Analysis and Comparison of Exact and Approximate Bin Packing Algorithms

Luka Tadic*,**, Petar Afric*, Lucija Sikic*, Adrian Satja Kurdija*, Vladimir Klemo*, Goran Delac*, Marin Silic*

*University of Zagreb, Faculty of Electrical Engineering and Computing,
Unska 3, 10000 Zagreb, Croatia

**Corresponding author, *email:* luka.tadic@fer.hr

***Abstract*—In this paper we describe the Bin packing problem (BPP) and evaluate standard state of the art approaches to its solution. Because BPP is NP-hard [1] , there is no exact algorithm which solves the problem in polynomial time. We describe several approximate algorithms and the exact Martello-Toth-Procedure (MTP) for solving BPP. Approximate algorithms are executed in polynomial time, but they do not give optimal solutions in general. MTP searches the solution space with the help of lower bounds and approximate algorithms to minimize the solution space. For approximate algorithms we use the worst-case-to-optimal-solution ratio as a measure of effectiveness. We evaluate the relative deviation from optimal solutions for all described approximate algorithms using instances of BPP for which proven optimal solutions are found. We compare the MTP algorithm performance to those results in order to demonstrate its effectiveness.**

***Index Terms*—Bin packing problem, discrete optimization, Martello-Toth-Procedure**

I. Introduction

The bin packing problem (BPP) is one of the most common optimization problems today. Every day millions of BPP instances are solved for the needs of parcel delivery companies. Because of this, increasingly efficient algorithms for solving this problem are needed to reduce cost and improve effectiveness. The problem can be defined as follows: n items of different sizes need to be packed into bins. The aim is to pack them in such a way that a minimal amount of bins is needed. The formal definition of the problem would be:

- n is a positive natural number representing the number of items.
- Items sizes are marked with a_1, a_2,...,a_n.
- There are $M = \inf$ bins marked S_1, S_2,...,S_M of size C.
- $c_{ij} = 1$ if item i is assigned to bin j.
- Find the minimal number of bins $B \ll M$ so that:
  - each item is assigned to only one bin:

$$\sum_{j=1}^{B} c_{ij} = 1, \forall i \in \{1, 2, ..., n\} \quad (1)$$

  - no bin capacity is exceeded:

$$\sum_{i=1}^{n} a_i \cdot c_{ij} \leq C, \forall j \in \{1, 2, ..., B\} \quad (2)$$

The described problem is a NP-hard [1] problem so often, in practice, it is not feasible to find the exact solution. Thus, a good enough solution is searched for. This is done using approximate algorithms and heuristics. In order to verify and compare the effectiveness of approximate algorithm a worst-case-to-optimal-solution ratio is used as a measure of their quality. When an exact solution can be found in real time the main aim is to reduce the amount of time needed for providing a solution. In this paper we implement effective, commonly used approximate algorithms and a state of the art exact Martello-Toth-Procedure (MTP) algorithm [2]. We compare their performance and demonstrate the effectiveness of the MTP algorithm when it can be used.

The rest of this paper is organized as follows. In Section II we present approximate approaches. In Section III we present lower bounds and reduction procedure used in MTP algorithm. In Section IV we present the MTP approach. In Section V we present the evaluation results of our experiments. In Section VI we summarize our work and present future steps in this work.

II. Approximate algorithms

Approximate algorithms provide good enough solutions for the BPP problem. For these algorithms a worst-case-to-optimal-solution ratio is defined as:

$$r(A) \geq \frac{A(I)}{z(I)}, \forall I \quad (3)$$

where I indicates the instance, $A(I)$ is the solution (number of bins required) of the instance provided by the approximate algorithm and $z(I)$ is the optimal solution of the instance.

This measure is often transformed into asymptotic form in order to provide more comparable results. Real number $r^{\infty}(A)$ is defined as minimal value so that for some positive number k

$$\frac{A(I)}{z(I)} \leq r^{\infty}(A), \quad \forall I : z(I) \geq k. \tag{4}$$

Worst-case-to-optimal-solution ratio shows how poor a heuristic algorithm can be when it builds the worst solution compared to the optimal. Value of $r(A)$ is always greater than or equal to 1. In II-A, II-B and II-C we describe the approximate algorithms First fit decreasing (FFF), Best fit decreasing (BFD) and Worst fit decreasing (WFD) implemented as part of our work and further used by the MTP algorithm.

A. First fit decreasing

FFD starts with all bins closed and all items unassigned and ordered in decreasing order by their sizes. It then repetitively selects an item and then tries to assign it to the first open bin with enough residual capacity. If there is no such box then an unopened bin is selected, opened and the item is assigned to it. This is repeated until all items are assigned.

The time complexity of this procedure is $O(n \log n)$ if specialized data structures are used. The expressions (5), (6), (7) and (8) hold for the solutions of the FFD algorithm (proven in [3]).

$$FFD(I) \leq \frac{11}{9} z(I) + 4 \tag{5}$$

When transformed to asymptotic form it becomes the expression show in (6).

$$r^{\infty}(FFD) = \frac{11}{9} \tag{6}$$

For instances where the item size is $a_i \in \left\langle 0, \frac{c}{2} \right]$ it can be shown that expressions (7) holds.

$$FFD(I) \leq \frac{71}{60} z(I) + c \tag{7}$$

For instances where the items size is $a_i \in \left\langle 0, \frac{c}{4} \right]$ it can be shown that expression (8) holds.

$$FFD(I) \leq \frac{23}{20} z(I) + c \tag{8}$$

B. Best fit decreasing

The best fit decreasing algorithm starts with all bins closed and all items unassigned and ordered in decreasing order by their sizes. It then repetitively selects next bin with lowest residual capacity if exists and assigns item to that bin. If no such bin exists then an unopened bin is selected, opened and the item is assigned to it. This is repeated until all items are assigned.

The time complexity of this procedure is $O(n \log n)$ if specialized data structures are used. The expressions (9), (10), (11) and (12) hold for the solutions of the BFD algorithm (proven in [3]).

$$BFD(I) \leq \frac{11}{9} z(I) + 4 \tag{9}$$

When transformed to asymptotic form it becomes the expression show in (10).

$$r^{\infty}(BFD) = \frac{11}{9} \tag{10}$$

For instances where the item size is $a_i \in \left\langle 0, \frac{c}{2} \right]$ it can be shown that expression (11) holds.

$$BFD(I) \leq \frac{71}{60} z(I) + c \tag{11}$$

For instances where the item size is $a_i \in \left\langle 0, \frac{c}{4} \right]$ it can be shown that expression (12) holds.

$$BFD(I) \leq \frac{23}{20} z(I) + c \tag{12}$$

C. Worst fit decreasing

The worst fit decreasing algorithm starts with all boxes closed and all items unassigned. It then repetitively selects an item and then tries to assign in to an open box with enough capacity which has highest residual capacity. If no such box exists then an unopened box is selected, opened and the item is assigned to it. This is repeated until all packages are assigned to a box. The time complexity of this procedure is $O(n \log n)$ if specialized data structures are used.

The expressions (13) and (14) hold for the solutions of the WFD algorithm (proven in [4]).

$$WFD(I) \leq \frac{11}{9} z(I) + 4 \tag{13}$$

When transformed to asymptotic form it becomes the expression shown in (14).

$$r^{\infty}(WFD) = \frac{11}{9} \tag{14}$$

III. Lower bounds and reduction

For purposes of MTP algorithm, using general description of BPP described in I, a possible mathematical formulation of the problem is:

minimize

$$z = \sum_{i=1}^{n} y_i \tag{15}$$

subject to

$$\sum_{j=1}^{n} w_j x_{ij} \leq c y_j, \quad i \in N = \{1, ..., n\} \tag{16}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j \in N \tag{17}$$

$$y_i = 0 \ or \ 1, \quad i \in N \tag{18}$$

$$x_{ij} = 0 \ or \ 1, \quad i \in N, j \in N \tag{19}$$

where

$$y_i = \begin{cases} 1, & \text{if bin i is used;} \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

$$x_{ij} = \begin{cases} 1, & \text{if item j is assigned to bin i;} \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

It is supposed that the weights w_j and c are positive integers, and that:

$$w_j < c \ , \forall j \in N. \tag{22}$$

With this mathematical model the following lower bounds for number of needed bins are determined.

A. Lower bound L_1

The continuous relaxation of the BPP given by

$$0 \leq \ y_i \leq 1, \qquad i \in N \tag{23}$$

$$0 \leq x_{ij} \leq 1, \qquad i \in N, j \in N, \tag{24}$$

can be solved as

$$z = \sum_{i=1}^{n} \frac{w_i}{c}$$

which gives lower bound L_1 as

$$L_1 = \left\lceil \sum_{j=1}^{n} w_j / c \right\rceil$$

.

