© International Journal of Combinatorial Optimization Problems and Informatics, Vol. 4, No. 3, Sep-Dec 2013, pp. 2-10. ISSN: 2007-1558.

A Parallel Nash Genetic Algorithm for the 3D Orthogonal Knapsack Problem

Daniel Soto[1], Wilson Soto[1,2] and Yoan Pinzón[1]
[1]*Research Group on Algorithms and Combinatorics (ALGOS-UN)*
Universidad Nacional de Colombia
dsoto.7@gmail.com, {wesotof,ypinzon}@unal.edu.co
[2]*Intelligent Systems and Spatial Information Research Group (SIGA)*
Universidad Central - Colombia
wsotof@ucentral.edu.co

Abstract. The Three-Dimensional Knapsack Packing Problem consists of finding the maximum profit for packing a subset of boxes in a larger box (packing box). The boxes to pack are rectangular but of different sizes. This problem is a variant of the Knapsack Problem and therefore its computational complexity is $\mathcal{NP}$-Complete. This article shows a parallel genetic algorithm based on the Island Model and the Nash equilibrium theory for solving the 3D Knapsack Packing Problem assuming both, orthogonally packing and rotation prevention. Computational experiments comparing the developed algorithm with previous approach indicate that the results are very hopeful for three-dimensional problem.

Keywords: Genetic Algorithms, Island Model, Knapsack Problem, Nash Equilibrium theory.

1 Introduction

The packing problem is a type of the combinatorial optimization problem related with the problem of how to optimally pack objects into a container. The objective is to find the densest packing of objects in a container or, in other cases, packing the objects in the least possible number of containers. For this type of problem, we are given the containers (usually two- or three-dimensional) and a set of objects with different sizes that we intend to pack into the containers.

This problem has many real life applications, from cutting problems where the idea is to optimize the material or space, to various problems relating to transport and packaging of goods.

Particular cases of packing problems are the Bin Packing Problem and the Knapsack Problem. These types of combinatorial problems are known to be $\mathcal{NP}$-Complete and extremely difficult to solve in practice.

Researches in theory of algorithms have determined that is unlikely to find a polynomial algorithm for combinatorial problems. Instead, in recent years, the evolutionary algorithms have proved to be highly efficient for solution of combinatorial problems.

Evolutionary algorithms (EA) are powerful optimization methods inspired by biological evolution such as, reproduction, recombination, mutation and selection. The most popular method of EA is the genetic algorithm (GA). They have successfully been used for solving various difficult real-world problems.

A new algorithm approach called PNGA (Parallel Nash Genetic Algorithm) for 3D Packing Knapsack Problem is suggested in this paper.

This article is organized as follows. Section 2 introduces some basic concepts needed throughout the paper. The previous works related with the 3D orthogonally packing knapsack problem appears in Section 3. In Section 4 we present our proposed algorithm. Section 5 is devoted to compare our obtained results to previous results in the literature, and finally, our conclusions are presented in Section 6.

Received Nov 20, 2013 / Accepted Dec 19, 2013

2 Preliminaries

2.1 Genetic Algorithms

Genetic algorithms are one of the three paradigms of evolutionary algorithms. Evolutionary algorithms are stochastic optimization methods that use mechanisms inspired by biological evolution. The idea is to represent the solutions by simulating individuals engaging in combinations and competitions so that the fittest or best individuals prevail aiming a more evolved population.

Particularly, genetic algorithms are methods based on biological mechanisms, such as, Mendel's laws and Darwin's fundamental principle of natural selection, and they simulate the process of natural population evolution according to the natural selection principle of survival of the fittest. An instance of the problem is represented by individuals or chromosomes, which create a set of solutions. For the chromosomes, the simulated genetics changes are crossovers and mutations. This changes are applied to achieve a set of solutions increasingly more adapted to the problem. These adaptations are then evaluated by a quality factor called fitness.

The Algorithm 1 shows the general structure of a genetic algorithm, where *g* is the iteration number and *P* is the population or set of individuals [8].

