An Exact Method to Solve the Multi-Trip Vehicle Routing Problem with Multi Time Windows

Rodolphe Giroudeau, Florent Hernandez, Olivier Naud, Frédéric Semet

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

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 windows $[a_i, b_i]$
- 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

Problem statement

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
- Tabu search : Taillard, Laporte and Gendreau (1996), Brandao and Mercer (1998)
- Genetic algorithm : Salhi and Petch (2004)
- Decomposition approach : Battarra, Monaci and Vigo (2009)

Exact methods for a variant where a limit duration is imposed on the trip

- Azi et al (2007 and 2010) and Macedo et al (2011)
  - Limit duration decrease the complexity that allows the use of a specific strategy
  - $\Rightarrow$ In our problem there is no limit duration

MTVRPTW vs VRPTW

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

VRPTW	MTVRPTW

- Visit all customers (graph covering)
- 1 demand and 1 service time per customer
- 1 time windows per customer
- 1 cost and 1 travel time between each customer

VRPTW

- unlimited fleet
- 1 vehicle = 1 route

MTVRPTW

- limited number of vehicles
- 1 vehicle = many trips

A set covering problem

Like VRPTW

- Linear relaxation of explicit formulation is very weak
- $\Rightarrow$ Formulation where variables represent trips

MTVRPTW $\neq$ VRPTW

- Temporal constraints appear between two trips
- $\Rightarrow$ Trips must be located in time

MTVRPTW

- Trips definition is extended
- $\Rightarrow$ Structure definition

Definitions of *structure* and *trip*

Structure definition

A structure is defined by:

- sequence of visited customers
- length / cost
- duration
- time window $[\mathcal{A}, \mathcal{B}]$ where $\mathcal{A}$ is the depot earliest departure time and $\mathcal{B}$ 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

- Ω a set of feasible trips, fixed in time
- θ_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

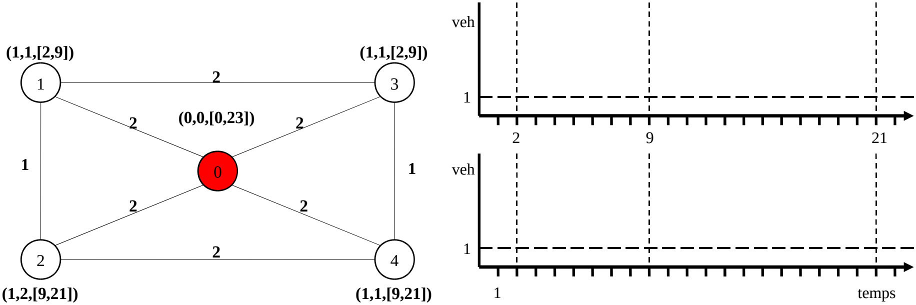
$$minimize \sum_{r_k \in \Omega} c_k \theta_k$$

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

$$\theta_k \in IN \qquad (r_k \in \Omega)$$

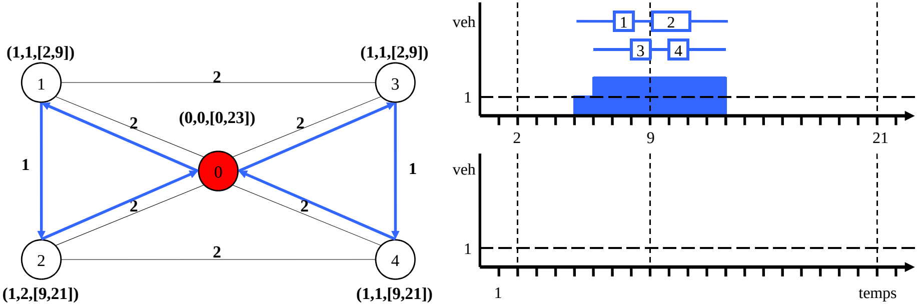
How to model the temporal constraints ?

