Interior Point Methods
and Column Generation

Jacek Gondzio
Email: J.Gondzio@ed.ac.uk
URL: http://www.maths.ed.ac.uk/~gondzio

Bromont, June 2012
Outline

- **Part 1: IPMs for Optimization**
  - IPM tricks: log barrier, central path
  - polynomial complexity
  - optimal partition

- **Part 2: Warmstarting IPMs**

- **Part 3: Column Generation with IPM**
  - cutting stock problem
  - vehicle routing problem with time windows

- **Conclusions**
Part 1:

Interior Point Methods for Optimization
Shocking mathematical concept:
Take \textit{linear} optimization problem and add \textit{nonlinear} function to the objective.

A step against common sense and centuries of mathematical practice:

\textbf{“nonlinearize” linear problem}
Logarithmic barrier

\[- \ln x_j \]

“replaces” the inequality

\[ x_j \geq 0 \, . \]

Observe that

\[
\min e^{- \sum_{j=1}^{n} \ln x_j} \quad \Longleftrightarrow \quad \max \prod_{j=1}^{n} x_j
\]

The minimization of \(- \sum_{j=1}^{n} \ln x_j\) is equivalent to the maximization of the product of distances from all hyper-planes defining the positive orthant: it prevents all \(x_j\) from approaching zero.

Bromont, June 2012
LP Problem: \[ \min c^T x \quad \text{s.t.} \quad Ax = b, \quad x \geq 0. \]

LP Barrier Prob: \[ \min c^T x - \mu \sum_{j=1}^{n} \ln x_j \quad \text{s.t.} \quad Ax = b. \]

Lagrangian: \[ L(x, y, \mu) = c^T x - y^T (Ax - b) - \mu \sum_{j=1}^{n} \ln x_j, \]

Stationarity: \[ \nabla_x L(x, y, \mu) = c - A^T y - \mu X^{-1} e = 0 \]
\[ \nabla_y L(x, y, \mu) = Ax - b = 0. \]

Denote: \[ s = \mu X^{-1} e, \quad \text{i.e.} \quad XSe = \mu e. \]
Complementarity in the Interior Point Method

The first order optimality conditions (FOC)

\[ Ax = b, \]
\[ A^T y + s = c, \]
\[ XSe = \mu e, \]
\[ x, s \geq 0, \]

where \( X = \text{diag}\{x_j\}, \ S = \text{diag}\{s_j\} \) and \( e = (1, \ldots, 1) \in \mathbb{R}^n \).

Analytic centre (\( \mu \)-centre): a (unique) point \( (x(\mu), y(\mu), s(\mu)) \), \( x(\mu) > 0, \ s(\mu) > 0 \) that satisfies FOC.

The interior point method gradually reduces the complementarity products

\[ x_j \cdot s_j \approx \mu \rightarrow 0 \quad \forall j = 1, 2, \ldots, n. \]
Interior Point Methods

Theory: convergence in $O(\sqrt{n})$ or $O(n)$ iterations

Practice: convergence in $O(\log n)$ iterations

Expected number of IPM iterations:

<table>
<thead>
<tr>
<th>Problem Dimension</th>
<th>LP</th>
<th>QP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>5 - 10</td>
<td>5 - 10</td>
</tr>
<tr>
<td>10,000</td>
<td>10 - 15</td>
<td>10 - 15</td>
</tr>
<tr>
<td>100,000</td>
<td>15 - 20</td>
<td>10 - 15</td>
</tr>
<tr>
<td>1,000,000</td>
<td>20 - 25</td>
<td>15 - 20</td>
</tr>
<tr>
<td>10,000,000</td>
<td>25 - 30</td>
<td>15 - 20</td>
</tr>
<tr>
<td>100,000,000</td>
<td>30 - 35</td>
<td>20 - 25</td>
</tr>
<tr>
<td>1000,000,000</td>
<td>35 - 40</td>
<td>20 - 25</td>
</tr>
</tbody>
</table>

... but one iteration may be expensive!
Complementarity \[ x_j \cdot s_j = 0 \quad \forall j = 1, 2, \ldots, n. \]

Simplex Method guesses an optimal partition:

For basic variables, \( s_B = 0 \) and
\[ (x_B)_j \cdot (s_B)_j = 0 \quad \forall j \in \mathcal{B}. \]

For non-basic variables, \( x_N = 0 \) hence
\[ (x_N)_j \cdot (s_N)_j = 0 \quad \forall j \in \mathcal{N}. \]

Interior Point Method uses \( \varepsilon \)-mathematics:

Replace \( x_j \cdot s_j = 0 \quad \forall j = 1, 2, \ldots, n \)
by \( x_j \cdot s_j = \mu \quad \forall j = 1, 2, \ldots, n. \)

Force convergence \( \mu \to 0. \)
First Order Optimality Conditions

Simplex Method:

\[ Ax = b \]
\[ A^T y + s = c \]
\[ XSe = 0 \]
\[ x, s \geq 0. \]

Interior Point Method:

\[ Ax = b \]
\[ A^T y + s = c \]
\[ XSe = \mu e \]
\[ x, s \geq 0. \]

Basic: \( x > 0, s = 0 \)
Nonbasic: \( x = 0, s > 0 \)

Bromont, June 2012
Part 2:

Warmstarting IPMs
A need to solve a sequence of similar problems

- column generation
- cutting plane methods
- subproblems in the block-angular LPs (Dantzig-Wolfe decomp., Benders decomp.)
- B&B, (and B&Cut, B&Cut&Price, etc)
- SQP
- any sequence of similar problems
  example: computing efficient frontier in Markowitz portfolio optimization
Warm Starts Which method should be used?
- Simplex Method, or
- Interior Point Method.

When is the Simplex Method better?
→ few indices change optimal partition
B & B, adding one cut in CPM, etc.

When is the Interior Point Method better?
→ many indices change optimal partition
adding many cuts in CPM,
dealing with a general change of problem data, etc

Conjecture:
The more changes in the (large) problem
the more attractive IPM-based warm starts are.

Bromont, June 2012
Difficulty of IPM Warm Starts

Modified Problem

Original Problem
Warm Starting in 1990+

Goffin & Vial et al., development of ACCPM 1990+
G. & Sarkissian, development of PDCGM in 1995

ACCPM  Analytic Centre Cutting Plane Method
PDCGM  Primal-Dual Column Generation Method
Warmstarting Heuristic

Idea: Start close to the (new) central path, not close to the (old) solution
Warm Start with $\mu$-centres

Old Problem: \[ \min \ c_0^T x + \frac{1}{2} x^T Q_0 x \]
\[ \text{s.t. } A_0 x = b_0, \]
\[ x \geq 0, \]

New Problem: \[ \min \ c^T x + \frac{1}{2} x^T Q x \]
\[ \text{s.t. } A x = b, \]
\[ x \geq 0, \]

We assume:
\[ c \approx c_0, \ Q \approx Q_0, \ A \approx A_0, \ b \approx b_0. \]
Warm Starting in 2000+

Colombo & Grothey, follow-up reports in 09,10
Benson & Mahanta, report in 2009
Ordonez & Waltz, report in 2009
Lemma. Let \((x, y, s) \in \mathcal{N}_{-\infty}(\gamma_0)\) for problem (LP) then the full Newton step \((\Delta x, \Delta y, \Delta s)\) in the perturbed problem (\(\tilde{\text{LP}}\)) is feasible and
\[
(x + \Delta x, y + \Delta y, s + \Delta s) \in \tilde{\mathcal{N}}_{-\infty}(\gamma)
\]
provided that
\[
\delta_{bc} = \|\xi_c\|_2 + \|A^T(AA^T)^{-1}\xi_b\|_2 \leq \|P\|_{\infty} \frac{\gamma_0}{1 + 1/\gamma^\mu},
\]
where
\[
P = I - S^{-1}A^T(AXS^{-1}A^T)^{-1}AX, \quad \xi_b = \tilde{b} - Ax, \quad \xi_c = \tilde{c} - A^Ty - s.
\]
LOQO vs OOPS warmstarting NETLIB problems


→ Unblocking technique ...