Algorithm 1. Generic Genetic Algorithm

```
g ← 0
population {P(g)}
evaluation {P(g)}
While gmax do
   P'(g) ← selection_parents {P(g)}
   P'(g) ← crossover {P'(g)}
   P'(g) ← mutation {P'(g)}
   evaluation {P'(g)}
   P(g+1) ← P'(g) + P(g)
   g ← g+1
EndWhile
```

The genetic algorithms are generally able to find good solutions in a reasonable time but in some cases the problems can have a huge search space requiring new options in order to achieve the goal of keeping the time reasonable. For that reason, there has been a recent steadily growing interest in this research field, focusing especially on parallel implementations [5].

2.2 Island Model

The Island Model is a class of genetic algorithms designed with the parallel paradigm. This model basically consists of a physical distribution of the population. The population is divided into several disjoint sub-populations, each of which is executed with a sequential genetic algorithm allowing for occasional exchange of individuals between sub-populations (migrations) according to given parameters (number or percentage of individuals to migrate) [8]. The main idea is to create diversity in all sub-populations periodically and avoiding the premature convergence to a local optimum (see Algorithm 2).

2.3 Nash Equilibrium

Game theory is a formal study designed to produce the optimal decisions of players in conflict. The decisions are strategic since it considers the actions taken by others players. One of the theories for finding the solution of a simultaneous decision game where the players have entire information is the Nash equilibrium.

Algorithm 2. Generic Parallel Genetic Algorithm

```
AGP
1   gi ← 0
2   population {Pi(gi)}
3   evaluation {Pi(gi)}
4   While gmax do
5       Pi'(gi) ← selection_parents {Pi(gi)}
6       Pi'(gi) ← crossover {Pi'(gi)}
7       Pi'(gi) ← mutation {Pi'(gi)}
8       evaluation {Pi'(gi)}
9       Pi'(gi) ← migration {Pi'(gi) + AGP[Pi(gi)]}
10      P(gi+1) ← Pi'(gi) + Pi(gi)
11      gi ← gi+1
12  EndWhile
```

The Nash equilibrium of a game is a deal between players, so that, if one of the players changes their strategies, this player gains nothing while others keep theirs. That is, if some player no achieves the deal and it was made unilaterally, risks earning below what he would have earned under the deal, however, in some cases, the deal may not be the most beneficial for the players, for instance, the Prisoners' dilemma [7].

A given game with a finite number of players and strategies has at least a Nash equilibrium, even if it involves some objective probabilities of game by the players. Thus, Nash equilibrium states that if it were possible to coordinate the actions of the players the outcome for all would be the best.

Formally, for an optimization problem with *G* objectives let *E* be the search space for the optimization of its own criteria and *F* the search space for the optimization of its own criteria given all others criteria determined for the others players. The strategy formed by the pair (x, y) is Nash equilibrium if and only if (1):

$$f_E(\bar{x},\bar{y}) = \inf_{x \in E} f_E(x,\bar{y})$$
$$f_F(\bar{x},\bar{y}) = \inf_{y \in F} f_F(\bar{x},y) \qquad \textbf{(1)}$$

We can define: $u = (u_1,\ldots,u_G)$ as Nash equilibrium if and only if (2):

$$\forall i, \forall v_i \; J_i\,(u_1,\ldots,u_{i-1},u_i,u_{i+1},\ldots,u_G) \le J_i\,(u_1,\ldots,u_{i-1},v_i,u_{i+1},\ldots,u_G) \qquad \textbf{(2)}$$

2.4 3D Packing Knapsack Problem

The 3D Packing Knapsack Problem (3DKP) consists of finding the maximum profit for packing a subset of boxes *i* (each box with dimensions a_i, h_i, l_i and positioned in the coordinates [x_i, y_i, z_i]) in a larger box (packing box) with dimensions *A, H* and *L*. The mathematical formulation of the problem (3) can be represented like in [4] where the binary decision variables are: δ_i that represents if the box was introduced in the packing box and k_{ij} (left), m_{ij} (right), n_{ij} (bottom), p_{ij} (top), q_{ij} (behind) and r_{ij} (front) to indicate the relative spatial position of the boxes *i-j* avoiding overlaps.

