Approximation Algorithms for 2-Dimensional Packing Problems

Klaus Jansen

Universität Kiel

Overview

- Introduction
- Algorithms NFDH, FFDH
- Algorithm by Kenyon, Remila
- Algorithm by Jansen, Solis-Oba
- Open problems

2D packing problem

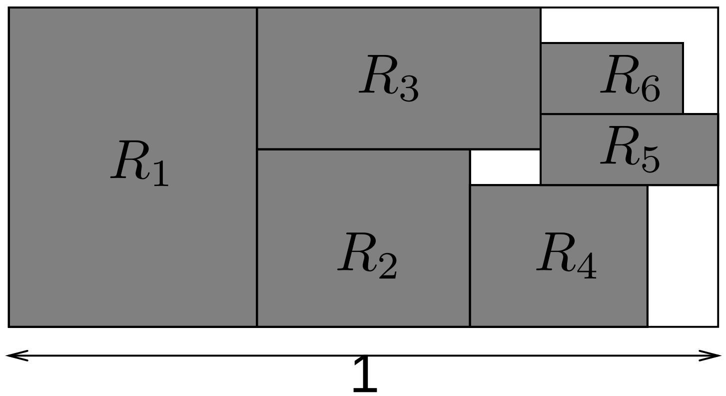
Given:

- n rectangles $R_j = (w_j, h_j)$ of width $w_j \leq 1$ and height $h_j \leq 1$,
- a strip of width 1 and unbounded height.

Problem: pack the n rectangles into the strip (without overlap and rotation) while minimizing the total height used.

Complexity: NP-hard (contains bin packing as special case).

Example



$R_1 = (7/20, 9/20)$, $R_2 = (3/10, 1/4)$, $R_3 = (2/5, 1/5)$, $R_4 = (1/4, 1/5)$, $R_5 = (1/4, 1/10)$, $R_6 = (1/5, 1/10)$. Here we have $OPT((R_1, \ldots, R_6)) = h(R_1) = 9/20$.

Application I

- **Cutting Stock**: cutting patterns of objects out of a large strip of material (like cloth, paper or steel).
- **Goal**: minimize the waste of material.

Application II

- **Scheduling**: compute a schedule for a set of jobs each requiring a certain number of resources (machines, processors or memory locations).
- **Goal**: minimum length of the schedule (minimum makespan).

Application III

- **VLSI Design**: placement of modules on a chip.
- **Goal**: minimum area of the chip.

Approximation algorithms

for an optimization problem are methods which for each instance L of the problem compute efficiently a **feasible solution** with provable performance guarantee.

We compare

$A(L)$ the height computed by algorithm A.
$OPT(L)$ the minimum height among all solutions.

Absolute performance ratio

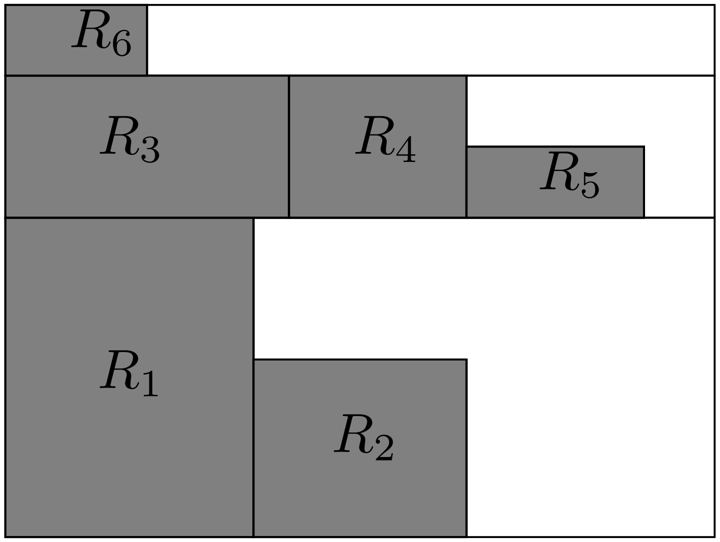
Worst Case:

For all instances L we have:

$$A(L) \leq aOPT(L)$$

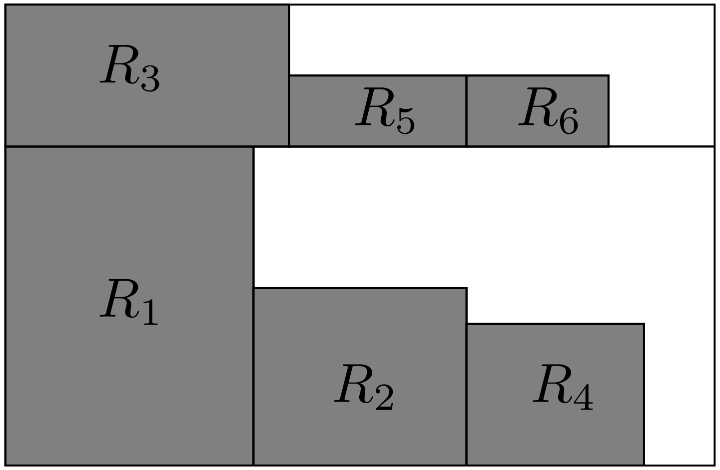
Goal: $a \geq 1$ should be close to 1.

Algorithm I: NFDH



Here we have $NFDH(L) = 3/4$.

Algorithm II: FFDH



Here we have $FFDH(L) = 13/20$.