Average savings:

- LOQO (B&S, 2007) 20–30 %
- OOPS (G&G, 2008) 50–70 %

Bromont, June 2012
Part 3:

Primal-Dual Column Generation Method

Joint work with two PhD students:

Pablo Gonzalez-Brevis and Pedro Munari

Bromont, June 2012
Column Generation (CG)

MP

RMP

newRMP
Column Generation (CG)

Consider an LP, called the master problem (MP):

\[
\begin{align*}
  z^* &:= \min \sum_{j \in N} c_j \lambda_j, \\
  \text{s.t.} & \sum_{j \in N} a_j \lambda_j = b, \\
  & \lambda_j \geq 0, \quad \forall j \in N.
\end{align*}
\]

- \( N \) is too big;
- The columns \( a_j \) are implicit elements of \( \mathcal{A} \);
- We know how to generate them!
**CG:** Restricted master problem (RMP): $\overline{N} \subset N$

$$z_{RMP} := \min \sum_{j \in \overline{N}} c_j \lambda_j,$$

s.t. $$\sum_{j \in \overline{N}} a_j \lambda_j = b,$$

$$\lambda_j \geq 0, \quad \forall j \in \overline{N}.$$ 

- Optimal $\bar{\lambda}$ for the RMP $\Rightarrow$ feasible $\hat{\lambda}$ for the MP;
- $\hat{\lambda}_j = \bar{\lambda}_j$, $\forall j \in \overline{N}$, and $\hat{\lambda}_j = 0$ otherwise;
- Hence, $z^* \leq z_{RMP} = UB$ (Upper Bound).
- How to know it is optimal?
  - Call the oracle!
**CG:**

- Oracle: check the feasibility of the dual $\overline{u}$;
- Reduced costs: $s_j = c_j - \overline{u}^T a_j$, $\forall j \in N$;
- But the columns are not explicit and, hence,
  $$z_{SP} := \min\{c_j - \overline{u}^T a_j | a_j \in A\}.$$  
- (we reset $z_{SP} := 0$, if $z_{SP} > 0$);
- Lower Bound: $LB = z_{RMP} + \kappa z_{SP} \leq z^*$, where
  $$\kappa \geq \sum_{i \in N} \lambda_i^*,$$
- If $z_{SP} < 0$, then new columns are generated;
- Otherwise, an optimal solution of the MP was found!
Appealing features of IPMs

- Use IPM to solve the RMP: → no degeneracy issues
- Terminate RMP solution early: → get stable dual solution $\bar{u}$
PDCGM Algorithm Parameters: $\varepsilon_{\text{max}}, D, \delta, \kappa$

1. set $\text{LB} = -\infty$, $\text{UB} = \infty$, $\text{gap} = \infty$, $\varepsilon = 0.5$;
2. while ($\text{gap} > \delta$) do
3. find a well-centred $\varepsilon$-optimal $(\tilde{\lambda}, \tilde{u})$ of the RMP;
4. $\text{UB} = \tilde{z}_{RMP}$;
5. call the oracle with the query point $\tilde{u}$;
6. $\text{LB} = \kappa\tilde{z}_{SP} + b^T\tilde{u}$;
7. $\text{gap} = (\text{UB} - \text{LB})/(1 + |\text{UB}|)$;
8. $\varepsilon = \min\{\varepsilon_{\text{max}}, \text{gap}/D\}$;
9. if $(\tilde{z}_{SP} < 0)$ then add new columns into the RMP;
10. end(while)
CSP: Column Generation Formulation

Gilmore and Gomory (1961) formulation:

\[
\begin{align*}
\min & \quad \sum_{p \in P} \lambda_p, \\
\text{s.t.} & \quad \sum_{p \in P} a_p \lambda_p \geq d, \\
& \quad \lambda_p \geq 0 \text{ and integer, } \forall p \in P.
\end{align*}
\]

- Columns are cutting patterns;
- We do not need to enumerate all of them;
- They can be dynamically generated knapsack problem.
VRPTW: Column Generation Formulation

Desrochers et al. (1992):

\[
\min \sum_{p \in P} c_p \lambda_p \\
\text{s.t. } \sum_{p \in P} a_p \lambda_p = 1, \\
\lambda_p \in \{0, 1\}, \quad \forall p \in P.
\]

- Columns are possible vehicle paths;
- The columns can be dynamically generated shortest path problem with resource constraints.
Computational experiments

Solving LP relaxations

Comparison of:

- Standard column generation (**SCG**):
  - simplex-type methods of IBM/CPLEX v.12.1.