Trip succession



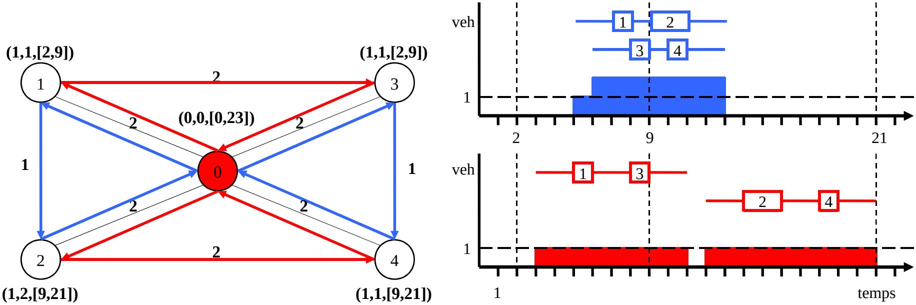
- 1 vehicle with a capacity of 2

Trip succession



- 1 vehicle with a capacity of 2
- VRPTW : Solution cost 10, not feasible

Trip succession



- 1 vehicle with a capacity of 2
- VRPTW : Solution cost 10, not feasible
- MTVRPTW : Solution cost 12, feasible

A set covering formulation for MTVRPTW

- Ω a set of feasible trips, fixed in time
- θ_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 δ_t

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

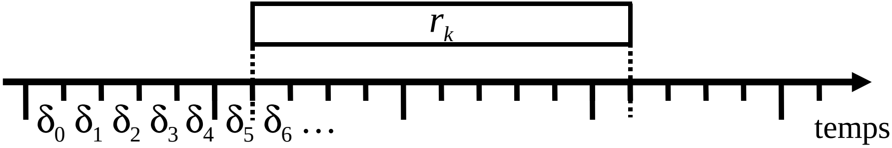
$$\sum_{r_k \in \Omega} a_{ik} \theta_k \geq 1 \qquad (i \in V \setminus \{0\})$$

$$\theta_k \in IN \qquad (r_k \in \Omega)$$

$$\sum_{r_k \in \Omega} b_{tk} \theta_k \leq U \qquad (\forall \delta_t)$$

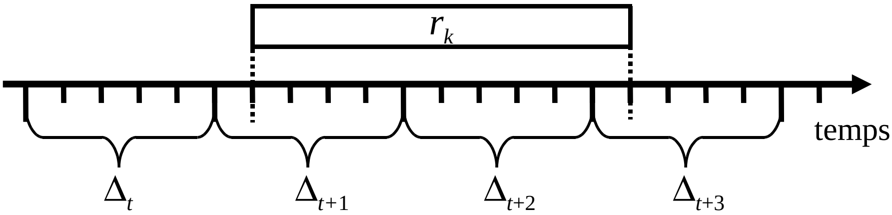
One time constraint by instant ?

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



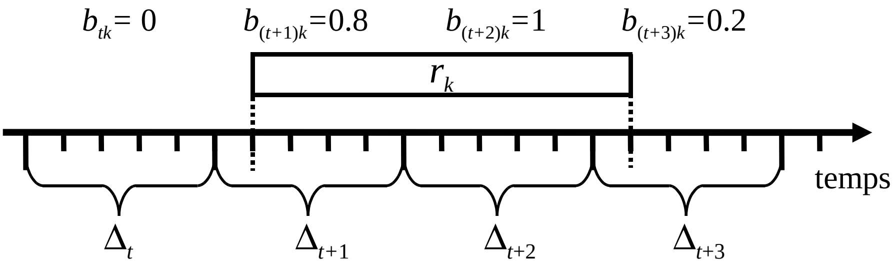
One time constraint by instant ?

- one time interval δ_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



One time constraint by instant ?

- one time interval δ_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 Δ_t occupied by trip r_k



A set covering formulation for MTVRPTW

- Ω a set of feasible trips, fixed in time
- θ_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 Δ_t

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

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

$$\theta_k \in IN \qquad (r_k \in \Omega)$$

$$\sum_{r_k \in \Omega} b_{tk} \theta_k \leq U \qquad (\forall \Delta_t)$$

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

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

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

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

Ω is a set of feasible trips, fixed in time

- too many variables

- $\Rightarrow$ adressed by column generation

What is the column generation ?

Inspiration

Simplexe algorithm

- Only basic varaibles 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.

Column generation

Problem decomposition

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

Reduced cost

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

- λ_i dual value associated to customer i
- μ_t dual value associated to time interval Δ_t

Dynamic programming :

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

Dynamic programming :

- labels $= L_{num} = (T^1_{num}, \cdots, T^n_{num})$
- each node has a label list
- during label extension
  - create a new label and insert it in correponding 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

Dynamic programming :

- labels = $L_{num} = (T^1_{num}, \cdots, T^n_{num})$
- each node has a label list
- during label extension
  - create a new label and insert it in correponding 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

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

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 $\Rightarrow$ 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 μ_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 $\Rightarrow$ 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 μ_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 μ_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 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

