Efficient meta-heuristics for multi-and-many objective Bin Packing Problems

André Rocha - andre.da.rocha@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

May 2020

Abstract

The Bin Packing Problem (BPP) is a specific combinatorial problem inside Cutting and Packing Problems (C&P), having a large impact in several industries with practical applications from packing boxes, to scheduling and resource allocation. The main goal of the BPP is to minimize the number of entities where the items will be placed, finding the best way of storing them.

Today, as computing capacity and processing speed increases, all industries are moving towards data collection and optimization. Therefore, more data can be analysed and better results achieved.

The BPP has been studied since the 60s, but, just recently started to appear in papers focusing on its multi-and-many objective formulations, balancing different combinations and trade-offs of minimizing other goals important to areas such as cargo loading or virtual memory allocations, being a good example of the computational advances impact in the different industries.

This paper describes the background of several Combinatorial Problems, contextualizing the BPP inside the Operations Research (OR) field, providing knowledge about other C&P and their usage. Furthermore, it introduces the PlatEMO [17] framework together with its combinatorial multi-and-many objective problems suite, that are used as a parallelism to study and introduce: the 3D-BPP with many-and-multi objectives; the meta-heuristics selected to be studied and to solve the problem; and the metrics used to evaluate its solutions. Lastly, the reader is presented with the comparison of the different meta-heuristics studied during in this paper and their application to the multi-and-many objective Bin Packing Problem (mBPP) developed.

Keywords: Combinatorial Problems; Cutting and Packing; Bin Packing; Meta-heuristics; Evolutionary Algorithms.

1. Introduction

1.1. Contextualization and motivation

Throughout history, human-beings always felt the necessity of improving the performance of their tasks. It was in 1947, after World War II, that the Simplex Algorithm for Linear Programming (LP), one of the biggest tools of OR, was presented by George Dantzig. One of the biggest applications of OR in the industrial processes was related with production and cutting processes. So, since the early stages, there was always a focus on specific problems related to C&P as the Knapsack Problem (KP), named and improved by Tobias Dantzing (father of George Dantzig); followed by the Cutting Stock Problem (CSP), involving cutting processes such as paper, glass or metal sheets with mass production; and the BPP, used for the necessity of packing and loading the products to transport. Later, it was found that the BPP could have other applications involving every problem that wanted to assign/pack a list of small items into one or several large objects, such as scheduling or resource allocation. Nowadays, the BPP is receiving a lot of attention mainly due to its direct application in computer science, more specifically in the field of data handling and data allocation in a virtual memory, turning this process more efficient, reducing the costs. In this thesis, with a well-established taxonomy of the different, several techniques to optimize BPP will be presented. From these techniques, a special emphasis will be given meta-heuristics that will be applied in different dimensions to a developed mBPP PlatEMO [17] - a framework used to study Evolutionary Algorithms (EAs) - in a MatLab environment.

1.2. Objective

The main goal of this work is studying efficient meta-heuristics for mBPP and comment on their benchmark performance. The followings goals were targeted:

1. Describe and contextualize intensively the BPP highlighting its differences and similarities to other C&P;

2. Elaborate a literature review with the state-of-the-art of meta-heuristics, heuristics and exact algorithms used to solve Single BPP;

3. Elaborate a literature review with the state-of-the-art of multi-objective Evolutionary Algorithms (MOEAs);

4. Elaborate a report/comment about the performance results of the different MOEA based on experiments in different Combinatorial Problems, mainly the BPP;

2. Cutting and Packing Problems

2.1. Taxonomy

The most used taxonomy of C&P was developed by Wäscher et al. (2007) that tried to identify all possible problems combinations through a detailed criteria [18].

2.1.1 Parameters, Criteria and Problem type

Before presenting the taxonomy, it is important to define several parameters in its definition. In C&P it is given two sets of elements:

1. a set of large objects (input, supply): $I = \{1, \ldots, m\}$;
2. a set of small items (output, demand): $J = \{1, \ldots, n\}$.

Some or several of these small items form subsets $J' \subset J$ that will be assigned to the large objects according to the following:

1. all items of the subset lie entirely within the large object;
2. the small items do not overlap.

Following the parameters, five main criteria used to distinguish the different C&P problems are: dimensionality; kind of assignment; assortment of large objects; assortment of small items; and shape of the small items.

Promptly, the reader is able to identify and distinguish the different kinds of problem based on hundreds of combinations of those parameters. The most importants are:

1. Knapsack Problem
   This problem is an output maximization, where all the dimensions of large objects are fixed and the small items are strongly heterogeneous. The objects in this problem are called containers or knapsacks. In the Classic 1D-KP, the items value and size are called profit and weight (c_j and w_j). The main goal is to select one or more disjoint subsets of items, so that its weight sum does not overcome the maximum weight capacity of the container (W), and its profit is maximized. This problem has just one container and it is characterized by a binary decision variable that indicates if item j is used ($x_j = 1$) or not ($x_j = 0$).

   LP formulation

   $$\max \sum_{j \in J} c_j x_j \tag{1a}$$

   subject to:

   $$\sum_{j \in J} w_j x_j \leqslant W \tag{1b}$$

   $$x_j \in \{0, 1\} \qquad \forall j \in J \tag{1c}$$

2. Cutting Stock Problem
   This problem is an input minimization, where all the dimensions of large objects are fixed and the small items are weakly heterogeneous. The objects in this problem are called stock rolls and have different patterns or possible combination of cuts. In the Classic 1D-CSP, the items have a certain length and demand (l_j and d_j). The main goal of this kind of problems is to cut standard-sized pieces of stock material, with a length L, into smaller pieces of different specified lengths. The amount of each pattern used to cut the pieces (y_i) should be determined in order to reduce the wasted material. The number of rolls of item j cut in pattern i is denoted as x_{ij} and the feasible patterns follow $\sum_{j \in J} l_j x_{ij} \leqslant L, \forall i \in I$.