- Primal dual column generation (**PDCGM**):
  - interior point solver HOPDM.

- Analytic centre cutting plane (**ACCPM**):
  - open-source solver OBOE/COIN-OR.
## Cutting stock problem

<table>
<thead>
<tr>
<th>Cuts</th>
<th>Class</th>
<th>SCG</th>
<th>PDCGM</th>
<th>ACCPM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>iters</td>
<td>iters</td>
<td>iters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>time</td>
<td>time</td>
</tr>
<tr>
<td>10</td>
<td>Small</td>
<td>150</td>
<td>102</td>
<td>253</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>2.1</td>
<td>26.1</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>251</td>
<td>158</td>
<td>368</td>
</tr>
<tr>
<td></td>
<td></td>
<td>77.0</td>
<td>18.3</td>
<td>148.7</td>
</tr>
<tr>
<td>50</td>
<td>Small</td>
<td>71</td>
<td>63</td>
<td>277</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.1</td>
<td>3.8</td>
<td>106.3</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>134</td>
<td>97</td>
<td>400</td>
</tr>
<tr>
<td></td>
<td></td>
<td>58.2</td>
<td>23.1</td>
<td>277.6</td>
</tr>
<tr>
<td>100</td>
<td>Small</td>
<td>54</td>
<td>54</td>
<td>308</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2</td>
<td>7.3</td>
<td>221.8</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>101</td>
<td>82</td>
<td>449</td>
</tr>
<tr>
<td></td>
<td></td>
<td>67.8</td>
<td>31.5</td>
<td>525.2</td>
</tr>
</tbody>
</table>

262 instances:
178 small \((m \leq 199)\), 84 large \((m \geq 200)\)

http://www.tu-dresden.de/~capad/
### CSP: Larger Instances: BPP-U09??? family

<table>
<thead>
<tr>
<th>Instance</th>
<th>m</th>
<th>SCG-Iter</th>
<th>SCG-Time</th>
<th>PDCGM-Iter</th>
<th>PDCGM-Time</th>
<th>ACCPM-Iter</th>
<th>ACCPM-Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>U09498</td>
<td>1005</td>
<td>548</td>
<td>12947</td>
<td>293</td>
<td>5678</td>
<td>762</td>
<td>21254</td>
</tr>
<tr>
<td>U09513</td>
<td>975</td>
<td>518</td>
<td>9904</td>
<td>267</td>
<td>4277</td>
<td>779</td>
<td>19362</td>
</tr>
<tr>
<td>U09528</td>
<td>945</td>
<td>541</td>
<td>9173</td>
<td>276</td>
<td>4924</td>
<td>740</td>
<td>15920</td>
</tr>
<tr>
<td>U09543</td>
<td>915</td>
<td>506</td>
<td>7798</td>
<td>263</td>
<td>3724</td>
<td>723</td>
<td>13449</td>
</tr>
<tr>
<td>U09558</td>
<td>885</td>
<td>482</td>
<td>5585</td>
<td>265</td>
<td>2730</td>
<td>683</td>
<td>10861</td>
</tr>
<tr>
<td>U09573</td>
<td>855</td>
<td>473</td>
<td>4771</td>
<td>230</td>
<td>2054</td>
<td>672</td>
<td>9794</td>
</tr>
<tr>
<td>U09588</td>
<td>825</td>
<td>467</td>
<td>4950</td>
<td>247</td>
<td>1649</td>
<td>658</td>
<td>9376</td>
</tr>
<tr>
<td>U09603</td>
<td>795</td>
<td>465</td>
<td>3962</td>
<td>237</td>
<td>1668</td>
<td>627</td>
<td>7504</td>
</tr>
</tbody>
</table>

Bromont, June 2012
### Vehicle routing problem with time windows

<table>
<thead>
<tr>
<th>Cuts</th>
<th>Class</th>
<th>SCG iters</th>
<th>SCG time</th>
<th>PDCGM iters</th>
<th>PDCGM time</th>
<th>ACCPM iters</th>
<th>ACCPM time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Small</td>
<td>26</td>
<td>0.3</td>
<td>22</td>
<td><strong>0.2</strong></td>
<td>94</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>67</td>
<td>6.2</td>
<td>38</td>
<td><strong>2.6</strong></td>
<td>122</td>
<td>5.7</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>188</td>
<td>114.1</td>
<td>73</td>
<td><strong>41.6</strong></td>
<td>171</td>
<td>92.1</td>
</tr>
<tr>
<td>100</td>
<td>Small</td>
<td>12</td>
<td>0.2</td>
<td>17</td>
<td>0.2</td>
<td>92</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>26</td>
<td>3.0</td>
<td>23</td>
<td><strong>1.7</strong></td>
<td>120</td>
<td>5.8</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>65</td>
<td>42.4</td>
<td>38</td>
<td><strong>21.5</strong></td>
<td>166</td>
<td>87.5</td>
</tr>
</tbody>
</table>

87 instances:
29 small \( (n = 25) \), 29 med \( (n = 50) \), 29 large \( (n = 100) \)

http://www2.imm.dtu.dk/~jla/solomon.html
### VRPTW: Larger Instances

<table>
<thead>
<tr>
<th>Instance</th>
<th>n</th>
<th>SCG</th>
<th>PDCGM</th>
<th>ACCPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_1_2_1</td>
<td>200</td>
<td>57</td>
<td>45</td>
<td>423</td>
</tr>
<tr>
<td>C_1_2_1</td>
<td>200</td>
<td>85</td>
<td>29</td>
<td>169</td>
</tr>
<tr>
<td>RC_1_2_1</td>
<td>200</td>
<td>67</td>
<td>57</td>
<td>385</td>
</tr>
<tr>
<td>R_1_4_1</td>
<td>400</td>
<td>131</td>
<td>84</td>
<td>636</td>
</tr>
<tr>
<td>C_1_4_1</td>
<td>400</td>
<td>137</td>
<td>53</td>
<td>272</td>
</tr>
<tr>
<td>RC_1_4_1</td>
<td>400</td>
<td>189</td>
<td>113</td>
<td>521</td>
</tr>
<tr>
<td>R_1_6_1</td>
<td>600</td>
<td>222</td>
<td>118</td>
<td>897</td>
</tr>
<tr>
<td>C_1_6_1</td>
<td>600</td>
<td>183</td>
<td>48</td>
<td>482</td>
</tr>
<tr>
<td>RC_1_6_1</td>
<td>600</td>
<td>258</td>
<td>150</td>
<td>923</td>
</tr>
</tbody>
</table>
Integer optimization
J. Gondzio  

**Integer VRPTW solved to optimality**  
**Branch-Price-and-Cut, Pedro Munari’s PhD**  

<table>
<thead>
<tr>
<th>Problem</th>
<th>DLH08 cuts</th>
<th>nodes</th>
<th>time</th>
<th>I-PDCGM cuts</th>
<th>nodes</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>C103</td>
<td>0</td>
<td>1</td>
<td>28</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>C104</td>
<td>0</td>
<td>1</td>
<td>86</td>
<td>0</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>RC103</td>
<td>262</td>
<td>5</td>
<td>541</td>
<td>162</td>
<td>5</td>
<td>429</td>
</tr>
<tr>
<td>RC104</td>
<td>437</td>
<td>21</td>
<td>11773</td>
<td>251</td>
<td>7</td>
<td>3436</td>
</tr>
<tr>
<td>R103</td>
<td>53</td>
<td>1</td>
<td>20</td>
<td>15</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>R104</td>
<td>391</td>
<td>11</td>
<td>3103</td>
<td>216</td>
<td>7</td>
<td>949</td>
</tr>
</tbody>
</table>

DLH08: Desaulniers, Lessard & Hadjar,  
*Transportation Science* **42** (2008) **387-404.**


Bromont, June 2012
Conclusions

A completely new perspective is needed to exploit the insight offered by IPMs in a number of combinational optimization applications:

- column generation
- cutting plane methods
- B & B, (and B & Cut, B & Cut & Price, etc)

**Warmstarting works well** in the CG context: problems are re-optimized in **3-5** IPM iterations
References


Example: Cutting Stock Problem (CSP)

A set $N$ of large pieces of wood of length $W$ is given. We need to cut them into smaller pieces. We need $d_j$ units of small piece $j \in M$ of length $w_j$.