2.5 Packing Algorithm

The insertion of the boxes in the container is in the LFB (left-front-bottom) order. The packing boxes begin in the point LFB of the container and continue finding available spaces with the scheme LR (left toward right), FB (front toward back) and BU (bottom toward up) (see Fig. 1).

Given a finite sequence of unique box-ids (i_1, i_2, ..., i_n), the packing algorithm evaluates the dimensions of the box i_1 with respect to the container dimensions. If the first box i_1 is incorporated in the container, it is at point LFB. For the second box i_2 the same evaluation is conducted, if possible, the box is positioned at the right of the box i_1, if no, at the right (LR) is evaluated behind (FB) of box i_1, and still, if no possible, evaluates top (BU) of box i_1. If the box cannot be positioned after these evaluations, the box is eliminated from the sequence. The same procedure is repeated, until the whole sequence of boxes is finished. The final sequence is the set of boxes incorporated in the container.

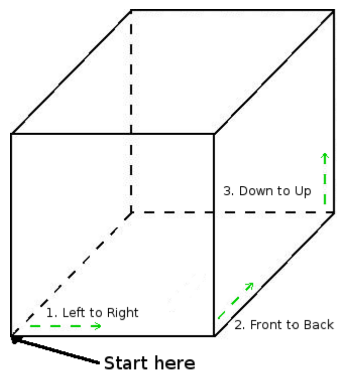

Fig. 1. Container with the point LFB and the directions for finding available spaces

3 Previous works

One of the most important works related to the Multi-Dimensional Packing Problems is in [1]. Specially, this work presents a classification into the Multi-Dimensional Bin Packing Problems, the Multi-Dimensional Knapsack Problems and the Container Loading Problem.

The Multi-Dimensional Bin Packing Problem consists in a set of boxes and an unlimited number of containers, where the task is to pack the boxes orthogonally in the minimum possible number of containers, assuming that the boxes cannot rotate. This problem is also known as the 3D-BP (*Three-Dimensional Orthogonal Bin Packing problem*) or 3D-SBSBPP (*Three-Dimensional Single Bin-Size Bin Packing Problem*).

The Multi-Dimensional Knapsack Problems consists in a set of boxes having a profit and a container of fixed size, where the task is to pack a subset of boxes orthogonally inside the container such that profit is maximized. The boxes cannot be rotated. This problem is also known as 3D-KP (*Three-Dimensional Orthogonal Knapsack problem*) or 3D-SLOPP (*Three-Dimensional Single Large Object Placement problem*).

The Container Loading Problem consists in a set of boxes and a container of fixed size, where the task is to pack a subset of boxes orthogonally inside the container, maximizing the volume used. This problem is also known as 3D-CLP (Three-Dimensional Container Loading problem). This problem usually can rotate boxes.

Notably, the 3D-KP problem has been under-researched and is largely unexplored, despite the fact some important contributions are: a branch and bound algorithm [4], a heuristic simulated annealing approach [2][3] and the GRASP model (Greedy Adaptive Search Procedure) [6].

4 A Proposed Algorithm

The proposed genetic algorithm is based on the Island Model and the Nash Equilibrium theory. The Algorithm creates an initial random population that, for the specific problem, it refers to the packing of boxes in the container until its capacity is reached.

The chromosome is designed as a structure containing the sequence of boxes packed in the container. A gene is located on a chromosome and in our design it represents a box (see Example 1). The structure also contains the calculated values of fitness for each objective associated to the problem.

Example 1. (Representation of the chromosome) Given a container with a volume of 700000 and a set of boxes of size 20, a possible solution is given by the following chromosome: [18 5 2 16 8 0 13 10 15 3] with associated values (633672 / 606422). In this case, the chromosome represents 10 boxes with total profit of 633672 and total volume of 606422.