Absolute performance ratio

$$A(L) \leq a\,OPT(L) \qquad \text{for all } L$$

3 NFDH **(Coffman et al.)**

2.7 FFDH **(Coffman et al.)**

2.5 **(Sleator)**

2 **(Schiermeyer, Steinberg)**

Asymptotic performance ratio

$$A(L) \leq a\,OPT(L) + b \qquad \text{for all } L$$

Goal: Ratio a close to 1.

An **AFPTAS** is a family of approximation algorithms $\{A_\epsilon | \epsilon > 0\}$ where $A_\epsilon(L) \leq (1+\epsilon)OPT(L) + b$ (and b does not depend on $OPT(L)$).

Asymptotic performance ratio

$$A(L) \leq a\,OPT(L) + b \qquad \text{for all } L$$

2	NFDH **(Coffman et al.)**
1.7	FFDH **(Coffman et al.)**
$4/3$	**(Golan)**
$5/4$	**(Baker et al.)**
$1 + \epsilon$	AFPTAS **(Kenyon, Remila)**

Algorithm by Kenyon, Remila

Theorem: **(Kenyon, Remila, FOCS 1996)**

There is an algorithm A which, given a list L of n rectangles and a positive number ϵ, produces a packing of L into a strip of width 1 and height

$$A(L) \leq (1+\epsilon)OPT(L) + 4/\epsilon^2.$$

The running time of A is polynomial in n and $1/\epsilon$.

Main ideas of the algorithm

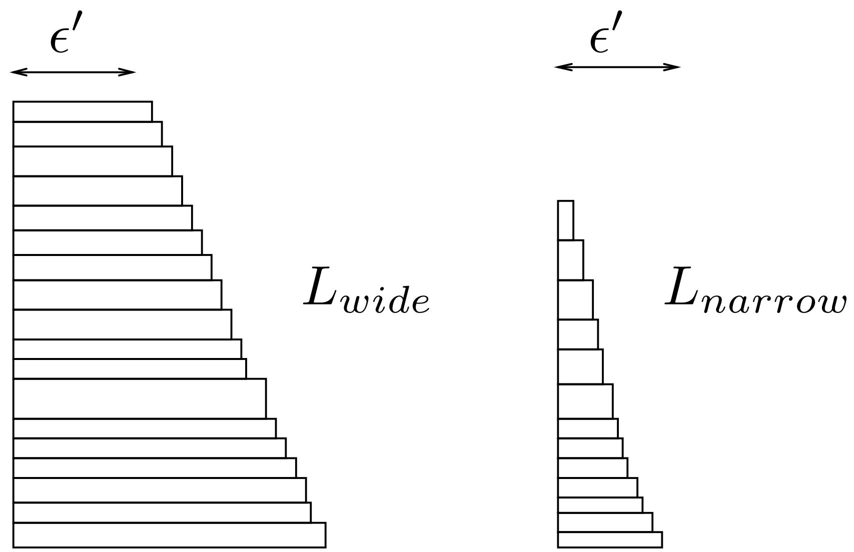
(a) partition the list L into narrow and wide rectangles,

(b) round the wide rectangles to obtain a constant number of widths,

(c) solve a linear program to pack the wide rectangles,

(d) use a modified version of NFDH to pack the narrow rectangles.

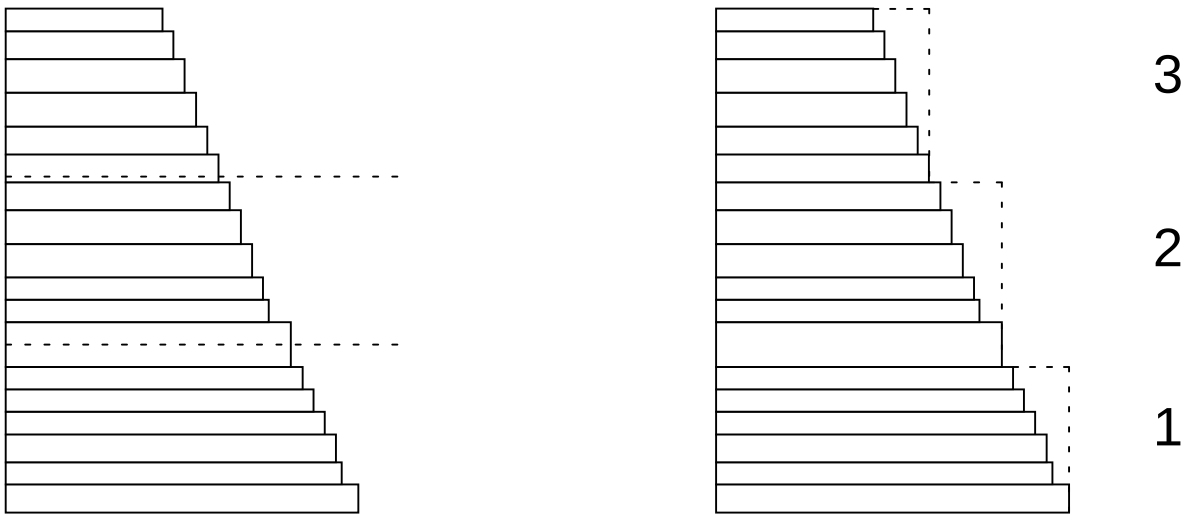
Partition of L into wide and narrow rectangles



