

## TP2 – How data structures affect models

### 1 An illustration with the admissible cells problem

Given a grid rectangular grid  $T^{m \times n}$  of boxes called *cells*. Some of these cells are *admissible*, the others are *non-admissible*. Also are given  $m + n$  non-negative integers  $l_0 \dots l_{m-1}$ ,  $c_0 \dots c_{n-1}$ . The goal is to fill in the admissible cells with integers such as:

- The sum of the numbers allocated on the admissible cells on row  $i$  should be less or equal than  $l_i$ ;
- The sum of the numbers allocated on the admissible cells on column  $j$  should be less or equal than  $c_j$ ;
- The total sum of all these numbers should be maximum.

**Example:** Let:

$$\begin{array}{ccccc} (m = 4) & l_0 = 9 & l_1 = 10 & l_2 = 15 & l_3 = 2 \\ (n = 5) & c_0 = 7 & c_1 = 5 & c_2 = 9 & c_3 = 4 & c_4 = 8 \end{array}$$

and the set of admissible cells is:

$$A = \{(0, 0), (0, 1), (1, 0), (1, 2), (1, 3), (2, 1), (2, 4), (3, 2), (3, 4)\}$$

where the first index corresponds to the row, second index to the column number.

	$C_0$	$C_1$	$C_2$	$C_3$	$C_4$
$R_0$					
$R_1$					
$R_2$					
$R_3$					

**Figure 1:** Only the blank cells can be filled with a value.  $L_i$  is the limit value of the sum of cells in row  $i$ ,  $C_j$  is the limit value of the sum of cells in column  $j$

## 2 Easy example then more complex ones

### 2.1 Data files reader API

In TP2 folder you can find `tp2_read_data_files.py`. There are functions to extract admissible cells from files in `data` directory, and a function to parse command line's argument. It is not necessary to understand how they work, just how to use them.

Especially, the function `get_adm_cells_data` lets you to iterate over it to get admissible cells data from each file. You can use it as following:

```
for adm_cells, row_limits, col_limits in get_adm_cells_data():
    # adm_cells: list of tuple (row_i, col_j)
    #   corresponding to admissible cells
    # row_limits: list of row limits (int)
    # col_limits: list of column limits (int)
    solve_admissible_cells(adm_cells, row_limits, col_limits)
```

In fact, `get_adm_cells_data` *yield* the three lists instead of returning them. It allows to iterate over the function in a stream way.

### 2.2 Let play!

#### Question 1.

Use the pre-structured Python3 program in `TP_BASE/TP0/base_model.py` and write the admissible cells model.

#### Question 2.

Run your program with:

```
(.venv_38) python3.8 your_prog.py
```

What is the solution?

In order to test more complex instances, you can run your program like this:

```
(.venv_38) python3.8 your_prog.py --all-data > tp2_all_results.log
```

#### Question 3.

Run the program for all the data and plot the CPU time according number of number of variables, and according the number of constraints, with the method of your choice.

**Tip:** you can use CTRL+F on `tp2_all_results.log` and search for `FILE` word in order to find very quickly statistics.