisation of the problem benchmark instances. The inclusion of other features is expected to result in a more robust classification of the various problem instances. Possible future work might also involve incorporating additional test problems in the analyses carried out in this paper. The author is aware of other existing problem generators, such as the 2DCPackGen of Silva *et al.* [45] which may be employed to generate various problem instances. Incorporating such additional test problems may facilitate identification of other important features prevailing in the data, and also render the methodology more realistic and robust in the sense of being able to accommodate a large variety of problem instances. This research direction together with an extension of the framework in order to improve on the prediction accuracy is the focus in an ongoing project of the author.

Finally, it remains for future investigation the adjustment of the framework, by omitting the cluster analysis step and directly applying classification techniques to classify the instances based on their characteristics and best performing algorithms, and assess the effect of this adaptation on the accuracy of the prediction result.

# References


[1] Aggarwal CC & Reddy CK (Eds), 2013, *Data clustering: algorithms and applications*, CRC press, Boca Raton (FL).

[2] Babu AR & Babu NR, 1999, *Effective nesting of rectangular parts in multiple rectangular sheets using genetic and heuristic algorithms*, International Journal of Production Research, **37(7)**, pp. 1625–1643.

[3] Bach FR & Jordan MI, 2004, *Learning spectral clustering*, Advances in Neural information processing systems, **16**, pp. 305–312.

[4] Barr RS, Golden BL, Kelly JP, Resende Mauricio GC & Stewart WR, 1995, *Designing and reporting on computational experiments with heuristic methods*, Journal of Heuristics, **1(1)**, pp. 9–32.

[5] Beasley J, 1985, *Algorithms for unconstrained two-dimensional guillotine cutting*, Journal of the Operational Research Society, **36(4)**, pp. 297–306.

[6] Bengtsson B, 1982, *Packing rectangular pieces — a heuristic approach*, The Computer Journal, **25(3)**, pp. 353–357.

[7] Berkey J & Wang P, 1987, *Two-dimensional finite bin-packing algorithms*, Journal of the Operational Research Society, **38(5)**, pp. 423–429.

[8] Berkhin P, 2006, *A survey of clustering data mining techniques*, Grouping multidimensional data, pp. 25–71.

[9] Bortfeldt A & Gehring H, 2006, *New Large benchmark instances for the two-dimensional strip packing problem with rectangular pieces*, Proceedings of the $39^{th}$ Annual Hawaii International Conference on System Sciences, Kauai (HI), p. 30b.

[10] Burke E & Kendall G, 1999, *Applying simulated annealing and the no fit polygon to the nesting problem*, pp, 453–464 in Foo N (Eds), Advanced Topics in Artificial Intelligence, Springer, Berlin (Heidelberg).

[11] Burke E, Kendall G & Whitwell G, 2004, *A new placement heuristic for the orthogonal stock-cutting problem*, Operations Research, **52(4)**, pp. 655–671.

[12] Caliński T & Harabasz J, 1974, *A dendrite method for cluster analysis*, Communications in Statistics-theory and Methods, **3(1)**, pp. 1–27.

[13] Charrad M, Ghazzali N, Boiteau V, Niknafs A & Charrad MM, 2014, *Package NbClust*, Journal of Statistical Software, **61**, pp. 1–36.

[14] Chen Z & Chen J, 2018, *An Effective Corner Increment-Based Algorithm for the Two-Dimensional Strip Packing Problem*, IEEE Access, **6**, pp. 72906–72924.

[15] Christofides N & Whitlock C, 1977, *An algorithm for two-dimensional cutting problems*, Operations Research, **25(1)**, pp. 30–44.

[16] Dagli C & Poshyanonda P, 1997, *New approaches to nesting rectangular patterns*, Operations Research, **8(3)**, pp. 177–190.

[17] Davies D & Bouldin D, 1979, *A cluster separation measure*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **1(2)**, pp. 224–227.

[18] Dunn JC, 1974, *Well-separated clusters and optimal fuzzy partitions*, Journal of Cybernetics, **4(1)**, pp. 95–104.

[19] ESICUP, 2015, *Datasets 2D-Rectangular*, [Online], [Cited July $7^{\text{th}}$, 2020], Available from http://www.euro-online.org/websites/esicup/data-sets/.

[20] Ester M, Kriegel H, Sande J & Xu X, 1996, *A density-based algorithm for discovering clusters in large spatial databases with noise*, Proceedings of the $2^{nd}$ International Conference on Knowledge Discovery and Data Mining, pp. 226–231.

[21] Ferreira EP & Oliveira JF, 2005, *Algorithm based on graphs for the non-guillotinable two-dimensional packing problem*, Proceedings of the $2^{nd}$ ESICUP Meeting, Southampton, no page.

[22] Guyon I & Elisseeff A, 1974, *An introduction to variable and feature selection*, Journal of Machine Learning Research, **3**, pp. 1157–1182.

[23] Hall N & Posner ME, 2007, *Performance prediction and preselection for optimization and heuristic solution procedures*, Operations Research, **55(4)**, pp. 703–716.

[24] Hifi M, 1998, *Exact algorithms for the guillotine strip cutting/packing problem*, Computers and Operations Research, **25(11)**, pp. 925–940.

[25] Hifi M, 1999, *The strip cutting and packing problem: Incremental substrip algorithms-based heuristics*, Pesquisa Operacional, **19(2)**, pp. 169–188.

[26] Hopper E & Turton B, 2002, *Problem generators for rectangular packing problems*, Studia Informatica Universalis, **2(1)**, pp. 123–136.

[27] Hopper E & Turton B, 2001, *An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem*, European Journal of Operational Research, **128(1)**, pp. 34–57.

