Palletizing with adjacency constrain

The objective is to implement an optimization model for the palletizing problem extended by **constraints**. The constraints should have a form of two lists:

1. a list of pairs of element which have to be adjacent in the arrangement
2. a list of pairs of element which cannot be adjacent in the arrangement

The ***benchmarks/data*** directory contains the files with the data for the problem. The data files are in the format:

```
3   5
1   4
2   6
4   7
10  5
3   6

0   1
5   4
3   2

1   2
```

The files must be divided into three sections, each section is separated by a blank line **\n**:

- The first section contains the dimensions of each element. The first number is the width and the second is the height.
- The second section contains the index of pairs of elements which have to be adjacent, these shapes start from index 0 (0 element (3,5), 1 element (1,4) and so on).
- The third section contains the index pairs of elements which cannot be adjacent.

Thus, the dimensions of elements will be: **[(3, 5), (1, 4), (2, 6), (4, 7), (10, 5), (3, 6)]**

the adjacency elements will be: **[(0, 1), (5, 4), (3, 2)]**

and the non-adjacency elements will be: **[(1, 2)]**

Note: the adjacency and non-adjacency elements are indexed from 0, where 0 is the first element in the list and so on.

Solution

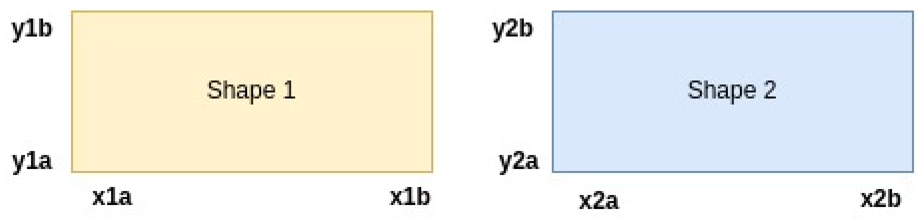
It is possible to solve this problem in two ways:

1. The basic one, without constraints
2. The extended one, with constraints

To solve this problem with constrains, several constraints were added to the model. Given two shapes with the initial and final coordinates like:

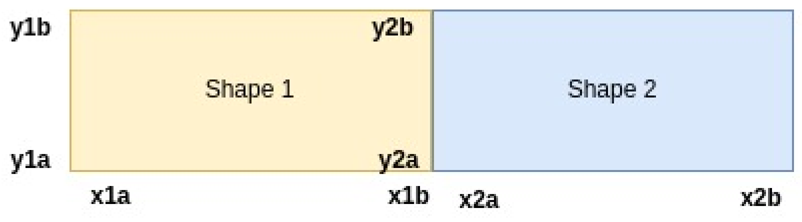
- shape1: [(x1a, y1a), (x1b, y1b)]
- shape2: [(x2a, y2a), (x2b, y2b)]

Adjacent shapes

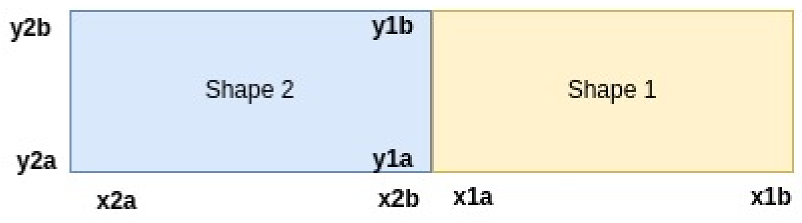


To these shapes be adjacent, one of the following conditions must be met:

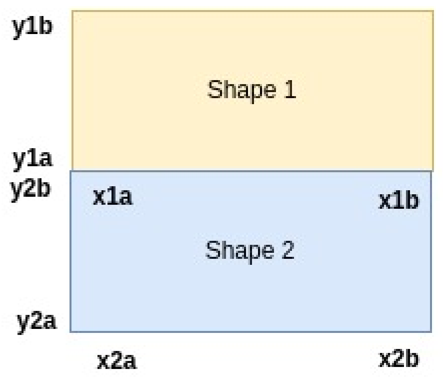
- x1b == x2a



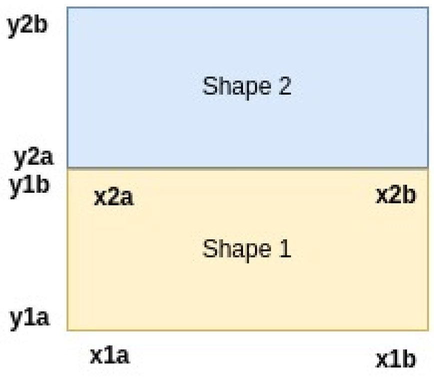
- x1a == x2b



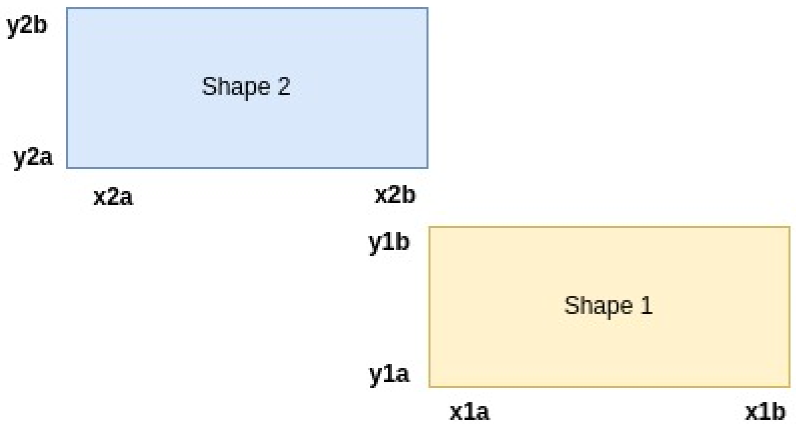
- y2b == y1a

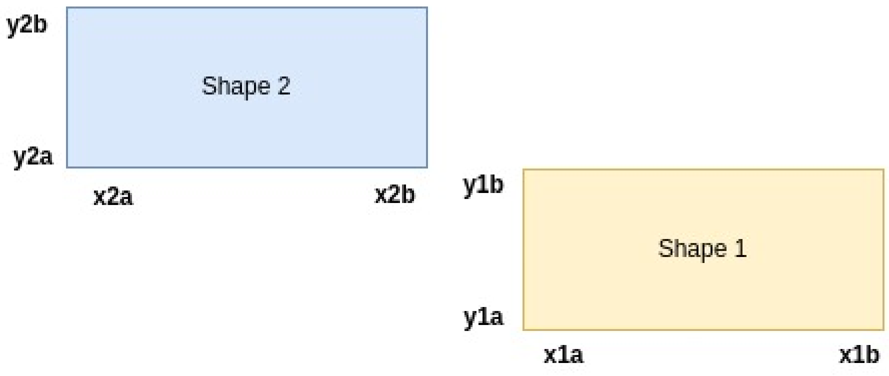


- y1b == y2a



however, this is not enought once the shapes may remain non-adjacent.





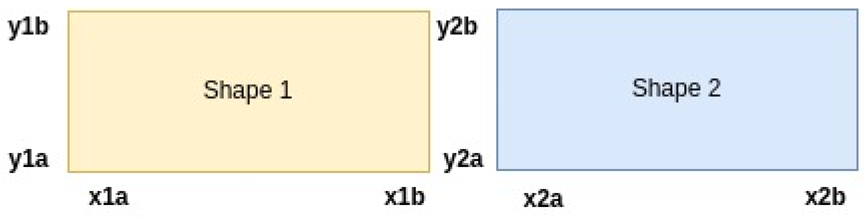
In these examples the second condition is met (**x1a==x2b**), but the shapes are non adjacent, in the first image and in the second image, the fourth condition is achieved. Thus, stricter conditions need to be added to the model:

- **x1b == x2a ^ y2b > y1a ^ y2a < y1b**
- **x1a == x2b ^ y2b > y1a ^ y2a < y1b**
- **y2b == y1a ^ x2b > x1a ^ x2a < x1b**
- **y1b == y2a ^ x2b > x1a ^ x2a < x1b**

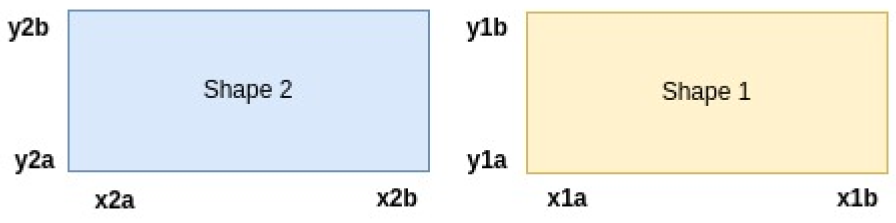
Non Adjacent shapes

To these shapes be non adjacent, one of the following conditions must be met:

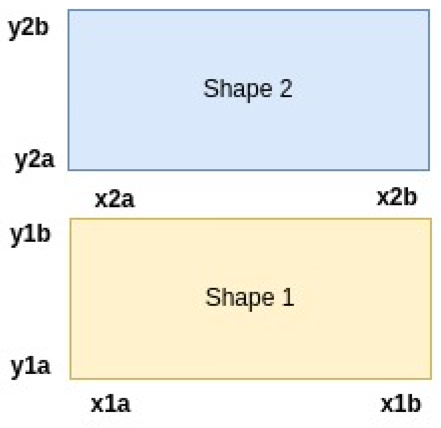
- **x1b < x2a**



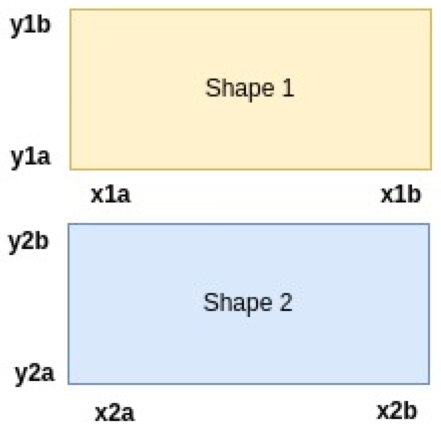
- **x2b < x1a**



- **y1b < y2a**



- **y2b < y1a**

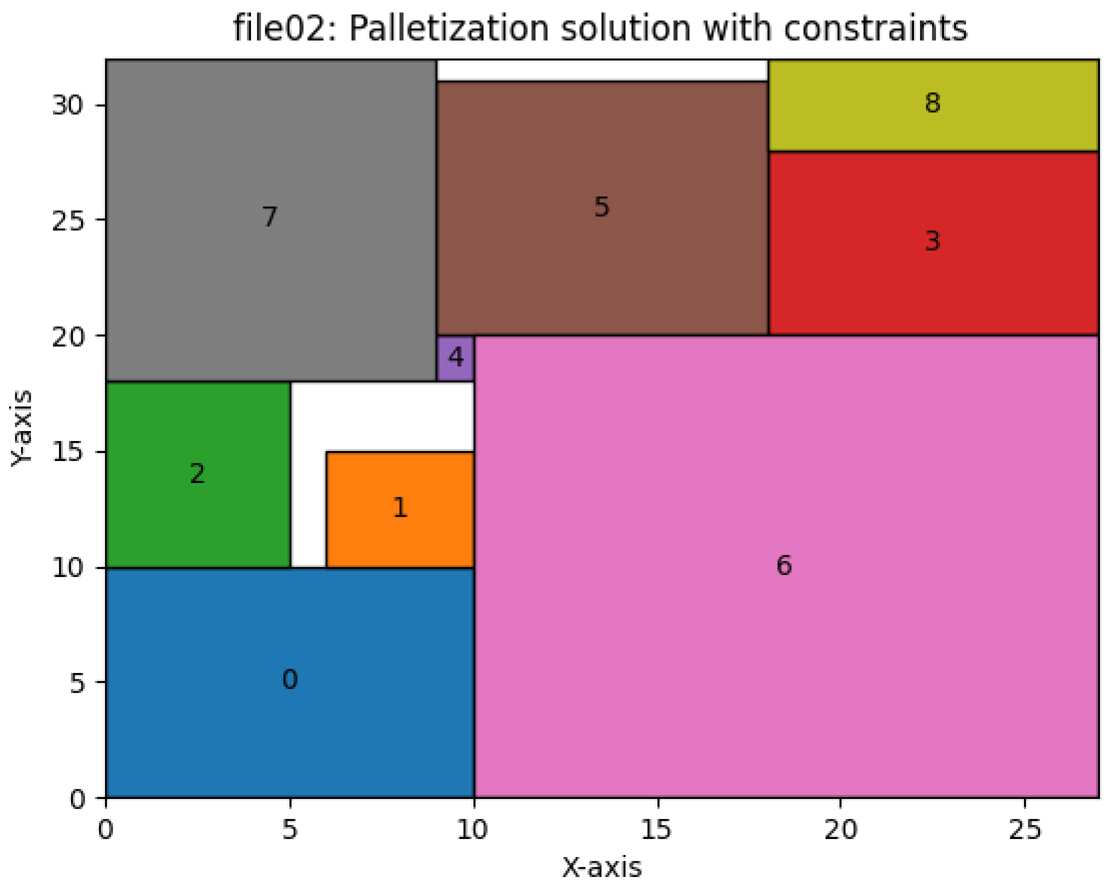


In this case the conditions are so strict that it is not necessary to add more conditions to the model.