The proposed algorithm uses the packing algorithm and the designed chromosomes. Besides, the chromosome is useful for verifying that the dimensions of each box that is introduced in the container.

$$
\begin{array}{lll}
\text{maximize} & z = \sum_{i=1}^{n} p_i \, \delta_i & \\
\text{s.t.} & k_{ij} + m_{ij} + n_{ij} + p_{ij} + q_{ij} + r_{ij} \geq \delta_i + \delta_j - 1 & i, j = 1,\ldots,n \\
& x_i - x_j + Ak_{ij} \leq A - a_i & i, j = 1,\ldots,n \\
& x_j - x_i + Am_{ij} \leq A - a_j & i, j = 1,\ldots,n \\
& y_i - y_j + Hn_{ij} \leq H - h_i & i, j = 1,\ldots,n \\
& y_j - y_i + Hp_{ij} \leq H - h_j & i, j = 1,\ldots,n \\
& z_i - z_j + Lq_{ij} \leq L - l_i & i, j = 1,\ldots,n \\
& z_j - z_i + Lr_{ij} \leq L - l_j & i, j = 1,\ldots,n \\
& 0 \leq x_i \leq A - a_i & i = 1,\ldots,n \\
& 0 \leq y_i \leq H - h_i & i = 1,\ldots,n \\
& 0 \leq z_i \leq L - l_i & i = 1,\ldots,n \\
& k_{ij}, m_{ij}, n_{ij}, p_{ij}, q_{ij}, r_{ij} \in \{0,1\} & i, j = 1,\ldots,n \\
& \delta_i \in \{0,1\} & i = 1,\ldots,n \\
& x_i, y_i, z_i \geq 0 & i = 1,\ldots,n
\end{array}
\qquad (3)
$$

Then by the configured number of iterations the algorithm performs the stage crossover where the type of crossover is exclusive for the two cases. In the first case, One-point crossover with 80% probability and the second case with uniform crossover and 20% probability until four children are obtained for each pair of randomly selected parents. Afterwards, in the mutation stage, there are two types of inclusive mutation: a mutation by exchange with 10% probability and a replacement mutation with 30% probability. Once the mutation stage is evaluated, is evaluated whether it is possible to introduce more boxes in the container.

Each individual of the population has two fitness types. The first fitness value corresponds to the sum of profits of each box, and second value corresponds to the empty volume in the container or packing box.

Each sub-population created by the algorithm has a specific task, the even sub-populations attempt to minimize the empty volume and the odd sub-populations attempt to maximize the total profit of the boxes introduced in the container.

For each even iteration some individuals are randomly selected and exchanged between sub-populations, this exchange is based on the Nash Equilibrium. Finally, all the individuals (population) are sorted by their fitness value where the first individual is the solution, in other words, the global maximum of the optimization.

The Algorithm 3 summarizes the proposed 3DKP-PNGA (3DKP – Parallel Nash Genetic Algorithm) algorithm.

The parallel implementation is using OpenMP [9] with a dynamic allocation and a structure of shared memory that contains the generated local populations and a shared area for the exchanges between all the sub-populations.

Algorithm 3. Algorithm 3DKP-PNGA

```
   Input: Individuals number and iterations number
   Output: Individual with best fitness in all the sub-populations
1  Create 4 sub-populations ◄ (For each processor core)
2  Create initial population (individuals number) for each
   sub-population
3  While (i <= iterations number) do ◄ (For each sub-population)
4      Individual crossover
5      Individual evaluation - fitness
6      Individual selection
7      If (i is even) then
8          Exchange individuals between
                     sub-populations ◄ (Nash Equilibrium)
9      EndIf
10 EndWhile
11 Sort the individuals of the sub-populations by fitness values
```

5 Computational experiments

The proposed algorithm was run on a 4MB memory machine with Debian 7.1 operating system and an AMD Quad-Core 1.4MHz processor. After ten iterations, Table 1 and Figure 2 show the summarized results. In [10], the source code and the used dataset are available.

