Combinatorial Optimization
CoContest Semester Project Assignment: **TurkeyBox™**

Industrial Informatics Department
Czech Technical University in Prague
https://industrialinformatics.fel.cvut.cz/

March 1, 2025

Abstract

This document introduces the assignment for the CoContest semester project.

1 Motivational Example

Theodor Krocan operates a successful business delivering orders (without any questions) to its parcel box at the Krůtí Hora. Since the parcel box is unmanned, has limited capacity, and the shipments come only once a month, Theodor needs to reason about which customers' orders he will fulfill to maximize his profit. Namely:

- Parcel box consists of a list of lockers of different sizes, each with its own security code.
- Each customer orders a list of items. Multiple items can be stored inside a single locker, but they have to fit given the locker's size.
- One locker can contain only items of a single customer.[1]
- Each customer can be assigned multiple lockers.

The customer pays a partial price for each item delivered. However, if Theodor is able to fulfill their entire order, the customer will provide an additional bonus/bribe to incentivize such priority.

This problem will be solved in two phases. In the first phase (**optimal**), all the lockers have the same width, and all the items have the same width as well. Thus, it is only necessary to determine to which locker the item is assigned, while the sum of the item's heights cannot exceed the locker's height.

In the second phase (**threshold**, **ranking**), the lockers are rectangular with arbitrary width and height, and so are the items. Thus, not only does Theodor need to determine to which locker he will put the item, but he also needs to consider whether the assigned items will fit not only the height of the locker but also its width. Also, he must determine the position of each item to ensure they fit into the locker (no overlap).

2 Formal Problem Statement

You are given list of M lockers, $(B_1, \ldots, B_M)$ and list of N customers $(C_1, \ldots, C_N)$. Each customer i is associated with their bonus pay P_i and list of n_i items $(O^i_1, \ldots, O^i_k, \ldots, O^i_{n_i})$. Each item's

[1] Otherwise, rogue Pilsnerians would steal each others' stuff

partial price is p^i_k. Let $\pi^i_k \in \{0, 1, \ldots, M\}$ be assignment of item O^i_k to one of the lockers (or none, when $\pi^i_k = 0$). All parameters are non-negative integers.

The task is to determine the assignment of subset of the items to lockers π^i_k (with 0 means unassigned) so Theodor maximizes:

$$A = \sum_{i=1}^{N} \sum_{k:\pi^i_k \neq 0} p^i_k + \sum_{i=1}^{N} P_i \cdot [\![\text{all items of } C_i \text{ are delivered}]\!]$$

2.1 Optimal

Each locker B_m is associated with its height H_m and each item O^i_k with its height h^i_k. A feasible assignment needs to satisfy:

- Single customer per locker

$$(\pi^i_k \neq \pi^j_l) \vee (\pi^i_k \cdot \pi^j_l = 0) \quad \forall i, j \in \{1, \ldots, N\} : i \neq j, \forall k \in \{1, \ldots, n_i\}, \forall l \in \{1, \ldots, n_j\}$$

- Locker is not overfilled

$$\sum_{\pi^i_k : \pi^i_k = m} h^i_k \leq H_m \quad \forall m \in \{1, \ldots, M\}$$

2.2 Threshold and Ranking

Each rectangular locker B_m is associated not only with height H_m but also with width W_m, and similarly each rectangular item O^i_k with its height h^i_k and width w^i_k. Your solutions have to determine the position of each item O^i_k's bottom-left corner x^i_k (width axis), y^i_k (height axis) so that the items do not overlap. Since the items **can be rotated** by 90 degrees in your solution (width essentially becomes height and vice versa), each item is also associated with flag r^i_k (that you will set to 1 if rotated, 0 if not). A feasible assignment needs to satisfy:

- Single customer per locker

$$(\pi^i_k \neq \pi^j_l) \vee (\pi^i_k \cdot \pi^j_l = 0) \quad \forall i, j \in \{1, \ldots, N\} : i \neq j, \forall k \in \{1, \ldots, n_i\}, \forall l \in \{1, \ldots, n_j\}$$