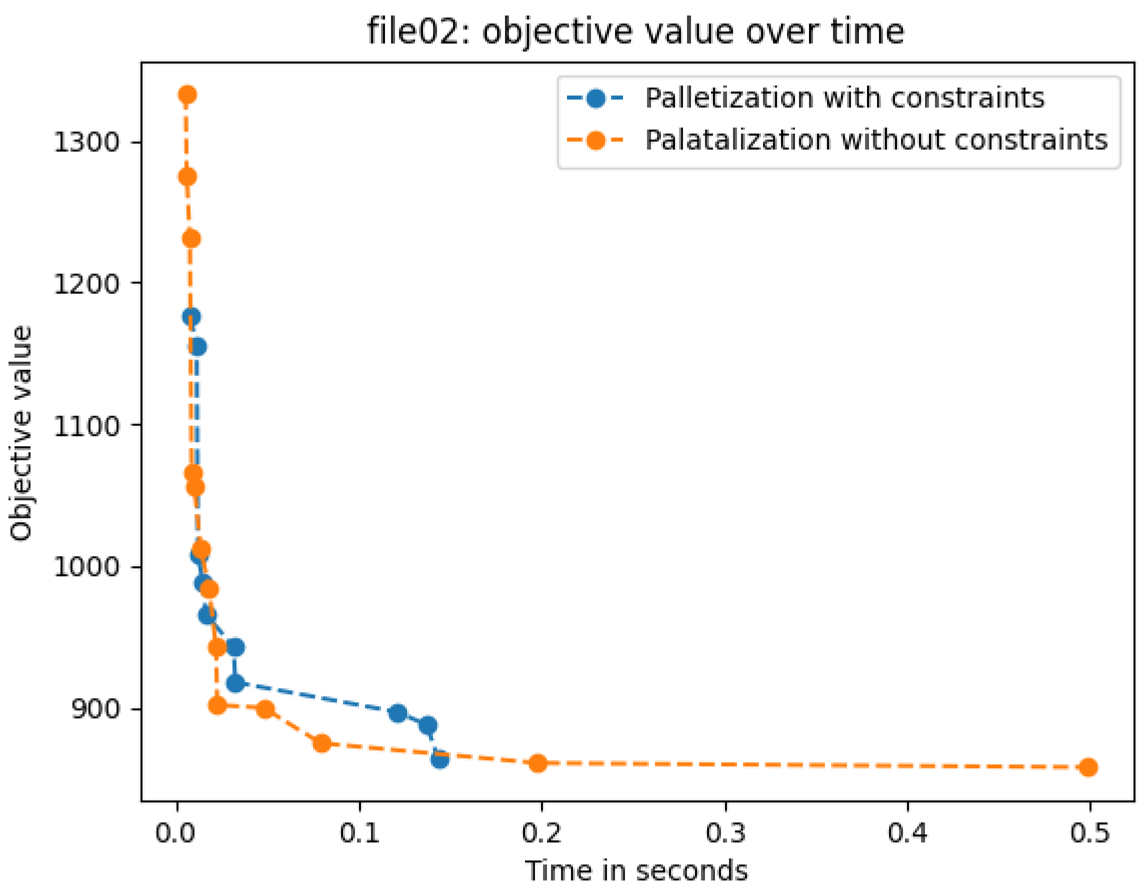
Results

The solutions are stores in the ***benchmarks/*** directory and inside the folder with the same name as the file used to run the program. Inside this folder, two directories are created, one for the solution without constraints and the other for the solution with constraints. Each directory contains two files:

- The first one contains the position of each element in the arrangement, the area of the arrangement, the max values of **X** and **Y** coordinates, the **area** and the **occupation rate**.
- The second one is an image of the arrangement.



Inside the directory with the same name as the file used to run the program, there is an image that shows the optimization progress over time for the solution **with** and **without** constraints.



How to run the program

It is possible to run the program with a file created by your own, you just have to put the correct path, to the file, like in the command below:

```
python3 palletizationWithConstraints.py -f benchmarks/data/file01.txt
```

If you don't want to create your own file, you can use one of the files in the ***benchmarks/data*** directory. The command above will run the program with the file ***file01.txt***.

Results with my files

	With Constraints		Without contraints	
File	**Area**	**Occupation (%)**	**Area**	**Occupation (%)**
file00.txt	532	97.6	527	98.5
file01.txt	130	97.7	130	97.7
file02.txt	864	96.6	858	97.3
file03.txt	15158	98	15000	99

file04.txt	5047	97	4998	98
file05.txt	1950	99.3	1943	99.7
file06.txt	560	97.1	550	98.9
file07.txt	4416	**97.9**	4450	**97.2**
file08.txt	59118	97.7	58680	98.5
file09.txt	110	100	110	100

Note
More results can be found inside benchmarks directory.

Author

Work done by João Farias
Politechnika Rzeszowska, June 2023.