# Combinatorial Optimization - INFO-F-424 Project

Renaud Chicoisne
renaud.chicoisne@ulb.ac.be

Jérôme de Boeck
Jerome.De.Boeck@ulb.be

## Problem description

The *Bin Packing* problem (BP)is a combinatorial problem with application in industry in which a set of products $P$ must be packed in a set of boxes $B$. Each product $p \in P$ has a size $s_p > 0$ and each box has a capacity $c > 0$, all boxes have the same capacity. The goal is to find a packing minimizing the number of boxes used. We consider $s_p \leq c$ and $|P| = |B|$ in order to guarantee the existence of a solution.

Let us define the variables $x$ and $y$ as follows:

$$y_b := \begin{cases} 1 & \text{If box } b \in B \text{ is used} \\ 0 & \text{Otherwise} \end{cases}$$

$$x_{pb} := \begin{cases} 1 & \text{If product } p \in P \text{ is packed in box } b \in B \\ 0 & \text{Otherwise} \end{cases}$$

BP can be cast as the following Integer Programming problem (IP)

$$\min_{x,y} \quad \sum_{b \in B} y_b \tag{1}$$

$$\text{s.t.} \quad \sum_{p \in P} x_{pb} \leqslant c y_b, \qquad \forall b \in B \tag{2}$$

$$\sum_{b \in B} x_{pb} = 1, \qquad \forall p \in P \tag{3}$$

$$x_{pb} \in \{0,1\} \qquad \forall p \in P, b \in B \tag{4}$$

$$y_b \in \{0,1\} \qquad \forall b \in B \tag{5}$$

In this project, we propose heuristic methods based on Linear Programming relaxations (LP) and implement the techniques studied throughout the course to find an optimal solution.

## Tasks

Throughout all the different tasks of this project, you can limit the solving time to 10 minutes.

### Solve the problem with glpk

1. We first ask you to solve the linear relaxation of the problem with glpk in a python script `bp.py` using pyomo. Your script must contain a function `solve_bp(instance_name) : return (obj,x,y)` where `instance_name` is the name of an instance in the same folder (including the extension) and the return value is a tuple tuple containing the optimal value `obj` of the linear relaxation and an optimal solution `x` that is a list of lists where `x[p][b]` $= x_{pb}$ and `y` is a list where `y[b]` $= y_b$. Perform numerical experiments illustrating the solving times.

2. Propose a heuristic method that repairs the solution of the LP relaxation of the problem. Explain your choices and report the gap to LP relaxation.

### Implement a Branch & Bound procedure

1. In the same file `bp.py`, implement a function `branch_and_bound(instance_name, branching_scheme ,valid_inequalities,time_limit)` which performs a Branch & Bound procedure for BP where `instance_name` is the name of an instance in the same folder and `branching_scheme` is an integer representing a branching scheme you design. Parameter `valid_inequalities` is an integer that will be set to 0 for now. A maximum time limit for the branching procedure in seconds is provided with `time_limit`. The choice of variables to branch on is left to you as well as the order to treat the nodes of the Branch & Bound tree. Explain your choices on these last points without going in to theoretical justifications. At each node of the B & B tree, you can use your repair heuristic previously developed to find feasible solutions. Provide at least two branching scheme.

2. Represent the solving times of the B & B procedure by comparing different branching choices. Report for which instances an optimal solution is found. Compare the quality of the best solution found in the B & B with the repaired solution of the LP relaxation. For instances not solved to optimality, report the maximum gap to optimality.

### Add cutting planes in the B & B

1. Add general valid inequalities to each node of the B & B procedure. These inequalities must be added in function `branch_and_bound(instance_name, branching_scheme,valid_inequalities,time_limit)` if the parameter `valid_inequalities` is positive. Several different valid inequalities can be used in different configurations. As for the parameter `branching_scheme`, the value of parameter `valid_inequalities` indicates the configuration used to add valid inequalities. Propose at least two configurations adding valid inequalities. Each configuration choice must be explained.

2. Solve BP with the different configurations proposed and compare the solving times with a B & B procedure not using valid inequalities. For unsolved instances, report if the valid inequalities improve the best solution found or reduces the maximum gap to optimality.

## Goal of the project

The goal of this project is not specifically to solve large instances but to make you implement techniques which are contained in linear solver to better understand their challenges (such as the branching order in the B & B or which valid inequalities to consider).

The projects will be mostly evaluated on the proper use of the techniques seen throughout this course with proper justifications in your report.

## Pyomo

All you code must be written in a python file `bp.py` using the Pyomo library. Details on how to install Pyomo are provided here, on Modelling features here and Examples here. You are free to work with a concrete model or an abstract model.

The solver `glpk` is to be used and can be installed as detailed here for Linux, here for Mac Os X and here for Windows.

Questions over the functionalities can be asked on the UV. The use of classes is allowed but the functions asked for in this project must remain functions outside of any class.

## Instances

An extensive number of instances are provided. There is no need to solve all of them, try to solve to optimality the largest possible instances. The instances are provided in a format compatible with `DataPortal` in order to upload them directly in a Pyomo `AbstractModel`. You are also free to use a `ConcreteModel` and parse the instance files.

## General instructions

The project must be made by groups of two. Please send a mail for your group containing the names and ids of the students to jdeboeck@ulb.ac.be by March 18$^{\text{th}}$. If you do not find a partner, please contact me before the deadline. All students who have not taken contact by March 18$^{\text{th}}$ are considered as not doing the project.

The deadline for the project is May 13$^{\text{rd}}$ at 2 pm. For each group send a `zip` file containing your python script and the report in `pdf` format at Jerome.De.Boeck@ulb.be. The report should be no more than 7 pages long.