[28] Imahori S & Yagiura M, 2010, *The best-fit heuristic for the rectangular strip packing problem: An efficient implementation and the worst-case approximation ratio*, Computers and Operations Research, **37(2)**, pp. 325–333.

[29] Jain A & Dubes RC, 1988, *Algorithms for clustering data*, Prentice-Hall, River (NJ).

[30] Jain A, Murty MN & Flynn PJ, 1999, *Data clustering: A review*, ACM Computing Surveys, **31(3)**, pp. 264–323.

[31] Jakobs S, 1996, *On genetic algorithms for the packing of polygons*, European Journal of Operational Research, **88(1)**, pp. 165–181.

[32] Júnior AN, Silva E, Gomes AM, Soares C & Oliveira JF, 2019, *Data mining based framework to assess solution quality for the rectangular 2D strip-packing problem*, Expert Systems with Applications, **118**, pp. 365–380.

[33] Leung S & Zhang D, 2011, *A fast layer-based heuristic for non-guillotine strip packing*, Expert Systems with Applications, **38(10)**, pp. 13032–13042.

[34] Leung S, Zhang D & Sim KM, 2011, *A two-stage intelligent search algorithm for the two-dimensional strip packing problem*, European Journal of Operational Research, **215(1)**, pp. 57–69.

[35] Martello S & Vigo D, 1998, *Exact solution of the two-dimensional finite bin packing problem*, Management Science, **44(3)**, pp. 388–399.

[36] Murtagh F, 1983, *A survey of recent advances in hierarchical clustering algorithms*, The Computer Journal, **26(4)**, pp. 354–359.

[37] Nagesh SC, 2019, *Decision Tree Algorithm – Explained*, [Online], [Cited: 7 July 2020], Available: http://towardsdatascience.com/decision-tree-algorithm-explained-83beb6e78ef4.

[38] Perez J, Frausto J, Cruz L & Fraire H, 2004, *A machine learning approach for modeling algorithm performance predictors*, International Conference on Modeling Decisions for Artificial Intelligence, Barcelona, pp. 70–80.

[39] Rakotonirainy RG, 2018, *Metaheuristic solution of the two-dimensional strip packing problem*, PhD Thesis, University of Stellenbosch, Stellenbosch.

[40] Rakotonirainy RG & Van Vuuren JH, 2020, *Improved metaheuristic for the two-dimensional strip packing problem*, Applied Soft Computing, **92**, no page.

[41] Rice J, 1976, *The algorithm selection problem*, Advances in computers, **15**, pp. 65–118.

[42] Rokach L & Maimon OZ, 2008, *Data mining with decision trees: theory and applications*, World Scientific, Singapore.

[43] Perez J, Frausto J, Cruz L & Fraire H, 1987, *Silhouettes: a graphical aid to the interpretation and validation of cluster analysis*, Journal of computational and applied mathematics, **20**, pp. 53–65.

[44] Santoyo AM, Ortega JP, Romero D & Reyes LC, 2015, *Towards a Characterization of Difficult Instances of the Bin Packing Problems*, IEEE Latin America Transactions, **13(7)**, pp. 2454–2462.

[45] Silva E, Oliveira JF & Wäscher G, 2014, *2DCPackGen: a problem generator for two-dimensional rectangular cutting and packing problems*, European Journal of Operational Research, **237(3)**, pp. 846–856.

[46] Smith-Miles K, Baatar D, Wreford B & Lewis R, 2014, *Towards objective measures of algorithm performance across instance space*, Computers and Operations Research, **45**, pp. 12–24.

[47] Smith-Miles K & Lopes L, 2012, *Measuring instance difficulty for combinatorial optimization problems*, Computers and Operations Research, **39(5)**, pp. 875–889.

[48] Smith-Miles K, Wreford B, Lopes L & Insani N, 2013, *Predicting metaheuristic performance on graph coloring problems using data mining*, Talbi EG (Eds), Hybrid metaheuristics, SCI, Heidelberg.

[49] Smith-Miles K, 2009, *Cross-disciplinary perspectives on meta-learning for algorithm selection*, ACM Computing Surveys, **41(1)**, pp. 1–25.

[50] Valenzuela CL & Wang PY, 2001, *Heuristics for large strip packing problems with guillotine patterns: An empirical study*, Proceedings of the $4^{th}$ Metaheuristics International Conference, Porto, pp. 417–421.

[51] Van Vuuren JH & Rakotonirainy RG, 2018, *The 2D rectangular strip packing problem (clustered benchmarks)*, [Online], [Cited July $7^{\text{th}}$, 2020], Available from http://www.vuuren.co.za/main.php.

[52] Von Luxburg U, 2007, *A tutorial on spectral clustering*, Statistics and Computing, **17(4)**, pp. 395–416.

[53] Wei L, Hu Q, Leung S & Zhang N, 2017, *An improved skyline based heuristic for the 2D strip packing problem and its efficient implementation*, Computers and Operations Research, **80)**, pp. 113–127.

[54] Wei L, Qin H, Cheang B & Xu X, 2016, *An efficient intelligent search algorithm for the two-dimensional rectangular strip packing problem*, International Transactions in Operational Research, **23(1–2)**, pp. 65–92.

[55] Xu R & Wunsch D, 2005, *Survey of clustering algorithms*, IEEE Transactions on Neural Networks, **16(3)**, pp. 645–678.

[56] Yang S, Han S & He W, 2013, *A simple randomized algorithm for two-dimensional strip packing*, Computers and Operations Research, **40(1)**, pp. 1–8.

[57] Zelnik-Manor L & Perona P, 2005, *Self-tuning spectral clustering*, Advances in neural information processing systems, **17**, pp. 1601–1608.