Pseudo-polynomial formulations
for bin packing and cutting stock problems

Maxence Delorme\(^{(1)}\) Manuel Iori\(^{(2)}\)

\(^{(1)}\) DEI, University of Bologna, Italy,
\(^{(2)}\) DISMI, University of Modena and Reggio Emilia, Italy

COLGEN 2016 – Búzios
1. Introduction
2. Some C&P pseudo-polynomial formulations
3. How to improve?
4. Preliminary results on the Variable-Sized Bin Packing Problem
5. Conclusions and future research directions
Bin Packing and Cutting Stock

In this talk we study Cutting & Packing (C&P) problems from a computational point of view.

We focus on two (related) fundamental one-dimensional C&P problems:

- **Bin Packing Problem (BPP):** given $n$ items having width $w_j$ for $j = 1, \ldots, n$, and bins of capacity $c$, pack the items into the minimum number of bins.

- **Cutting Stock Problem (CSP):** given $m$ item types, each consisting of a demand $d_j$ of items of width $w_j$, and stocks of length $c$, cut the items from the minimum number of stocks.
Pseudo-polynomial formulations

Pseudo-polynomial formulations (position-indexed formulations, capacity-indexed formulations, time-indexed formulations, ...) model CSP and BPP instances by taking care of the position in which the item is cut from/packed into the bin:

- good lower bounds
- number of variables and constraints grows not only with $n$ but also with $c$

Intensively studied for C&P problems from the 1960s:

- as a basis for branch-and-price algorithms
- as stand-alone models for optimal problem solution
Pseudo-polynomial formulations

Pseudo-polynomial formulations (*position-indexed* formulations, *capacity-indexed* formulations, *time-indexed* formulations, ...) model CSP and BPP instances by taking care of the *position* in which the item is cut from/packed into the bin.

- good lower bounds
- number of variables and constraints grows not only with $n$ but also with $c$

Intensively studied for C&P problems from the 1960s:

- as a basis for branch-and-price algorithms
- as stand-alone models for optimal problem solution
Evolution of software for MILP

Pseudo-polynomial formulations are by far the main mathematical model in multi-dimensional C&P since Beasley (Operations Research, 1985).

For the 1-dim, case, from Delorme, Iori and Martello (EJOR, forthcoming):

- n. of selected CSP instances solved to optimality (average seconds, 2.66 GHz) using the ARC-FLOW formulation by Valério de Carvalho (Annals of OR, 1999) and different CPLEX versions

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>13 (366)</td>
<td>10 (420)</td>
<td>5 (570)</td>
<td>17 (268)</td>
<td>19 (162)</td>
<td>20 (65)</td>
<td>19 (117)</td>
<td>20 (114)</td>
</tr>
<tr>
<td>3600</td>
<td>16 (897)</td>
<td>15 (1210)</td>
<td>15 (2009)</td>
<td>20 (343)</td>
<td>20 (186)</td>
<td>20 (65)</td>
<td>19 (267)</td>
<td>20 (114)</td>
</tr>
</tbody>
</table>
Some C&P pseudo-polynomial formulations

Dynamic-Programming Flow Formulation (DP-FLOW)

Formally introduced for BPP only by Cambazard and O’Sullivan (LNCS, 2010)

- DP table: a BPP item $j$ is an horizontal layer, a weight $d$ is a vertical layer
- $x_{j,d,j+1,e} = \text{number of times arc } ((j, d), (j + 1, e)) \text{ is selected}$

Example ($c=9$):
Dynamic-Programming Flow Formulation (DP-FLOW)

Formally introduced for BPP only by Cambazard and O’Sullivan (LNCS, 2010)

- DP table: a BPP item $j$ is an horizontal layer, a weight $d$ is a vertical layer
- $x_{j,d,j+1,e} =$ number of times arc $((j,d),(j+1,e))$ is selected

$$\min \quad z$$

$$\sum_{((j,d),(j+1,e))\in \delta^+((j,d))} x_{j,d,j+1,e} - \sum_{((j-1,e),(j,d))\in \delta^-((j,d))} x_{j-1,e,j,d} = \begin{cases} z & \text{if } (j,d) = (0,0) \\ -z & \text{if } (j,d) = (n+1,c) \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{((j-1,d),(j,d+w_j))\in A} x_{j-1,d,j,d+w_j} = 1$$