$$L_{wide} = \{(x, y) \in L \mid x > \epsilon'\}$$

$$L_{narrow} = \{(x, y) \in L \mid x \leq \epsilon'\}$$

Rounding of the wide rectangles



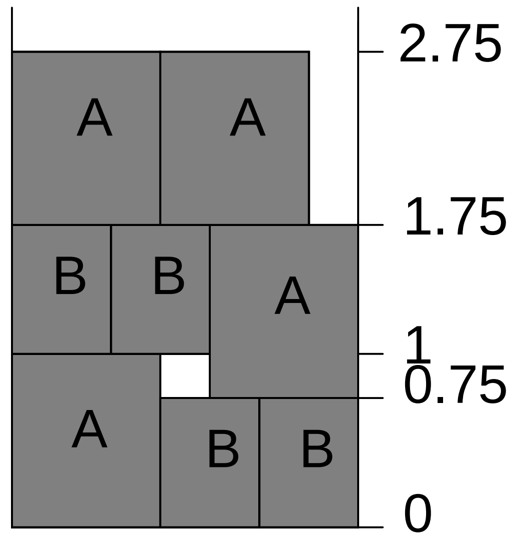
Round up each rectangle in group i to the widest rectangle in group i.

This leaves a constant number $m' \leq m = 4/\epsilon^2$ of widths.

Configurations

- a configuration C is a **multiset** $\{\alpha_{C,1} : w'_1, \ldots, \alpha_{C,m'} : w'_{m'}\}$ of widths with total sum $\sum_{i=1} \alpha_{C,i} w'_i \leq 1$,
- $\alpha_{C,i}$ denotes the number of occurrences of width w'_i in configuration C.

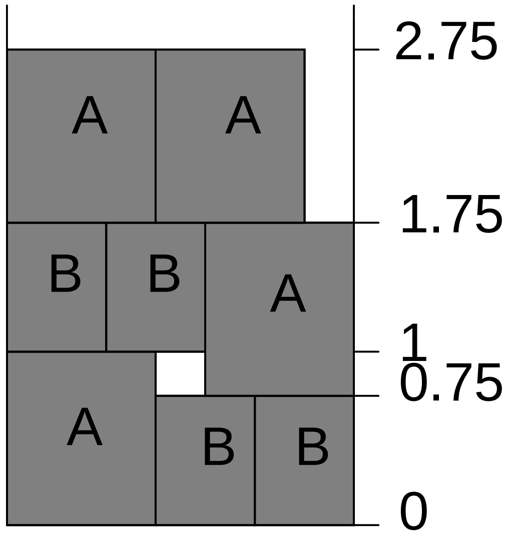
Example



Type A: 4 rectangles with width $3/7$ and height 1,

Type B: 4 rectangles with width $2/7$ and height $3/4$.

Configurations



1. configuration with 2 type A rectangles:

$C_1 = \{2 : 3/7, 0 : 2/7\}$.

2. configuration with 1 type A and 2 type B rectangles:

$C_2 = \{1 : 3/7, 2 : 2/7\}$.

Linear program

Use for each configuration C a **positive variable** $x_C \geq 0$ that represents the total height of configuration C in the solution.

Objective function: $\sum_C x_C$ is the total height of the packing.

Important inequality

β_i the sum of the heights of all rectangles with width w_i'.

$\alpha_{C,i}$ the number of occurrences of w_i' in configuration C.

Inequality: $\sum_C \alpha_{C,i} x_C \geq \beta_i$ for $i = 1, \ldots, m'$.

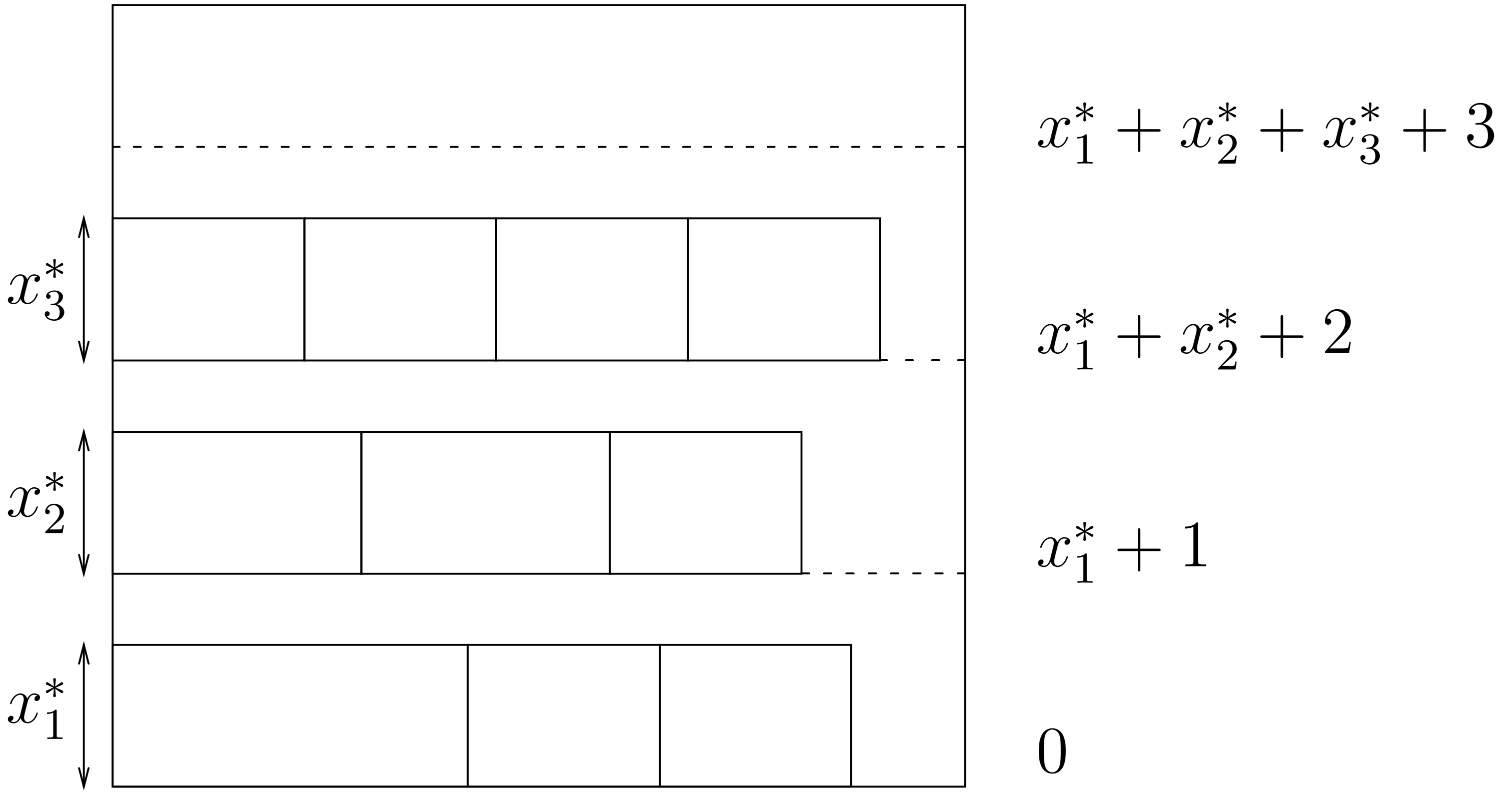
Idea: The configurations must reserve **enough space** for the rectangles with width w_i'.

Linear program (LP)

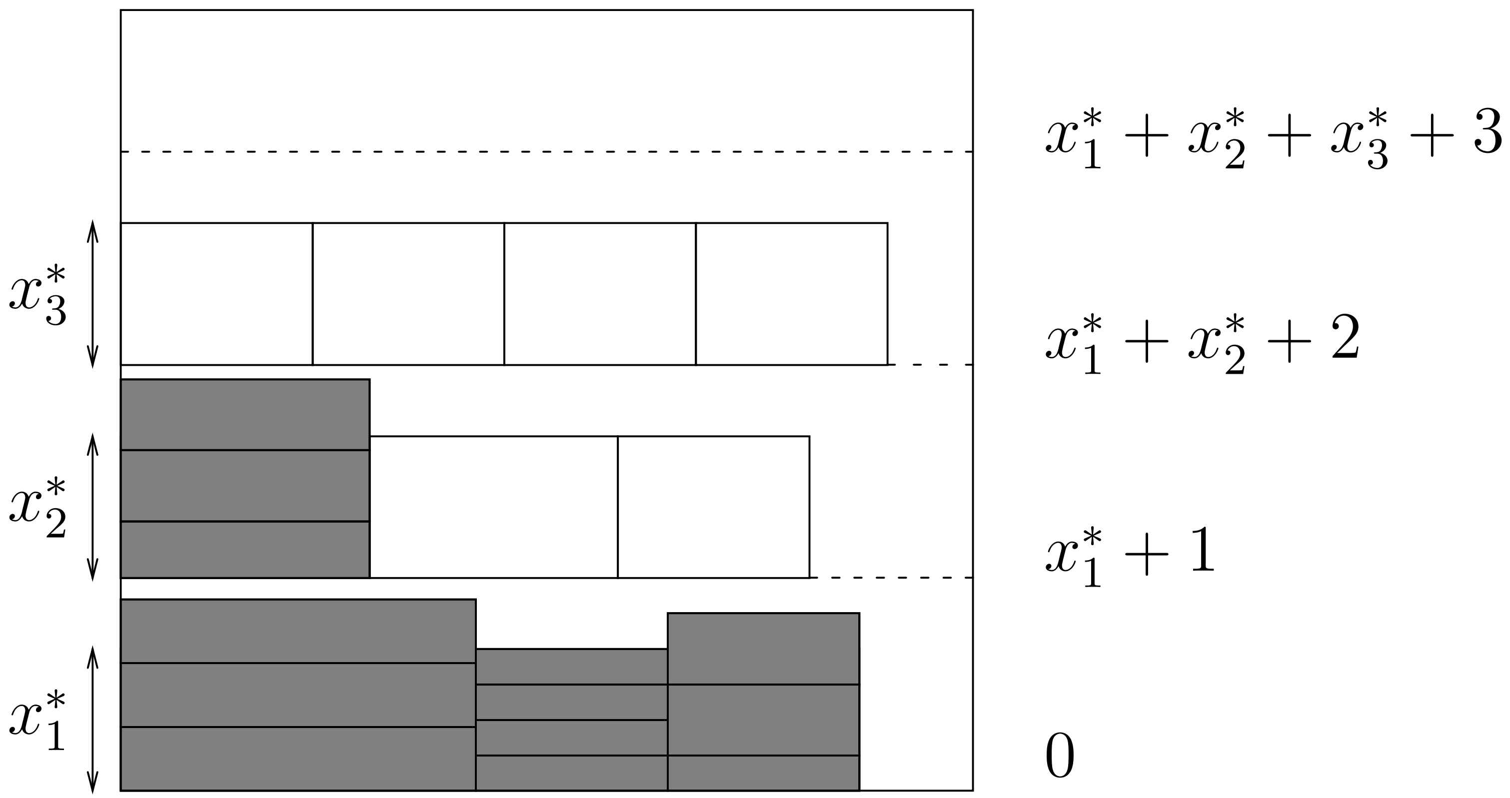
$$
\begin{array}{lll}
\min & \sum_C x_C & \\
\text{such that} & \sum_C \alpha_{C,i} x_C \geq \beta_i \quad i = 1, \ldots, m' & (1) \\
& x_C \geq 0 &
\end{array}
$$

Notice: this is a **relaxation** of the $2D$ packing problem (implicitly we allow to cut rectangles into pieces).

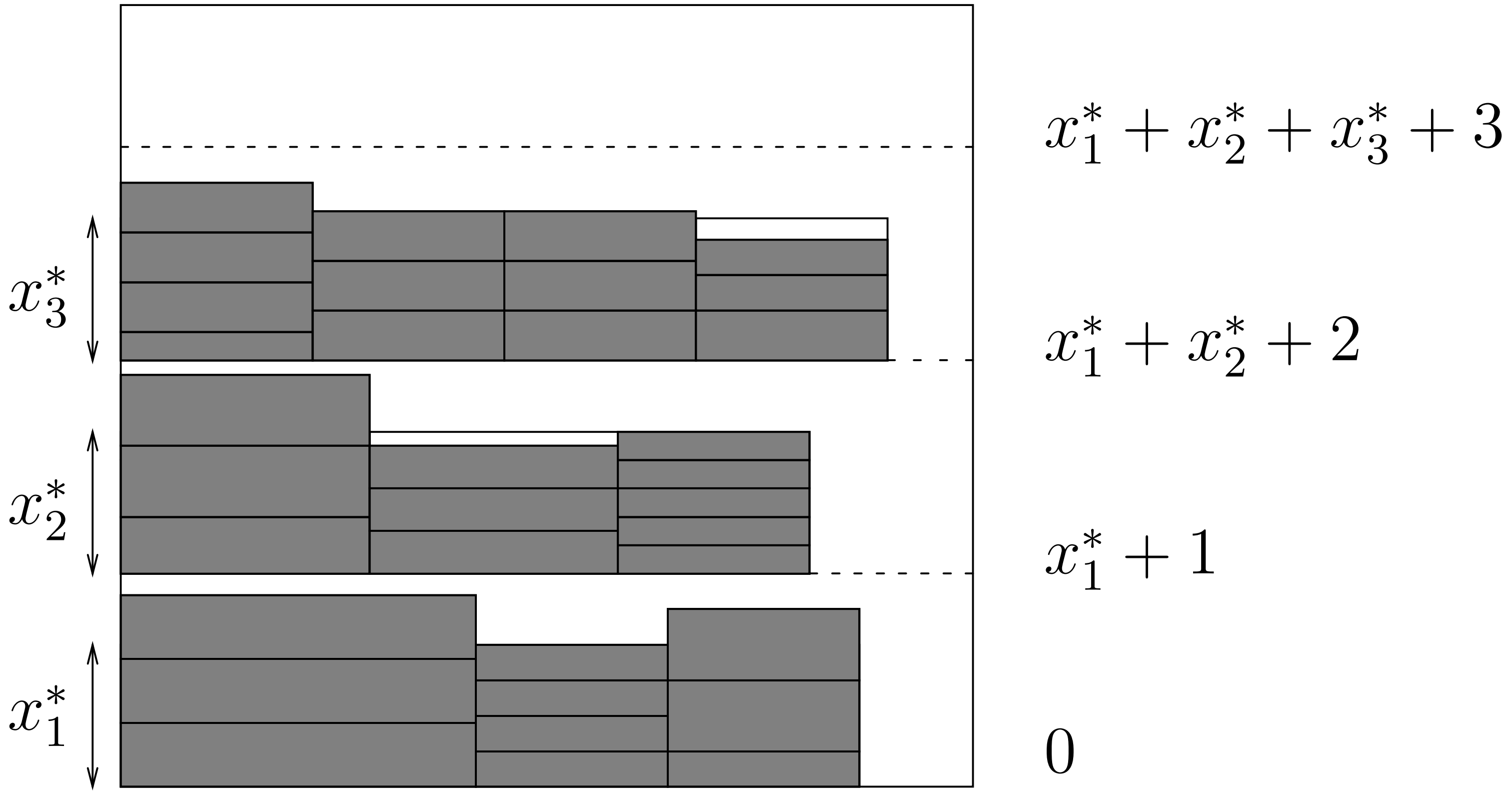
Step I (space generated by LP)



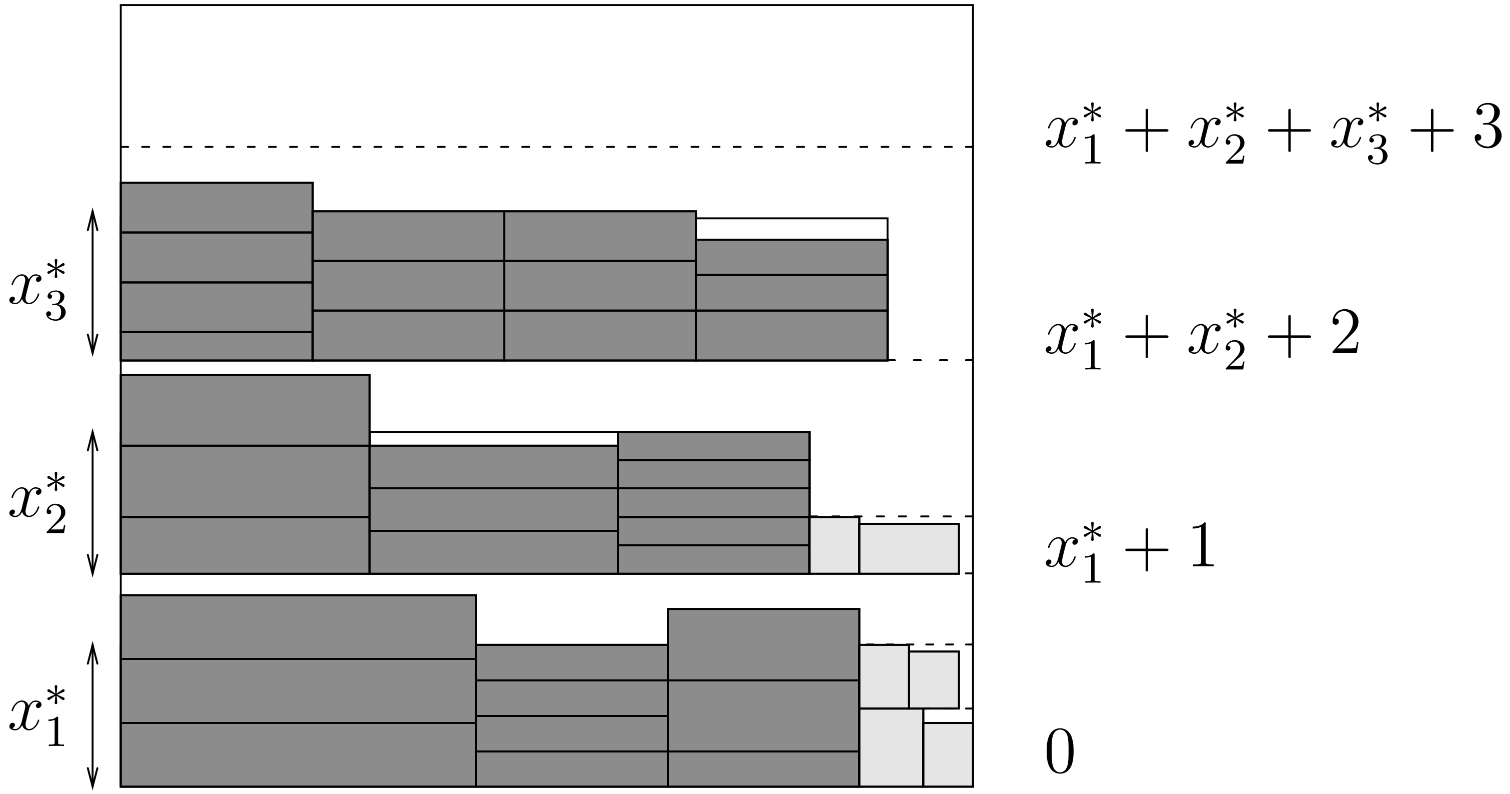
Step II (placing the wide rectangles)



Step III (after placing all wide rectangles)



Step IV (adding the narrow rectangles)



New Result

Theorem: **(Jansen, Solis-Oba 2006)**

There is an approximation algorithm A, which computes a packing into a strip of width 1 and height

$$A(L) \leq (1+\epsilon)OPT(L) + 1$$

for any $\epsilon > 0$.

2D knapsack problem

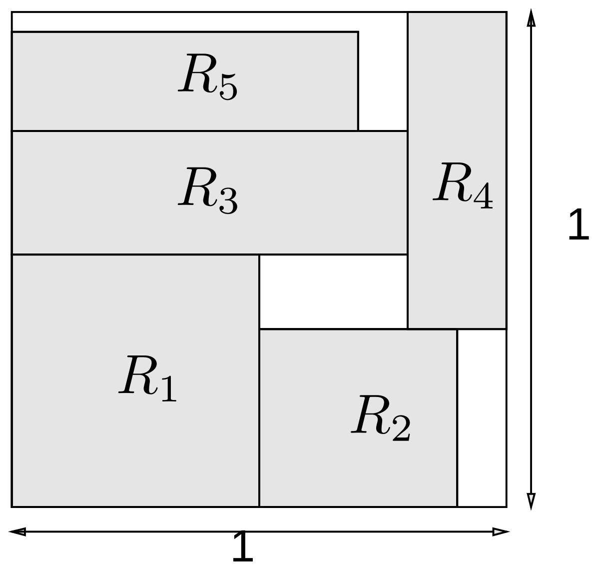
Given:

- n rectangles $R_i = (w_i, h_i)$ of width $w_i \leq 1$, height $h_i \leq 1$ and profit $p_i > 0$.

Find: a subset $R' \subset \{R_1, \ldots, R_n\}$ which can be packed into a square $[0,1] \times [0,1]$.

Goal: maximize the total profit $\sum_{R_i \in R'} p_i$.

Example



$R_1 = (1/2, 1/2)$, $R_2 = (2/5, 7/20)$, $R_3 = (4/5, 1/4)$, $R_4 = (1/5, 13/20)$, $R_5 = (7/10, 1/5)$, $R_6 = (4/5, 1/3)$ with $p_i = 1$.

New result

Theorem: **(Jansen, Solis-Oba 2006)** There is an algorithm B which finds a subset $R' \subset \{R_1, \ldots, R_n\}$ that can be packed into a rectangle of width 1 and height $1+\epsilon$ and has total profit

$$B(L) \geq (1-\epsilon)OPT(L),$$

where $OPT(L)$ is the maximum profit among all subsets (that fit into a square $[0,1] \times [0,1]$).