   LP formulation:

   $$\min \sum_{i \in I} y_i \tag{2a}$$

   subject to:

   $$\sum_{i \in I} x_{ij} y_i \geqslant d_j \qquad \forall j \in J \tag{2b}$$

   $$y_i \geqslant 0, \text{integer} \qquad \forall i \in I \tag{2c}$$

3. Bin Packing Problem
   This problem is an input minimization, where all the dimensions of large objects are fixed and the small items are strongly heterogeneous. The objects in this problem are often called containers or bins. Variants of the BPP diverge according to any possible constraint on the orientation or placement/positioning of the items, on-line or off-line problems (items can be reordered), fuzzy information, geometric or vector packing. In the Classic 1D-BPP

the size of the items is simply an integer. It is the same as packing items with a certain width in a bin with the same width, as if it was a chimney. Let an item height or weight be described as w_i and the maximum bin height/weight capacity denoted as C. If a bin i is used then $y_i = 1$, otherwise $y_i = 0$. If an item j is packed into a bin j then $x_{ij} = 1$, otherwise $x_{ij} = 0$.

LP formulation:

$$\min \sum_{i \in I} y_i \tag{3a}$$

subject to:

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{3b}$$

$$\sum_{j \in J} w_j x_{ij} \leqslant C y_i \qquad \forall i \in I \tag{3c}$$

$$x_{ij}, y_i \in \{0, 1\} \qquad \forall i \in I, \forall j \in J \tag{3d}$$

Comparing the KP, CSP and BPP LP formulations, it is easily checked that the main difference between:

- 1D-KP and 1D-BPP is that the first one maximizes the items packed according to its value and not all items are used (i.e., there are no demands) and the second one minimizes the bins used;
- 1D-CSP and 1D-BPP is that the first one uses the same item several times to accomplish a demand and the second one has a demand of exactly one for each item.

4. Open Dimension Problem
This problem is an input minimization, with only one large object where at least one dimension is variable, and the small items are arbitrary. The objects in this problem are called strips. The 2D Strip Packing Problem (2D-SPP) is the most known one. If having an amount of items, with width and height (w_j and h_j), to be packed into a strip with a fixed width (W), the unbounded height (H) of that strip for which all the items are packed should be determined. The bottom left corner of the strip is usually used as the origin of a xy-plane with x and y having the direction of the width and height of the strip respectively. The location of a rectangle is given by (x_j, y_j). If a rectangle j is located left considering a rectangle k, then $l_{jk} = 1$ with $x_j + w_j \leqslant x_k$, otherwise $l_{jk} = 0$. If a rectangle j is located below considering a rectangle k, then $b_{jk} = 1$ with $y_j + h_j \leqslant y_k$, otherwise $b_{jk} = 0$.

LP formulation:

$$\min H \tag{4a}$$

subject to:

$$l_{jk} + l_{kj} + b_{jk} + b_{kj} \geqslant 1 \qquad \forall j, k \in J, j \neq k \tag{4b}$$

$$0 \leqslant x_j \leqslant W - w_j \qquad \forall j \in J \tag{4c}$$

$$0 \leqslant y_j \leqslant H - h_j \qquad \forall j \in J \tag{4d}$$

$$x_j + w_j \leqslant x_k + W(1 - l_{jk}) \qquad \forall j, k \in J, j \neq k \tag{4e}$$

$$y_j + h_j \leqslant y_k + H(1 - b_{jk}) \qquad \forall j, k \in J, j \neq k \tag{4f}$$

$$l_{jk}, b_{jk} \in \{0, 1\} \qquad \forall j, k \in J, j \neq k \tag{4g}$$

The main difference between 2D-SPP and 2D-BPP is that the first one minimizes the unbounded height of the bin used, while the second one minimizes the bins used. Therefore, in the LP formulation of the 2D-SPP, all the items are packed in the same and only existing bin. It is important for the reader to know the basis of this problem because it can be used to solve higher dimensional BPP problems (2D or higher).

2.2. Upper and Lower Bounds

None of the problems stated above can be solved in polynomial time, unless $\mathcal{P} = \mathcal{NP}$. Hence, the quality of the solutions found through a heuristic can be evaluated through the comparison with a UB or LB. Moreover, UB/LB and dominance criteria can be used to obtain an enumerative algorithm for exact solution of the problem.

Two particular performance measures in BPP to evaluate UB and LB are the Absolute (ABS) and Asymptotic (ASY) Worst Case Performance Ratios (WCPRs) or approximation ratios that will be defined formally next.

