Multiple-type, two-dimensional finite bin packing problem

This is a mini project for Fundamental of Optimization course of SOICT – HUST

TABLE OF CONTENTS

01

Introduction

You can enter a subtitle
here if you need it

Problem

There are K trucks 1,2,...,K for transporting N packages of 1, 2, ..., N. Trucks have a container size of Wk * Lk. Each package I has size wi * li.

Packages that are placed in same container must not overlap. Assume that the number K can be large, leading to a large number of trucks that are not being used. The cost of using truck k is ck.

Find a way to put these package in the trucks so that the total cost is minimal.

Exact solutions

CP and MIP models

02

Denotation

N_items is the number of items given

Item i has width $\boldsymbol{w_i}$ and height $\boldsymbol{h_i}$

N_bins is the number of bins given

Bin j has width $\boldsymbol{W_j}$, height $\boldsymbol{H_j}$ and cost $\boldsymbol{C_j}$

Decision variable

* $X_{ij} = 1$: item i packed in bin j

$$\Rightarrow \sum_{i=1}^{\text{N\_items}} X_{ij} \geq 1 \iff Z_j = 1\text{: bin j has been used}$$

* $R_i = 1$: item i rotated 90 degree

Item's Coordinate

l_i, r_i, b_i, t_i: left, right, bottom and top coordinates of item i

* First way to approach:

- if item i not rotated: $R_i = 0$

$$\Rightarrow \begin{cases} \mathrm{r}_i = l_i + w_i \\ \mathrm{t}_i = b_i + h_i \end{cases}$$

- if item i rotated: $R_i = 1$

$$\Rightarrow \begin{cases} \mathrm{r}_i = l_i + h_i \\ \mathrm{t}_i = b_i + w_i \end{cases}$$

* Another way to approach:

- if item i not rotated: $R_i = 0$

$$\Rightarrow \begin{cases} \mathrm{w}_i = w_i \\ \mathrm{h}_i = h_i \end{cases}$$

- if item i rotated: $R_i = 1$

$$\Rightarrow \begin{cases} \mathrm{w}_i = h_i \\ \mathrm{h}_i = w_i \end{cases}$$

CP model - Constraints

Each item has to be packed in exactly 1 bin

$$\sum_{j=1}^{\text{N\_bins}} X_{ij} = 1 \text{ for } i \text{ in N\_items}$$

No two items overlap

$$\text{if } X_{i_1 j} = X_{i_2 j} = 1$$

$$r_{i_1} \leq l_{i_2} \text{ or } r_{i_2} \leq l_{i_1} \text{ or } t_{i_1} \leq b_{i_2} \text{ or } t_{i_2} \leq b_{i_1}$$

Items cannot exceed the bin

$$\text{if } X_{ij} = 1$$

$$\Rightarrow \begin{cases} \mathrm{w}_i \leq r_i \leq \mathrm{W}_j \\ \mathrm{h}_i \leq t_i \leq \mathrm{H}_j \end{cases}$$

MIP model - Constraints

Each item has to be packed in exactly 1 bin

$$\sum_{j=1}^{\text{N\_bins}} X_{ij} = 1 \text{ for } i \text{ in N\_items}$$

No two items overlap

if $X_{i_1 j} = X_{i_2 j} = 1$

$$r_{i_1} \leq l_{i_2} \text{ or } r_{i_2} \leq l_{i_1} \text{ or } t_{i_1} \leq b_{i_2} \text{ or } t_{i_2} \leq b_{i_1}$$

$\Rightarrow$ **To MIP:**

$$\begin{cases} \mathrm{r}_{i_1} \leq l_{i_2} + \mathrm{M} * (1 - (r_{i_1} \leq l_{i_2})) \\ \mathrm{r}_{i_2} \leq l_{i_1} + \mathrm{M} * (1 - (r_{i_2} \leq l_{i_1})) \\ \mathrm{t}_{i_1} \leq b_{i_2} + \mathrm{M} * (1 - (t_{i_1} \leq b_{i_2})) \\ \mathrm{t}_{i_2} \leq b_{i_1} + \mathrm{M} * (1 - (t_{i_2} \leq b_{i_1})) \\ (r_{i_1} \leq l_{i_2}) + (r_{i_2} \leq l_{i_1}) + (t_{i_1} \leq b_{i_2}) + (t_{i_2} \leq b_{i_1}) + (1 - (\mathrm{X}_{i_1 j} == X_{i_2 j}) * \mathrm{M} \geq 1 \\ (r_{i_1} \leq l_{i_2}) + (r_{i_2} \leq l_{i_1}) + (t_{i_1} \leq b_{i_2}) + (t_{i_2} \leq b_{i_1}) \leq (\mathrm{X}_{i_1 j} == X_{i_2 j}) * \mathrm{M} \end{cases}$$