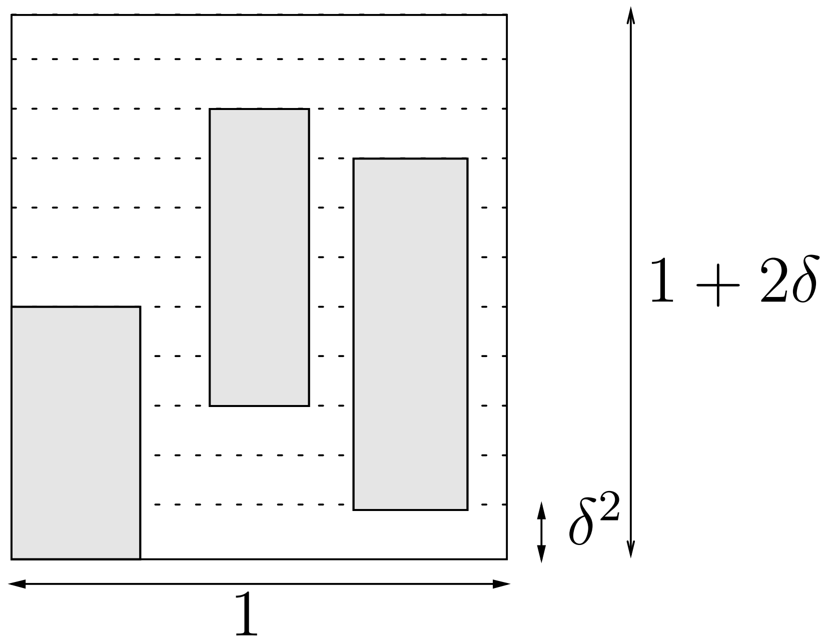
Main ideas for $2D$ knapsack problem

(a) eliminate a group of rectangles (with low profit) of width or height within $[\delta^s, \delta]$.

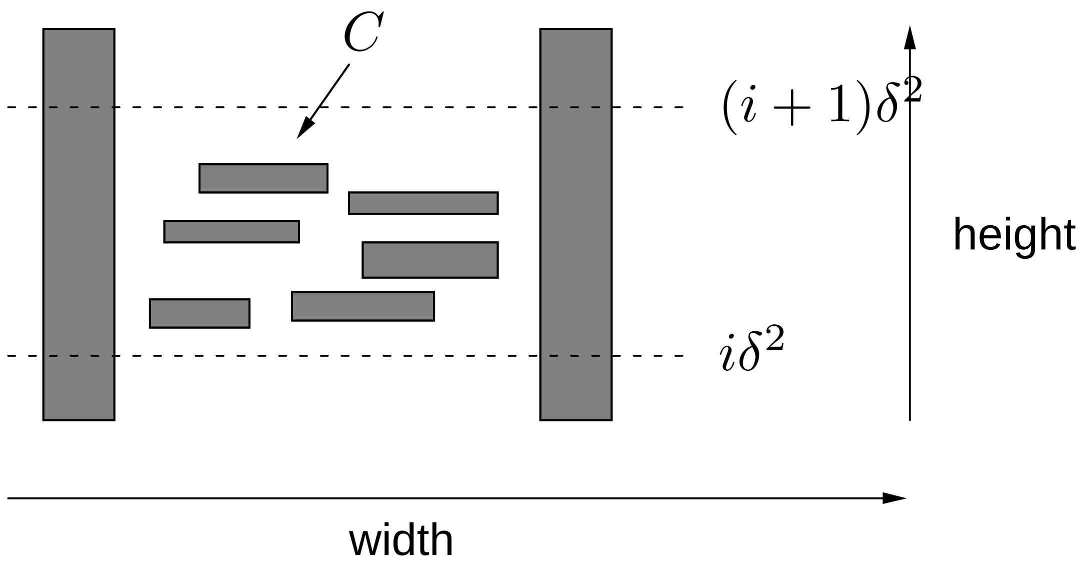
(b) partition the rectangles into tall and short rectangles and into wide and narrow rectangles.

(c) round up the height of each tall rectangle (with $h_i > \delta$) to a multiple of δ^2 and move these rectangles vertically.

Rounding of tall rectangles



Container for short rectangles



Each container contains at least one short, wide rectangle. Therefore, there is only a **constant number** of such containers.

Further ideas

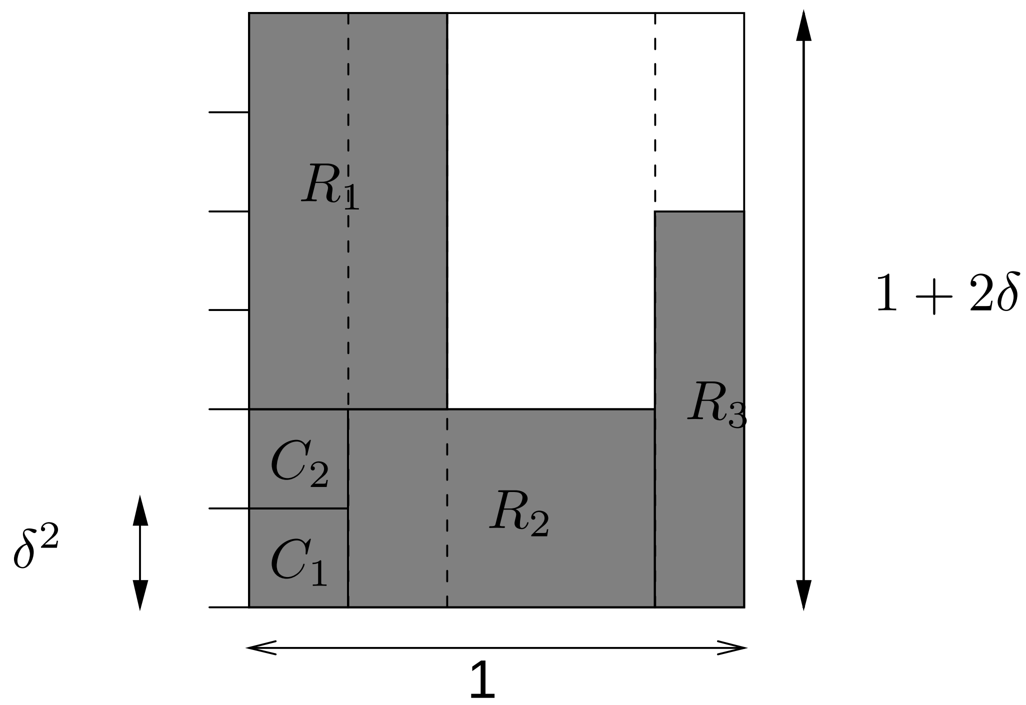
(a) pack short, wide rectangles only in containers,