B. Lower bound L_2

Given any instance I of BPP, and any integer α, $0 \leq \alpha \leq c/2$, using

$$J_1 = \{j \in N : w_j > c - \alpha\} \tag{25}$$

$$J_2 = \{j \in N : c - \alpha \geq w_j > c/2\} \tag{26}$$

$$J_3 = \{j \in N : c/2 \geq w_j \geq \alpha\}; \tag{27}$$

the following is defined:

$$L(\alpha) = |J_1| + |J_2| + max\left(0, \left\lceil \frac{\sum_{j \in J_3} w_j - (|J_2|c - \sum_{j \in J_2} w_j)}{c} \right\rceil\right). \tag{28}$$

Lower bound L_2 is then defined using (28) as

$$L_2 = max\{L(\alpha) : 0 \leq \alpha \leq c/2, \alpha \in Z\}. \tag{29}$$

By the dominance rule L_2 dominates over L_1.

C. Reduction

Reduction is used to reduce search space of possible solutions. Reductions described in MTP algorithm are based on the following dominance criterion.

Initially, a feasible set is defined as a subset $F \subseteq N$ such that $\sum_{j \in F} w_j \leq c$. Given two feasible sets F_1 and F_2, we say that F_1 dominates F_2 if the value of the optimal solution which can be obtained by imposing for a bin, say i^*, the values $x_{i*j} = 1$ if $j \in F_1$ and $x_{i*j} = 0$ if $j \notin F_1$, is no greater than the value that can be obtained by forcing the values $x_{i*j} = 1$ if $j \in F_2$ and $x_{i*j} = 0$ if $j \notin F_2$. A possible way to check such situations is using the following criterion

Dominance criterion Given two distinct feasible sets F_1 and F_2, if a partition of F_2 into subsets $P_1, ..., P_l$ and a subset $\{j_1, ..., j_l\}$ of F_1 exist such that $w_{j_h} \leq \sum_{k \in P_k} w_k$ for $h = 1, ..., l$, then F_1 dominates F_2.

If a feasible set F dominates all others, then the items of F can be assigned to a bin and removed from N, where N is defined at the begining of section III. Checking all such situations, however, is clearly impractical. The algorithm MTRP used in MTP limits the search to sets of cardinality not greater than 3 and avoids enumeration of useless sets. It considers the items according to decreasing weights and, for each item j, it checks for the existence of a feasible set F such that $j \in F$, with $|F| \leq 3$, dominating all feasible sets containing item j. Whenever such set is found, the corresponding items are assigned to a new bin and the search continues with the remaining items. It is assumed that, on input, the items are sorted in non-decreasing order. On output z^r gives the number of optimally filled bins, and for $j \in N$,

$$b_j = \begin{cases} 0, \text{if item j has not been assigned;} \\ \text{bin to wich it has been assigned, otherwise.} \end{cases}$$

Time complexity of MTRP is $O(n^2)$

D. Lower bound L_3

Reduction procedure MTRP can be used to determine a new lower bound L_3. Lower bound L_3 is determined with lower bound L_2 and procedure MTRP. After execution of procedure MTRP for an instance I of BPP, let z_1^r denote the output value of z^r, and $I(z_1^r)$ the corresponding residual instance, defined by item set $\{j \in N : b_j = 0\}$. It is obvious that a lower bound for I is given by $z^r + L(I(z_1^r))$, where $L(I(z_1^r))$ denotes the value of any lower bound for $I(z_1^r)$.(Note that $z_1^r + L(I(z_1^r)) \geq L((I))$.) Suppose now that $I(z_q^r)$ is relaxed in some way and MTRP is applied to the relaxed instance, producing the output value z_2^r and a residual relaxed instance $I(z_1^r, z_2^r)$. A lower bound for I is then $z_1^r + z_2^r + L(I(z_1^r, z_2^r))$. Iterating the process we obtain a series of lower bounds of the form

$$L_3 = z_1^r + z_2^r + ... + L(I(z_1^3, z_2^r, ...)).$$

Iterations are repeated while there are still unassigned items in residual relaxed instance of BPP. Procedure for determining lower bound L_3 uses lower bound L_2 as L. Items are sorted in non-increasing order. Time complexity for executing procedure for determining L_3 is $O(n^3)$. It is obvious that $L_3 \geq L_2$.

