An Exact Method to Solve the Multi-Trip Vehicle Routing Problem with Multi Time Windows
Rodolphe Giroudeau, Florent Hernandez, Olivier Naud, Frédéric Semet

To cite this version:
Rodolphe Giroudeau, Florent Hernandez, Olivier Naud, Frédéric Semet. An Exact Method to Solve the Multi-Trip Vehicle Routing Problem with Multi Time Windows. 5th International Workshop on Freight Transportation and Logistics (ODYSSEUS), May 2012, Mykonos Island, Greece. pp.9-12. lirmm-00702137

HAL Id: lirmm-00702137
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00702137
Submitted on 13 Nov 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
An Exact Method to Solve the Multi-Trip Vehicle Routing Problem with Time Windows

F. Hernandez

***
École Polytechnique de Montréal (MAGI) and CIRRELT, Canada
***

TRANSP-OR Seminar

August 2012
Summary

1. Problem statement
2. Branch and Price algorithm
3. Results
4. Conclusion
Problem statement

Instance

- an oriented graph $G = (V, A)$.
- $V = \{0, \cdots, n\}$ with 0 the depot and 1, $\cdots$, $n$ the customers
- a cost $c_{ij}$ and a travel time $t_{ij}$ for each arc $(i, j) \in A$
- for each customer $i \in \{1, \cdots, n\}$
  - demand $d_i$
  - service time $st_i$
  - a time window $[a_i, b_i]$
- $U$ vehicles allowed
- a capacity $Q$
- planning time horizon $[0, T]$

- Each customer must be visited within its time window
- vehicles may arrive earlier and wait before start the service
Objective:

Find a set of trips with minimal cost visiting all customers and respecting capacity and time windows constraints such that:

- two trips are not traveled at the same time by the same vehicle
- at most $U$ vehicles are used

- A trip is portion of a vehicle route issued from the depot and coming back to the depot
Litterature review

**Meta-heuristics**
- Fleishmann (1990): first idea of multi-trip
- Decomposition approach: Battarra, Monaci and Vigo (2009)

**Exact methods for a variant where a limit duration is imposed on the trip**
  - Limit duration decrease the complexity that allows the use of a specific strategy
  - ⇒ In our problem there is no limit duration
MTVRPTW vs VRPTW

MTVRPTW $\Rightarrow$ variant of the vehicle routing problem with time windows (VRPTW)

<table>
<thead>
<tr>
<th>VRPTW</th>
<th>MTVRPTW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visit all customers (graph covering)</td>
<td></td>
</tr>
<tr>
<td>1 demand and 1 service time per customer</td>
<td></td>
</tr>
<tr>
<td>1 time windows per customer</td>
<td></td>
</tr>
<tr>
<td>1 cost and 1 travel time between each customer</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VRPTW</th>
<th>MTVRPTW</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlimited fleet</td>
<td>limited number of vehicles</td>
</tr>
<tr>
<td>1 vehicle = 1 route</td>
<td>1 vehicle = many trips</td>
</tr>
</tbody>
</table>
A set covering problem

Like VRPTW
- Linear relaxation of explicit formulation is very weak
- ⇒ Formulation where variables represent trips

MTVRPTW ≠ VRPTW
- Temporal constraints appear between two trips
- ⇒ Trips must be located in time

MTVRPTW
- Trips definition is extended
- ⇒ Structure definition
Definitions of *structure* and *trip*

**Structure definition**

A structure is defined by:

- sequence of visited customers
- length / cost
- duration
- time window \([A, B]\) where \(A\) is the depot earliest departure time and \(B\) is the depot latest arrival time for which this structure is valid and its duration is minimal (i.e., minimum waiting time)

**Trip definition**

A trip is defined by a structure and:

- start and end times

Many trips with different schedules can be derived from every structure
A set covering formulation for VRPTW

- $\Omega$ a set of feasible trips, fixed in time
- $\theta_k$ indicates the number of times where trip $r_k$ is selected for covering, $c_k$ cost of trip $r_k$
- $a_{ik} = 1$ if the customer $i$ is visited by $r_k$, 0 else

$$\text{minimize} \sum_{r_k \in \Omega} c_k \theta_k$$

$$\sum_{r_k \in \Omega} a_{ik} \theta_k \geq 1 \quad (i \in V \setminus \{0\})$$

$$\theta_k \in \mathbb{IN} \quad (r_k \in \Omega)$$

How to model the temporal constraints?
1 vehicle with a capacity of 2
1 vehicle with a capacity of 2
VRPTW : Solution cost 10, not feasible
- 1 vehicle with a capacity of 2
- VRPTW: Solution cost 10, **not feasible**
- MTVRPTW: Solution cost 12, **feasible**
A set covering formulation for MTVRPTW

- $\Omega$ a set of feasible trips, fixed in time
- $\theta_k$ indicates the number of times where trip $r_k$ is selected for covering,
- $c_k$ cost of trip $r_k$
- $a_{ik} = 1$ if the customer $i$ is visited by $r_k$, 0 else
- $b_{tk} \in \{0, 1\}$ indicates if the trip $r_k$ includes the instant $\delta_t$

\[\text{minimize} \sum_{r_k \in \Omega} c_k \theta_k\]
\[\sum_{r_k \in \Omega} a_{ik} \theta_k \geq 1 \quad (i \in V \setminus \{0\})\]
\[\theta_k \in \mathbb{N} \quad (r_k \in \Omega)\]
\[\sum_{r_k \in \Omega} b_{tk} \theta_k \leq U \quad (\forall \delta_t)\]
One time constraint by instant?

- one time interval $\delta_t$ by instant $\Rightarrow b_{tk}$ are binary
- combinatorial explosion of constraint number related to temporal precision
One time constraint by instant?

- one time interval $\delta_t$ by instant $\Rightarrow b_{tk}$ are binary
- combinatorial explosion of constraint number related to temporal precision
- time interval length $= \text{minimal duration of possible trips}$

$$r_k = \Delta_t, \Delta_{t+1}, \Delta_{t+2}, \Delta_{t+3}$$
One time constraint by instant?

- one time interval $\delta_t$ by instant $\Rightarrow b_{tk}$ are binary
- combinatorial explosion of constraint number related to temporal precision
- time interval length = minimal duration of possible trips
- $\Rightarrow b_{tk} \in [0, 1]$ become the fraction of time interval $\Delta_t$ occupied by trip $r_k$

\[
\begin{align*}
  b_{tk} &= 0 \\
  b_{(t+1)k} &= 0.8 \\
  b_{(t+2)k} &= 1 \\
  b_{(t+3)k} &= 0.2 \\
\end{align*}
\]
A set covering formulation for MTVRPTW

- \( \Omega \) a set of feasible trips, fixed in time
- \( \theta_k \) indicates the number of times where trip \( r_k \) is selected for covering,
  - \( c_k \) cost of trip \( r_k \)
- \( a_{ik} = 1 \) if the customer \( i \) is visited by \( r_k \), 0 else
- \( b_{tk} \in [0, 1] \) indicates if the trip \( r_k \) includes the instant \( \Delta_t \)

\[
\begin{align*}
\text{minimize} & \quad \sum_{r_k \in \Omega} c_k \theta_k \\
\sum_{r_k \in \Omega} a_{ik} \theta_k & \geq 1 \quad (i \in V \setminus \{0\}) \\
\theta_k & \in \mathbb{N} \quad (r_k \in \Omega) \\
\sum_{r_k \in \Omega} b_{tk} \theta_k & \leq U \quad (\forall \Delta_t)
\end{align*}
\]
Two issues

Time interval length = minimal duration of possible trips

- relaxation of the problem

Ω is a set of feasible trips, fixed in time

- too many variables
Two issues

Time interval length = minimal duration of possible trips

- relaxation of the problem

⇒ if the solution found is not feasible then the problem have to be solved with one time interval $\delta_t$ by instant

$\Omega$ is a set of feasible trips, fixed in time

- too many variables
Two issues

**Time interval length** = minimal duration of possible trips

- relaxation of the problem

- if the solution found is not feasible then the problem have to be solved with one time interval $\delta_t$ by instant

**$\Omega$ is a set of feasible trips, fixed in time**

- too many variables

- addressed by column generation
What is the column generation?

Inspiration

Simplexe algorithm

- Only basic variables are interesting

How does a nonbasic variable becomes a basic variable?

- In minimisation case: only if its reduced cost is negative

Method

The column generation consists to solve iteratively two problems.

- the restricted master problem = problem restricted to a sub set of variables (basic and nonbasic)
- the pricing problem = Find new nonbasic variables that can become basic

Stop when the pricing problem don’t find new nonbasic variables that can improve the solution.
Problem decomposition

- Restricted master problem: Set covering problem where the integrity constraints are relaxed, reduced to a subset of variables $\Omega_w$
- Subproblem: Find a negative reduced cost variable $\Rightarrow$ Elementary shortest path problem with resource constraints (ESPPRC)
Column generation

Problem decomposition

- Restricted master problem: Set covering problem where the integrity constraints are relaxed, reduced to a subset of variables $\Omega_w$
- Subproblem: Find a negative reduced cost variable $\Rightarrow$ Elementary shortest path problem with resource constraints (ESPPRC)

Reduced cost

$$c_k^r = c_k - \sum_{i \in V \setminus \{0\}} a_{ik} \lambda_i + \sum_{\Delta_t} b_{tk} \mu_t$$

- $\lambda_i$: dual value associated to customer $i$
- $\mu_t$: dual value associated to time interval $\Delta_t$
Dynamic programming:

- labels = $L_{num} = (T_{num}^1, \cdots, T_{num}^n)$
- each node has a label list
- during label extension
  - create a new label and insert it in corresponding node label list
  - set resource consumption
  - check that the resource constraints are meet
- stop when no more label can be extended

Problem: too many generated labels
Solution: apply a dominance relation after each extension on corresponding label list
Dynamic programming:

- **labels** \( L_{num} = (T_1^{num}, \ldots, T_n^{num}) \)
- each node has a label list
- during label extension
  - create a new label and insert it in corresponding node label list
  - set resource consumption
  - check that the resource constraints are met
- stop when no more label can be extended

**Problem**

- too many generated labels
Dynamic programming:

- labels \( L_{num} = (T_{num}^1, \ldots, T_{num}^n) \)
- each node has a label list
- during label extension
  - create a new label and insert it in corresponding node label list
  - set resource consumption
  - check that the resource constraints are met
- stop when no more label can be extended

Problem

- too many generated labels

Solution

- apply a dominance relation after each extension on corresponding label list
Subproblem

Objective 1

- take into account the loading times

The loading times are at the depot before departure

- add a customer to a trip ⇒ delay time of service for the previous customers
Subproblem

Objective 1

- take into account the loading times

The loading times are at the depot before departure

- add a customer to a trip $\Rightarrow$ delay time of service for the previous customers
- $\Rightarrow$ extend the label in backward move
Subproblem

**Objective 1**
- take into account the loading times

The loading times are at the depot before departure
- add a customer to a trip \( \Rightarrow \) delay time of service for the previous customers
- \( \Rightarrow \) extend the label in backward move

**Objective 2**
- generate a trip with minimal reduced cost
Subproblem

Objective 1
- take into account the loading times

The loading times are at the depot before departure
- add a customer to a trip ⇒ delay time of service for the previous customers
- ⇒ extend the label in backward move

Objective 2
- generate a trip with minimal reduced cost

avoid the time interval associated with a dual value $\mu_t$ not null
- take into account all possible departure time
Subproblem

Objective 1
- take into account the loading times

The loading times are at the depot before departure
- add a customer to a trip ⇒ delay time of service for the previous customers
- ⇒ extend the label in backward move

Objective 2
- generate a trip with minimal reduced cost

avoid the time interval associated with a dual value $\mu_t$ not null
- take into account all possible departure time
  - too many labels
Subproblem

**Objective 1**
- take into account the loading times

The loading times are at the depot before departure
- add a customer to a trip \( \Rightarrow \) delay time of service for the previous customers
- \( \Rightarrow \) extend the label in backward move

**Objective 2**
- generate a trip with minimal reduced cost

Avoid the time interval associated with a dual value \( \mu_t \) not null
- take into account all possible departure time
  - too many labels
  - \( \Rightarrow \) group labels that represent the same structure and select a representative
Subproblem: Label groups and representative label

Label group definition
A group of labels is a set of labels that represent the same partial path and whose arrival-to-destination dates belong to the same time interval.

How to select a representative label
Two main rules:
- It can be dominated by another label if and only if all labels of its group are dominated by this label.
- It must accept all extensions accepted by at least one label of its group.
Subproblem: Label groups and representative label

Label group definition
A group of labels is a set of labels that represent the same partial path and whose arrival-to-destination dates belong to the same time interval.

How to select a representative label

- Two main rules:
  - It can be dominated by an other label if and only if all labels of its group are dominated by this label.
  - It must accept all extensions accepted by at least one label of its group.

In our previous studies

- To compare the time dependent reduced cost of two different labels $L_k$ and $L_k'$ during the dominance relation process we use this formula:
  
  $$c_c^r + \sum_{t=h_k}^{h_k'} \mu_t \leq c_k'^r$$
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- reduced cost formula: $c_k^r + \sum_{t=h_k'}^{t=h_k} \mu_t \leq c_{k'}^r$
- $c_1^r = 3$; $c_3^r = 5$; $c_2^r = c_3^r + 3.5 = 8.5$
- comparisons:

\[
\begin{align*}
\sum \mu_t &= 3.5 \\
L_2 &\stackrel{3}{\longrightarrow} \times \\
L_3 &\stackrel{3}{\longrightarrow} \times \\
L_1 &\stackrel{3}{\longrightarrow} \times
\end{align*}
\]
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- reduced cost formula: $c_k^r + \sum_{t=h_k}^{t=h_k'} \mu_t \leq c_k^{r'}$
- $c_1^r = 3; c_3^r = 5; c_2^r = c_3^r + 3.5 = 8.5$
- comparisons: $c_1^r + 4 < c_2^r$

\[ c_1^r = 3; \quad c_3^r = 5; \quad c_2^r = c_3^r + 3.5 = 8.5 \]

\[ \sum \mu_t = 4 \]

\[ \Delta_1 \quad \Delta_2 \quad \Delta_3 \quad \Delta_4 \]

\[ \text{time} \]

\[ L_1 \quad L_2 \quad L_3 \]
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- reduced cost formula: $c_k^r + \sum_{t=h_k}^{t=h_{k'}} \mu_t \leq c_{k'}^r$
- $c_1^r = 3$; $c_3^r = 5$; $c_2^r = c_3^r + 3.5 = 8.5$
- comparisons: $3 + 4 < 8.5$
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- reduced cost formula: $c_k^r + \sum_{t=h_k'}^{t=h_k} \mu_t \leq c_k'^r$
- $c_1^r = 3, c_3^r = 5, c_2^r = c_3^r + 3.5 = 8.5$
- comparisons:
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- Reduced cost formula: $c^r_k + \sum_{t=h_k}^{t=h_{k'}} \mu_t \leq c^r_{k'}$
- $c^r_1 = 3; \ c^r_3 = 5; \ c^r_2 = c^r_3 + 3.5 = 8.5$
- Comparisons: $c^r_1 + 4.5 > c^r_3$

\[ \sum_{\mu_t=4.5} \]

\[ \Delta_1 \Delta_2 \Delta_3 \Delta_4 \]

\[ \text{time} \]
Subproblem: Select a representative label

Select relative to the reduced cost

Let $L_2$ and $L_3$ in the same group (their represent the same partial path)

- reduced cost formula: $c'_k + \sum_{t=h_k'}^{t>h_k} \mu_t \leq c'_k$
- $c'_1 = 3$; $c'_3 = 5$; $c'_2 = c'_3 + 3.5 = 8.5$
- comparisons: $3 + 4.5 > 5$
Subproblem: Select a representative label

Select relative to the possible extensions

- it must accept all extensions accepted by at least one label of its group
Subproblem: Select a representative label

Select relative to the possible extensions

- it must accept all extensions accepted by at least one label of its group

New parameter: retardation of the arrival time to the depot
Subproblem: Select a representative label

Select relative to the possible extensions

- it must accept all extensions accepted by at least one label of its group
Subproblem: Select a representative label

Select relative to the possible extensions

- it must accept all extensions accepted by at least one label of its group
- all labels of the group have to take into account $\Rightarrow$ retardation
Subproblem: Select a representative label

Select relative to the possible extensions

- it must accept all extensions accepted by at least one label of its group
- all labels of the group have to take into account $\Rightarrow$ retardation

New parameter: retardation of the arrival time to the depot
New definition of representative labels

Définition

A path \( p \) from \( j \) to 0 is represented by the label:
\[
L_p = \left( c_p^r, h_p, q_p, d_p, rd_p, V_p^1, \cdots, V_p^n \right),
\]
where:
- \( c_p^r \) is the reduced cost of \( p \)
- \( h_p \) is the starting time of the service of \( j \)
- \( q_p \) is the carried quantity
- \( d_p \) is the arrival time to the depot
- \( rd_p \) is the possible retardation of the arrival time of the depot
- \( V_p^i = 1 \) if the customer \( i \) is unreachable by \( p \), 0 else

Initialisation

- One label is created at the end of the planning time horizon and one label at the beginning of each time interval with a not null dual value
Dominance relation

Label \( L_1 \) dominates label \( L_2 \) if and only if:

- \( q_1 \leq q_2 \) (carried quantity)
- All customers unreachable by \( L_1 \) are not by \( L_2 \) too
- \( h_2 + rd_2 \leq h_1 \) (starting time of the service)
- \( c_1^r + \sum_{t=h_2}^{h_1} \mu_t \leq c_2^r \)
Column generation scheme

Initial solution for master problem → Set of possible trips $\Omega$ → Restricted master problem

Addition of trip(s) in $\Omega$ → Determination of reduced cost function

Yes → Set of possible trips $\Omega$ → Trip? → Optimal solution

No → Sub-problem → Trip?
### Recall

<table>
<thead>
<tr>
<th>Branch and bound</th>
<th>Branch and Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplexe</td>
<td>Column generation</td>
</tr>
</tbody>
</table>

Branching on an arc $(i,j)$ with a fractional flow $x_{ij} = 1 \Rightarrow \theta_k = 0$ if trip $k$ visits customer $i$ or $j$ without using arc $(i,j)$.

$x_{ij} = 0 \Rightarrow \theta_k = 0$ if trip $k$ uses arc $(i,j)$.

Problem having all arcs with an integer flow ($x_{ij} \in \mathbb{Z}$) does not imply that the solution is integer ($\theta_k \in \{0, 1\}$) \Rightarrow call a repair strategy.
Branching on arcs

- select an arc \((i,j)\) with a fractional flow
- \(x_{ij} = 1 \Rightarrow \theta_k = 0\) if trip \(k\) visits customer \(i\) or \(j\) without using arc \((i,j)\)
- \(x_{ij} = 0 \Rightarrow \theta_k = 0\) if trip \(k\) uses arc \((i,j)\)
Recall

<table>
<thead>
<tr>
<th>Branch and bound</th>
<th>Branch and Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplexe</td>
<td>Column generation</td>
</tr>
</tbody>
</table>

Branching on arcs

- select an arc \((i,j)\) with a fractional flow
- \(x_{ij} = 1 \Rightarrow \theta_k = 0\) if trip \(k\) visits customer \(i\) or \(j\) without using arc \((i,j)\)
- \(x_{ij} = 0 \Rightarrow \theta_k = 0\) if trip \(k\) uses arc \((i,j)\)

Problem

- having all arcs with an integer flow \((x_{ij} \in \mathbb{I}\)) does not imply that the solution is integer \((\theta_k \in \{0, 1\})\)
## Recall

<table>
<thead>
<tr>
<th>Branch and bound</th>
<th>Branch and Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplexe</td>
<td>Column generation</td>
</tr>
</tbody>
</table>