Table 1. Results for some ep3d instances. 1D is the optimal solution of the one dimensional relaxed problem. Show the best case and worst case for each instance. Best time is the time it took before the best solution was encountered. The percentage is the relation between the heuristic solution and the 1D relaxed problem.

Instance	1D	Best	Worst	Time	Percentage 1D
ep3d-20-C-C-50	1026348	633672	606422	20	61.7%
ep3d-20-C-C-90	1834340	916241	892284	0.8	49.9%
ep3d-20-D-C-50	395916	239532	209776	52,2	60.5%
ep3d-20-U-C-50	4495440	3088676	2542564	278,4	68.7%
ep3d-20-D-R-50	240621	195937	154890	192,2	81.4%
ep3d-40-C-C-50	2065540	1265664	959300	564	61.2%
ep3d-60-C-C-50	3063219	1504980	1265648	60	49.1%

In Fig. 3 we show the comparison of the experimental results between our proposed algorithm (3DKP - PNGA) and the dataset authors (3DKP-RS). Both algorithms found the same best solutions. The best solution for the dataset authors represents the upper bound of the solutions. The difference between the two algorithms is at the lower bound, but the results of the proposed algorithm can be improved by updating the number of iterations or generations.

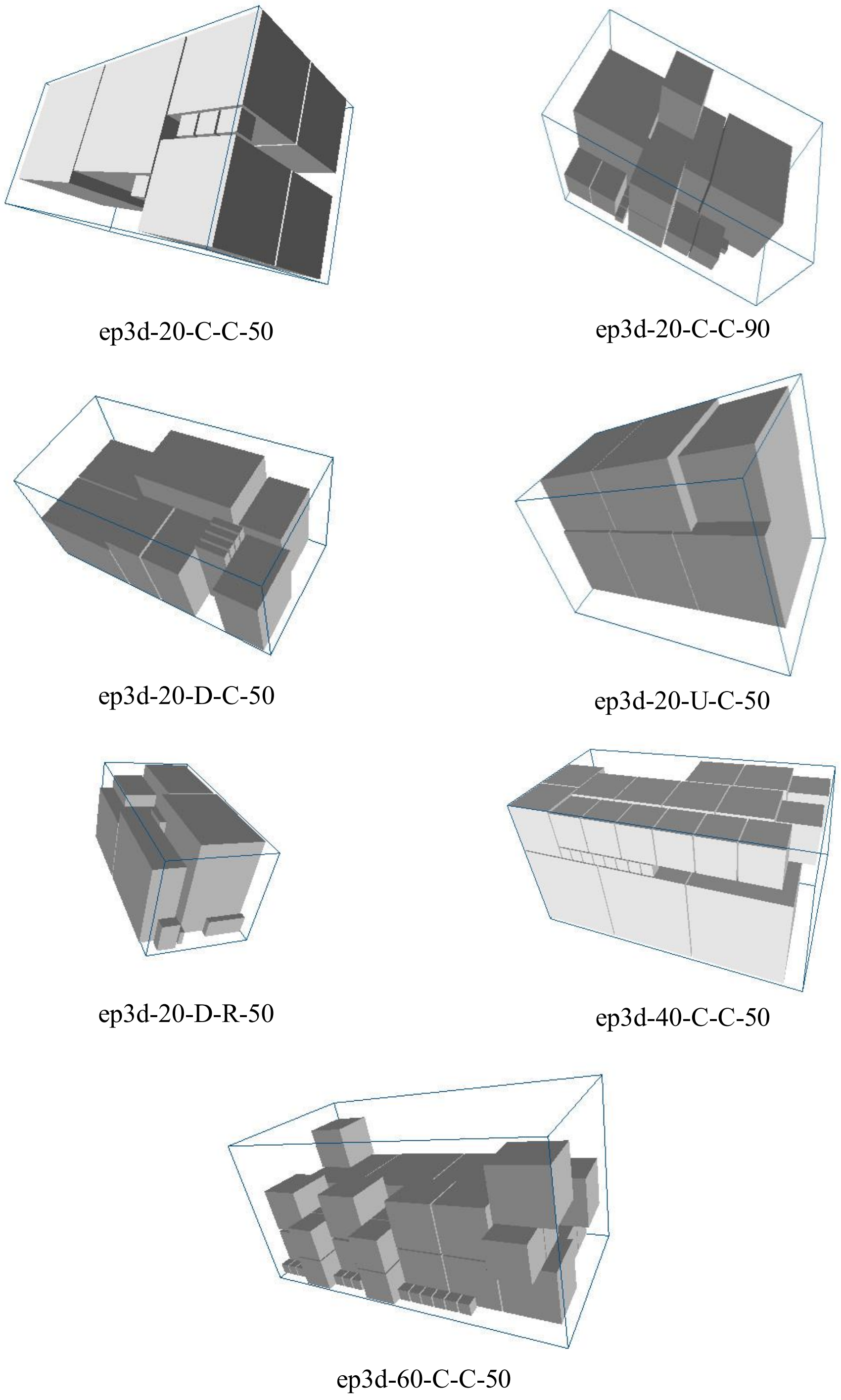

Fig. 2. The best results for 7 ep3d instances.