(b) determine a **constant number** K of tall rectangles with highest profit

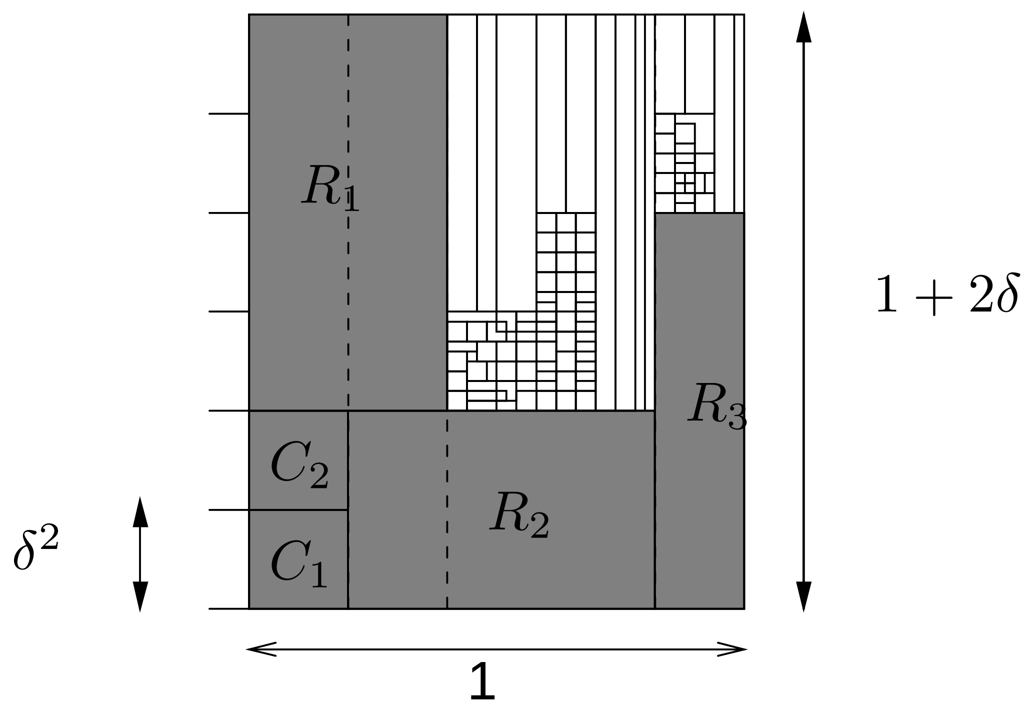
(c) compute packings for the K tall rectangles and the containers,

(d) use a **linear program** to place the remaining tall and short, narrow rectangles.

Packing of the K tall rectangles and containers



Adding the other rectangles



New 2D packing algorithm I

(1) use the algorithm by **Steinberg** to pack L into a strip of height $v \leq 2OPT_{Height}(L)$,

(2) guess approximately a value $v' \in [v/2, v]$,

(3) use the **2D knapsack algorithm** for the set of rectangles with scaled height $\bar{h}_i = h_i/v'$, width w_i and profit (or area) $p_i = \bar{h}_i w_i$.

2D packing algorithm II

(4) for v' with

$$OPT_{Height}(L) \leq v' \leq (1+\epsilon)OPT_{Height}(L)$$

there is a **packing of all scaled rectangles** into the square $[0,1] \times [0,1]$ with total area $F \leq 1$,

(5) our **2D knapsack algorithm** packs a subset $R' \subset R$ into the rectangle $[0,1] \times [0, 1+\epsilon]$ with total profit $\geq (1-\epsilon)OPT_{Profit}(L) = (1-\epsilon)F$,

The subset $R \setminus R'$ with profit or area $\leq \epsilon F \leq \epsilon$ remains unpacked.

2D packing algorithm III

(6) pack the subset $R \setminus R'$ (using the algorithm by Steinberg) into a rectangle of width 1 and height $\leq \epsilon + 1/v'$,

(7) the packing of the **scaled rectangles** gives a **total height** of $1 + 2\epsilon + 1/v'$.

Rescaling generates a **total height** of at most

$$\begin{aligned} & (1+2\epsilon)v' + 1 \\ \leq \quad & (1+2\epsilon)(1+\epsilon)OPT_{Height}(L) + 1 \\ \leq \quad & (1+5\epsilon)OPT_{Height}(L) + 1. \end{aligned}$$

Open questions

(1) is there an efficient algorithm A for the **2D packing problem** with

$$A(L) \leq a\, OPT_{Height}(L)$$

and $a < 2$?

(2) is there an efficient algorithm B for the **2D packing problem** with

$$B(L) \leq (1+\epsilon) OPT_{Height}(L) + b$$

and $b < 1$?