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

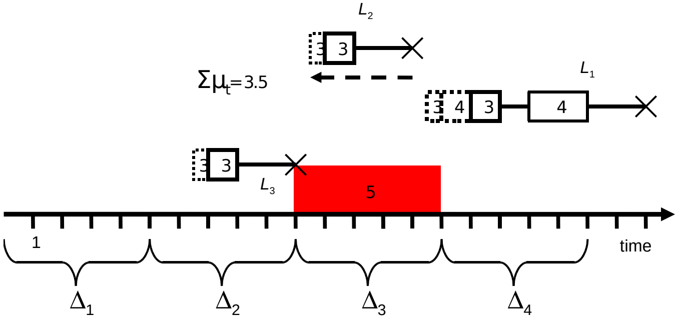
- To compare the time dependent reduced cost of two different labels L_k and $L_{k'}$ during the dominance relation process we use this formula
- $c_k^r + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c_{k'}^r$

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 : $c_k^r + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c_{k'}^r$
- $c_1^r = 3$; $c_3^r = 5$; $c_2^r = c_3^r + 3.5 = 8.5$
- comparaisons :

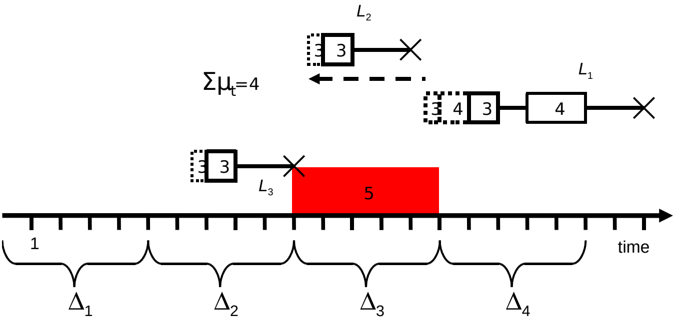


Subproblem: Select a representative label

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 : $c_k^r + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c_{k'}^r$
- $c_1^r = 3$; $c_3^r = 5$; $c_2^r = c_3^r + 3.5 = 8.5$
- comparaisons : $c_1^r + 4 < c_2^r$

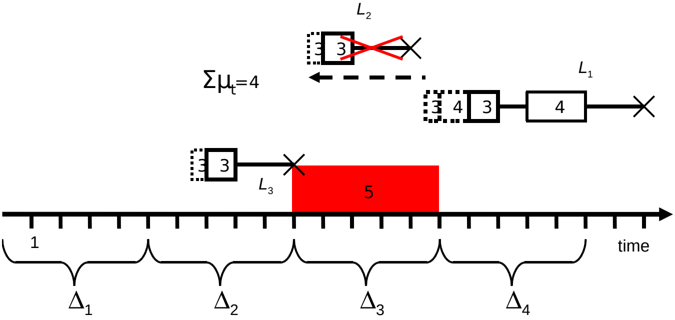


Subproblem: Select a representative label

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 : $c^r_k + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c^r_{k'}$
- $c^r_1 = 3$; $c^r_3 = 5$; $c^r_2 = c^r_3 + 3.5 = 8.5$
- comparaisons : $3 + 4 < 8.5$

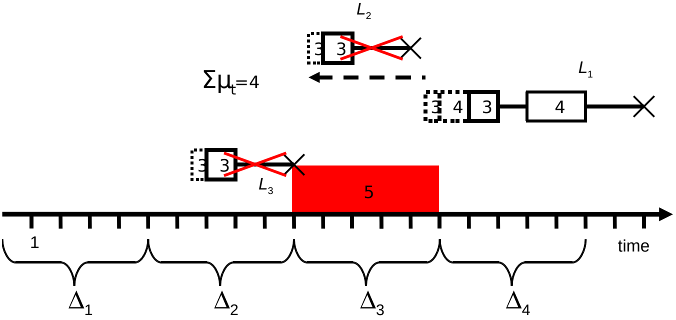


Subproblem: Select a representative label

Select relative to the reduced cost

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

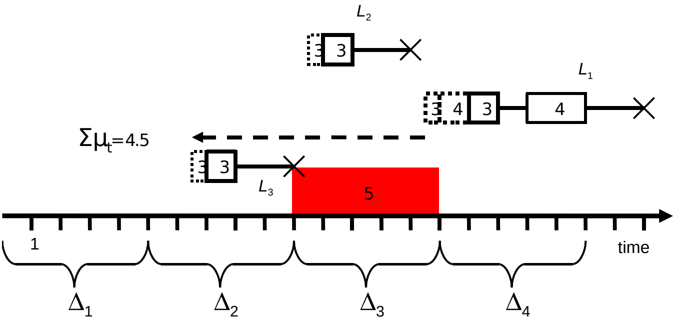
- reduced cost formula : $c^r_k + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c^r_{k'}$
- $c^r_1 = 3$; $c^r_3 = 5$; $c^r_2 = c^r_3 + 3.5 = 8.5$
- comparaisons :