Let I denote the set of the items and P a minimization problem. Let OPT denote an optimal offline algorithm, and A an arbitrary packing algorithm. Let $OPT(I)$ and $A(I)$ denote the number of bins created by them to pack the items of I, respectively. The ABS and ASY approximation ratio of algorithm A are defined as:

$$\overline{r}_{ABS}(A) = \sup_I \{\frac{A(I)}{OPT(I)} | I \text{ is an instance of } P\} \tag{5a}$$

$$\overline{r}^{\infty}_{ASY}(A) = \lim_{n \to \infty} \sup_I \{\frac{A(I)}{OPT(I)} | \\ I \text{ is an instance of } P \text{ with } OPT(I) \geqslant n\} \tag{5b}$$

In the case of a LB, $\underline{r}_{ABS}(L)$ and $\underline{r}^{\infty}_{ASY}(A)$ are defined by the $\inf$ and the $\sup$, respectively, in opposition to the UB. To create a new UB/LB, there is a comparison with previous ones in terms of performance and dominance relations, and a worst-case analysis is performed to assess the WCPRs.

3. Literature Review

To be easier to understand each algorithm mechanism, generally non-empty bins are classified as either open or closed, with open bins available for packing additional items and closed bins unavailable for packing permanently (cannot be reopened). A bin (necessarily empty) is opened when it receives its first item.

3.1. Exact algorithms

Exact algorithms are slower than heuristics because their search to find the optimal solution is exhaustive. For 1D-BPP, the best known B&B technique are Martello and Toth Procedure (MTP), using LBs to reduce the problem; BISON, with improved LBs and a Tabu Search for more efficiency; and an improved Bin Completion, conjugating LBs, UPs and a Heuristics defining bin completions through each node of the B&B tree. It is still important to mention the improved and revised Arc-flow algorithm, applying a multi-graph generalization to the original algorithm, a popular LP method to solve CSP and BPP merging B&B with BP and column generation. For 2D-BPP, Branch-and-Cut-and-Price (BCP) and column generation techniques introduced and applied to 2D Two-Stage Constrained Cutting Problems are a key for optimal solutions. In addition, other algorithms based on B&B techniques like the 2D-BPP nested branching scheme proposed are used. For 3D-BPP and higher dimensions, there are other B&B techniques and methods conjugated with LBs and heuristics that can provide exact solutions such as the method presented in Martello et al. (2000). Nonetheless, it is rare to run exact methods in problems with such complexity because the costs are extremely high. Therefore, there are not a lot of developments in this area [15, 6, 3, 14].

3.2. Heuristics

Heuristics use some underlying features of given problem, being a problem dependent techniques, deciding the best choice based on the current partial solution and the performance metric in each phase. Heuristics for Multidimensional BPP generally use: a shelf technique, also called level-oriented; or a technique based on two-phase algorithms, SPP solution and a later individual finite bin solution. The heuristics used for 1D-BPP are valid and used to build new heuristics applied to higher-dimensional BPP and the same is binding to other dimensions as well.

Heuristics with guarantees of performance

These well-studied algorithms provide a tight upper bound on the possible results, and the quality of the solutions provided by them can be measured using the WCPR.

For 1D-BPP, the most common heuristics are the Next Fit (NF), First Fit (FF), Best Fit (BF) and Worst Fit (WF), processing input items one by one in an arbitrary order. The difference stands in which bin the item will be places. The NF packs the next item in the last opened bin if it fits, otherwise a new bin is opened and the item packed in it. The FF packs the next item into the opened bin lowest indexed where it fits and if not suiting in any bin, a new one is opened. The BF packs the next item into the opened bin of greatest total weight (smaller residual capacity) where the item fits, following the index order of the bins and if not suiting in any bin, a new one is opened. The WF packs the next item into the opened bin of smallest total weight (emptiest) where the item fits, following the index order of the bins and if not suiting in any bin, a new one is opened. The Decreasing algorithms (NFD, FFD, BFD) behave, respectively, as NF, FF and BF algorithms with the input items sorted from the biggest to the smallest before starting the packing.

Table 1: 1D-BPP Heuristics with guarantees [16, 5, 10]

Algorithm	Time	$\overline{r}_{ABS}(A)$	$\overline{r}^{\infty}_{ASY}(A)$
NF	$\mathcal{O}(n)$	2	2
FF	$\mathcal{O}(n \log n)$	1.7	1.7
BF	$\mathcal{O}(n \log n)$	1.7	1.7
WF	$\mathcal{O}(n \log n)$	2	2
NFD	$\mathcal{O}(n \log n)$	Not Studied	$1.691\ldots$
FFD	$\mathcal{O}(n \log n)$	1.5	$1.222\ldots$
BFD	$\mathcal{O}(n \log n)$	1.5	$1.222\ldots$

According to SimchiLevi (1994), no polynomial time approximation algorithm for the 1D-BPP can have an ABS WCPR smaller than $\frac{3}{2}$ unless $\mathcal{P} = \mathcal{NP}$.

Before presenting the heuristics with guarantees related to the 2D-BPP, it's important to briefly describe 2D-SPP heuristics with guarantees due to the last being used as a middle step to solve the first in some cases. Then, for 2D-SPP the most used are Next Fit Decreasing Height (NFDH), First Fit Decreasing Height (FFDH) and Best Fit Decreasing Height (BFDH) and Bottom Left (BL) algorithm. The first three pack the items sorted by a non-increasing order of height and follow a shelf strategy using techniques similar to NF, FF and BF from 1D-BPP; while BL sorts the items by a non-increasing order of widths and packs the next item as close as possible to the bottom left corner without overlapping items.