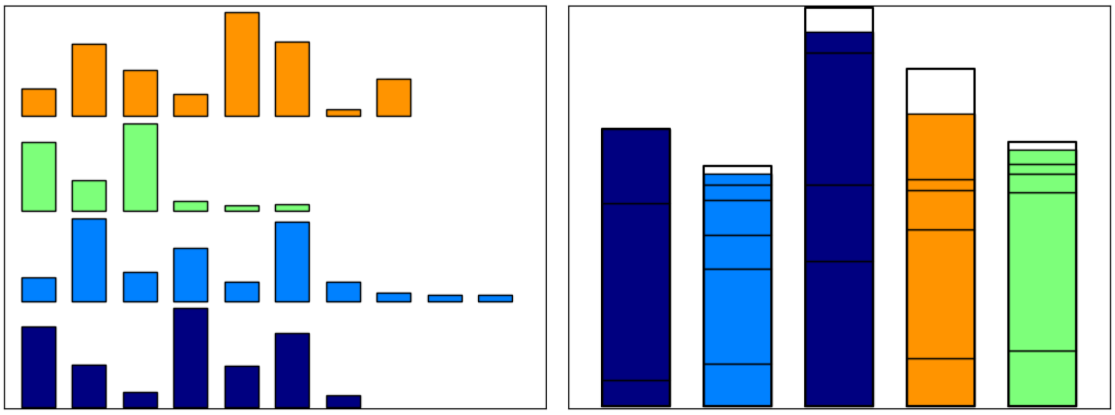
- Locker is not overfilled (including rotation, when the item's dimensions are swapped)

$$(\pi^i_k = m \ \wedge \ m > 0 \ \wedge \ r^i_k = 0) \implies (0 \leq x^i_k \leq W_m - w^i_k \ \wedge \ 0 \leq y^i_k \leq H_m - h^i_k)$$
$$\forall i \in \{1, \ldots, N\}, \forall k \{1, \ldots, n_i\}$$

$$(\pi^i_k = m \ \wedge \ m > 0 \ \wedge \ r^i_k = 1) \implies (0 \leq x^i_k \leq W_m - h^i_k \ \wedge \ 0 \leq y^i_k \leq H_m - w^i_k)$$
$$\forall i \in \{1, \ldots, N\}, \forall k \{1, \ldots, n_i\}$$

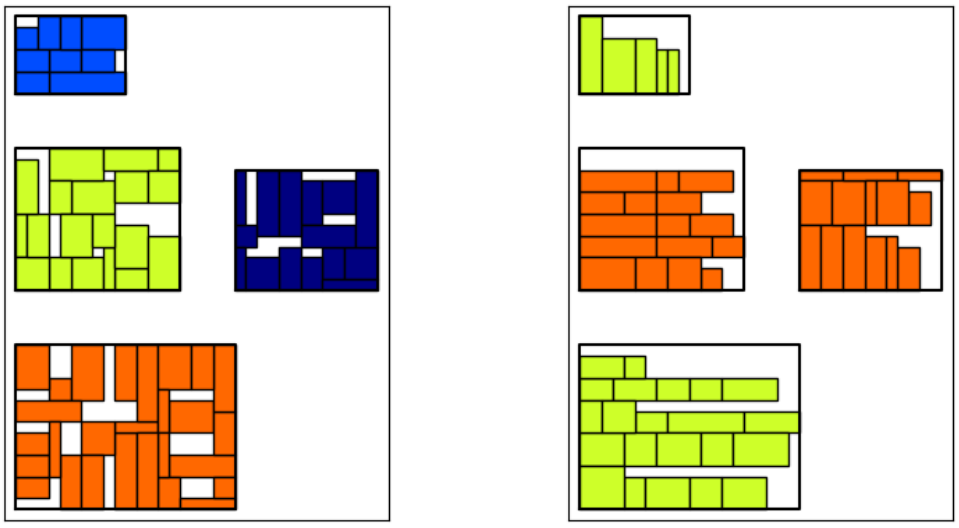
- Items in the same locker do not overlap (in 2D geometric sense)

$$(\pi^i_k = \pi^i_l \ \wedge \ \pi^i_k, \pi^i_l > 0) \implies \texttt{noOverlap}(O^i_k, O^i_l) \quad \forall i \in \{1, \ldots, N\}, \forall k, l \{\in 1, \ldots, n_i\} : k \neq l$$