IV. MTP ALGORITHM

Algorithm MTP is exact algorithm which uses branch and bound method with "first fit decreasing" branching strategy. The items are initially sorted according to decreasing weights. At each decision node, the first free item is assigned, in turn to the feasible initialized bins (by increasing index) and to a new bin. At any forward step, (a) procedures L_2 and L_3 are called to attempt to remove the node from the search space and reduce the current problem; (b) when no removing occurs, approximate algorithms FFD, BFD, and WFD are applied to the current problem, to try and improve the best solution so far. In addition, the following dominance criterion between decision nodes is used. When the current item j^* is assigned to a bin i^* whose residual capacity $\overline{c}_{i^*}$ is less than $w_j + w_n$, this assignment dominates all the assignments to i^* of items $j > j^*$ which do not allow the insertion of at least one further item. Hence such assignment "closes" bin i^*, in the sense that no item $j \in \{k > j^* : w_k + w_n > \overline{c}_{i^*}\}$ is assigned to bin i^*. The bin is "re-opened" when the first item $j > j^*$ for which $w_j + w_n \leq \overline{c}_{i^*}$ is considered. Since at any decision node the current residual capacities $\overline{c}_i$ of the bins are different, the computation of lower bounds L_2 and L_3 must take into account this situation. An easy way is to relax the current instance by adding one extra item of weight $c - \overline{c}_i$ to the free items for each initialized bin i, and by supposing that all bins have capacity c.

V. EVALUATION RESULTS

After implementing exact algorithm MTP and approximate algorithms FFD, WFD and BFD in programming language Python, programs are executed on several different instances of BPP. Used instances are Falkenauer [5], Scholl [6] and Schwerin [7].

A. Falkenauer

Instance of this dataset is divided into two classes of 80 instances each: the first class has uniformly distributed item sizes (set 'Falkenauer U') with n between 120 and 1000, and c = 150. The instances of the second class (set 'Falkenauer T') includes the so-called triplets, i.e., groups of three items (one large, two small) that need to be assigned to the same bin in any optimal packing, with n between 60 and 501, and c = 1000. Boxplot diagram which shows relative deviation from optimum for approximate algorithms FFD, BFD and BFD is showed in figure 1.

B. Scholl

Instances of this dataset are divided into three sets of 720, 480, and 10 samples, respectively, uniformly distributed instances with n between 50 and 500. The capacity c is between 100 and 150 (set 'Scholl 1'), equal to 1000 (set 'Scholl 2'), and equal to 100 000 (set 'Scholl 3'), respectively. Boxplot diagram which shows relative deviation from optimum for approximate algorithms FFD, BFD and BFD is showed in figure 3.

C. Schwerin

They are divided into two sets ('Schwerin 1' and 'Schwerin 2') of 100 instances each with n = 100 and n = 120, respectively, and c = 1000. Boxplot diagram which shows relative deviation from optimum for approximate algorithms FFD, BFD and BFD is showed in 2.