For 2D-BPP, the most used heuristics are Hybrid Next Fit (HNF), Hybrid First Fit (HFF) and Hybrid Best Fit (HBF). These heuristics are two-phase algorithms. During the first phase NFDH, FFDH and BFDH are used respectively. And on the second

Table 2: 2D-SPP Heuristics with guarantees [7, 2]

Algorithm	Time	$\overline{r}^{\infty}_{ASY}(A)$
NFDH	$\mathcal{O}(n \log n)$	2
FFDH	$\mathcal{O}(n \log n)$	1.7
BFDH	$\mathcal{O}(n \log n)$	1.7
BL	$\mathcal{O}(n^2)$	3

phase a NFD, FFD and BFD strategies are applied to pack the shelves into the finite bins.

Table 3: 2D Heuristics with guarantees [11, 4]

Algorithm	Time	$\overline{r}^{\infty}_{ASY}(A)$
HNF	$\mathcal{O}(n \log n)$	$3.382\ldots$
HFF	$\mathcal{O}(n \log n)$	2.125
HBF	$\mathcal{O}(n \log n)$	2.125

For any BPP with a dimension higher than two, all the previous algorithms continue to be valid and can be adapted. Although, the quality of the solutions decreases a lot for approximation algorithms. Due to this fact, there is no literature measuring the WCPRs for three or more dimensional BPP.

Heuristics without guarantees of performance

The following heuristics do not have an assessed and established performance. For 1D-BPP, even though there is some literature, guarantees of performance are used more often due to its efficiency. For 2D-BPP, there are two-phase algorithms like Hybrid Floor Ceiling (HFC) the sorts items by a non-increasing order of height and packs them in bottom line using a shelf technique from the left to the right and in a top line the inverse way, following a BFD strategy; the one-phase heuristics like Finite Next Fit (FNF) or Finite First Fit (FFF) that follow a NF and FF strategy; the Bottom Left Fill (BLF) applying BL; the Alternate Directions (AD) that is a non-level approach algorithm that uses LBs and BFD to get its solutions; among other known ones. Despite that, it is important to highlight that it is possible to create multiple hybrid algorithms using the meta-heuristic GA as a searching procedure and other common heuristic routine for BPPs as decoder to fill in the bins, turning them into an hybrid heuristic that can potentially increase its performance comparing with the parent heuristic. For 3D-BPP, it is a common strategy to make use of hybrid heuristics such as Height First Area Second (HA), that uses Unified Tabu Search (TS) approach as part of its core; and the Biased Random-Key Genetic Algorithm/Variable Neighborhood Descent (BRKGA/VND) linking-up variants of GA and Variable Neighborhood Search (VNS) meta-heuristics applied to 3D-BPPs [4, 13, 23].

3.3. Meta-heuristics

Meta-heuristics are methods applied to solve problems that work in a higher level framework efficiently looking for a feasible set, being a black box. These techniques can be applied to any problem without further information. The most important and relevant meta-heuristic to this work is the Genetic Algorithm (GA), that is a type of Evolutionary Algorithms. It inspired in biological factors and applies processes named reproduction, mutation, recombination (cross over) and selection. One used extension of GA is Memetic Algorithm (MA) where the individuals learn throughout their lifetime period and pass that information to next generations. Likewise, nowadays a lot of other meta-heuristics can be found and classified according to a taxonomy including local search and global search; single solution and population based; nature inspired, metaphor based and other inspiration; among others. The most studied and applied ones are Tabu Search (TS), Simulated Annealing (SA), Ant Colony Optimization, Particle Swarm Optimization, Greedy Randomized Adaptive Search Procedure, Variable Neighborhood Search and Guided Local Search; that cover a big part of the existing taxonomy of the state-of-the-art of meta-heuristics.

4. Multi-and-many Objective BPP

The natural development of optimization problems adapted to the real world gave origin to problems with multiple constraints and multiple conflicting objective functions. These type of of problems are called Multi-Objective Problems (MOPs) and they admit a set of optimal solutions called Pareto-set (PS). A problem with more than three objective functions is called Many-Objective Problem (MaOP). This designation raised from the solving difficulties associated with problems with a higher number of conflicting objectives applied in the real world.

Several methodologies can be used to solve MaOPs, such as No-preference, *A priori*, *A posteriori*, Interactive or Evolutionary Multi-Objective Optimization (EMO) methods [8]. The main differences and advantages of EMO compared to other approaches are that they do not use gradient-based searching procedures not getting stuck in close to optimal solutions and are able to generate several solutions at the same time (population), improving the processing speed of the search, presenting a wider range of solutions for better solving M(a)OP.

4.1. Algorithms

In order to design effective multi-objective Evolutionary Algorithms (MOEAs), the following aspects must be taken into consideration: convergence, distribution and elistism [20].

Through the years, many MOEAs were de-

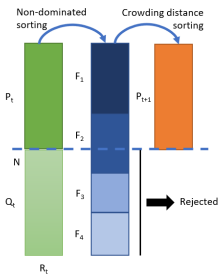


Figure 1: NSGA procedure

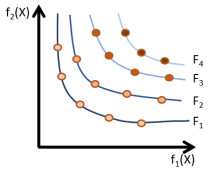


Figure 2: Fronts after non-dominated sorting

veloped, implemented and improved. Some old important ones and as a reference to the reader were Vector Evaluated Genetic Algorithm (VEGA), Nondominated Sorting Genetic Algorithm (NSGA), Strength Pareto Evolutionary Algorithm (SPEA), Pareto Envelope Based Selection Algorithm (PESA), Pareto Archived Evolution Strategy (PAES) and micro-GA. These algorithms are part of the MOPs state-of-the-art through the years and were intensively studied and applied by many authors.