$x_{j,d,j+1,e} \geq 0$ and integer
Aggregate Flow Formulation (ARC-FLOW)

Formally introduced for CSP by Valério de Carvalho (Annals of OR, 1999) as a basis for a B&P

- all DP horizontal layers are “aggregated” into a unique layer
- \( x_{de} = n \)\(^{\circ} \) of times arc \((d, e)\) is chosen (item of weight \(e - d\) or empty space)

\[\min z - \sum_{(d, e) \in \delta^-} x_{de} + \sum_{(e, f) \in \delta^+} x_{ef} = \begin{cases} z & \text{if } e = 0 \\ -z & \text{if } e = c \\ 0 & \text{if } e = 1, \ldots, c-1 \end{cases} \]

\[\sum_{(d, d+w_i) \in A} x_{de} \geq d_i \quad i = 1, \ldots, m \quad x_{de} \geq 0 \quad \text{and integer } \quad (d, e) \in A\]

Example (c=9):

- Item 4 repeated 2 times
- Item 3 repeated 2 times
- Item 2 repeated 2 times
Aggregate Flow Formulation (ARC-FLOW)

Formally introduced for CSP by Valério de Carvalho (Annals of OR, 1999) as a basis for a B&P

- all DP horizontal layers are “aggregated” into a unique layer
- $x_{de} = n^\circ$ of times arc $(d, e)$ is chosen (item of weight $e−d$ or empty space)

\[
\begin{align*}
\text{min} & \quad z - \sum (d, e) \in \delta^- (e) x_{de} + \sum (e, f) \in \delta^+ (e) x_{ef} \\
\text{s.t.} & \quad \sum (d, d+w_i) \in A x_{d, d+w_i} \geq d_i \quad i = 1, \ldots, m \\
& \quad x_{de} \geq 0 \text{ and integer } (d, e) \in A
\end{align*}
\]

Example ($c=9$):

- Example graph with 19 arcs (remove redundant arcs, sort items by non-increasing width)
- $0\rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9$
Aggregate Flow Formulation (ARC-FLOW)

Formally introduced for CSP by Valério de Carvalho (Annals of OR, 1999) as a basis for a B&P

- all DP horizontal layers are “aggregated” into a unique layer
- \( x_{de} = n \)° of times arc \((d, e)\) is chosen (item of weight \(e - d\) or empty space)

\[
\begin{align*}
\min z &= \sum_{(d, e) \in \delta^-} x_{de} + \sum_{(e, f) \in \delta^+} x_{ef} \\
&\quad \text{subject to } \sum_{(d, d+iw) \in A} x_{de} \geq d, i = 1, \ldots, m, \quad x_{de} \geq 0, \quad \text{integers } (d, e) \in A
\end{align*}
\]

Example (\(c=9\)):

- \(2 \times 2\)
- \(3 \times 2\)
- \(4 \times 2\)

Full graph (27 arcs)
Aggregate Flow Formulation (ARC-FLOW)

Formally introduced for CSP by Valério de Carvalho (Annals of OR, 1999) as a basis for a B&P

- all DP horizontal layers are “aggregated” into a unique layer
- \( x_{de} = n^o \) of times arc \((d, e)\) is chosen (item of weight \(e-d\) or empty space)

Example (\(c=9\)):

Reduced graph (19 arcs)

(remove redundant arcs, sort items by non-increasing width)
Aggregate Flow Formulation (ARC-FLOW)

Formally introduced for CSP by Valério de Carvalho (Annals of OR, 1999) as a basis for a B&P

- all DP horizontal layers are “aggregated” into a unique layer
- $x_{de} = n^o$ of times arc $(d, e)$ is chosen (item of weight $e-d$ or empty space)

$$\min \quad z$$

$$- \sum_{(d,e) \in \delta^-(e)} x_{de} + \sum_{(e,f) \in \delta^+(e)} x_{ef} = \begin{cases} z & \text{if } e = 0 \\ -z & \text{if } e = c \\ 0 & \text{if } e = 1, \ldots, c - 1 \end{cases}$$

$$\sum_{(d,d+w_i) \in A} x_{d,d+w_i} \geq d_i \quad i = 1, \ldots, m$$

$x_{de} \geq 0$ and integer \quad $(d, e) \in A$
ONE-CUT


- \( y_{pq} \) = number of times a piece of width \( p \) is cut into a left piece (item) of width \( q \) and a right piece (residual) of width \( p - q \)

\[
\begin{align*}
\text{min} & \quad z = \sum_{q \in W} c_{q} y_{q} \\
\text{s.t.} & \quad \sum_{p \in A}(q) y_{pq} + \sum_{p \in B}(q) y_{p} + q, p \geq d_{q} \\
& \quad \sum_{r \in C}(q) y_{qr} q \in (W \cup R) \setminus \{c\} \\
& \quad y_{pq} \geq 0 \text{ and integer} \quad p \in R, q \in W, p > q
\end{align*}
\]

\( y_{9,4} = 1 \)
\( y_{9,3} = 1 \)
\( y_{9,2} = 1 \)
ONE-CUT


- $y_{pq}$ = number of times a piece of width $p$ is cut into a left piece (item) of width $q$ and a right piece (residual) of width $p - q$

\[
\begin{align*}
9 & \\
4 & 5 & 9 \\
4 & 2 & 9 \\
3 & 6 & 9 \\
2 & 7 & 9 \\
2 & 4 & 9 \\
3 & 3 & 9 \\
4 & 2 & 9 \\
5 & 6 & 9 \\
3 & 6 & 9 \\
2 & 6 & 9 \\
1 & 6 & 9 \\
0 & 6 & 9 \\
\end{align*}
\]

\[
\begin{align*}
y_{6,4} &= 1 \\
y_{6,3} &= 1 \\
y_{6,2} &= 1 \\
\end{align*}
\]
ONE-CUT


- $y_{pq} =$ number of times a piece of width $p$ is cut into a left piece (item) of width $q$ and a right piece (residual) of width $p - q$
ONE-CUT


- \( y_{pq} \) = number of times a piece of width \( p \) is cut into a left piece (item) of width \( q \) and a right piece (residual) of width \( p - q \)

\[
\begin{align*}
\min \quad & z = \sum_{q \in W} y_{cq} \\
\text{s.t.} \quad & \sum_{p \in A(q)} y_{pq} + \sum_{p \in B(q)} y_{p+q,p} \geq d_q + \sum_{r \in C(q)} y_{qr} \quad q \in (W \cup R) \setminus \{c\} \\
& y_{pq} \geq 0 \text{ and integer} \quad p \in R, \ q \in W, \ p > q
\end{align*}
\]
Some C&P pseudo-polynomial formulations

Computational results for BPP/CSP

From Delorme, Iori and Martello (EJOR, forthcoming):

- n. of instances solved to optimality in one minute (average CPU seconds) for number of items n

<table>
<thead>
<tr>
<th>n</th>
<th>inst.</th>
<th>Pseudo-polynomial formulations</th>
<th>Branch(-and-cut)-and-price</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>DP-FLOW</td>
<td>ONE-CUT</td>
</tr>
<tr>
<td>50</td>
<td>165</td>
<td>165 (0.5)</td>
<td>165 (0.1)</td>
</tr>
<tr>
<td>100</td>
<td>271</td>
<td>271 (5.0)</td>
<td>271 (0.8)</td>
</tr>
<tr>
<td>200</td>
<td>359</td>
<td>292 (21.0)</td>
<td>358 (2.4)</td>
</tr>
<tr>
<td>300</td>
<td>393</td>
<td>243 (33.9)</td>
<td>385 (4.5)</td>
</tr>
<tr>
<td>400</td>
<td>425</td>
<td>193 (42.4)</td>
<td>408 (5.1)</td>
</tr>
<tr>
<td>500</td>
<td>414</td>
<td>169 (44.8)</td>
<td>394 (6.3)</td>
</tr>
<tr>
<td>750</td>
<td>433</td>
<td>120 (52.6)</td>
<td>401 (7.8)</td>
</tr>
<tr>
<td>1000</td>
<td>441</td>
<td>67 (56.4)</td>
<td>407 (8.1)</td>
</tr>
</tbody>
</table>

Total 2901 1520 (36.7) 2789 (5.0) 2840 (3.3) 1089 (42.4) 2152 (20.3) 2901 (0.2)

ONE-CUT and ARC-FLOW somehow competitive up to n=200
Computational results for BPP/CSP

From Delorme, Iori and Martello (EJOR, forthcoming):
- n. of instances solved to optimality in one minute (average CPU seconds) for bin capacity $c$

<table>
<thead>
<tr>
<th>$c$</th>
<th>inst.</th>
<th>DPFLOW</th>
<th>ONECUT</th>
<th>ARCFLOW</th>
<th>SCIP-BPP</th>
<th>VANCE</th>
<th>BELOV</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>223</td>
<td>205 (13.6)</td>
<td>223 (0.1)</td>
<td>223 (0.1)</td>
<td>86 (43.0)</td>
<td>162 (20.2)</td>
<td>223 (0.1)</td>
</tr>
<tr>
<td>75</td>
<td>240</td>
<td>208 (19.5)</td>
<td>240 (0.1)</td>
<td>240 (0.1)</td>
<td>96 (42.0)</td>
<td>176 (20.2)</td>
<td>240 (0.1)</td>
</tr>
<tr>
<td>100</td>
<td>234</td>
<td>185 (26.4)</td>
<td>234 (0.1)</td>
<td>234 (0.1)</td>
<td>89 (42.8)</td>
<td>172 (20.3)</td>
<td>234 (0.1)</td>
</tr>
<tr>
<td>120</td>
<td>241</td>
<td>168 (29.6)</td>
<td>241 (0.1)</td>
<td>241 (0.1)</td>
<td>91 (42.2)</td>
<td>181 (19.7)</td>
<td>241 (0.1)</td>
</tr>
<tr>
<td>125</td>
<td>251</td>
<td>174 (30.9)</td>
<td>251 (0.1)</td>
<td>251 (0.1)</td>
<td>101 (41.0)</td>
<td>192 (19.2)</td>
<td>251 (0.1)</td>
</tr>
<tr>
<td>150</td>
<td>240</td>
<td>143 (34.9)</td>
<td>240 (0.1)</td>
<td>240 (0.2)</td>
<td>95 (41.8)</td>
<td>181 (19.5)</td>
<td>240 (0.1)</td>
</tr>
<tr>
<td>200</td>
<td>246</td>
<td>127 (39.3)</td>
<td>246 (0.3)</td>
<td>246 (0.3)</td>
<td>99 (40.8)</td>
<td>184 (20.5)</td>
<td>246 (0.1)</td>
</tr>
<tr>
<td>300</td>
<td>237</td>
<td>79 (45.6)</td>
<td>237 (1.5)</td>
<td>237 (1.3)</td>
<td>80 (44.4)</td>
<td>172 (21.9)</td>
<td>237 (0.1)</td>
</tr>
<tr>
<td>400</td>
<td>245</td>
<td>71 (47.4)</td>
<td>245 (3.9)</td>
<td>245 (2.8)</td>
<td>95 (41.6)</td>
<td>184 (20.3)</td>
<td>245 (0.2)</td>
</tr>
<tr>
<td>500</td>
<td>243</td>
<td>56 (49.2)</td>
<td>241 (8.6)</td>
<td>242 (5.1)</td>
<td>77 (44.8)</td>
<td>179 (20.7)</td>
<td>243 (0.3)</td>
</tr>
<tr>
<td>750</td>
<td>249</td>
<td>55 (49.9)</td>
<td>211 (18.6)</td>
<td>229 (11.2)</td>
<td>91 (42.4)</td>
<td>183 (21.0)</td>
<td>249 (0.8)</td>
</tr>
<tr>
<td>1000</td>
<td>252</td>
<td>49 (51.5)</td>
<td>180 (25.3)</td>
<td>212 (17.6)</td>
<td>89 (42.5)</td>
<td>186 (20.7)</td>
<td>252 (0.1)</td>
</tr>
<tr>
<td>Total</td>
<td>2901</td>
<td>1520 (36.7)</td>
<td>2789 (5)</td>
<td>2840 (3.3)</td>
<td>1089 (42.4)</td>
<td>2152 (20.3)</td>
<td>2901 (0.2)</td>
</tr>
</tbody>
</table>

ONE-CUT and ARC-FLOW somehow competitive up to $c=400$
Equivalence of the formulations