(a) Instance with 4 customers (b) Optimal solution using 5 lockers

Figure 1: Example for 1D **optimal** phase



(a) Solution with objective value = 2303 (b) Solution with objective value = 2734

Figure 2: Solution examples for 2D **threshold and ranking** phase

3 Rules

If you decide to choose CoContest as your semestral project, then you are expected to implement a correct solver for TurkeyBox™ problem. The implementation will be submitted to BRUTE https://cw.felk.cvut.cz/brute/ where it will be automatically evaluated (the number of submissions is not limited). The grading is a combination of the ability to find good solutions and the achieved rank relative to other students (w.r.t. the objective function). Therefore, you can acquire a small number of points even if your solver is not very efficient relative to other students.

In BRUTE, you will find 3 tasks related to the contest. Each task has specific instances, rules, and grading. The contest is split into different tasks to avoid re-evaluation of the instances (which is time-consuming) and so that you can implement a specific solver for each task.

1. `SP_CC_O`: you have to formulate ILP model using Gurobi for the first phase (i.e., one-dimensional) problem. If your solver solves all the instances in this task optimally, then you will get **3 points** for this task. If the solver returns a suboptimal solution for **ANY** instance in this phase, then the evaluation of your solver is stopped, and you will get 0 points in this task.

2. `SP_CC_T`: the goal is to find the best possible feasible solution within the specified time limit for the second phase (i.e., two-dimensional) problem. The optimal solutions are not required, and you are encouraged to implement clever heuristics solving these instances. For each instance in this task, you will obtain some fraction of the point if your solution's cost is not worse than our threshold (**4 points** at max).

3. `SP_CC_R`: similarly as in `SP_CC_T`, in this task, we are also interested in finding the best possible feasible solution within the specified time limit. However, your solver's evaluation will depend on how good your solver is relative to other students' solvers, i.e., the number of points obtained will depend on your rank (**4 points** at max).

Some general contest rules also apply:

1. Usage of the single-purpose problem-specific solvers is prohibited (i.e., a MILP solver is allowed, but somebody's else code for solving "TurkeyBox™" like problem is not).

2. Every participant is required to write their own code. However, sharing ideas and discussing the problem is encouraged.

4 Input and Output Format

In `SP_CC_O`, your solver will be called as

```
$ ./your-solver PATH_INPUT_FILE PATH_OUTPUT_FILE
```

whereas in `SP_CC_T` and `SP_CC_R` we include a time limit

```
$ ./your-solver PATH_INPUT_FILE PATH_OUTPUT_FILE TIME_LIMIT
```

- `PATH_INPUT_FILE` and `PATH_OUTPUT_FILE`: similarly as in homeworks, these parameters represent the path to the input and output files, respectively (see below for a description of the file formats).

- `TIME_LIMIT`: a float representing the time limit in seconds given to your solver. Your solver will be killed after the time limit is reached, and you will be awarded 0 points. Hence, your solver's output is considered only if your program exits with status code 0 before it times out.

4.1 Optimal

The input file has the following form (we use one space as a separator between values on one line):

$$\begin{array}{llllllll}
M & N \\
n_1 & \dots & n_N \\
H_1 & H_2 & \dots & H_M \\
P_1 & p^1_1 & h^1_1 & p^1_2 & h^1_2 & \dots & p^1_{n_1} & h^1_{n_1} \\
\vdots \\
P_N & p^N_1 & h^N_1 & p^N_2 & h^N_2 & \dots & p^N_{n_N} & h^N_{n_N}
\end{array}$$

The output file has the following format:

$$\begin{array}{l}
obj \\
\pi^1_1 \\
\vdots \\
\pi^1_{n_1} \\
\pi^2_1 \\
\vdots \\
\pi^N_{n_N}
\end{array}$$

where *obj* is the total profit. All values are integer.