Then, to solve MaOPs several new different approaches emerged. Examples of these new approaches are based on: decomposition; indicator; diversity modification; dominance modification; and preference.

Concluding, the algorithms to be tested in this work will be: MOEA based on Decomposition (MOEA/D), Nondominated Sorting Genetic Algorithm III (NSGA-III), Hypervolume Estimation Algorithm (HypE), Knee Point-Driven Evolutionary Algorithm (KnEA), Grid-based Evolutionary Algorithm (GrEA) and Two-Archive Algorithm 2 (TwoArch2). These procedures were selected due to their variety of approaches used and its implementation will be framed using PlatEMO [17] in a MatLab environment. All the original papers are recommended for a better understanding.

MOEA/D decomposes the MaOP in N scalar optimization sub-problems with a corresponding weighting vectors, making use of pre-defined search directions and running multiple searches simultaneously in the different directions [21].

NSGA-III uses an elitist approach and a similar basic procedure of combination and evaluation applied to the parent and offspring populations. The non-dominated sorted approach is applied together with a crowding scheme. Later, it creates several fronts of sorted non-dominated solutions and ranks them accordingly [9].

HypE attempts to estimate, through Monte Carlo simulation, the ranking of individuals induced by the Hypervolume (HV) indicator, having a guidance according to it and reducing significantly the computational costs. It also uses the same non-dominated sorting scheme as NSGA-III [1].

KnEA also makes use of the non-dominated sorting technique and the corresponding fronts as well. The main difference to others is the introduction of the knee point criterion, which accelerates the convergence of the population to the PF, maintaining the diversity of the solutions [22].

GrEA is based on grid division, that is defined through a parameter, and proposes the usage of a Grid Crowding Distance (GCD) metric. This methodology exploits the potential of a grip to pressure the selection process, making solutions converging faster to the PF, ensuring the maintenance of an uniform spread and following the same PF strategy as NSGA-III [19].

TwoArch2 stands for a low-complexity method with two archives directed for convergence and diversity separately, denominated CA and DA respectively. Each archive is treated in a different way with their own selection principles: indicator-based ($I_{\epsilon+}$ in CA) and Pareto-based (L_p-norm-based crowding distance metric in DA). It applies crossing over between the two archives, however, mutation is exclusive of CA [12].

4.2. Benchmark Problems

In the optimization's field, it is common to use a test suite, i.e., a set of different tests, in order to test the different algorithms mechanisms and functionality. However, an algorithm cannot be proved to be better than other simply basing its better performance in a higher number of problems. Hence, it is more accurate to state a algorithm as a good algorithm solving a particular type of problem instead of looking for its average results.

Briefly describing, the state-of-the-art of benchmarking in MOP, the most known suites are ZDT, DTLZ, WFG, and Pareto-box problem. It is still important to highlight MaF benchmark designed for CEC'2018 MaOP, containing several artificial problems covering most the real-world problems to be tested by MaOPs, manipulating some existing test suites turning them more complex.

Nonetheless, to this dissertation it was selected an alternative suit offered by the PlatEMO [17] framework and that suits better our problem: the Combinatorial MOP suit. It is composed by multi-objective Knapsack Problem (MOKP), multi-objective Quadratic Assignment Problem (mQAP) and multi-objective Traveling Salesman Problem (MOTSP).

The KP was explained in the first part of this article. The QAP belongs to the family of Facility Locations Problems. Its main purpose is to optimize the assignment of a group of facilities to a group of locations based on the cost of the materials transported and the distance between two different loca-

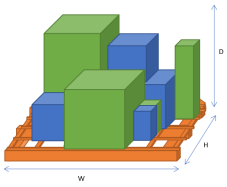


Figure 3: Sketch of a pallet loading

tions. The TSP is one of the most famous Combinatorial Problems in OR and looks for an optimization about the best order path to be followed by a salesman between different cities without repeating them and returning to the first city in the end.

During the optimization of the MaOPs in PlatEMO, the framework only uses one solution vector. Thus the problems are encoded as follows: MOKP is encoded with 0,1 in a binary procedure; mQAP is encoded in a way that allows the permutation of its variables; and MOTSP is encoded as mQAP allowing the permutation between cities.

4.3. Performance Metrics

When an algorithm tries to solve a particular known problem with a specific solution, there is the need to rank its performance through performance indicators (measures or metrics). These indicators focus on two main aspects presented to design a good algorithm: convergence towards the PF and distribution/diversity of the solutions. As a matter of fact, certain metrics are used as a complement to assess the fitness and process the elitism of some algorithms. The performance metrics to be used during this work will be Inverted Generational Distance (IGD), an indicator that calculates the average Euclidean distance between two sets of solutions; and Hypervolume (HV), an indicator that calculates the space dominated by a set of solutions, given a reference point. These metrics were selected due to covering both convergence and diversity efficiently.

4.4. MaOP applied to the BPP

To increase the objectivity of the problem and for better understanding of the reader, the built BPP will be based in a freight for shipping.

In this example, the large objects dimensions are defined by the pallets to be shipped and the transportation mean container, $W \times H \times D$. The small items have regular box shape with dimensions $w_j \times h_j \times d_j$, mass m_j and CM in the center of the box. Also, the items have an associated cost if not packed, c_j.

