2D Cutting and Packing of Rectangular Pieces

Everything You Always Wanted To Know About 2D Rectangular Packing & Cutting (But Were Afraid To Ask)

Ramón Alvarez-Valdés
University of Valencia. Spain
Outline

• Introduction and classification
• 2D problems
  – Cutting Stock Problem: Column Generation
  – Strip Packing Problem: Branch & Bound
  – Pallet Loading Problem: Branch & Cut

OBJECTIVE:
DESCRIBE AND DISCUSS STRATEGIES FOR EXACT ALGORITHMS
Cutting & Packing

Large objects

¿Patterns?

Small objects

Stock sheets

Pieces
Characteristics

2. Type of assignment

All the sheets with the maximum value of pieces

All the pieces with minimum number of sheets

All the sheets with the maximum value of pieces
Cutting patterns
Cutting patterns: constraints

Orthogonal cuts

Non-orthogonal cuts
Cutting patterns: constraints

Guillotine cuts

Non-guillotine cuts
Cutting patterns: constraints

2-staged cutting

Multi-staged

piece

trimming

strip
Cutting patterns: constraints

Normalized cuts

Initial cutting pattern

Normalized cutting pattern
Cutting patterns: constraints

Fixed orientation

Piece \((l,w) \neq (w,l)\)
Classification of problems

1. Dimension (1, 2, 3)

2. Objective
   - Maximize output (fixed input)
   - Minimize input (fixed output)

3. Types of small objects (pieces, boxes,....)
   - Identical
   - Weakly heterogeneous
   - Strongly heterogeneous

4. Types of large objects (stock sheets, containers,....)
   - One object
   - Several objects (identical or different)

5. Shape of the small objects (pieces)
   - Rectangular
   - Irregular
## Summary of research efforts

### MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>IDENTICAL</th>
<th>DIFFERENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>UNIQUE</td>
<td>IIPP</td>
<td>SMALL-PACK</td>
</tr>
<tr>
<td></td>
<td>IDENTICAL</td>
<td>MILOPP/MIKP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MHLOPP/MHKP</td>
<td></td>
</tr>
</tbody>
</table>

- **1995-2005**
- **2006-2011**

### MINIMIZE INPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>WEAKLY HETEROGENEOUS</th>
<th>STRONGLY HETEROGENEOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>IDENTICAL</td>
<td>SSSCSP</td>
<td>SBSBPP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MSSCSP/RCSP</td>
<td>MBSBPP/RBPP</td>
</tr>
<tr>
<td>LARGE OBJECT OF VARIABLE DIMENSIONS</td>
<td></td>
<td>ODP</td>
<td></td>
</tr>
</tbody>
</table>
## Summary of research efforts

### MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>FIXED DIMENSIONS</th>
<th>SMALL OBJECTS</th>
<th>IDENTICAL</th>
<th>DIFFERENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALL</td>
<td>UNIQUE</td>
<td>IIPP</td>
<td></td>
<td>SLO-PACK</td>
</tr>
<tr>
<td>OBJECTS</td>
<td>IDENTICAL</td>
<td>MILOPP/MIKP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MHLOPP/MHKP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **1995-2005**
- **2006-2011**
Summary of research efforts

MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>IDENTICAL</th>
<th>DIFFERENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>UNIQUE</td>
<td>IIPP</td>
<td>SLOPP/SKP</td>
</tr>
<tr>
<td>IDENTICAL</td>
<td>MILOPP/MIKP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIFFERENT</td>
<td>MHLOPP/MHKP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 1995-2005
- 2006-2011
## Summary of research efforts

### MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>IDENTICAL</th>
<th>DIFFERENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>UNIQUE</td>
<td>IIPP</td>
<td>MILOPP/MIKP</td>
</tr>
<tr>
<td></td>
<td>IDENTICAL</td>
<td></td>
<td>MHLOPP/MHKP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 1995-2005
- 2006-2011
## Summary of research efforts

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>WEAKLY HETEROGENEOUS</th>
<th>STRONGLY HETEROGENEOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>IDENTICAL</td>
<td>SSSCSP</td>
<td>SBSBPP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MSSCSP/RCSP</td>
<td>MBSBPP/RBPP</td>
</tr>
<tr>
<td>LARGE OBJECT OF VARIABLE DIMENSIONS</td>
<td></td>
<td></td>
<td>ODP</td>
</tr>
</tbody>
</table>
## Summary of research efforts

### MINIMIZE INPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>WEAKLY HETEROGENEOUS</th>
<th>STRONGLY HETEROGENEOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>IDENTICAL</td>
<td>SSSCSP</td>
<td>SBSBPP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MSSCSP/RCSP</td>
<td>MBSBPP/RBPP</td>
</tr>
<tr>
<td>LARGE OBJECT OF VARIABLE DIMENSIONS</td>
<td></td>
<td>QDP</td>
<td></td>
</tr>
</tbody>
</table>
## Summary of research efforts

### MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>WEAKLY HETEROGENEOUS</th>
<th>STRONGLY HETEROGENEOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>IDENTICAL</td>
<td>SSSCSP</td>
<td>SBSBPP</td>
</tr>
<tr>
<td>DIFFERENT</td>
<td>MSSCSP/RCSP</td>
<td>MBSBPP/RBPP</td>
<td></td>
</tr>
<tr>
<td>LARGE OBJECT OF VARIABLE DIMENSIONS</td>
<td></td>
<td>ODP</td>
<td></td>
</tr>
</tbody>
</table>
The two-dimensional guillotine cutting stock problem

- Given a set of **stock sheets**, with known dimensions and **costs**

- and a set of **pieces**, with known dimensions and **demands**
The two-dimensional guillotine cutting stock problem

• The problem is:
  – how many sheets to cut
  – and in which way to cut them
  
  – to satisfy the demands of pieces completely
  – with minimum cost of sheets
    (minimum waste, if value = area)
The Gilmore–Gomory scheme

- Formulate the problem as a **linear programming** problem

- Solve the linear problem by using a **column generation** procedure

- **Round** to an integer solution
Linear programming formulation

Min \( \sum_{q \in Q} c_q x_q \)

s.t. \( \sum_{q \in Q} a_{iq} x_q \geq d_i, \quad i = 1, \ldots, m \)

\( x_q \geq 0, \) integer, \( \forall q \in Q \)

\( Q = \) set of cutting patterns

\( x_q = \) number of times we use pattern \( q \)

\( a_{iq} = \) number of times piece \( i \) appears in pattern \( q \)

\( d_i = \) demand of piece \( i \)

\( c_q = \) cost of pattern \( q \) (cost of sheet)

\( \pi_i = \) dual price of constraint \( i \)
A column of the formulation
Column generation procedure

1. - Generate an **initial set** of m patterns Q’, one for each type of piece

2. - Solve the **linear relaxation** of the problem over the set of variables Q’
3.- For each stock sheet $p$ solve the subproblem:

$$z_p = \text{Max} \sum_i \pi_i a_i$$

s.t. \( \{a_1, \ldots, a_m\} \) is a cutting pattern for sheet $S_p$

If, for some $p$, $z_p > c_p$, add the column to Q’ and go back to Step 2.
Otherwise, the process stops.
Subproblem of Step 3

• Given a stock sheet

• and a set of pieces

• decide how many pieces of each type to cut
• in order to maximize the total value of pieces cut
Solving subproblem 3

- Dynamic Programming
  (Gilmore-Gomory, Beasley)

- Heuristic algorithms
  - **Constructive**
  - **GRASP**
  - **Tabu Search**
## Comparative results

Waste percentages and running times of integer solutions

<table>
<thead>
<tr>
<th>Method</th>
<th>Waste</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructive</td>
<td>4.93</td>
<td>218</td>
</tr>
<tr>
<td>GRASP</td>
<td>3.73</td>
<td>873</td>
</tr>
<tr>
<td>Tabu Search</td>
<td>3.57</td>
<td>2351</td>
</tr>
<tr>
<td>DP-Beasley</td>
<td>3.38</td>
<td>7870</td>
</tr>
</tbody>
</table>
## Summary of research efforts

### MAXIMIZE OUTPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>IDENTICAL</th>
<th>DIFFERENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>UNIQUE</td>
<td>IIPP</td>
<td>SLO-PASK</td>
</tr>
<tr>
<td></td>
<td>IDENTICAL</td>
<td></td>
<td>MILOPP/MIKP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td></td>
<td>MHLOPP/MHKP</td>
</tr>
</tbody>
</table>

- **1995-2005**
- **2006-2011**

### MINIMIZE INPUT

<table>
<thead>
<tr>
<th>LARGE OBJECTS</th>
<th>SMALL OBJECTS</th>
<th>WEAKLY HETEROGENEOUS</th>
<th>STRONGLY HETEROGENEOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED DIMENSIONS</td>
<td>IDENTICAL</td>
<td>SSSCSP</td>
<td>SBSBPP</td>
</tr>
<tr>
<td></td>
<td>DIFFERENT</td>
<td>MSSCSP/RCSP</td>
<td>MBSBPP/RBPP</td>
</tr>
<tr>
<td>LARGE OBJECT OF VARIABLE DIMENSIONS</td>
<td></td>
<td>ODP</td>
<td></td>
</tr>
</tbody>
</table>
Bidimensional Strip Packing Problem

Min $H$

$n$ rectangular pieces

$W$
Solution methods

• **Exact algorithms**
  – Guarantee optimality
  – Can be very time-consuming

• **Heuristics & metaheuristics**
  – Do not guarantee optimality
  – Obtain good solution in reasonable computing times
Exact Algorithms: Branch & Bound

- Branching strategy
- Dominance/symmetry criteria
- Lower bounds

- Martello, Monaci & Vigo (IJOC, 2003)
- Alvarez-Valdes, Parreño & Tamarit (C&OR, 2008)
- Boschetti & Montaletti (OR, 2010)
- Côté, Dell’Amico, Iori (ESICUP, 2012)
- Arahori, Imamichi & Nagamochi (C&OR, 2012)
Preprocessing: Fixing pieces
Branch & Bound: branching strategy

Root node (empty solution)
Branch & Bound: branching strategy

Contour

Corner points
(Martello et al., 2003)
Branch & Bound: dominance
Branch & Bound: dominance

- More efficient use of space
Branch & Bound: dominance

- Avoid studying equivalent solutions
Branch & Bound: symmetry

Best solution

Best solution -1 \(\Rightarrow\)

\(Waste = P\)

\(Waste \leq P/2\)

\(Waste \leq P/2\)
Effect of dominance and symmetry criteria
Implicit enumeration: bounding

If we already have a solution with total waste = 4, we can fathom this branch

Waste = 6
Simple lower bounds

- Bound based on the area of pieces:

\[ L_0 = \max \left\{ \max_i \{h_i\}, \left[ \sum_{i} \frac{w_i h_i}{W} \right] \right\} \]

- Improved by solving knapsack problems at each level
Simple lower bounds

- Bound $L_2$ proposed by Martello et al. with some modifications

\[ J_1 = \{ j \in J : w_j > W - \alpha \} \]
\[ J_2 = \{ j \in J : W - \alpha \geq w_j > W/2 \} \]
\[ J_3 = \{ j \in J : W/2 \geq w_j \geq \alpha \} \]

\[
L(\alpha) = \sum_{j \in J_1 \cup J_2} h_j + \max \left( 0, \left( \sum_{j \in J_3} w_j h_j - \left( \sum_{j \in J_2} (W - w_j) h_j \right) \right) \right) / W
\]
Simple lower bounds

- Bound $L_2$ proposed by Martello et al. with some modifications

$J_1 = \{ j \in J : w_j > W - \alpha \}$

$J_2 = \{ j \in J : W - \alpha \geq w_j > W/2 \}$

$J_3 = \{ j \in J : W/2 \geq w_j \geq \alpha \}$

$$L(\alpha) = \sum_{j \in J_1 \cup J_2} h_j + \max \left( 0, \left[ \left( \sum_{j \in J_3} w_j h_j - K \left( \sum_{j \in J_2} (W - w_j) h_j \right) \right) / W \right] \right)$$
Dual feasible functions (Fekete&Schepers)

Function $u : [0,1] \rightarrow [0,1]$ is called **dual feasible** if

for any set $S$ of non-negative real numbers: $\sum_{x \in S} x \leq 1 \Rightarrow \sum_{x \in S} u(x) \leq 1$

$u^{(k)} : [0,1] \rightarrow [0,1]$

$x \rightarrow \begin{cases} 
  x & \text{if } x(k+1) \in Z \\
  \left\lfloor (k+1)x \right\rfloor \frac{1}{k} & \text{otherwise}
\end{cases}$
Dual feasible functions (Fekete&Schepers)

\[ U^{(\varepsilon)} : [0,1] \rightarrow [0,1] \]

\[ x \rightarrow \begin{cases} 
1 & \text{if } x > 1 - \varepsilon \\
0 & \text{if } x < \varepsilon \\
x & \text{if } \varepsilon \leq x \leq 1 - \varepsilon
\end{cases} \]
Dual feasible functions (Fekete&Schepers)

\[ \phi^\epsilon(x) : [0,1] \rightarrow [0,1] \]

\[ x \rightarrow \begin{cases} 
1 - \frac{(1-x)\epsilon^{-1}}{\epsilon^{-1}} & \text{if } x > \frac{1}{2} \\
\frac{1}{\epsilon^{-1}} & \text{if } \epsilon \leq x \leq \frac{1}{2} \\
0 & \text{if } x < \epsilon
\end{cases} \]

Improved by Carlier et al.:

\[ f_2^k(x) : [0,W] \rightarrow \left[ 0,2\left\lfloor \frac{W}{k} \right\rfloor \right] \]

\[ x \rightarrow \begin{cases} 2\left( \left\lfloor \frac{W}{k} \right\rfloor - \left\lfloor \frac{W-x}{k} \right\rfloor \right) & \text{if } x > \frac{1}{2}W \\
\left\lfloor \frac{W}{k} \right\rfloor & \text{if } x = \frac{1}{2}W \\
2\left\lfloor \frac{x}{k} \right\rfloor & \text{if } x < \frac{1}{2}W
\end{cases} \]
Example of using a dual feasible function

- Strip: \( W=10 \)
- Pieces:

<table>
<thead>
<tr>
<th>( w_i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>( h_i )</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

Area bound \( = \left\lfloor \frac{8 \times 4 + 5 \times 5 + 4 \times 5 + 2 \times 1}{10} \right\rfloor = \left\lfloor \frac{79}{10} \right\rfloor = 8 \)

- Function \( f_{3,2} \) by Carlier et al. (\( k=3 \)) \( f^k_2: [0, 10] \rightarrow [0, 6] \)

\[
x \rightarrow \begin{cases} 
2 \left( \left\lfloor \frac{W}{k} \right\rfloor - \left\lfloor \frac{W-x}{k} \right\rfloor \right) & \text{if } x > \frac{1}{2}W \quad x = 8 \rightarrow 6 \\
\left\lfloor \frac{W}{k} \right\rfloor & \text{if } x = \frac{1}{2}W \quad x = 5 \rightarrow 3 \\
2 \left\lfloor \frac{x}{k} \right\rfloor & \text{if } x < \frac{1}{2}W \quad x = 4 \rightarrow 2 \\
\end{cases}
\]

Area bound (DFF) \( = \left\lfloor \frac{6 \times 4 + 3 \times 5 + 2 \times 5 + 0 \times 1}{6} \right\rfloor = \left\lfloor \frac{49}{6} \right\rfloor = 9 \)
Lower bounds: Problem relaxations

- $K_{NCBP}$
- $1_{NCBP}$
- $1_{CBP}$
- $2_{SP}$
1-Contiguous Bin Packing Problem (1-CBP)
An integer formulation for the 1-CBP

\[ x_{ij} = \begin{cases} 
1 & \text{if piece } i \text{ has its bottom side at height } j \\
0 & \text{otherwise} 
\end{cases} \quad i = 1, 2, \ldots, n ; \quad j = 1, 2, \ldots, UB - h_i + 1 \]

\[ x_{n+1,j} = \begin{cases} 
1 & \text{if some piece } i \text{ reaches level } j \\
0 & \text{otherwise} 
\end{cases} \quad j = LB, \ldots, UB \]

Min \quad \sum_{j} j x_{n+1,j}

s.t. \quad \sum_{j} x_{ij} = 1 \quad \forall i = 1, 2, \ldots, n

\[ \sum_{i} w_i \sum_{t=j-w_i+1}^{j} x_{ij} \leq W \quad \forall j = 1, 2, \ldots, UB \]

\[ \sum_{j=\tau}^{UB-h_i} x_{ij} + \sum_{k=LB}^{\tau+h_i-1} x_{n+1,k} \leq 1 \quad \forall i = 1, 2, \ldots, n \]

\[ \forall \tau = LB - h_i + 1, \ldots, UB - h_i \]
An integer formulation for the 1-CBP

- Large number of variables
  - gcut04: $n=50$, $W=250$, $UB=3077$ → 150000 variables

- Long running times
Optimization → Decision

\[ x_{ij} = \begin{cases} 
1 & \text{if piece } i \text{ has its bottom side at height } j \\
0 & \text{otherwise}
\end{cases} \quad i = 1, 2, \ldots, n ; \quad j = 1, 2, \ldots, UB - h_i + 1 \]

Can all the pieces be packed into a height \( K \)?

s.t.

\[ \sum_{j} x_{ij} = 1 \quad \forall i = 1, 2, \ldots, n \]

\[ \sum_{i} w_i \sum_{t=j-w_j+1}^{j} x_{ij} \leq W \quad \forall j = 1, 2, \ldots, UB \]

\[ \sum_{j=\tau}^{UB-h_i} x_{ij} + \sum_{k=LB}^{\tau+h_i-1} x_{n+1,k} \leq 1 \quad \forall i = 1, 2, \ldots, n \]

\[ \forall \tau = LB - h_i + 1, \ldots, UB - h_i \]
Rotating strip and pieces
Binary search

- Step 0.- Initialization:  \( K_1 = LB, \ K_2 = UB, \ K = \left\lfloor \frac{K_1 + K_2}{2} \right\rfloor \)

- Step 1.- Solve decision problem with target height \( K \)

  - **Yes** (or TimeLimit reached): Adjust upper limit
    \[
    K_2 = K, \quad K = \left\lfloor \frac{K_1 + K_2}{2} \right\rfloor
    \]
    \[
    LB = K + 1
    \]

  - **No**: Adjust lower limit
    \[
    K_1 = K, \quad K = \left\lfloor \frac{K_1 + K_2}{2} \right\rfloor
    \]
# Binary search on instance gcut04

<table>
<thead>
<tr>
<th>Lower (K1)</th>
<th>Upper (K2)</th>
<th>Target K</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>2926</td>
<td>3189</td>
<td>3057</td>
<td>Yes (TimeLimit)</td>
</tr>
<tr>
<td>2926</td>
<td><strong>3057</strong></td>
<td>2991</td>
<td>Yes (TimeLimit)</td>
</tr>
<tr>
<td>2926</td>
<td>2991</td>
<td>2958</td>
<td>No</td>
</tr>
<tr>
<td><strong>2958</strong></td>
<td>2991</td>
<td>2974</td>
<td>No</td>
</tr>
<tr>
<td>2974</td>
<td>2991</td>
<td>2982</td>
<td>No</td>
</tr>
<tr>
<td><strong>2982</strong></td>
<td>2991</td>
<td>2986</td>
<td>No</td>
</tr>
<tr>
<td>2986</td>
<td>2991</td>
<td>2988</td>
<td>No</td>
</tr>
<tr>
<td>2988</td>
<td>2991</td>
<td>2989</td>
<td>No</td>
</tr>
<tr>
<td><strong>2989</strong></td>
<td>2991</td>
<td>2990</td>
<td>No</td>
</tr>
</tbody>
</table>

**LB = 2991**
## Results of the new lower bound

<table>
<thead>
<tr>
<th>Instance</th>
<th>Optimal solution</th>
<th>1-CBP bound</th>
<th>Orientation</th>
<th>Time</th>
<th>Martello bound</th>
<th>Martello time</th>
</tr>
</thead>
<tbody>
<tr>
<td>cgcut02</td>
<td>66*</td>
<td>63</td>
<td>V</td>
<td>60.07</td>
<td>63</td>
<td>2178.39</td>
</tr>
<tr>
<td>cgcut03</td>
<td>670*</td>
<td>654</td>
<td>H</td>
<td>40.70</td>
<td>636</td>
<td>2996.11</td>
</tr>
<tr>
<td>gc02</td>
<td>1191*</td>
<td>1184</td>
<td>H</td>
<td>80.23</td>
<td>1133</td>
<td>3393.61</td>
</tr>
<tr>
<td>gc03</td>
<td>1803</td>
<td>1803</td>
<td>H</td>
<td>0.70</td>
<td>1803</td>
<td>0.00</td>
</tr>
<tr>
<td>gc04</td>
<td>3002*</td>
<td>2991</td>
<td>H</td>
<td>42.37</td>
<td>2934</td>
<td>3944.81</td>
</tr>
<tr>
<td>ngcut01</td>
<td>23</td>
<td>23</td>
<td>H</td>
<td>0.05</td>
<td>23</td>
<td>0.05</td>
</tr>
<tr>
<td>ngcut02</td>
<td>30</td>
<td>30</td>
<td>H</td>
<td>0.03</td>
<td>30</td>
<td>11.31</td>
</tr>
<tr>
<td>ngcut04</td>
<td>20</td>
<td>20</td>
<td>V</td>
<td>0.03</td>
<td>20</td>
<td>0.00</td>
</tr>
<tr>
<td>ngcut06</td>
<td>31</td>
<td>31</td>
<td>H</td>
<td>15.45</td>
<td>31</td>
<td>727.20</td>
</tr>
<tr>
<td>ngcut08</td>
<td>33</td>
<td>33</td>
<td>H</td>
<td>0.11</td>
<td>33</td>
<td>53.09</td>
</tr>
<tr>
<td>ngcut09</td>
<td>50</td>
<td>49</td>
<td>H</td>
<td>20.03</td>
<td>49</td>
<td>1129.01</td>
</tr>
<tr>
<td>ngcut10</td>
<td>80</td>
<td>80</td>
<td>V</td>
<td>0.09</td>
<td>80</td>
<td>0.18</td>
</tr>
<tr>
<td>ngcut11</td>
<td>52</td>
<td>52</td>
<td>H</td>
<td>24.70</td>
<td>52</td>
<td>483.01</td>
</tr>
<tr>
<td>ngcut12</td>
<td>87</td>
<td>87</td>
<td>H</td>
<td>0.16</td>
<td>87</td>
<td>0.00</td>
</tr>
<tr>
<td>ht02</td>
<td>20</td>
<td>20</td>
<td>H</td>
<td>9.36</td>
<td>20</td>
<td>623.53</td>
</tr>
<tr>
<td>ht08</td>
<td>30</td>
<td>30</td>
<td>V</td>
<td>40.05</td>
<td>30</td>
<td>630.17</td>
</tr>
</tbody>
</table>
Lower bounds: Problem relaxations

- K_NCBP
- 1_NCBP
- 1_CBP
- 2SP
1-Non-Contiguous Bin Packing Problem (1-NCBP)
k-NCBP: k-Non-Contiguous Bin Packing Problem
A new approach for the SPP  
(Côté, Dell’Amico, Iori, ESICUP 2012)

We model the SPP as a MILP by using:

\[ x_{jp} = 1 \text{ if } j \text{ packed with left border in } p, \ 0 \text{ otherwise} \]
\[ y_{jr} = 1 \text{ if } j \text{ packed with bottom border in } r, \ 0 \text{ otherwise} \]
\[ z = \text{height of the packing} \]

We obtain:

\[ \text{min } z \]
\[ \sum_{j \in N} \sum_{p \in W(j,q)} h_{j}x_{jp} \leq z \quad q \in W \]
\[ \sum_{p \in W} x_{jp} = 1 \quad j \in N \]
\[ \sum_{r \in H} y_{jr} = 1 \quad j \in N \]
\[ \sum_{p \in W(j,q)} x_{jp} + \sum_{p \in W(k,q)} x_{kp} + \sum_{r \in H(j,t)} y_{jr} + \sum_{r \in H(k,t)} y_{kr} \leq 3 \quad j, k \in N, q \in W, t \in H \]
\[ x_{jp} \in \{0, 1\} \quad j \in N, p \in W \]
\[ y_{jr} \in \{0, 1\} \quad j \in N, r \in H \]
We take care of the $x$ first ...
A new approach for the SPP
(Côté, Dell’Amico, Iori, ESICUP 2012)

... and then of the y

Suppose an integer solution $S = \{z^s, x^s_{jp}\}$ is found to $P|\text{cont}|C_{\text{max}}$.

We check if $S$ is $y-$feasible by defining:

$$p^s_j = \sum_{p \in W} p \cdot x^s_{jp} = \text{coordinate of left border of } j \text{ in solution } S$$

and solving the following slave problem:

$$(y-\text{check}) \quad \sum_{r \in H} y_{jr} = 1 \quad j \in N$$

$$\sum_{j \in N: q \in W(j, p^s_j)} \sum_{r \in H(j,t)} y_{jr} \leq 1 \quad q \in W, t \in H, t \leq z^s$$

$$y_{jr} \in \{0, 1\} \quad j \in N, r \in H, r \leq z^s$$
A new approach for the SPP
(Côté, Dell’Amico, Iori, ESICUP 2012)

**Complexity of the \(y\)-check**

Usually we can provide solutions in a very quick time.

![A difficult \(y\)-check instance](image)

If \(S\) is feasible for the slave, then it is feasible for SPP. Otherwise we found a violated *Benders’ cut*:

\[
\sum_{j \in N} c_{j,p_j} \leq |N| - 1
\]
Computational results

Instances by Berkey&Wang, Martello&Vigo
10 classes de instances, of different characteristics and sizes
Constructive algorithms: Bottom-Left
Constructive algorithms : Bottom-Left-Fill
Constructive algorithms : Best-Fit

- Burke, Kendall, Whitwell, Operations Research 2005
Constructive algorithms: Scoring

- Leung, Zhang, Sim, EJOR 2011
Metaheuristics

- **Based on piece orderings**
  - Gómez & De la Fuente (2000): GA(BL)
  - Iori et al. (2003): GA (TP2SP)
  - Lesh et al. (2004): BLD* (BLD)
  - Belov et al. (2007): BS (BLR)
  - Belov et al. (2007): SVC(SubKP)
  - Burke et al. (2008): BF +SA(BLF)
  - Leung et al. (2011): LS, SA (Scoring)
  - Burke et al. (2011): Squeaky Wheel (BF)
  - Wei et al. (2011): Tabu Search (Spread)

- **Using the solution layout**
  - Iori et al. (2003): TS
  - Beltrán et al. (2004): GRASP+VND
  - Bortfeldt (2005): GA
  - Alvarez-Valdés et al. (2008): GRASP
Squeaky Wheel

Priority space

Penalize

New ordering

Solution space
Definition of movements: $N_1$

- **Block reduction**
  - Choose the block to reduce
  - Move the remaining blocks to the container’s nearest corner, measured by the lexicographic distance.
  - Update the list $L$ of maximal spaces.
  - Fill the empty maximal spaces by applying the constructive algorithm with the Best-volume objective function.
Definition of movements: $N_2$

- **Column insertion**
  - Choose the space
  - Choose the box to be inserted
  - Put box B into the corner of S nearest to a corner of the container
  - Choose a possible direction for building a column of boxes
  - Remove the overlapping boxes of the container
  - Update the list L of maximal spaces
  - Fill the empty maximal spaces by applying the constructive algorithm
Definition of movements: $N_3$

- **Box insertion**
  - Choose a box to insert $B$
  - Choose the space $S$ to insert this piece
  - Choose the position of $B$ in $S$
  - Remove the overlapping boxes of the container
  - Update the list $L$ of maximal spaces.
  - Fill the empty maximal spaces by applying the constructive algorithm
Definition of movements: $N_4 - N_5$

- **Emptying a region**
  - Take a first space $S_1$
  - From among the spaces smallest than $S_1$, take a second space $S_2$
  - Create the smallest parallelepiped $P$ containing $S_1$ and $S_2$
  - Remove all the boxes overlapping with $P$
  - Update the list $L$ of maximal spaces
  - Fill the empty spaces by applying the constructive algorithm
Definition of movements: $N_5-N_6$

- Emptying a region
  - Take a first space $S_1$
  - From among the spaces smallest than $S_1$, take a second space $S_2$
  - Create the smallest parallelepiped $P$ containing $S_1$ and $S_2$
  - Remove all the boxes overlapping with $P$
  - Update the list $L$ of maximal spaces
  - Fill the empty spaces by applying the constructive algorithm
Computational results

- Martello et al. & Berkey et al. instances
  (500 instances, 20-100 pieces)
Computational results

- Hopper & Turton instances (20-160 pieces)
The pallet loading problem

Problem arising in factories:
Loading a pallet with one type of product
Problem definition

The problem becomes two-dimensional
Problem definition

The problem becomes two-dimensional
Problem definition

How many boxes can be loaded into the Pallet?
Branch and Cut

Linear formulation + branching constraints

+ cuts

Fathomed if:
- integer solution is found
- solution value lower than best known solution

Otherwise, further branching
Branch and Cut

- Linear formulation
- Branching and fixing variables
- Separation procedures
Let us consider the instance \((105, 70, 12, 7)\)
Let us assume that \(UB = 87\)
and we have a heuristic solution with 84 boxes
Definition of variables

Horizontal variable

Vertical variable
Definition of variables

$$20\ 14\ 4\ 3$$

$$(L-h)(W-w)$$

(horizontal variables)
Reduction of variables: Raster points

Raster Points

Only linear combinations of box dimensions
Reduction of variables: Raster points

The number of variables is reduced from 346 to 202
Reduction of variables: Dominance

20 14 4 3
Covering constraints

The sum of variables covering a unit square must be less than or equal to 1
Branch and Cut: Linear formulation

\[ \begin{align*} 
\text{Max} & \quad \sum_{(i, j) \in V} v_{ij} + \sum_{(k, l) \in H} h_{kl} \\
\text{s.t.} : & \quad \sum_{(i, j) \in V_p} v_{ij} + \sum_{(k, l) \in H_p} h_{kl} \leq 1 \quad \forall p \in R \\
& \quad \sum_{(i, j) \in V} v_{ij} + \sum_{(k, l) \in H} h_{kl} \leq UB \\
\end{align*} \]

+ Symmetry constraints

\[ \begin{align*} 
v_{ij} &= \{0, 1\} \quad (i, j) \in V \\
h_{kl} &= \{0, 1\} \quad (k, l) \in H \\
\end{align*} \]
Transformation into a maximal independent set problem

Two vertices are adjacent, if their variables overlap  
Each fractional variable of the linear solution defines a vertex
Edges related to the maximal allowable waste in the solution

20 14 4 3

If the waste is greater than the maximal allowable waste, there is an edge between the vertices of these pieces.
Edges associated to dominance criteria
Edges associated to dominance criteria

20 14 4 3
Graph of the fractional solution

Independent set ↔ Feasible layout of boxes
Separation procedures

- Cliques

- Odd cycles

(We are not using other classes of facets and valid inequalities for the maximal independent set)
The identification procedure combines two strategies, depending on the value of the minimum degree of the vertices:

- Greedy heuristic ($d(v) \geq 25$)

\[ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leq 1 \]
Cliques in the graph of the solution
Odd cycles

\[
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 \leq \left\lfloor \frac{9}{2} \right\rfloor = 4
\]

- Minimum spanning tree
- Hole identification (Hoffman y Padberg (1985))
- In this problem, it is highly unlikely to have violated inequalities involving only the cycle variables
Odd cycles in the graph of the solution
Odd cycle lifting

- Procedure developed by Padberg (1972)
- Starting from a hole, solve a series of integer linear problems to obtain the coefficients for the remaining variables

\[
\sum_{j \in S} x_j + \sum_{j \in T} \beta_j x_j \leq \frac{1}{2}(|S| - 1)
\]

- Heuristic lifting procedure, including new variables, one at a time, if some conditions are satisfied
Graph of the fractional solution

\[ \sum_{i=1}^{9} x_i + x_{10} + x_{11} \leq 4 \]
Branching based on variables

\[ x_i = 1 \]

The variables adjacent to variable \( i \), are set to 0

\[ x_i = 0 \]
Branching on constraints

$$\sum x_i = 0$$

$$\sum x_i = 1$$

Covering constraints
Some conclusions

• A lot of hard work
• Many previous results (on the same or related problems)
• Some new good ideas
  +
• Much more powerful integer linear codes
• Much more powerful computers

GREAT TIMES ARE WAITING!
2D Cutting and Packing of Rectangular Pieces

Everything You Always Wanted To Know About 2D Rectangular Packing & Cutting (But Were Afraid To Ask)

Ramón Alvarez-Valdés
University of Valencia. Spain