A.I.R.O. Winter 2005
Cortina d'Ampezzo, February 3, 2005

# A branch-and-price algorithm for the bi-dimensional level strip packing problem

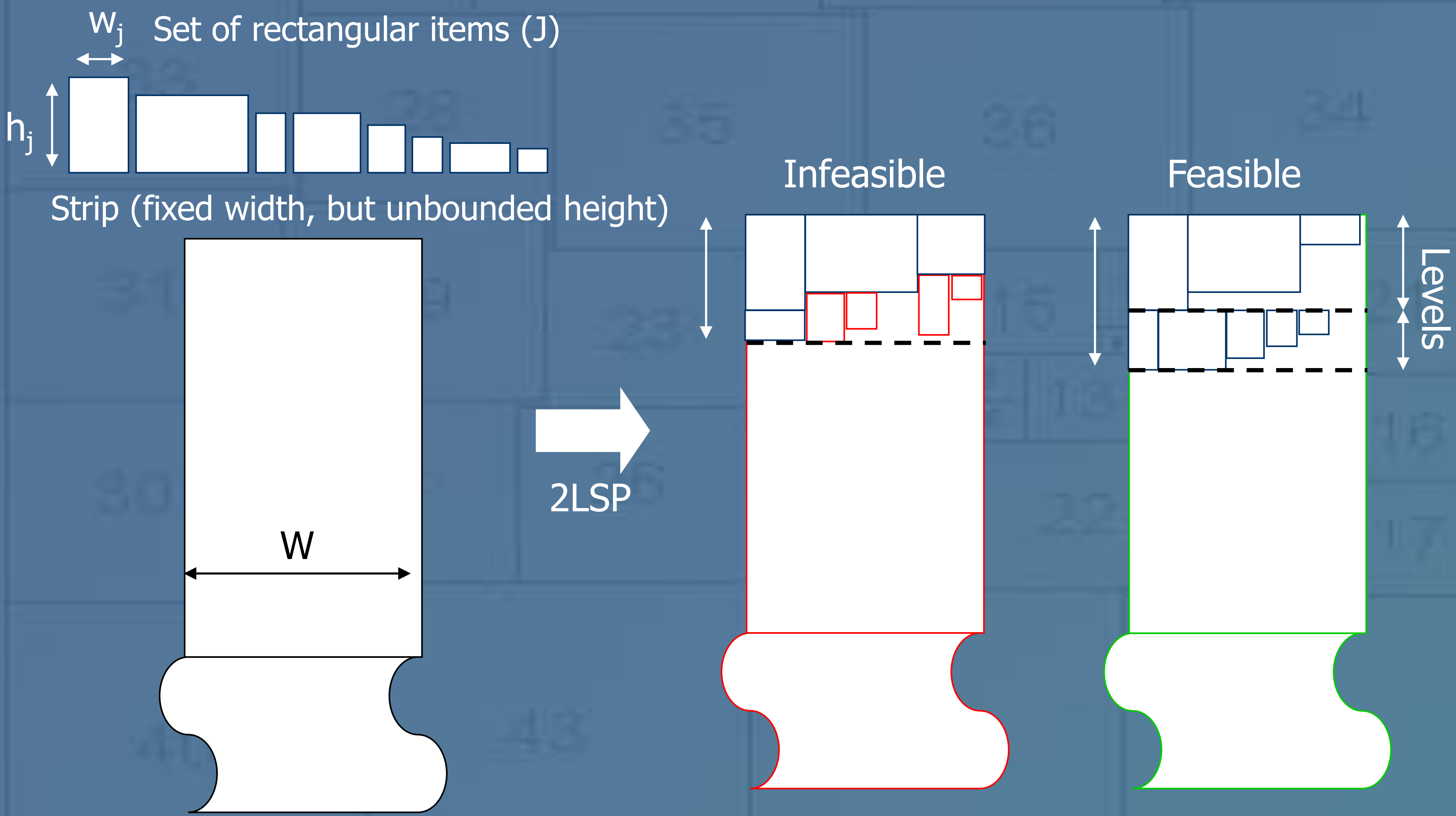
A. Bettinelli, A. Ceselli, G. Righini
Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione

email: righini@dti.unimi.it

# Outline of the talk

- The bi-dimensional level strip packing problem (2LSPP)
  - Literature review
  - Formulations
- A branch-and-price algorithm
  - Dual bound:
    column generation, the penalized knapsack problem
  - Primal bounds
  - Branching rules
- Experimental results

# Problem description



# Literature review

- Complexity: 2LSPP is strongly NP-Hard (1BP)
- Previous work:
  - ILP formulations, dual bounds
    - Lodi, Martello, Monaci "Two-dimensional packing problems: a survey", EJOR 141, 2002
    - Lodi, Martello, Vigo "Models and bounds for two-dimensional level packing problems", JCO 8, 2004
  - Approximation algorithms (Coffman et al., 1980)
    - Best Fit, Decreasing Height
    - First Fit, Decreasing Height
    - Next Fit, Decreasing Height

# ILP formulation

$$\min \sum_{j \in J} h_j x_{jj}$$

Assumption: the items are ordered by non-increasing $h_j$ values

$$s.t.$$

$$\begin{cases} \sum_{j \in J} x_{ij} = 1 & \forall i \in J \\ \sum_{i \in J} w_i x_{ij} \le W x_{jj} & \forall j \in J \\ h_i x_{ij} \le h_j & \forall i, j \in J \\ x_{ij} \in \{0,1\} & \forall i, j \in J \end{cases}$$

$h_i x_{ij} \le h_j \;\; \forall i, j \in J \longrightarrow x_{ij} = 0 \;\; \forall i > j$

# CCLP formulation

$$\min \sum_{i\in I} d_{ij} x_{ij} + \sum_{j\in J} f_j x_{jj}$$

$s.t.$

$$\begin{cases} \sum_{j\in J} x_{ij} = 1 & \forall i \in J \\ \sum_{i\in I} a_{ij} x_{ij} \le c_j x_{jj} & \forall j \in J \\ x_{ij} \in \{0,1\} & \forall i \in I, \forall j \in J \end{cases}$$

$$d_{ij} = 0 \qquad \forall i \in I, \forall j \in J$$

$$f_j = h_j \qquad \forall j \in J$$

$$c_j = W \qquad \forall j \in J$$

$$a_{ij} = \begin{cases} w_i & \text{(if i} \le \text{j)} \\ +\infty & \text{(otherwise)} \end{cases}$$

$$\forall i \in I, \forall j \in J$$

# Set covering formulation

$$\min v = \sum_{k \in K} h^k z^k$$

$s.t.$

$$\begin{cases} \sum_{k \in K} x_j^k z^k \geq 1 & \forall j \in J \\ z^k \in \{0,1\} & \forall k \in K \end{cases}$$

$\forall k \in K:$

$$\begin{cases} \sum_{j \in J} w_j x_j^k \leq W \\ x_j^k \in \{0,1\} \; \forall j \in J \end{cases}$$

$$h^k = \max_{j \in J | x_j^k = 1} \{h_j\}$$

# B&P - Dual bound

- LP relaxation by column generation:

$$\min v = \sum_{k \in K} h^k z^k$$

$$\sum_{k \in K} x_j^k z^k \geq 1 \; \forall j \in J \quad (\lambda_j)$$

$$0 \leq z^k \leq 1 \qquad z^k \in \{0,1\} \qquad \forall k \in K$$

$$\forall k \in K: \quad \begin{cases} \sum_{j \in J} w_j x_j^k \leq W \\ x_j^k \in \{0,1\} \forall j \in J \end{cases} \qquad h^k = \max_{j \in J | x_j^k = 1} \{h_j\}$$

$$rc(k) = h^k - \sum_{j \in J} \lambda_j x_j^k$$

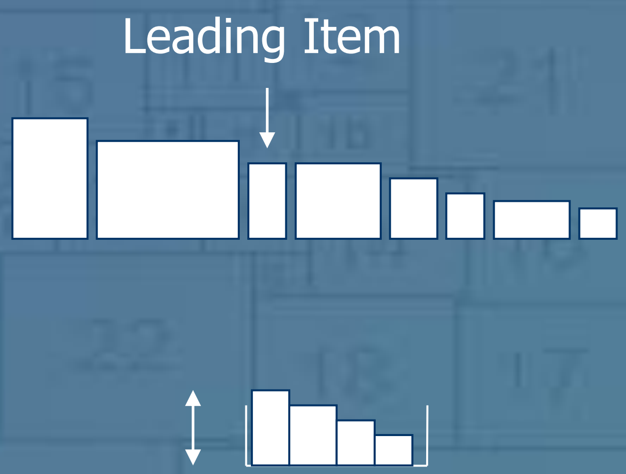
# B&P – Pricing problem

$$\min\left[-\sum_{j\in J}\lambda_j x_j^k + \max_{j\in J|x_j^k=1}\{h_j\}\right]$$

$$s.t.\begin{cases}\sum_{j\in J} w_j x_j^k \le W\\ x_j^k \in \{0,1\} \quad \forall j\in J\end{cases}$$

$$\min -\sum_{j\in J}\lambda_j x_j^k + \eta$$

$$s.t.\begin{cases}\sum_{j\in J} w_j x_j^k \le W\\ h_j x_j^k \le \eta \quad \forall j\in J\\ x_j^k \in \{0,1\} \quad \forall j\in J\\ \eta \ge 0\end{cases}$$