Thus, using the formulation of 1D-BPP, the LP formulation below can be achieved:

$$\min \sum_{i \in I} s_i \tag{6a}$$

$$\min \frac{\sum_{j \in J} m_j (z_j + d_j/2) r_{ij}}{\sum_{j \in J} m_j r_{ij}} \qquad \forall i \in I \tag{6b}$$

$$\min W/2 - \frac{\sum_{j \in J} m_j (x_j + w_j/2) r_{ij}}{\sum_{j \in J} m_j r_{ij}} \qquad \forall i \in I \tag{6c}$$

$$\min H/2 - \frac{\sum_{j \in J} m_j (y_j + h_j/2) r_{ij}}{\sum_{j \in J} m_j r_{ij}} \qquad \forall i \in I \tag{6d}$$

$$\min \sum_{j \in J} c_j (1 - \sum_{i \in I} r_{ij}) \tag{6e}$$

subject to:

$$\sum_{i \in I} r_{ij} \leqslant 1 \qquad \forall j \in J \tag{6f}$$

$$r_{ij} \leqslant s_i \qquad \forall i \in I, \forall j \in J \tag{6g}$$

$$l_{jk} + l_{kj} + b_{jk} + b_{kj} + f_{jk} + f_{kj} \geqslant p_{jk} \qquad \forall j,k \in J, j \neq k \tag{6h}$$

$$0 \leqslant x_j \leqslant W - w_j \qquad \forall j \in J \tag{6i}$$

$$0 \leqslant y_j \leqslant H - h_j \qquad \forall j \in J \tag{6j}$$

$$0 \leqslant z_j \leqslant D - d_j \qquad \forall j \in J \tag{6k}$$

$$x_j + w_j \leqslant x_k + W(2 - l_{jk} - p_{jk}) \qquad \forall j,k \in J, j \neq k \tag{6l}$$

$$y_j + h_j \leqslant y_k + H(2 - b_{jk} - p_{jk}) \qquad \forall j,k \in J, j \neq k \tag{6m}$$

$$z_j + d_j \leqslant z_k + D(2 - f_{jk} - p_{jk}) \qquad \forall j,k \in J, j \neq k \tag{6n}$$

$$s_i, r_{ij}, l_{jk}, b_{jk}, f_{jk}, p_{jk} \in \{0,1\} \qquad \forall i \in I, \forall j,k \in J, j \neq k \tag{6o}$$

The goal of this model is to minimize the number of pallets with objective function 6a, distributing the boxes efficiently and minimizing the spare space between the boxes; to maximize the weight distribution, balancing the CM of each pallet by minimizing its distance to the floor and center of the pallet in the z, x and y axes with objective functions 6b, 6c and 6d, respectively; and to minimize the cost of items not packed through objective function 6e.

In order to be able to run the algorithms in the mBPP created for the framework used, an encoding and initialization process of the solution vector was performed. The encoding of a set of items is represented by b_j - an integer between $[0, n]$, that only can be zero if the variant of the problems includes a cost for non packed items; x_j, with a value between $[0, W]$; x_j, with a value between $[0, H]$ and x_j, with a value between $[0, D]$. To evaluate the minimizing functions the LB and UB of the objective solutions is taken in account.

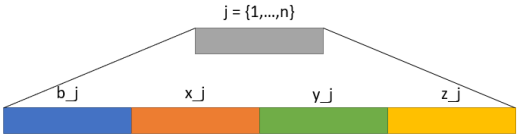


Figure 4: mBPP encoding

The standard number of items used in the problem the same as in MOKP given the problem similarity and the constants width, height and depth of each item ($w_j \times h_j \times d_j$) are initialized randomly, while the constants width, height and depth of each bin ($W \times H \times D$) are calculated as the function of the sum of the width, height and depth of all items ($w_j \times h_j \times d_j$) divided by the LB expected.

5. Experimental Analysis

The hardware used to run all the simulations was a Processor Intel ®Core ™i7-2600 CPU @ 3.40 GHz x 8, and the software was the UBUNTO 18.04.3 LTS - 64 bit, with Matlab R2018b - v9.5.0.944444 running PlatEMO 2.3 framework.

The general parameter settings common to all algorithms defined for the experiences are a compromise between the efficacy and efficiency to obtain satisfactory results in a short time frame and were the following: Population size - $N = 100$; Number of objectives to test - $M = \{2, 3, 4, 5\}$; Number of runs - 30; Number of evaluations - $N \times 100$; and Number of variables specific for each problem - $D(MOKP) = 250, D(mQAP) = 10, D(MOTSP) = 30$.

The individual parameter settings defined according to the original literature of each algorithm were the following: Number of neighbours (MOEA/D) - $N/10$; Number of Samples (HypE) - 10000; Rate of kneww points (KnEA) - 0.5; Number of dividions (GrEA) - $M = \{2, 3, 4, 5\} : div = \{45, 15, 10, 9\}$; CA size and Fractional distance (TwoArch2) - N and $1/M$.

5.1. Performance comparison for MOKP, mQAP and MOTSP

The following analysis were performed with the help of the results' tables and the notched box plots that can be consulted in Appendix A of the main dissertation document. When analysing the box plot, it is relevant that the notch displays the 95% confidence interval around the median and that if two boxes' notches do not overlap there is 'strong evidence' their medians differ. Another interesting statistic tool used is the Wilcoxon rank-sum test, that was used to provide information on 0.05 significance level of the differences of results between two competing algorithms.