Minimize the number of units of large pieces of wood.

Define binary variable $y_i$ which takes value 1 if $i$-th large piece of wood is cut and 0 if it is not used.

Define integer variable $x_{ij}$ which determines the number of units of small piece of wood $j \in M$ obtained by cutting the large piece $i \in N$. 
Cutting Stock Problem (CSP)

Kantorovich’s formulation:

\[
\begin{align*}
\min \quad & \sum_{i \in N} y_i \\
\text{s.t.} \quad & \sum_{i \in N} x_{ij} \geq d_j \quad \forall j \in M, \\
\sum_{j \in M} x_{ij} w_j & \leq W y_i \quad \forall i \in N, \\
\end{align*}
\]

\(y_i \in \{0, 1\} \quad \forall i \in N,
\]

\(x_{ij} \geq 0 \text{ and integer} \quad \forall j \in M, \forall i \in N.
\]

LP relaxation gives very weak bound.
Vehicle Routing Problem with Time Windows

A company delivers goods to customers $i \in C$. The company has vehicles $k \in V$ and each of them starts at a depot, travels to several customers and returns to the depot. The visit of vehicle $k$ to customer $i$ needs to take place in a specific time window: $a_i \leq s_{ik} \leq b_i$, where $s_{ik}$ is the time when vehicle $k$ reaches customer $i$.

Objective: Minimize the total cost of delivery.

Define binary variable $x_{ijk}$ which takes value 1 if vehicle $k$ travels from customer $i$ to customer $j$ ($k \in V$, $i, j \in C$) and takes value zero otherwise.
Vehicle Routing Problem with Time Windows

Constraints:

Exactly one vehicle leaves customer $i$:

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1, \quad \forall i \in C$$

Vehicle capacity constraint:

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq q, \quad \forall k \in V$$

Each vehicle leaves the depot and returns to it:

$$\sum_{j \in N} x_{0jk} = 1 \quad \text{and} \quad \sum_{j \in N} x_{i(n+1)k} = 1, \quad \forall k \in V$$
VRPTW: Constraints (continued)

Time-window constraint

\[ s_{ik} + t_{ij} - M(1 - x_{ijk}) \leq s_{jk}, \quad \forall i, j \in N, \forall k \in V. \]

Since \( x_{ijk} \) is binary the above constraint has the following meaning: If \( x_{ijk} = 1 \) (vehicle \( k \) travels from customer \( i \) to customer \( j \)) then

\[ s_{ik} + t_{ij} \leq s_{jk} \]

that is, the arrival time of vehicle \( k \) to customer \( j \) is greater than or equal the sum of time when vehicle \( k \) arrives to customer \( i \) and the time \( t_{ij} \) it takes to travel from \( i \) to \( j \). Otherwise (if \( x_{ijk} = 0 \)) the presence of “big” \( M \) guarantees that the constraint is always inactive.
VRPTW

\[
\min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk}
\]

s.t.
\[
\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1, \quad \forall i \in C,
\]
\[
\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq q, \quad \forall k \in V,
\]
\[
\sum_{j \in N} x_{0jk} = 1, \sum_{i \in N} x_{i(n+1)k} = 1, \quad \forall k \in V,
\]
\[
\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{jhk} = 0, \quad \forall h \in C, \forall k \in V,
\]
\[
s_{ik} + t_{ij} - M(1 - x_{ijk}) \leq s_{jk}, \quad \forall i, j \in N, \forall k \in V,
\]
\[
a_i \leq s_{ik} \leq b_i, \quad \forall i \in N, \forall k \in V,
\]
\[
x_{ijk} \in \{0, 1\}, \quad \forall i, j \in N, \forall k \in V.
\]