4.2 Threshold and Ranking

The input file has the following form (we use one space as a separator between values on one line):

$$\begin{array}{llllllll}
M & N \\
n_1 & \dots & n_N \\
W_1 & H_1 & W_2 & H_2 & \dots & W_M & H_M \\
P_1 & p^1_1 & w^1_1 & h^1_1 & \dots & p^1_{n_1} & w^1_{n_i} & h^1_{n_1} \\
\vdots \\
P_N & p^N_1 & w^N_1 & h^N_1 & \dots & p^N_{n_N} & w^N_{n_N} & h^N_{n_N}
\end{array}$$

The output file has the following format:

$$\begin{array}{llll}
obj \\
\pi^1_1 & x^1_1 & y^1_1 & r^1_1 \\
\vdots \\
\pi^1_{n_1} & x^1_{n_1} & y^1_{n_1} & r^1_{n_1} \\
\pi^2_1 & x^2_1 & y^2_1 & r^2_1 \\
\vdots \\
\pi^N_{n_N} & x^N_{n_N} & y^N_{n_N} & r^N_{n_N}
\end{array}$$

where *obj* is the total profit. Note that when the item is not assigned ($\pi^i_j = 0$), values x, y, r may be arbitrary. Nevertheless, all values are integer.

Optimal Example

This example corresponds to the motivation example Fig 1.

Input:

```
5 4
7 10 6 8
105 91 151 128 100
82 26 55 21 29 2 10 24 67 18 28 30 50 6 8
32 22 16 23 56 5 20 15 36 19 13 4 54 13 13 14 6 6 4 21 4
57 8 47 19 21 30 60 20 7 3 4 15 5
53 24 18 18 49 3 31 9 15 20 70 17 50 25 4 26 25
```

Output:

```
502
3
3
1
1
1
3
3
2
0
0
2
2
0
2
2
0
2
0
5
5
5
5
5
4
4
0
4
0
0
4
4
```

Threshold / Ranking Example

This example corresponds to the motivation example Fig 2 (for clarity, long lines are separated).

Input:

```
4 5
27 21 27 27 30
20 15 15 13 13 11 10 7
69 15 4 1 8 7 2 43 2 4 19 2 3 23 3 4 44 4 2 32 3 3 36 3 3 34 6 2 32 2 6 19
    3 5 45 2 2 29 3 2 40 1 5 22 4 3 40 1 4 41 4 3 25 4 3 45 3 3 31 6 2 20
    3 4 18 5 2 38 2 7 32 2 6 48 2 6 23 5 1 15 3 4
64 16 5 2 8 6 2 27 4 4 48 4 2 48 4 3 34 2 2 23 4 4 23 2 7 38 2 7 41 3 3 49
    5 3 25 4 3 9 6 2 46 3 2 14 3 2 48 3 2 35 3 2 35 6 2 15 3 2 42 3 2 20 2 6
93 15 4 3 25 3 5 46 3 3 23 4 3 9 2 3 23 3 3 18 2 2 15 6 2 10 4 3 49 2 4 12 3
    3 36 2 2 11 7 2 23 3 4 9 4 2 36 1 5 49 2 7 23 3 3 32 3 3 21 2 4 17 1 4
    21 3 3 37 3 3 34 3 3 17 4 3 36 2 3 18 1 5
73 28 2 3 30 3 2 30 4 3 42 3 3 11 1 4 18 4 3 17 7 2 8 2 7 40 5 2 30 1 4 29
    4 3 43 3 5 46 3 2 12 4 2 24 3 3 39 3 2 41 4 3 47 3 3 27 3 2 38 5 2 40
    5 3 26 2 3 15 4 4 45 4 2 39 2 2 42 2 5 28 3 3
944 33 2 2 36 7 2 43 5 2 46 3 2 30 5 3 28 4 1 41 2 5 21 2 4 10 7 2 30 3 4
    36 2 6 38 2 6 24 3 2 24 3 4 16 5 1 33 2 3 14 1 4 8 4 2 10 4 1 42 3 3
    20 2 6 24 4 2 48 3 2 48 3 3 42 3 4 27 1 5 26 4 2 33 7 2 17 2 2 34 5 2
```