5.1.1 Computational costs comparison

Regarding the time comparison, it can be affirmed that in general KnEA performs the best results on the 2 and 3-objective MOKP, mQAP and MOTSP and have approximately the same performance as NSGAIII in the other dimensions of the problems. Despite that, HypE highlights its high computational costs, being really slow compared to the other algorithms. Also, it can be classified as a general previous-last performing the MOEA/D algorithm, even though it can get closer to the performance of TwoArch2 in some problems 4-objective problems. The Wilcoxon rank-sum test confirms the outstanding results of KnEA for all problems and dimensions, except for the 4-objective MOKP where NSGAIII performs better, and for the 4-objective mQAP and MOTSP where these two algorithms have sensibly the same performance. It also confirms the evident high computational time of HypE, and it is curious to notice that MOEA/D out-ranks TwoArch2 for 4-objective MOKP and 5-objective MOTSP. Summing up, similar good performances are expected for NSGAIII, KnEA and GrEA when applied to the mBPP, and a poorer one from MOEA/D and HypE.

5.1.2 IGD comparison

Regarding IGD, analysing MOKP, it is easily identified that TwoArch2 is the dominant algorithm, followed by a good performance of NSGAIII. However, the others do not perform that good, mainly MOEA/D that decreases its comparable performance as the objective dimensions increase. Furthermore, for mQAP conclusions cannot be taken for a 2-objective problem because all algorithms perform relatively the same.Although, as we increase the objective dimension, TwoArch2 and NSGAIII start having a better result for IGD metric compared to the other algorithms, and MOEA/D decreases, once again, its performance. For the last problem, MOTSP, NSGAIII outstands its IGD values compared to the other algorithms, followed by TwoArch2, while the MOEA/D performance follows the one from the other problems, decreasing as the dimensionality increases. The Wilcoxon rank-sum test confirms problem dependent performance for the best ranked algorithms, TwoArch2 and NSGAIII and lower performance of MOEA/D that decreases with the dimensionality, loosing significance through the Wilcoxon rank-sum test, while it performs as good as other algorithms with less objectives. Summing up, for the mBPP it is expected a similar good performance for the IGD metric values from NSGAIII and TwoArch2, while MOEA/D will fall short of the other algorithms.

5.1.3 HV comparison

Regarding HV, analysing MOKP, it can be identified a solid comparative performance on the HV metric for HypE and a good performance from GrEA that starts decreasing as the dimensionality is increased. Differently, neither TwoArch2, NSGAIII nor MOEA/D present good HV results. Also, for mQAP it can be pointed out a similar good performance from HypE, GrEA and TwoArch2 for all the dimensions. Although, KnEA and MOEA/D stay behind the other algorithms using this metric. For the last problem, MOTSP, HypE has an outstanding HV values compared to the other algorithms, NSGAIII and KnEA fall short of the other algorithms' results, being the first the worst one. The Wilcoxon rank-sum test highlights the outstanding of HypE, and the good performance of GrEA for mQAP, where both algorithm get similar results in significance. Furthermore, other Wilcoxon rank-sum tests were done and pointed out a compareble good performance of KnEA for MOKP, TwoArch2 for mQAP, and MOEA/D for MOTSP, excluding HypE and GrEA algorithms. Summing up, it was expected that HypE would outperform most of the algorithms because of being an indicator based algorithm focusing on HV, even though it not always happened. Nontheless, for the mBPP it is expected a similar HV perfomance to the MOKP given the problems similarity.

5.2. Performance comparison for mBPP

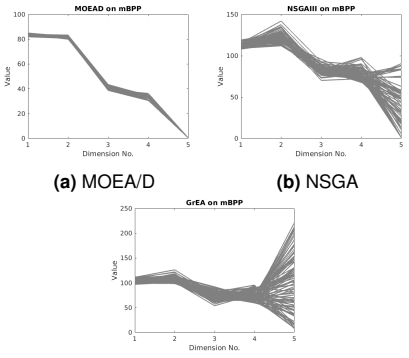
As in the previous subsection, the following analysis were performed with the help of the results' tables and the notched box plots that can be consulted in Appendix A of the main dissertation document.

5.2.1 Time comparison

Setting up a parallelism with the previous subsection and analysing the corresponding table and box plot, it is interesting to notice that NSGAIII, KnEA, GrEA and TwoArch2 have a low computational costs and as expected MOEA/D and HypE have bigger ones. Nonetheless, and comparing with the other studied combinatorial problems, the relative difference is bigger for MOEA/D in mBPP.

5.2.2 IGD comparison

Setting up a parallelism with the previous subsection and analysing the corresponding table and box plot, NSGAIII and TwoArch2 present good values for IGD, and it is important to highlight that as the dimensionality is increased, KnEA and GrEA approximate the good performance of the previous algorithms. Differently, MOEA/D and HypE rank the lowest overall.



(a) MOEA/D **(b)** NSGA

(c) GrEA

Figure 5: 5-objective mBPP solutions

The approximation of GrEA and TwoArch2 as the dimensionality increases to the top-ranked algorithms can be observed through the Wilcoxon rank-sum rank test presented in the main document.

5.2.3 HV comparison

Setting up a parallelism with the previous subsection and analysing the corresponding table and box plot, besides MOEA/D exception, the relative ranking of the results is very similar to the one expecting with HypE having the best HV values due to the reason explained before, and NSGAIII having a lower performance together with TwoArch2. Thus, MOEA/D surprisingly excels in is HV values. This happens due to the way the algorithm is implemented to perform crossover, mutation and selection of the next generation for mBPP, generating solutions closer to each other and not performing a broader search as can be seen in Figure 5. Therefore, the non-diversification of the solutions that provides a good HV result can be translated in a poor IGD result.