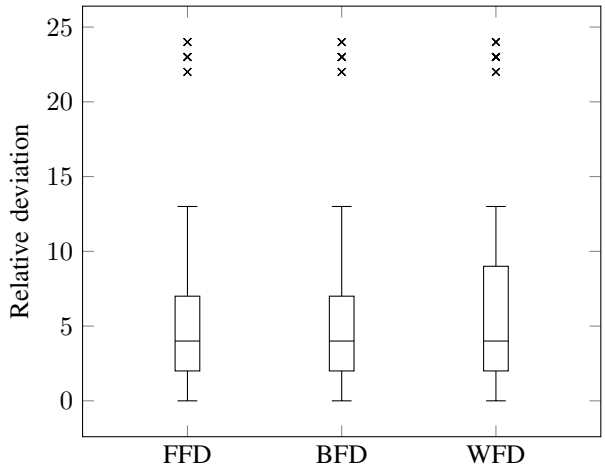

Fig. 1: Box plot for instances from Falkenauer set

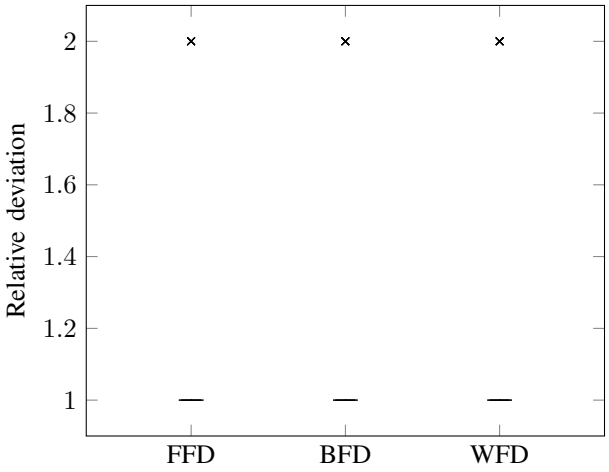

Fig. 2: Box plot for instances from Schwerin set

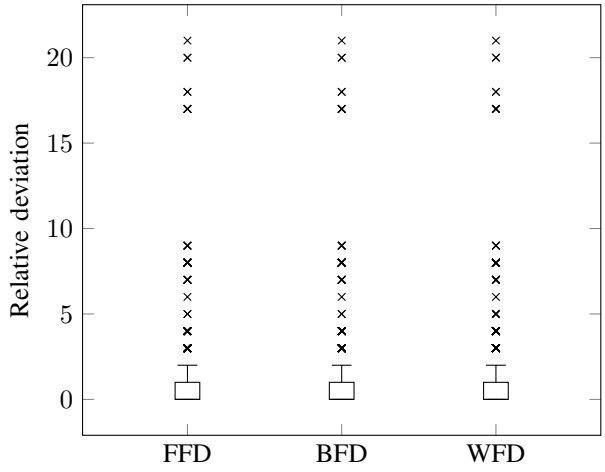

Fig. 3: Box plot for instances in Scholl set

D. MTP

Results of exact algorithm MTP are determined by executing algorithm for each instance set for 5 minutes are showed in table I.

TABLE I: Results of executing algorithm MTP

Instance	Number of optimal solutions found	Total number of problems
Falkenauer	107	160
Scholl	1115	1210
Schwerin	195	200

VI. Conclusion

BPP is NP-hard problem, so often in practice, it is not feasible to find the exact solution. Thus, a good enough solution is searched for. We presented three approximate algorithms FFD, WFD and BFD and showed box plot of their relative deviation from optimum. When an exact solution is required the main aim is to reduce the amount of time needed for providing a solution. In this paper we implement a state of the art exact MTP algorithm and demonstrate its effectiveness.

Acknowledgment

The authors acknowledge the support of the Croatian Science Foundation for this research through the Reliable Composite Applications Based on Web Services (IP-2018-01-6423) research project.

This research has been partly supported by the European Regional Development Fund under the grants KK.01.1.1.01.0009 (DATACROSS) and KK.01.2.1.01.0111 (OperOSS). The Titan X Pascal used for this research was donated by the NVIDIA Corporation.

References

[1] *Bin-Packing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 426–441. [Online]. Available: https://doi.org/10.1007/3-540-29297-7_18

[2] S. Martello and P. Toth, *Knapsack problems : algorithms and computer implementations*. Chichester : Wiley, 1990. [Online]. Available: https://lib.ugent.be/catalog/rug01:000223124

[3] D. S. Johnson, A. J. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham, "Worst-case performance bounds for simple one-dimensional packing algorithms." *SIAM J. Comput.*, vol. 3, no. 4, pp. 299–325, 1974. [Online]. Available: http://dblp.uni-trier.de/db/journals/siamcomp/siamcomp3.html#JohnsonDUGG74

[4] D. S. Johnson, "Fast algorithms for bin packing," *Journal of Computer and System Sciences*, vol. 8, no. 3, pp. 272 – 314, 1974.

[Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022000074800267

[5] E. Falkenauer, “Falkenauer bpp instances dataset.” [Online]. Available: http://or.dei.unibo.it/sites/or.dei.unibo.it/files/instances/Solutions.rar

[6] A. Scholl, R. Klein, and C. Jürgens, “Scholl bpp instances dataset.” [Online]. Available: http://or.dei.unibo.it/sites/or.dei.unibo.it/files/instances/Scholl.rar

[7] P. Schwerin and G. Wäscher, “Schwerin bpp instances dataset.” [Online]. Available: http://or.dei.unibo.it/sites/or.dei.unibo.it/files/instances/Schwerin.rar