Items cannot exceed the bin

if $X_{ij} = 1$

$$\Rightarrow \begin{cases} \mathrm{w}_i \leq r_i \leq \mathrm{W}_j \\ \mathrm{h}_i \leq t_i \leq \mathrm{H}_j \end{cases}$$

$\Rightarrow$ **To MIP:**

$$\begin{cases} \mathrm{r}_i \leq (1 - \mathrm{X}_{ij}) * \mathrm{M} + W_j \\ \mathrm{t}_i \leq (1 - \mathrm{X}_{ij}) * \mathrm{M} + H_j \end{cases}$$

Heuristic

03

The heuristic algorithm

Sorting

Bins by “density” (cost/area), in descending order

Items by longer side in descending order

Destination Bin

Bin First Fit rule – pack the item into the bin with the lowest index (after the process of sorting bins from I); in other words, pack in the first bin that the item fits

Destination Free Rectangle of a Bin

Best Short Side Fit rule – choose a free rectangle where the shorter remainder side after insertion is minimized; in other words, minimize the length of the shorter leftover side, ties broken with best longer side (longer leftover side is minimized)

Packing process

Concept of free rectangles: A list of free rectangles represents the free space of the bin. In the Guillotine algorithm, these rectangles are pairwise disjoint

Heuristic algorithms: Combine two algorithms, including Guillotine and Maximal Rectangles

Guilotine

- Pack the item into the first free rectangle of the bin, which means the bin itself, starting with its bottom-left corner.

- For each insertion, split the initial free rectangle into smaller free rectangle(s), which are tracked in a list.

- Whenever a new item is inserted into the bin, choose a free rectangle (with rule II.2) and place the item into its bottom-left corner, then split the chosen rectangle using the Guillotine split rule to produce at most two new rectangles.

- Merge some free rectangles into larger ones if possible.

* Splitting rule: Best Short Side rule - split by horizontal axis if the free rectangle's width is less than its height; otherwise, split by vertical axis.

* Rectangle merging: if there exists a pair of neighboring rectangles F_i and F_j such that $F_i \cup F_j$ can be exactly represented by a single bigger rectangle, merge these two into one.

Maximal Rectangles

- Rather than choosing one of the two split axes like in the Guillotine algorithm, the Maximal Rectangles algorithm picks both split axes at the same time to ensure that the largest possible rectangular areas are present in the list of free rectangles.

- Because the free rectangles are no longer pairwise disjoint like in the Guillotine algorithm, any free rectangle that intersects the area occupied by the newly inserted item is split to remove the intersection.

- Delete every free rectangle which is fully overlapped by others in the list.

Statistics

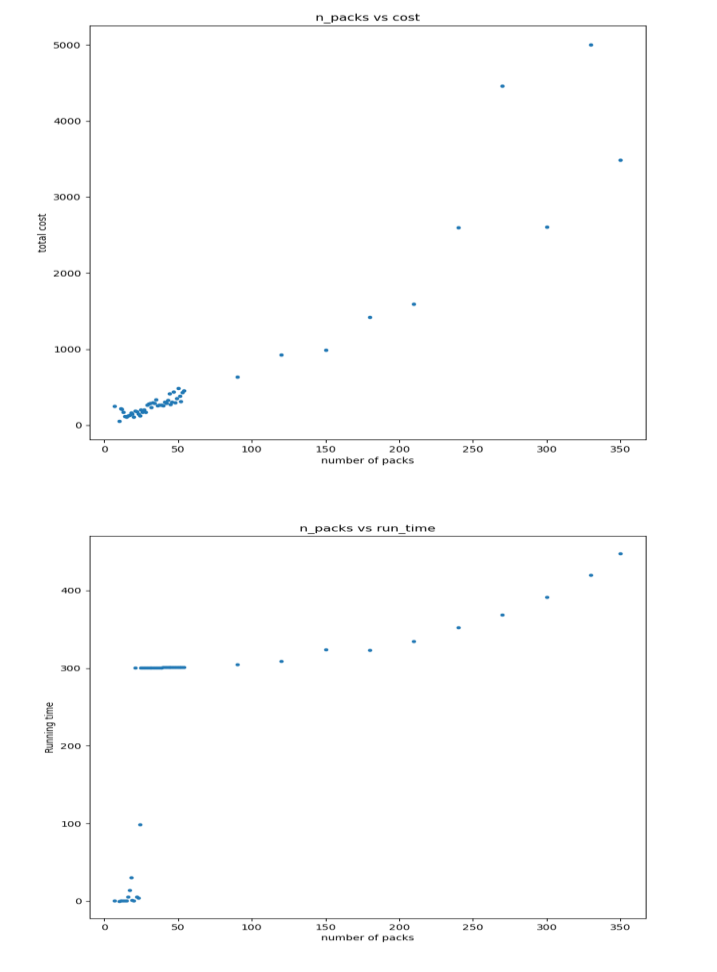
You can enter a subtitle here if you need it