6. Final remarks and future work

With a technological world, BPP, initially focused on the traditional industry of mass production, has become one of the most important combinatorial problems. Nowadays, it is possible to order any sort of product from any point of the world, and, while other branches of OR focus on how to deliver the products efficiently using the appropriate channels, BPP allow us to efficiently pack the products accordingly to its constraints, decreasing the volume needed for it and consequently reducing costs and resources. At the same time, in a data era, where each time more memory access is needed to perform tasks and analyze behaviours, BPP help us to efficiently create dynamic memories with an accurate packing of data in today's virtual machines.

With this dissertation, some concern was raised about efficient meta-heuristic that could be used

to solve these type of combinatorial problems, and now the next steps are towards studying and combine them, creating new heuristics and hybrid meta-heuristics to have better solutions to our problems. There is already a very efficient and solid work towards the 1D that can be taken as an example for future development of higher dimensions techniques. It is interesting to notice that a big part of the literature written recently is improving the previous bounds and creating more efficient and accurate schemes. Although, the research still focus a lot in other type of Combinatorial Problems not often mentioning the BPP.

Concluding the analysis of the chosen problems in the previous Section, it is possible now to help the reader on how to select a good algorithm. Firstly, it is important to point out that the best algorithm to choose will always depend on the type of problems in hand and how many objective dimensions exist. Secondly, certain algorithms will have more affinity with a certain type of problems than others. Therefore, it is always important to study the state-of-the-art of the problem to be studied. And, finally, the algorithm must be applied to known problems and analysed with different metrics as done before. This analysis will be helpful to understand the behaviour of the different algorithms when acting in different environments.

Another interesting point to be implemented in the future is the development of a complementary platform (patch) to be implemented in PlatEMO framework or build a new framework from scratch allowing the comparison and analysis of exact methods, heuristics and meta-heuristics simultaneously. It is important though to study other types of C&P together with the mBPP because the study of them as whole can complement their own state-of-the-art and find different approaches and solutions in each other due to its similarity and root.

Finally, in order to implement the future works of this topic in the real world industries, it is crucial to study the way the information will be processed and presented to the Decision Makers and to the final users in order to be efficiently implemented, depending on the area where the problem solution is going to be performed.

References

[1] J. Bader and E. Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19:45–76, 2011.

[2] B. S. Baker, E. G. Coffman, Jr., and R. L. Rivest. Orthogonal Packings in Two Dimensions. *SIAM Journal on Computing*, 9(4):846–855, 1980.

[3] G. Belov. Problems, models and algorithms in one-and two-dimensional cutting. *Technischen Universität Dresden*, 2003.

[4] J. O. Berkey and P. Y. Wang. Two-Dimensional Finite Bin-Packing Algorithms. *Journal of the Operational Research Society*, 1987.

[5] J. Boyar, G. Dósa, and L. Epstein. On the absolute approximation ratio for First Fit and related results. *Discrete Applied Mathematics*, 160(13):1914–1923, 2012.

[6] F. Brandão and J. P. Pedroso. Bin packing and related problems: General arc-flow formulation with graph compression. *Computers & Operations Research*, 69:56–67, 2016.

[7] E. Coffman, M. Garey, D. Johnson, and R. Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(4), 1980.

[8] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, NY, USA, 2001.

[9] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2014.

[10] G. Dósa and J. Sgall. Optimal analysis of best fit bin packing. *Lecture Notes in Computer Science*, 8572(1):429–441, 2014.

[11] J. B. G. Frenk and G. Galambos. Hybrid next-fit algorithm for the two-dimensional rectangle bin-packing problem. *Computing*, 39:201–217, 1987.

[12] Handing Wang, Licheng Jiao, and Xin Yao. Two_Arch2: An Improved Two-Archive Algorithm for Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 19(4):524–541, 2014.

[13] A. Lodi, S. Martello, and D. Vigo. Heuristic algorithms for the three-dimensional bin packing problem. *European Journal of Operational Research*, 141(2):410–420, 2002.

[14] S. Martello, D. Pisinger, and D. Vigo. The Three-Dimensional Bin Packing Problem. *Operations Research*, 48, 2000.

[15] E. L. Schreiber and R. E. Korf. Improved bin completion for optimal bin packing and number partitioning. *International Joint Conference on Artificial Intelligence*, pages 651–658, 2013.

[16] D. Simchi-Levi. New worst-case results for the bin-packing problem. *Naval Research Logistics*, 41(4):579–585, 1994.

[17] Y. Tian, R. Cheng, X. Zhang, and Y. Jin. Platemo: A matlab platform for evolutionary multi-objective optimization [educational forum]. *IEEE Computational Intelligence Magazine*, 12(4):73–87, 2017.

[18] G. Wäscher, H. Haußner, and H. Schumann. An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130, 2007.

[19] S. Yang, M. Li, X. Liu, and J. Zheng. A grid-based evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 17(5):721–736, 2013.

[20] X. Yu and M. Gen. *Introduction to Evolutionary Algorithms*. Springer, London, UK, 2010.

[21] Q. Zhang and H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[22] X. Zhang, Y. Tian, and Y. Jin. A knee point-driven evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 19(6):761–776, 2015.

[23] A. Zudio, D. H. da Silva Costa, B. P. Masquio, I. M. Coelho, and P. E. D. Pinto. BRKGA/VND Hybrid Algorithm for the Classic Three-dimensional Bin Packing Problem. *Electronic Notes in Discrete Mathematics*, 66:175–182, 2018.