# B&P – Pricing algorithm

A. Ceselli, G. Righini, “An optimization algorithm for the penalized knapsack problem”, technical report (2005).

1. Preprocessing
   (removal of candidate leading items)
2. LP bounds ( $O(N^2)$ time)
3. Best bound search for the optimal leading item:
   1. Solution of a KP
   2. Domination criterion
   3. Bounding criterion

# B&P – Primal bounds

- First Fit, Decreasing Height heuristic (Coffman et al. '80):
  - O(N log N) time
  - Worst case (for the 2SP)

    $$FFDH(I) <= 17/10\ OPT(I) + 1$$
  - Very effective in our case
- Greedy heuristic
  - Sort columns by non increasing value of the corresponding variable
  - Select columns in a greedy way, until a feasible covering is found

# B&P - Branching rules

- Incompatibilities (Ryan and Foster):

$$\exists i, j \in J \mid \quad 0 < \sum_{k \in K \mid x_i^k = 1 \wedge x_j^k = 1} z^k < 1$$

$$\rightarrow \quad x_i^k + x_j^k \leq 1$$

$$\rightarrow \quad x_i^k - x_j^k = 0$$

- Two-level branching:
  - First level:
    fix the $x_{jj}$ variables (leading items)
  - Second level:
    solve the remaining GAP
    (Savelsbergh, Ceselli and Righini)
  - Best first search policy

# Experiments

- Dataset 1: 38 instances for 2SP taken from the literature (Lodi, Martello, Monaci 2002)
- Dataset 2: 500 instances for 2BP taken from the literature (Lodi, Martello, Vigo 2004)

- C++ programming language
- Pentium IV 1.6 GHz CPU, 512 MB RAM
- Linux operating system
- Time limit of 1 hour

- Benchmarks:
  - CPLEX 8.1 general purpose solver (Lodi, Martello, Vigo formulation)
  - Reformulation as CCLP, solved with a B&P algorithm for the CPMP (Ceselli, Righini, "A branch-and-price algorithm for the CPMP", Networks, 2005).

# Experimental results: dual bounds

- Comparison of LP, combinatorial (CUT) and CG bounds: dataset 1

| Instance Class | LP bound | CUT bound | CG bound |
|---|---|---|---|
| BENG (10) | 93,25% | 99,53% | 95,31% |
| GCUT (4) | 85,09% | 89,43% | 99,86% |
| NGCUT (12) | 87,43% | 95,79% | 94,90% |
| CGCUT (3) | 95,34% | 95,34% | 93,05% |
| HT (9) | 92,20% | 99,60% | 95,91% |
| Avg. | 90,66% | 95,94% | 95,81% |

# Experimental results: dual bounds

- Comparison of LP, combinatorial (CUT) and CG bounds: dataset 2

| Class | LP bound | CUT bound | CG bound |
|---|---|---|---|
| 1 | 91,27% | 93,63% | 97,71% |
| 2 | 92,20% | 99,00% | 94,54% |
| 3 | 88,05% | 91,03% | 97,04% |
| 4 | 92,01% | 98,45% | 96,20% |
| 5 | 88,10% | 90,60% | 97,81% |
| 6 | 91,32% | 98,21% | 96,22% |
| 7 | 85,43% | 87,83% | 99,31% |
| 8 | 91,39% | 94,58% | 95,47% |
| 9 | 80,82% | 82,23% | 99,96% |
| 10 | 90,76% | 95,20% | 97,44% |
| Avg. | 89,13% | 93,08% | 97,17% |

| N | LP bound | CUT bound | CG bound |
|---|---|---|---|
| 20 | 86,67% | 93,69% | 96,24% |
| 40 | 88,19% | 92,78% | 97,21% |
| 60 | 89,55% | 92,90% | 97,38% |
| 80 | 89,75% | 92,30% | 97,34% |
| 100 | 91,51% | 93,71% | 97,68% |
| Avg. | 89,13% | 93,08% | 97,17% |

# Optimal solutions: dataset 1

- Branch-and-price and CPLEX 8.1 (used as an IP solver)

| | Branch and Price | | CPLEX 8.1 | |
|---|---|---|---|---|
| Instance Class | gap | time (s) | gap | time (s) |
| BENG (10) | 4,20% (7 inst.) | 32,50 (3 inst.) | 4,96% (7 inst.) | 24,47 (3 inst.) |
| GCUT (4) | 0.00% | 2,29 | 0.00% | 1,06 |
| NGCUT (12) | 0.00% | 0,05 | 0.00% | 0,09 |
| CGCUT (3) | 0.00% | 2,52 | 0.00% | 67,01 |
| HT (9) | 0.00% | 4,34 | 0.00% | 10,14 |

# Further developments

- Implementation issues (columns pool management, tree management, primal heuristics …).
- Experimental analysis on ORLib instances and new random instances.
- Handling rotations of the items.
- Algorithm for the PKP with incompatibility constraints.