## Branching on arcs

- select an arc \((i,j)\) with a fractional flow
- \(x_{ij} = 1 \Rightarrow \theta_k = 0\) if trip \(k\) visits customer \(i\) or \(j\) without using arc \((i,j)\)
- \(x_{ij} = 0 \Rightarrow \theta_k = 0\) if trip \(k\) uses arc \((i,j)\)

## Problem

- having all arcs with an integer flow \((x_{ij} \in \mathbb{IN})\) does not imply that the solution is integer \((\theta_k \in \{0, 1\})\)

- \(\Rightarrow\) call a repair strategy
Case when the flow matrix is integer and some variables fractional

- The flow matrix is such that every customer has an unique successor
- The flow matrix represents a set of structures
- In the actual fractional solution some structures are represented by several trips with different time positions

We consider the following issue:

- Is it possible to assign a single time position to every structure?
- Equivalent to determining the existence of an integer solution supported by the integer flow matrix

Solved using a VRPTW modeling

- Nodes: structures
- Arcs: feasible successions
Instance of VRPTW
- customers
  - service time
  - demande
  - time window
- arcs \((i, j)\)
  - travel time
  - cost

Our branching problem
- structures
  - duration
- time window
- arcs \((i, j)\)
  - cost of the structure \(j\)
Instance of VRPTW
- customers
  - service time
  - demande
  - time window
- arcs \((i, j)\)
  - travel time
  - cost

Our branching problem
- structures
  - duration
  - 0
  - time window
- arcs \((i, j)\)
  - 0
  - cost of the structure \(j\)
### Instance of VRPTW
- customers
  - service time
  - demande
  - time window
- arcs \((i, j)\)
  - travel time
  - cost

### Our branching problem
- structures
  - duration
  - 0
  - time window
- arcs \((i, j)\)
  - 0
  - cost of the structure \(j\)

---

**We solve it with:**
- Standard branch and price scheme
if a solution of the VRPTW is found

- Update the upper bound
- Prune the node

If no VRPTW solution exists

- Select an arc not involved in previous branching constraints
- Branch on this arc
Implementation

Hardware and software
- **Language:** C++
- **Solver:** GLPK (open source)
- **Computer:** Intel Core2Duo E7300 2.66GHz, 3Gb RAM

Benchmarks
- based on Solomon’s instances: R2, RC2, C2
- 25 customers: 2 vehicles allowed
- 50 customers: 4 vehicles allowed
- computation time limited to 30h per instance
### Instances with 25 customers

<table>
<thead>
<tr>
<th>Instances</th>
<th>% GAP</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>C</td>
<td>0.55</td>
<td>4.85</td>
</tr>
<tr>
<td>R</td>
<td>0.44</td>
<td>4.70</td>
</tr>
<tr>
<td>RC</td>
<td>3.52</td>
<td>8.98</td>
</tr>
</tbody>
</table>

- 25/27 instances closed
- Great variation of computation times and GAPs
## Results

### Instances with 50 customers

<table>
<thead>
<tr>
<th>Instances</th>
<th>% GAP</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>R201-50</td>
<td>1.58</td>
<td>237</td>
</tr>
<tr>
<td>R202-50</td>
<td>2.42</td>
<td>78880</td>
</tr>
<tr>
<td>R205-50</td>
<td>2.39</td>
<td>24062</td>
</tr>
<tr>
<td>RC201-50</td>
<td>2.24</td>
<td>662</td>
</tr>
<tr>
<td>RC202-50</td>
<td>2.40</td>
<td>99346</td>
</tr>
</tbody>
</table>

- Only 5/27 instances closed
- Great variation of computation times
Conclusion

- Master problem: time constraint between trips
  - time aggregation
  - all solution found with time aggregation are feasible that imply a good relaxation of time constraint
- Sub-problem: time dependent reduced cost
  - appropriate dynamic programming $\Rightarrow$ representative labels

Perspectives

- Problem with a great temporal dependence
  - mainly solved with a structural branching scheme
  - $\Rightarrow$ improve the branching scheme with temporal branching politic
Thank you for your attention

florent.hernandez@cirrelt.ca