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

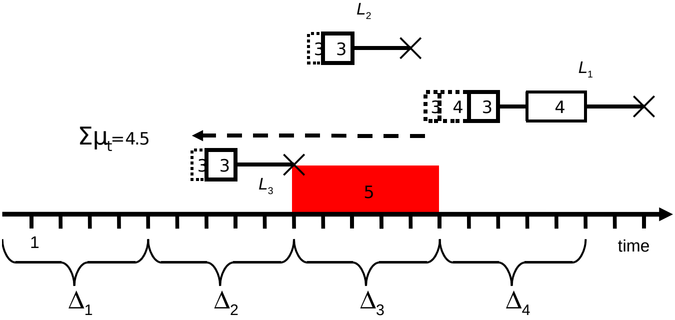


Subproblem: Select a representative label

Select relative to the reduced cost

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

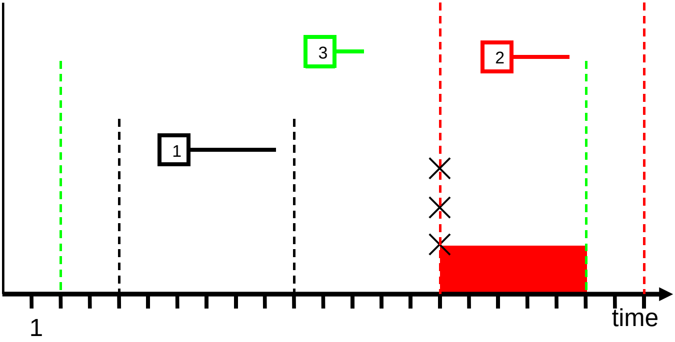
- reduced cost formula : $c^r_k + \sum_{t>h_{k'}}^{t=h_k} \mu_t \leq c^r_{k'}$
- $c^r_1 = 3$; $c^r_3 = 5$; $c^r_2 = c^r_3 + 3.5 = 8.5$
- comparaisons : $3 + 4.5 > 5$



Subproblem: Select a representative label

Select relative to the possible extensions

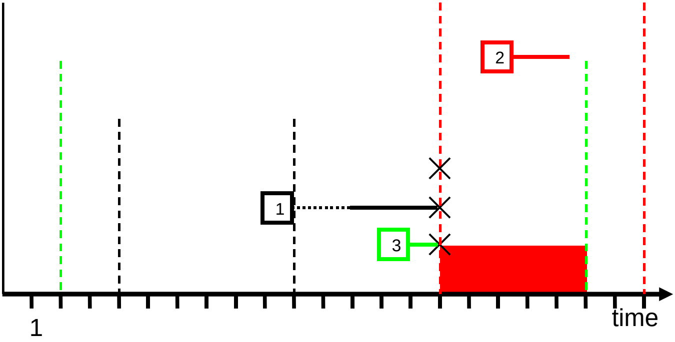
- it must accept all extensions accepted by at least one label of its group



Subproblem: Select a representative label

Select relative to the possible extensions

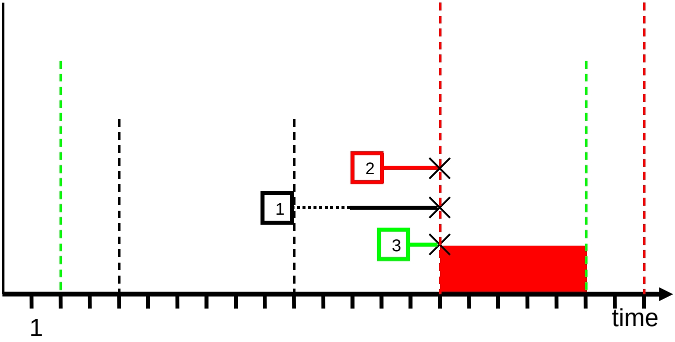
- it must accept all extensions accepted by at least one label of its group



Subproblem: Select a representative label

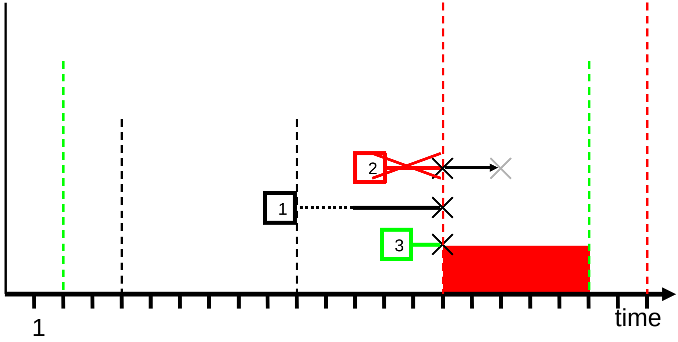
Select relative to the possible extensions

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



Select relative to the possible extensions

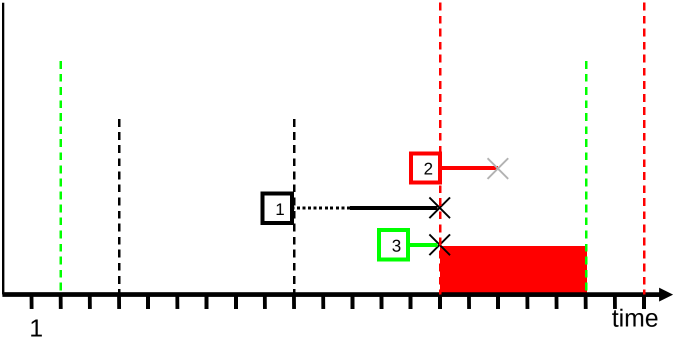
- it must accept all extensions accepted by at least one label of its group
- all labels of the group have to take into account ⇒ retardation



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:
$L_p = (c_p^r, h_p, q_p, d_p, rd_p, V_p^1, \cdots, V_p^n)$, 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

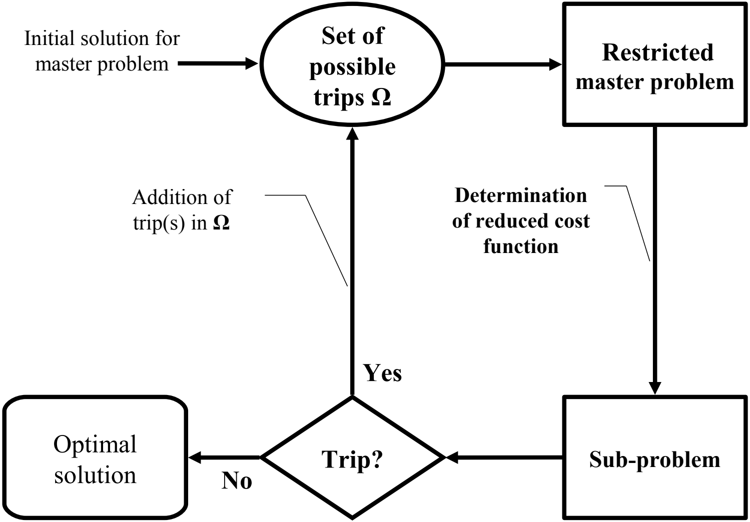
- One label is created at the end of the planning time horizon and one label at the beginning of each time interval with a not null dual value

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)
- $c_1^r + \sum_{t>h_2}^{t=h_1} \mu_t \leq c_2^r$

Column generation scheme



Recall

Branch and bound	Branch and Price
Simplexe	Column generation

Branch and Price

Recall

Branch and bound	Branch and Price
Simplexe	Column generation

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

Branch and bound	Branch and Price
Simplexe	Column generation

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 ($x_{ij} \in IN$) does not imply that the solution is integer ($\theta_k \in \{0,1\}$)

Branch and Price

Recall

Branch and bound	Branch and Price
Simplexe	Column generation

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 ($x_{ij} \in IN$) does not imply that the solution is integer ($\theta_k \in \{0,1\}$)

- $\Rightarrow$ call a repair strategy

Branch and Price

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

Branch and Price: case of an integer flow matrix

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

Branch and Price: case of an integer flow matrix

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

Branch and Price: case of an integer flow matrix

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

Branch and Price: case of an integer flow matrix

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

Results

Instances with 25 customers

Instances	% GAP			Time (sec)		
	Min	Max	Avg	Min	Max	Avg
C	0.55	4.85	2.27	12	371	170
R	0.44	4.70	2.41	22	3769	1006
RC	3.52	8.98	5.41	9	20038	12537

- 25/27 instances closed
- great variation of computation times and GAPs

Instances with 50 customers

Instances	% GAP	Time (sec)
C	-	-
R201-50	1.58	237
R202-50	2.42	78880
R205-50	2.39	24062
RC201-50	2.24	662
RC202-50	2.40	99346

- Only 5/27 instances closed
- great variation of computation times

Conclusion

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