All proposed formulations have the same continuous lower bound value as that provided by the *Gilmore and Gomory (Operations Research, 1963)* model (GG)

**Proposition (Valério de Carvalho (Annals of OR, 1999))**

*ARC-FLOW is equivalent to GG*

**Proposition**

*DP-FLOW is equivalent to ARC-FLOW*

Proof Skipped

**Proposition**

*ONE-CUT is equivalent to ARC-FLOW*

Proof based on decomposition of non-negative flows into paths and cycles
ARC-FLOW is included in ONE-CUT

We use an LP ARC-FLOW solution $[\bar{x}_{de}]$ to build an LP ONE-CUT solution $[\bar{y}_{pq}]$

First step, we transform ARC-FLOW into an equivalent Normal ARC-FLOW by replacing, for any $d$, original loss arcs $(d, d + 1)$ with $(d, c)$
ARC-FLOW is included in ONE-CUT

We use an LP ARC-FLOW solution \([\bar{x}_{de}]\) to build an LP ONE-CUT solution \([\bar{y}_{pq}]\)

Second step, we build \([\bar{y}_{pq}]\) using the normal Arc-Flow solution as:

\[
\bar{y}_{c-d,e-d} = \bar{x}_{de} \quad \forall (d,e) \in A, \ e \neq c
\]

ARC-FLOW LP solution \((z = 1.67)\):

Resulting ONE-CUT LP solution:

\[
\bar{y}_{9,3} = 1.67 \\
\bar{y}_{6,3} = 0.67 \\
\bar{y}_{6,2} = 1.00
\]

Example \((c=9)\):

\[
\begin{array}{c}
\text{4} \\
\times 1 \\
\text{3} \\
\times 3 \\
\text{2} \\
\times 1
\end{array}
\]
ONE-CUT is included in ARC-FLOW

We use an LP ONE-CUT solution \([\bar{y}_{pq}]\) to build an LP ARC-FLOW solution \([\bar{x}_{de}]\)

Set all \(\bar{x}_{d,e} = 0\);
For all \(\bar{y}_{p,q} > 0\) do

\[
\bar{x}_{c-p-c-p+q} += \bar{y}_{p,q};
\]

\[
\bar{x}_{c-p+q,c} += \bar{y}_{p,q};
\]

if \(p \neq c\)

\[
\bar{x}_{c-p,c} -= \bar{y}_{p,q};
\]

endif
end do;
ONE-CUT is included in ARC-FLOW

We use an LP ONE-CUT solution \([\tilde{y}_{pq}]\) to build an LP ARC-FLOW solution \([\tilde{x}_{de}]\)

Set all \(\tilde{x}_{d,e} = 0\);

For all \(\tilde{y}_{p,q} > 0\) do

\[
\begin{align*}
\tilde{x}_{c-p,c-p+q} & = \tilde{y}_{p,q}; \\
\tilde{x}_{c-p+q,c} & = \tilde{y}_{p,q}; \\
\text{if } p \neq c \\
\tilde{x}_{c-p,c} & = -\tilde{y}_{p,q}; \\
\text{endif}
\end{align*}
\]

end do;

Example (\(c=9\)):

\[
\begin{align*}
\tilde{y}_{9,2} & = 1.00 \\
\tilde{y}_{7,4} & = 1.00 \\
\tilde{y}_{9,3} & = 0.67 \\
\tilde{y}_{6,3} & = 0.67 \\
\tilde{y}_{9,4} & = 0.50 \\
\tilde{y}_{5,4} & = 0.50
\end{align*}
\]

LP solution for ONE-CUT (\(z = 2.17\)):
ONE-CUT is included in ARC-FLOW

We use an LP ONE-CUT solution \([\bar{y}_{pq}]\) to build an LP ARC-FLOW solution \([\bar{x}_{de}]\)

Set all \(\bar{x}_{d,e} = 0\);

For all \(\bar{y}_{p,q} > 0\) do

\[
\bar{x}_{c-p,c-p+q} \leftarrow \bar{y}_{p,q};
\]

\[
\bar{x}_{c-p+q,c} \leftarrow \bar{y}_{p,q};
\]

if \(p \neq c\)

\[
\bar{x}_{c-p,c} \leftarrow \bar{y}_{p,q};
\]

endif

end do;

Example \((c=9)\):

\[
\begin{array}{c}
\text{\(\bar{y}_{9,2} = 1.00\)} & \text{\(\bar{y}_{9,3} = 0.67\)} & \text{\(\bar{y}_{9,4} = 0.50\)} & \text{\(\bar{y}_{5,4} = 0.50\)} \\
\text{\(\bar{y}_{7,4} = 1.00\)} & \text{\(\bar{y}_{6,3} = 0.67\)} & \text{\(\bar{y}_{9,4} = 0.50\)} & \text{\(\bar{y}_{5,4} = 0.50\)} \\
\end{array}
\]

Resulting LP solution for ARC-FLOW:
How to improve?

ONE-CUT is included in ARC-FLOW

We use an LP ONE-CUT solution \([\bar{y}_{pq}]\) to build an LP ARC-FLOW solution \([\bar{x}_{de}]\)

Set all \(\bar{x}_{d,e} = 0\);
For all \(\bar{y}_{p,q} > 0\) do
\[\begin{align*}
\bar{x}_{c-p,c-p+q} &= \bar{y}_{p,q}; \\
\bar{x}_{c-p+q,c} &= \bar{y}_{p,q}; \\
&\text{if } p \neq c \\
\bar{x}_{c-p,c} &= \bar{y}_{p,q};
\end{align*}\]
end if
end do;

Example \((c=9)\):

\[
\begin{array}{c}
\text{4} \times 2 \\
\text{3} \times 3 \\
\text{2} \times 1
\end{array}
\]

LP solution for ONE-CUT \((z = 2.17)\):

\[
\begin{align*}
\bar{y}_{9,2} &= 1.00 \\
\bar{y}_{7,4} &= 1.00 \\
\bar{y}_{9,3} &= 0.67 \\
\bar{y}_{6,3} &= 0.67 \\
\bar{y}_{9,4} &= 0.50 \\
\bar{y}_{5,4} &= 0.50
\end{align*}
\]

Resulting LP solution for ARC-FLOW:

\[
\begin{array}{c}
0 \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
8 \\
9
\end{array}
\]
How to improve?

ONE-CUT is included in ARC-FLOW

We use an LP ONE-CUT solution \([\bar{y}_{pq}]\) to build an LP ARC-FLOW solution \([\bar{x}_{de}]\)

Set all \(\bar{x}_{d,e} = 0\);
For all \(\bar{y}_{p,q} > 0\) do
  \(\bar{x}_{c-p,c-p+q} += \bar{y}_{p,q};\)
  \(\bar{x}_{c-p+q,c} += \bar{y}_{p,q};\)
  if \(p \neq c\)
    \(\bar{x}_{c-p,c} -= \bar{y}_{p,q};\)
  endif
end do;

Example \((c=9)\):

\[
\begin{array}{c}
\text{Orange} & \times 2 \\
\text{Red} & \times 3 \\
\text{Green} & \times 1 \\
\end{array}
\]

LP solution for ONE-CUT \((z = 2.17)\):

\[
\begin{array}{c|c|c}
\bar{y}_{9,2} & = 1.00 & \text{Green} \\
\bar{y}_{7,4} & = 1.00 & \text{Orange} \quad \text{Red} \\
\bar{y}_{9,3} & = 0.67 & \text{Red} \\
\bar{y}_{6,3} & = 0.67 & \text{Orange} \quad \text{Red} \\
\bar{y}_{9,4} & = 0.50 & \text{Orange} \\
\bar{y}_{5,4} & = 0.50 & \text{Orange} \quad \text{Green} \\
\end{array}
\]

Resulting LP solution for ARC-FLOW:
**ONE-CUT is included in ARC-FLOW**

We use an LP ONE-CUT solution $[\bar{y}_{pq}]$ to build an LP ARC-FLOW solution $[\bar{x}_{de}]$.

ONE-CUT does not forbid recutting an item

$$\bar{y}_{9,6} = 2$$

$$\bar{y}_{6,3} = 1$$

Which would result into a negative flow in ARC-FLOW

We solve this by using an equivalent Normal ONE-CUT formulation where we add

$$\sum_{p \in B(q)} y_{p+q,p} \geq \sum_{r \in C(q)} y_{qr} \quad q \in W$$
Improving ARC-FLOW with a Multi-Graph

Use of a multi-graph to possibly increase arcs length

- *Brandão and Pedroso (Computers & Operations Research, 2016)*
- \( \chi^j_{de} = n^o \) of times item \( j \) is packed as an arc \( (d, e) \)
- enlarge width of each arc \( (d, e, j) \) by solving a subset sum problem
- remove arcs \( (d, e, j) \) if there is another arc \( (d', e', j) \) s.t. \( d' \geq d, e' \leq e \).
Improving ARC-FLOW with a Multi-Graph

Use of a multi-graph to possibly increase arcs length

- \( Brandão and Pedroso (Computers & Operations Research, 2016) \)
- \( x^j_{de} = n^o \) of times item \( j \) is packed as an arc \((d, e)\)
- enlarge width of each arc \((d, e, j)\) by solving a subset sum problem
- remove arcs \((d, e, j)\) if there is another arc \((d', e', j)\) s.t. \( d' \geq d, e' \leq e \).

Example \((c=9)\):

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
<th>Multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td>\times 2</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>\times 2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>\times 2</td>
</tr>
</tbody>
</table>

Multi-graph

\[ 0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \]
Improving ARC-FLOW with a Multi-Graph

Use of a multi-graph to possibly increase arcs length

- Brandão and Pedroso (Computers & Operations Research, 2016)
- $x_{de}^j = n^\circ$ of times item $j$ is packed as an arc $(d, e)$
- enlarge width of each arc $(d, e, j)$ by solving a subset sum problem
- remove arcs $(d, e, j)$ if there is another arc $(d', e', j)$ s.t. $d' \geq d$, $e' \leq e$.

Example ($c=9$):
Improving ARC-FLOW with a Multi-Graph

Use of a multi-graph to possibly increase arcs length

- Brandão and Pedroso (Computers & Operations Research, 2016)
- $x^j_{de} = n^\circ$ of times item $j$ is packed as an arc $(d, e)$
- enlarge width of each arc $(d, e, j)$ by solving a subset sum problem
- remove arcs $(d, e, j)$ if there is another arc $(d', e', j)$ s.t. $d' \geq d$, $e' \leq e$.

Example ($c=9$):

```
0 1 2 3 4 5 6 7 8 9
```

Multi-graph

```
Example (c=9):

4 x 2
3 x 2
2 x 2
```
Improving ARC-FLOW with a Multi-Graph

Use of a multi-graph to possibly increase arcs length

- Brandão and Pedroso (Computers & Operations Research, 2016)
- $x^j_{de} = n^o$ of times item $j$ is packed as an arc $(d, e)$
- Enlarge width of each arc $(d, e, j)$ by solving a subset sum problem
- Remove arcs $(d, e, j)$ if there is another arc $(d', e', j)$ s.t. $d' \geq d$, $e' \leq e$.

Example ($c=9$):

- 4 (x 2)
- 3 (x 2)
- 2 (x 2)

-Multi-graph

8 arcs (instead of the original 27)

- **normal patterns** align items to the left to decrease candidate optimal solutions
- **Meet-in-the-Middle (MIM) patterns** align items either to the left or to the right according to a threshold cut value $t$
Improving ARC-FLOW with Meet-in-the-Middle Patterns

Key note features of the MIM patterns:

- computed by easy DP that also finds the best threshold cut value $t$
- same time complexity than normal patterns
- size is never higher and usually much smaller
- further reduction criteria
- when used in ARC-FLOW for CSP, 50% average reduction in the number of arcs and 97% max reduction
The case of Variable-Sized (and Cost) BPP (VSBPP)

VSBPP is the BPP variant in which each bin $k \in K$ has capacity $c_k$ and cost $\gamma_k$.

ARC-FLOW is easily adapted by including new arcs $(0, c_{\text{max}} - c_k)$ for each $k \in K$ (slight variation of Valério de Carvalho (EJOR, 2002)).

Proposition

$\text{ONE-CUT(VSBPP)}$ is equivalent to $\text{ARC-FLOW(VSBPP)}$.

Improvements based on Multi-graph and MIM can be directly used.
### Preliminary computational results

Number of VSBPP/BPP instances solved in 1 hour (3.10GHz, 4 cores, Cplex 12.6)

<table>
<thead>
<tr>
<th>Instance</th>
<th>$c_{\text{max}}$</th>
<th>#inst</th>
<th>ARC-FLOW</th>
<th>Multi-Graph</th>
<th>MIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monaci</td>
<td>150</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Perboli 2</td>
<td>300</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Perboli 3</td>
<td>120</td>
<td>480</td>
<td>480</td>
<td>480</td>
<td>480</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Belov 1</td>
<td>10000</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>87</td>
<td>57</td>
<td>50</td>
</tr>
<tr>
<td>Belov 2</td>
<td>10000</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>68</td>
<td>53</td>
<td>50</td>
</tr>
<tr>
<td>AI 200</td>
<td>2500</td>
<td>50</td>
<td>49</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>233</td>
<td>45</td>
<td>50</td>
</tr>
<tr>
<td>AI 400</td>
<td>10000</td>
<td>50</td>
<td>3</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3480</td>
<td>3333</td>
<td>10</td>
</tr>
<tr>
<td>ANI 200</td>
<td>2500</td>
<td>50</td>
<td>27</td>
<td>50</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2064</td>
<td>30</td>
<td>32</td>
</tr>
<tr>
<td>ANI 400</td>
<td>10000</td>
<td>50</td>
<td>0</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3602</td>
<td>3465</td>
<td>2</td>
</tr>
<tr>
<td>Sum/Avg</td>
<td>1140</td>
<td>1019</td>
<td>1060</td>
<td>267 K</td>
<td>1055</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1055</td>
<td>776</td>
<td>1034</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3602</td>
<td>3465</td>
<td>931</td>
</tr>
</tbody>
</table>

Preliminary results on the Variable-Sized Bin Packing Problem
Don’t blame capacity

Pseudo-polynomial formulations are interesting both in theory and in practice.

Of course, it is inadvisable to use them if capacity is too large.

Studying combinations with B&P is of great interest.
Conclusions and future research directions

MILP solver matters

Preliminary results on the $P||\sum w_j C_j$, by *Arthur Kramer, Dell’Amico, Iori* (2016, work in progress)

- Selected instances solved with ARC-FLOW and 300 seconds (at 2.4GHz)

<table>
<thead>
<tr>
<th>LP method</th>
<th>Solver</th>
<th>#inst</th>
<th>#opt</th>
<th>#nodes</th>
<th>seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual Simplex</td>
<td>Gurobi 6.5.1</td>
<td>260</td>
<td>212</td>
<td>0.2</td>
<td>81.6</td>
</tr>
<tr>
<td></td>
<td>Cplex 12.6.2</td>
<td>260</td>
<td>195</td>
<td>0.5</td>
<td>90.6</td>
</tr>
<tr>
<td>Primal Simplex</td>
<td>Gurobi 6.5.1</td>
<td>260</td>
<td>250</td>
<td>5.6</td>
<td>38.3</td>
</tr>
<tr>
<td></td>
<td>Cplex 12.6.2</td>
<td>260</td>
<td>224</td>
<td>8.9</td>
<td>58.6</td>
</tr>
<tr>
<td>Barrier</td>
<td>Gurobi 6.5.1</td>
<td>260</td>
<td>258</td>
<td>13.2</td>
<td>26.9</td>
</tr>
<tr>
<td></td>
<td>Cplex 12.6.2</td>
<td>260</td>
<td>225</td>
<td>22.0</td>
<td>54.5</td>
</tr>
</tbody>
</table>
Conclusions and future research directions

Rectangle packing

In two-dimensional orthogonal C&P problems, good computational results have been obtained by using pseudo-polynomial models in primal decomposition methods:

- Côté, Dell’Amico, Iori, Combinatorial Benders’ Cuts for the Strip Packing Problem, (Operations Research, 2014)

Use B&P as one of the engines in this kind of approaches? (Several instances with 20 items are open problems since decades)
The MIRUP conjecture

MIRUP conjecture holds for both BPP and CSP and states that the difference between $z_{opt}$ and the optimal rounded-up GG solution value is at most 1.

- Kartak, Ripatti, Scheithauer, Kurz, Minimal proper non-IRUP instances of the one-dimensional cutting stock problem, Discrete Applied Mathematics (2015)

Can pseudo-polynomial formulations help in this context?
Thank you very much for your attention