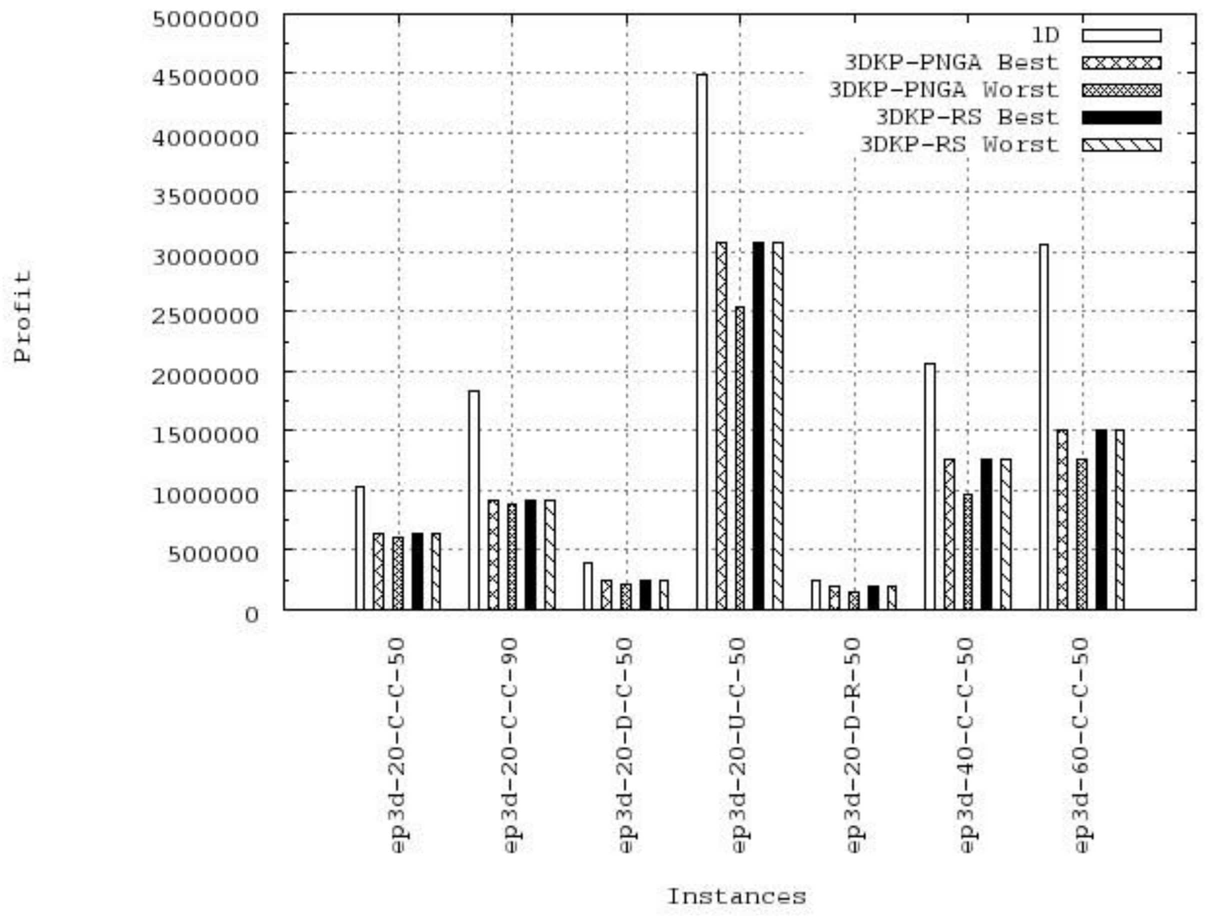

Fig. 3. Results for each data set with the best and the worst solution for the proposed algorithm 3DKP – PNAG and 3DKP-RS [3] (based in simulated annealing) and the relaxation of the 1-Dimension (1D) problem.

6 Conclusions

An algorithm for the 3D packing knapsack problem has been presented. The algorithm is based on the evolutionary algorithms, specifically a parallel genetic algorithm that has two additional characteristics, the Island Model and the Nash Equilibrium theory.

The performance of the proposed algorithm on the dataset used was similar to the upper bound of the solutions. The proposed algorithm found the solutions in a reasonable time. Other contribution of this work is a new variant for the 3D packing knapsack problem.

In our future work, we hope to run the experiments on more datasets and to implement other types of heuristics to improve the quality of solutions and looking to enhance the scientific understanding in this research area.

Acknowledgments

This work was supported by: SEGUNDA CONVOCATORIA INTERNA DE INVESTIGACIÓN - UNIVERSIDAD CENTRAL [Project No. 20301081-2013].

References

1. Crainic, T.G., Perboli, G., Tadei, R.: Recent Advances in Multi-dimensional Packing Problems. In: New Technologies - Trends, Innovations and Research. Constantin Volosencu, Rijeka, Croatia. pp. 91--110 (2012).
2. Egeblad, J.: Heuristics for Multidimensional Packing Problems. PhD Thesis. University of Copenhagen (2008).
3. Egeblad, J., Pisinger, D.: Heuristic approaches for the two- and three-dimensional knapsack packing problem. J. Comput. Oper. Res. 36 (4), 1026--1049 (2009).
4. Fekete, S. P., Schepers, J.: A combinatorial characterization of higher-dimensional orthogonal packing, J. Mathematics of Operations Research 29(2): 353--368 (2004).
5. Paz, E. C.: A Survey of Parallel Genetic Algorithms. J. Calculateurs Paralleles, Reseaux et Systems Repartis. 10 (2), 141--171 (1998).
6. Perboli, G., Crainic, T. G., Tadei, R.: An efficient metaheuristic for multi-dimensional multi-container packing, In: Seventh annual IEEE Conference on Automation Science and Engineering IEEE CASE 2011, Trieste, Italy. pp. 1--6 (2011).
7. Periaux, J., Sefrioui, M.: Nash Genetic Algorithms: Examples and Applications. In: Congress on Evolutionary Computation CEC00, CA, USA. pp. 509--516 (2000).
8. Torres E.: Análisis y Diseño de Algoritmos Genéticos Paralelos. PhD Thesis. Universidad de Malagá (1999).
9. The OpenMP® API specification for parallel programming. http://openmp.org/wp/
10. David Pisinger's optimization codes. http://www.diku.dk/~pisinger/codes.html.