# A branch-and-price algorithm for the bin-packing problem with conflicts

Ruslan Sadykov[1], François Vanderbeck[1,2]

[1] Equipe RealOpt, INRIA Bordeaux — Sud-Ouest, 351 cours de la Libération, 33405 Talence, France
Ruslan.Sadykov@inria.fr
[2] Université Bordeaux I, Institut de Mathématique, 351 cours de la Libération, 33405 Talence, France
fv@math.u-bordeaux1.fr

**Mots-Clés** : *branch-and-price, bin-packing, knapsack, conflicts*

## 1 Introduction

In the Bin Packing Problem with Conflicts (BPPC), we are given a set $V = \{1, 2, \ldots, n\}$ of items, each item $i$ having a non-negative weight $w_i$, and an infinite number of identical bins of weight capacity $W$. We are also given a conflict graph $G = (V, E)$, where $E$ is a set of edges such that $(i, j) \in E$ when $i$ and $j$ are in conflict. Items in conflict cannot be assigned to the same bin. The aim of the BPPC is to assign all items to the minimum number of bins, while ensuring that the total weight of the items assigned to a bin does not exceed the bin weight capacity and that no bin contains items in conflict.

The problem generalizes both the Bin Packing Problem and the Vertex Coloring Problem. It arises in many real-world applications such as examination scheduling, parallel computing, database storage and product delivery.

To our knowledge of the literature, there are two previous computational study on the BPPC : Gendreau et al. [1] evaluated six heuristics and lower bounds for the problem. An exact algorithm based on the branch-and-price approach was proposed by Muritiba et al. [3]. All but 10 test instances with up to 1000 items from the standard test set proposed in [1] were solved to optimality.

## 2 Branch-and-price approach

To solve the problem exactly, we adopt the branch-and-price approach as in [3]. We define $B$ as the set of all the item subsets representing feasible bins. Each bin $b \in B$ is associated with a binary variable $\lambda_b$ having value 1 if and only if the bin is selected. The BPPC can be formulated as the following set covering problem :

$$\min \left\{ \sum_{b \in B} \lambda_b : \quad \sum_{b \in B:\, i \in b} \lambda_b \geq 1, \forall i \in V, \quad \lambda_b \in \{0, 1\}, \forall b \in B \right\}. \tag{1}$$

To solve (1), we used a generic branch-and-price solver BaPCod being developed within the INRIA research team RealOpt. In our implementation, the only part specific to the problem is an oracle to solve the pricing subproblem, which is the knapsack problem with conflicts.

In the standard test instances used in [1] and [3], the conflict graph is an interval graph. To solve it we could have used the dynamic programming algorithm by Pferschy and Schauer [4] to solve the problem in the case when the conflict graph is chordal. However its high complexity $O(nW^2)$ does not allow one to use it in practice. Another option is the exact algorithm developed by Hifi and Michrafy[2] for the problem with an arbitrary conflict graph. But it based on a MIP solver and is not suited to be called many times during a column generation procedure. Therefore, we developed our own algorithm for the knapsack problem with an interval conflict graph. This algorithm is similar to the standard knapsack dynamic programming algorithm and run in the same time $O(nW)$.

To test our branch-and-price algorithm, we used 10 instances which were not solved in [3] within 1 hour. We could solve to optimality 7 instances within the same time limit 1 hour and 1 instance within 2 hours of computation time. For the remaining 2 instances, we could find better solutions than in [3]. To our opinion, the following ingredients of our algorithm contributed the most to its efficiency.
- The fast algorithm to solve the pricing problem.
- The generic branching scheme which does not break the interval conflict graph structure of the pricing problem.
- The generic "diving" (Limited Discrepancy Search) heuristic allowed us to solve 6 instances without branching.

## 3 Research perspectives

We are now working on the algorithm for solving the pricing problem for the case with an arbitrary conflict graph. This oracle should allow us to tackle general instances where the conflict graph is not an interval graph.

## Références

[1] M. Gendreau, G. Laporte, and F. Semet. Heuristics and lower bounds for the bin packing problem with conflicts. *Computers and Operations Research*, 31 :347–358, 2004.

[2] M. Hifi and M. Michrafy. Reduction strategies and exact algorithms for the disjunctively constrained knapsack problem. *Computers and Operations Research*, 34 :2657–2673, 2007.

[3] A. E. Fernandes Muritiba, M. Iori, E. Malaguti, and P. Toth. Algorithms for the bin packing problem with conflicts. *INFORMS Journal on Computing*, to appear, 2009.

[4] U. Pferschy and J. Schauer. The knapsack problem with conflict graphs. *Journal of Graph Algorithms and Applications*, 13 :233–249, 2009.