04

CP model

Out of 58 tests, the model can solve 56, with a 300-second time limitation

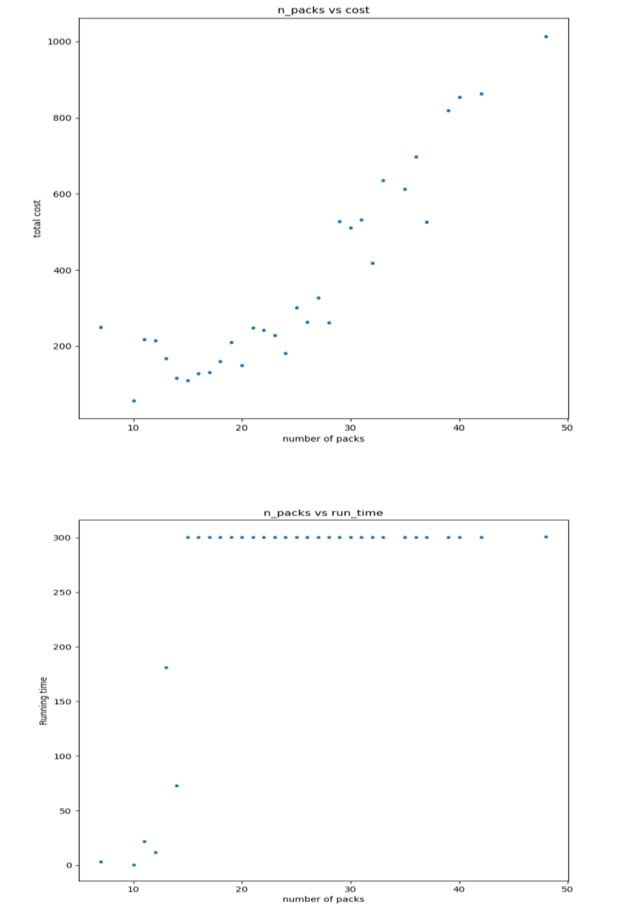
The objective values (costs) of solved instances are excellent in general



MIP model

Out of 58 tests, the model can solve 32, with a 300-second time limitation

The objective values (costs) of solved instances are quite good in general

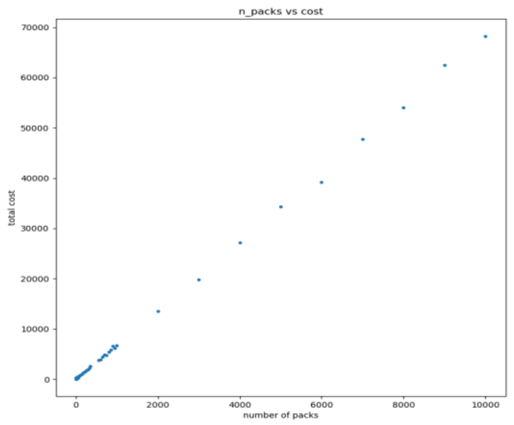


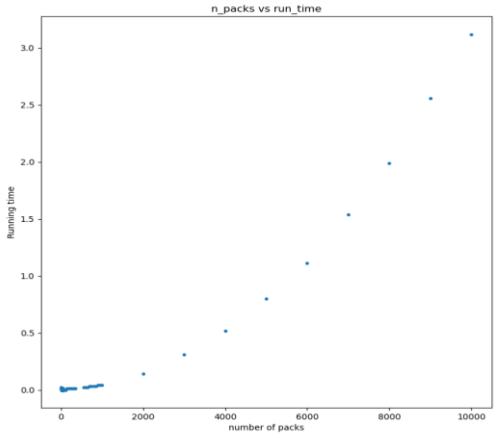
Heuristic

It can solve all 75 tests without a time limitation

The objective values (costs) of instances are good

The running time form an exponential curve with the number of rectangles (test size)





MEET THE TEAM

Phan Dinh Nhat

Nguyen Huu Duan

Chu Minh Ha

Do Quang Minh

CYBER SECURITY

Thanks for attention!

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik