Constrained vehicle routing and shortest path problems

Guy Desaulniers

Polytechnique Montréal and GERAD, Canada

Column Generation School 2014
Paris, France, March 10–14, 2014

Outline

1. Part I: Vehicle routing with time windows
2. Part II: Split delivery routing with time windows
3. Part III: Other shortest path problems with resource constraints

Part I

Vehicle routing problem with time windows

Outline of Part I: VRPTW

1. VRPTW definition
2. Arc-flow formulation
3. Path-flow formulation
4. Branch-price-and-cut
   - Master problem
   - Subproblem
   - Linear relaxation computational results
   - Reaching the elementary bound
   - Branching
   - Cutting
   - Integer solution computational results

Outline

Capacitated vehicle routing problem (CVRP)

Definition

- Given
  - unlimited number of identical vehicles with a given capacity, housed in a single depot
  - set of customers with known demands (all pickups or all deliveries)
- Find vehicle routes such that
  - all customer demands are met
  - each customer is visited by a single vehicle
  - each route starts and ends at the depot
  - each route satisfies vehicle capacity
  - total cost (distance) is minimized

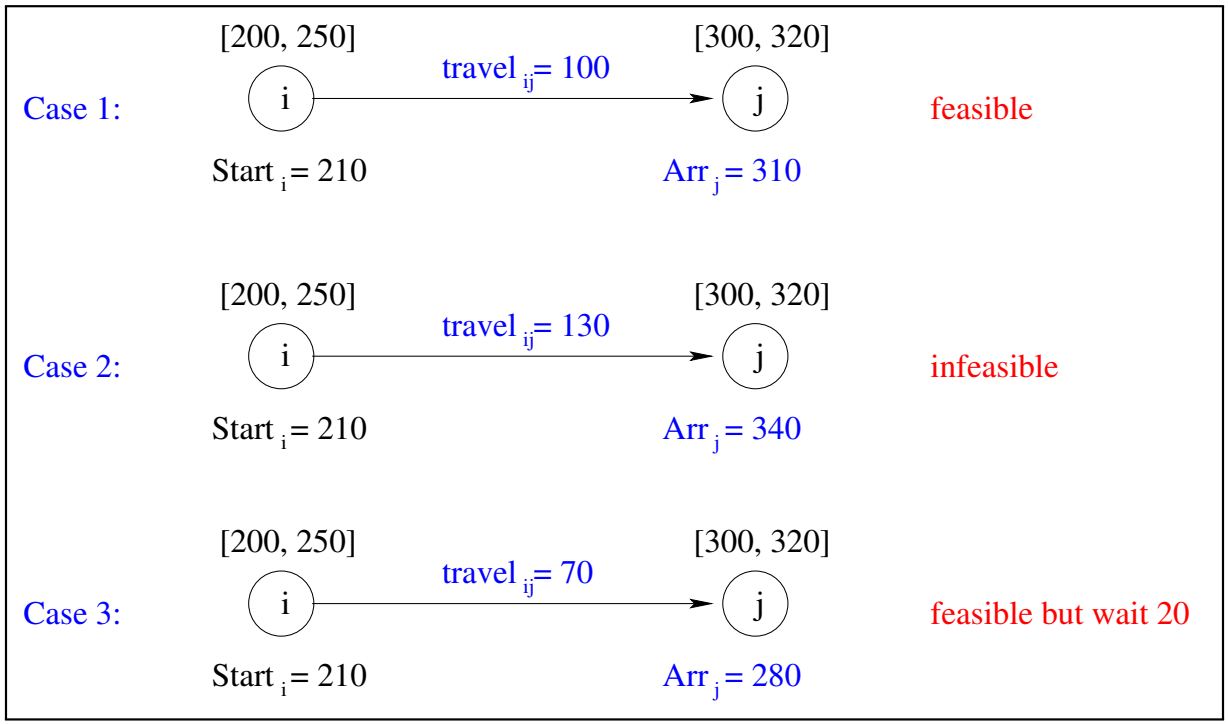
Vehicle routing problem with time windows (VRPTW)

Definition

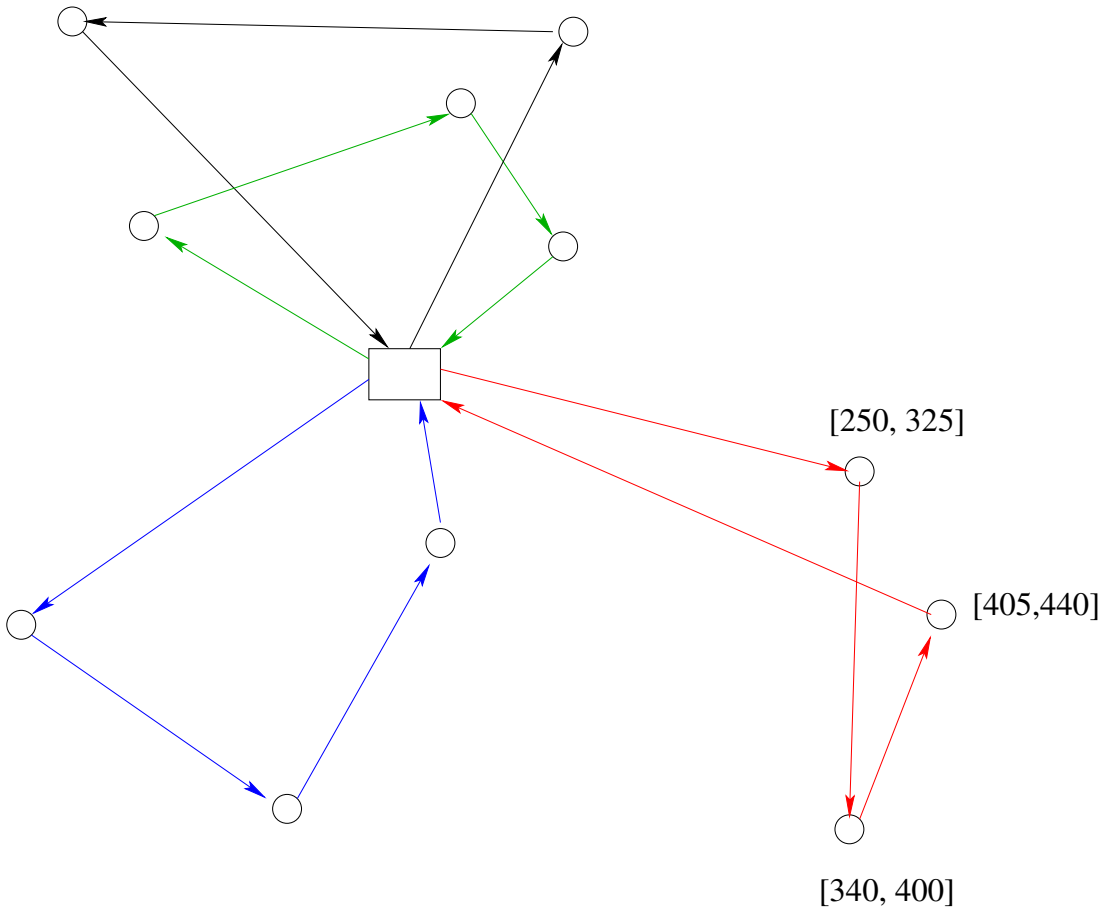
- Given
  - unlimited number of identical vehicles with a given capacity, housed in a single depot
  - set of customers with known demands
  - a time window for each customer
- Find vehicle routes such that
  - all customer demands are met
  - each customer is visited by a single vehicle
  - each route starts and ends at the depot
  - each route satisfies vehicle capacity and time windows
  - total cost (distance) is minimized

Time windows

- At every customer, service must start within its time window
- Hard time windows
  - Infeasible if upper bound is exceeded
  - Must wait until lower bound if vehicle arrives too early



Example of a solution



Outline

Notation

Customer set $N = \{1, \ldots, n\}$

- demand q_i
- time window $[a_i, b_i]$

Depot represented by two nodes $\{o, d\}$

- no demand: $q_o = q_d = 0$
- planning horizon: $[a_o, b_o] = [a_d, b_d]$

Node set $\bar{N} = N \cup \{o, d\}$

Vehicle set K

- $|K|$ is sufficiently large to be unrestrictive
- capacity Q

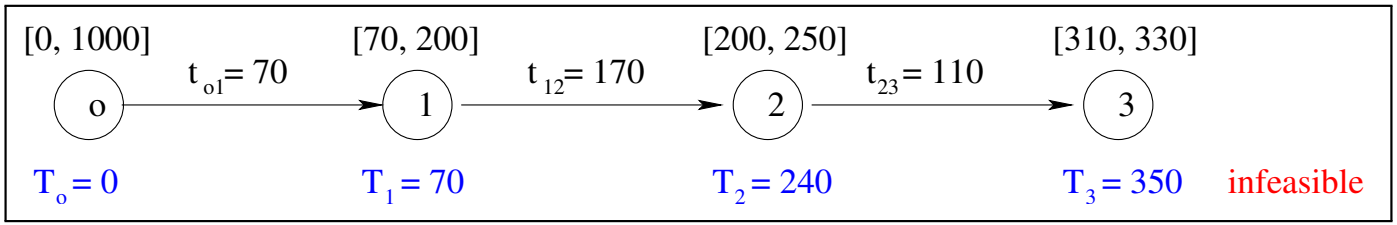
Arc set $A \subset \bar{N} \times \bar{N}$

- travel time t_{ij} (including service time at i)
- travel cost c_{ij}
- $(i, j) \in A$ if $q_i + q_j \leq Q$ and $a_i + t_{ij} \leq b_j$

Route feasibility

Network $G = (\bar{N}, A)$

- Every feasible route corresponds to an o-d path in G
- However, NOT all feasible o-d paths in G correspond to a feasible route



Decision variables

Arc-flow variables for all $(i,j) \in A$, $k \in K$

$$X_{ij}^k = \begin{cases} 1 & \text{if vehicle } k \text{ uses arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$$

Time variables for all $i \in \bar{N}$, $k \in K$

$$T_i^k = \text{start of service of vehicle } k \text{ at node } i$$

Remark

T_i^k is irrelevant if vehicle k does not visit node i

Formulation (1)

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} X_{ij}^k \qquad (1) \quad \text{total cost}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k = 1, \qquad \forall\, i \in N \qquad (2) \quad \text{visit each customer}$$

$$\sum_{(o,j) \in A} X_{oj}^k = \sum_{(i,d) \in A} X_{id}^k = 1, \qquad \forall\, k \in K \qquad (3) \quad \text{from and to depot}$$

$$\sum_{(i,j) \in A} X_{ij}^k - \sum_{(j,i) \in A} X_{ji}^k = 0, \qquad \forall\, k \in K, i \in N \qquad (4) \quad \text{flow conservation}$$

Formulation (2)

$$\sum_{(i,j)\in A} q_i X_{ij}^k \leq Q, \qquad \forall\, k \in K \qquad (5) \qquad \text{vehicle capacity}$$

$$X_{ij}^k (T_i^k + t_{ij} - T_j^k) \leq 0, \qquad \forall\, k \in K, (i,j) \in A \qquad (6) \qquad \text{time computation}$$

$$a_i \leq T_i^k \leq b_i, \qquad \forall\, k \in K, i \in \bar{N} \qquad (7) \qquad \text{time windows}$$

$$X_{ij}^k \in \{0,1\}, \qquad \forall\, k \in K, (i,j) \in A \qquad (8) \qquad \text{binary requirements}$$

Remarks

Set partitioning vs set covering

When costs and travel times satisfy the triangle inequality, set partitioning constraints (2)

$$\sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k = 1, \quad \forall\, i \in N$$

can be replaced by set covering constraints

$$\sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k \geq 1, \quad \forall\, i \in N.$$

- Reduces the dual space in half
- Can speed up solution process

Linearization

Constraints (6)

$$X_{ij}^k(T_i^k + t_{ij} - T_j^k) \leq 0, \quad \forall\, k \in K, (i,j) \in A$$

are nonlinear, but can be linearized as

$$T_i^k + t_{ij} - T_j^k \leq M_{ij}(1 - X_{ij}^k), \quad \forall\, k \in K, (i,j) \in A$$

where $M_{ij} = b_i + t_{ij} - a_j$ is a large constant

Yields a very weak linear relaxation

Loading variables

Constraints (5)

$$\sum_{(i,j)\in A} q_i X_{ij}^k \leq Q, \qquad \forall\, k \in K$$

can be rewritten in the same format as constraints (6)–(7) by introducing additional loading variables L_i^k:

$$X_{ij}^k (L_i^k + q_j - L_j^k) \leq 0, \qquad \forall\, k \in K, (i,j) \in A \tag{9}$$

$$0 \leq L_i^k \leq Q, \qquad \forall\, k \in K, i \in \bar{N} \tag{10}$$

where L_i^k is the total load of vehicle k up to node i (if visited)

Outline

Dantzig-Wolfe decomposition

Block diagonal structure

- Constraints (3)–(8) are separable by vehicle
- They define the feasible region of an elementary shortest path problem with resource constraints for two resources: time and load
- Elementarity is implied by the time variables (assuming there are no cycles of negative duration)
- No extreme rays
- Every extreme point corresponds to a feasible o-d path in G

Notation

Ω^k: set of extreme points (feasible paths in G including the empty o-d path, indexed $p = 0$) for vehicle k

c_p^k: cost of feasible path $p \in \Omega^k$

v_{pi}^k: binary parameter equal to 1 if path $p \in \Omega^k$ visits customer i

θ_p^k: binary variable equal to 1 if path $p \in \Omega^k$ is selected

Disaggregated formulation

$$\min \quad \sum_{k \in K} \sum_{p \in \Omega^k} c_p^k \theta_p^k \qquad (11) \quad \text{total cost}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{p \in \Omega^k} v_{pi}^k \theta_p^k = 1, \qquad \forall\, i \in N \qquad (12) \quad \text{visit each customer}$$

$$\sum_{p \in \Omega^k} \theta_p^k = 1, \qquad \forall\, k \in K \qquad (13) \quad \text{one route per vehicle}$$

$$\theta_p^k \in \{0, 1\}, \qquad \forall\, k \in K, p \in \Omega^k \qquad (14) \quad \text{binary requirements}$$

Remarks

- For the VRPTW, binary requirements on the X_{ij}^k arc-flow variables are equivalent to binary requirements on the θ_p^k path-flow variables
- Because the vehicles are all identical, the sets Ω^k, $k \in K$, are also identical and the costs c_p^k and parameters v_{pi}^k do not depend on k
- The vehicles (subproblems) can then be aggregated to yield an aggregated formulation
  - less variables
  - less symmetry
  - new notation: Ω, c_p, v_{pi}, $\theta_p = \sum_{k \in K} \theta_p^k$

Aggregated formulation

$$\min \quad \sum_{p \in \Omega} c_p \theta_p \qquad (15) \quad \text{total cost}$$

$$\text{s.t.} \quad \sum_{p \in \Omega} v_{pi} \theta_p = 1, \qquad \forall\, i \in N \qquad (16) \quad \text{visit each customer}$$

$$\sum_{p \in \Omega} \theta_p = |K|, \qquad (17) \quad \text{one route per vehicle}$$

$$\theta_p \in \{0, 1\}, \qquad \forall\, p \in \Omega \setminus \{0\} \qquad (18) \quad \text{binary requirements}$$

$$\theta_0 \geq 0, \text{integer} \qquad (19) \quad \text{integrality requirements}$$

Final path-flow formulation

Because $|K|$ is unrestrictive and Ω contains the empty o-d path, constraint (17) can be dropped

$$\min \quad \sum_{p \in \Omega} c_p \theta_p \qquad (20) \quad \text{total cost}$$

$$\text{s.t.} \quad \sum_{p \in \Omega} v_{pi} \theta_p = 1, \quad \forall\, i \in N \qquad (21) \quad \text{visit each customer}$$

$$\theta_p \in \{0,1\}, \quad \forall\, p \in \Omega \setminus \{0\} \qquad (22) \quad \text{binary requirements}$$

$$\theta_0 \geq 0, \text{integer} \qquad (23) \quad \text{integrality requirements}$$

This formulation could have been proposed directly, that is, without decomposing the arc-flow formulation

Outline

Master problem (MP)

- The integer master problem corresponds to the path-flow formulation (20)–(22)
- The master problem is the linear relaxation of this formulation and is solved by column generation

Dual variables and reduced cost

$\pi_i \geq 0$, $i \in N$: dual variables associated with constraints (21)

$\pi_o = 0$: defined for notational conciseness

Reduced cost $\bar{c}_p$ of variable θ_p:

$$\bar{c}_p = c_p - \sum_{i \in N} v_{pi}\pi_i = \sum_{(i,j) \in p} (c_{ij} - \pi_i) = \sum_{(i,j) \in p} \bar{c}_{ij}$$

$\bar{c}_{ij} = c_{ij} - \pi_i$: arc "reduced" cost

Subproblem formulation

$$\min \quad \sum_{(i,j)\in A} \bar{c}_{ij} X_{ij} \tag{24}$$

$$\text{s.t.} \quad \sum_{(o,j)\in A} X_{oj} = \sum_{(i,d)\in A} X_{id} = 1, \tag{25}$$

$$\sum_{(i,j)\in A} X_{ij} - \sum_{(j,i)\in A} X_{ji} = 0, \qquad \forall\, i \in N \tag{26}$$

$$X_{ij}(L_i + q_j - L_j) \leq 0, \qquad \forall\, (i,j) \in A \tag{27}$$

$$0 \leq L_i \leq Q, \qquad \forall\, i \in \bar{N} \tag{28}$$

$$X_{ij}(T_i + t_{ij} - T_j) \leq 0, \qquad \forall\, (i,j) \in A \tag{29}$$

$$a_i \leq T_i \leq b_i, \qquad \forall\, i \in \bar{N} \tag{30}$$

$$X_{ij} \in \{0,1\}, \qquad \forall\, (i,j) \in A \tag{31}$$

Remarks

- Index k was dropped
- Constraints (25)–(31) correspond to constraints (3)–(4) and (6)–(10)
- Constraints (27)–(30) are resource constraints
  - A resource is a quantity that varies along a path and its value must fall within a prespecified resource window at each node
  - Restrict path feasibility
- Objective (24) seeks to minimize path reduced cost
- The subproblem is an elementary shortest path problem with resource constraints (ESPPRC)
  - Strongly NP-hard problem
  - A relaxation such as the following (non-elementary) shortest path problem with resource constraints (SPPRC) can be used

SPPRC

- Same definition as ESPPRC except that cycles are allowed (example: path $p = o - 1 - 2 - 3 - 1 - 5 - 6 - 3 - d$ contains cycles $1 - 2 - 3 - 1$ and $3 - 1 - 5 - 6 - 3$)
- Columns with $v_{pi} \geq 2$ can be generated
- Enlarged set of columns Ω to consider in model (20)–(22)
- Model (20)–(22) remains valid because set partitioning constraints (21) and binary requirements (22) forbid columns with $v_{pi} \geq 2$ to be part of an integer solution
- Yields a weaker lower bound
  - For the path p above, $\theta_p = 0.5$ completely covers customers 1 and 3

Labeling algorithm

- SPPRC is usually solved by dynamic programming, more precisely, by a labeling algorithm
- In such an algorithm, partial paths start at the source node o and are represented by multi-dimensional resource vectors, called labels
- Starting from an initial label associated with node o, labels are propagated forwardly using resource extension functions (REFs) through network G (backward labeling is also possible)
- Resource windows are checked at each node to discard infeasible partial paths
- To avoid enumerating all feasible paths, a dominance rule is applied to discard unpromising labels
- Multiple labels are associated with each node
- Provides shortest paths from o to all nodes $i \in \bar{N}$

Labels for the VRPTW

- A label E_i^p representing a partial path p from node o to a node i contains three components
  - Z_i^p: (reduced) cost of p
  - L_i^p: load accumulated along p
  - T_i^p: (earliest) start of service time at i
- $E_i^p = (Z_i^p, L_i^p, T_i^p)$ is feasible if

$$L_i^p \in [0, Q] \quad \text{and} \quad T_i^p \in [a_i, b_i]$$

- No restrictions are imposed on the cost component Z which is also viewed as an unrestricted resource
- Whenever unambigous, indices p and i may be omitted in the following

Resource extension functions

- Path p (label E_i) can be extended by appending arc $(i,j) \in A$ to it
- The resulting path is represented by a label $E_j = (Z_j, L_j, T_j)$ whose components are computed using the following REFs

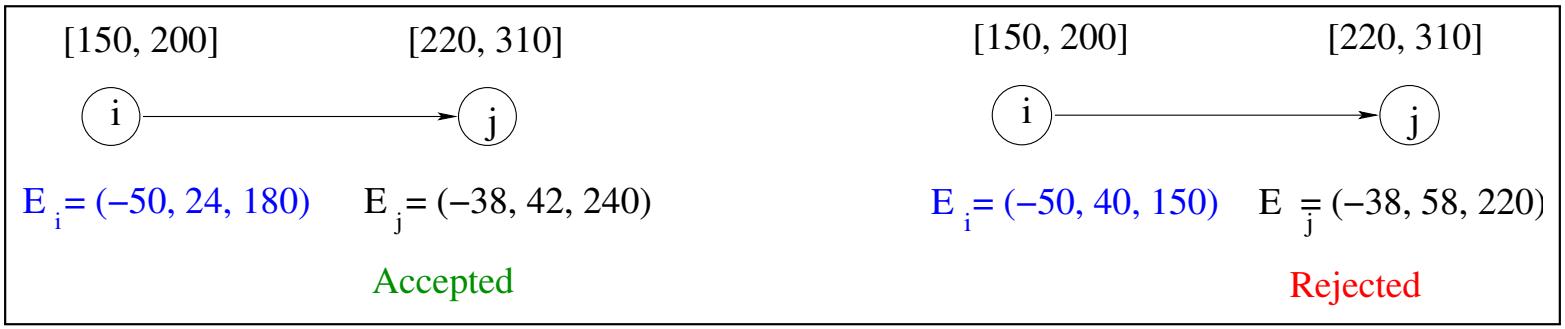
$$\begin{aligned} Z_j &= f_{ij}^{cost}(E_i) = Z_i + \bar{c}_{ij} \\ L_j &= f_{ij}^{load}(E_i) = L_i + q_j \\ T_j &= f_{ij}^{time}(E_i) = \max\{a_j, T_i + t_{ij}\} \end{aligned}$$

- The REFs $f_{ij}^{time}()$ ensure that time window lower bounds are always respected
- Therefore, E_j is feasible if $L_j \leq Q$ and $T_j \leq b_j$

Label extension: example

Consider

- a label $E_i = (Z_i, L_i, T_i)$ for a path ending in node i
- $[a_i, b_i] = [150, 200]$, $[a_j, b_j] = [220, 310]$
- $\bar{c}_{ij} = 12$, $t_{ij} = 60$, $q_j = 18$, $Q = 50$



Dominance

Principle

- Consider two labels (E^1 and E^2) representing two feasible partial paths ending at the same node
- E^2 cannot yield a shortest *o*-*d* path if
  - Every feasible extension of E^2 is also a feasible extension of E^1
  - For every such extension, the cost of the path resulting from the extension of E^1 is less than or equal to the cost of the path resulting from the extension of E^2
- In this case, we say that label E^1 dominates label E^2, which can then be discarded
- In general, these conditions cannot be verified easily

When all REFs are non-decreasing, the following (more restrictive) dominance rule can be applied

Dominance rule

Label E^1 dominates label E^2 if $E^1 \leq E^2$ componentwise (that is, $Z^1 \leq Z^2$, $L^1 \leq L^2$ and $T^1 \leq T^2$). In this case, E^2 can be discarded.

When all components are equal, keep one of the two labels

Example

- Consider three labels
  $E^1 = (-32, 25, 200)$, $E^2 = (-27, 34, 230)$,
  $E^3 = (-10, 20, 180)$
- E^1 dominates E^2
- E^1 does not dominate E^3 and vice versa (keep both labels)

Algorithm

Notation

$\mathcal{U}_i$: set of unprocessed labels at node i

$\mathcal{P}_i$: set of processed labels at node i

$i(E)$: last node of the path associated with E

$DOM(\mathcal{U}_j, \mathcal{P}_j)$: dominance algorithm applied to labels in $\mathcal{U}_j$ and $\mathcal{P}_j$ that returns a possibly reduced set $\mathcal{U}_j$

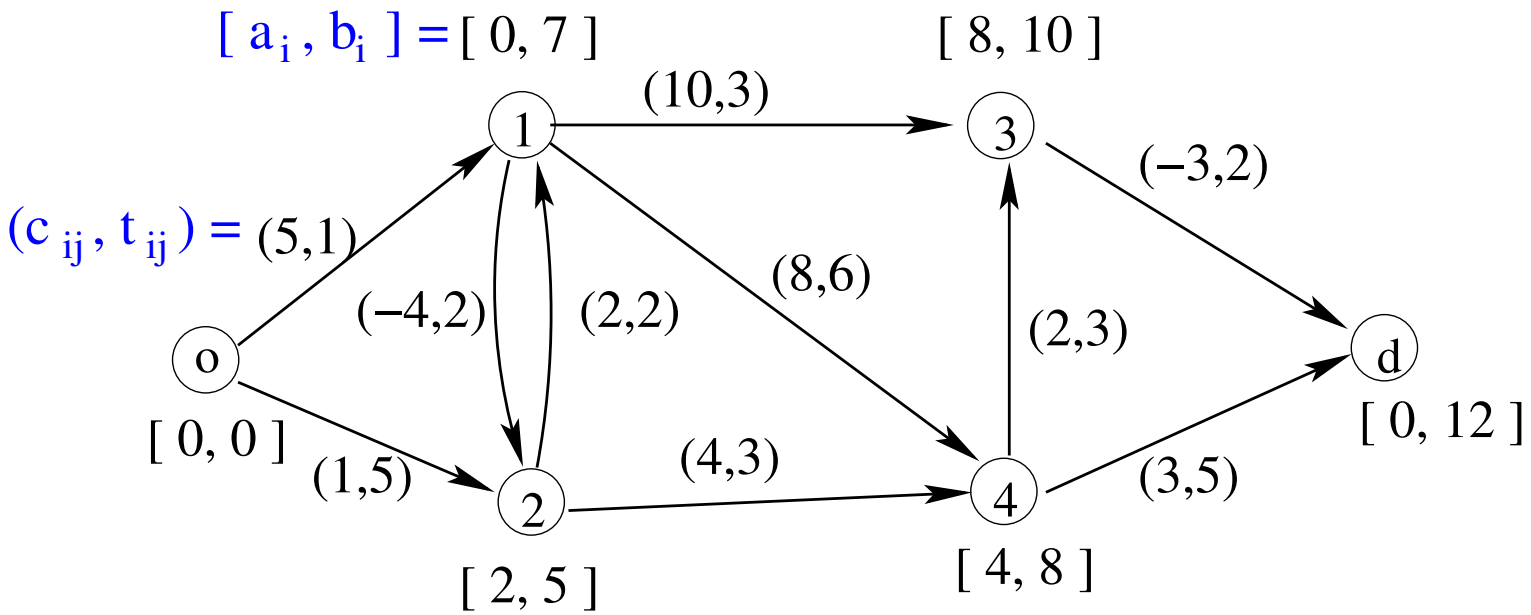
Algorithm 1 : Generic SPPRC labeling algorithm

Set $\mathcal{U}_i = \mathcal{P}_i = \emptyset$ for all $i \in \bar{N}$ except $\mathcal{U}_o = \{(0, 0, a_o)\}$
while $\bigcup_{i \in \bar{N}} \mathcal{U}_i \neq \emptyset$ **do**
  Choose a label $E \in \bigcup_{i \in \bar{N}} \mathcal{U}_i$ and remove E from $\mathcal{U}_{i(E)}$
  for all arcs $(i(E), j) \in A$ **do**
    Extend E along $(i(E), j)$ using the REFs to create a new label F
    if F is feasible **then**
      Add F to $\mathcal{U}_j$
      $\mathcal{U}_j = DOM(\mathcal{U}_j, \mathcal{P}_j)$
  Add E to $\mathcal{P}_{i(E)}$
Filter $\mathcal{P}_d$ to find a shortest o-d path

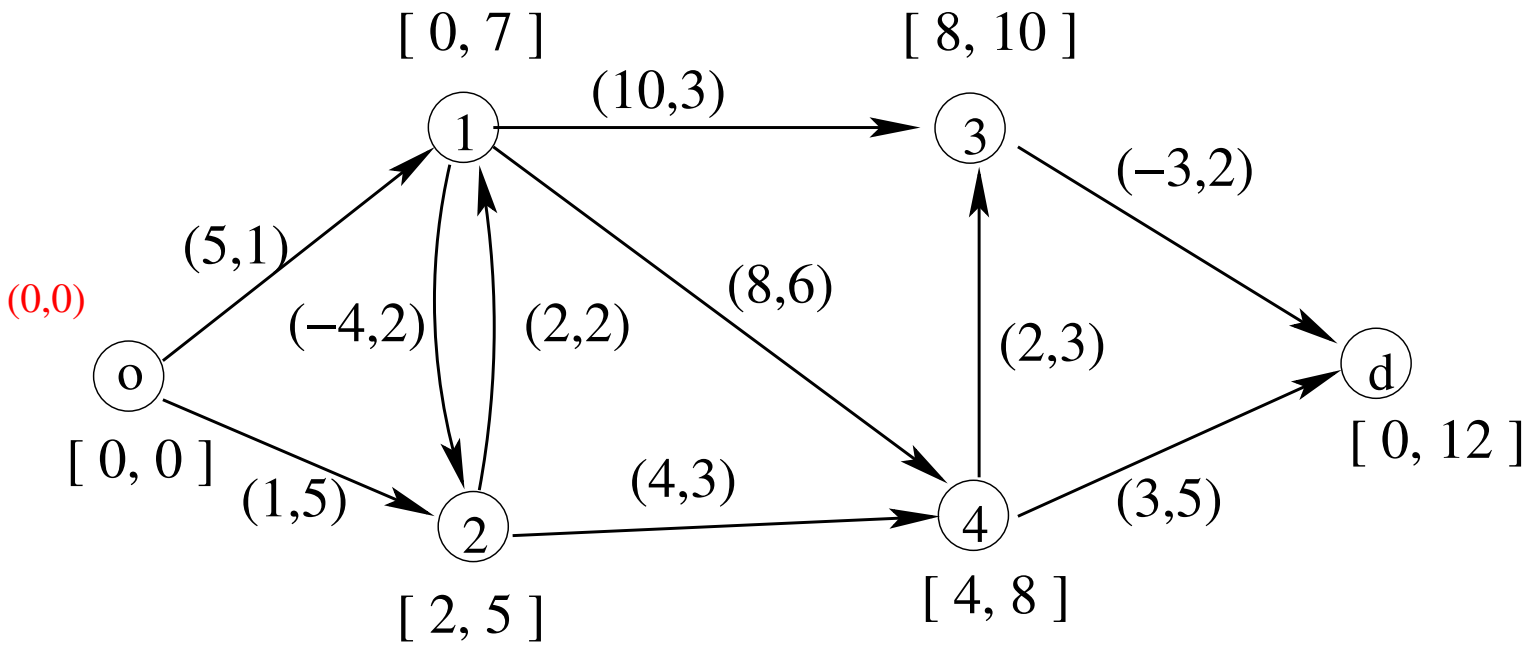
Remarks

- If there exists a resource with a positive consumption on every arc (e.g., time), then label E can be chosen in Step 3 as the one with the minimal value for this resource
  - Yields a pseudo-polynomial algorithm for the VRPTW
  - State is given by node, load and time values
  - Dimension of state space $= O(|N| \times Q \times W)$ where W is the maximum width of a resource window
  - Using this label selection rule, a state cannot be visited more than once
- In Step 8, dominance can be applied only when needed
- A pulling algorithm (labels are extended along arcs (i, j) for a fixed j) might be more efficient because it allows to apply dominance on a subset of new labels associated with the same node

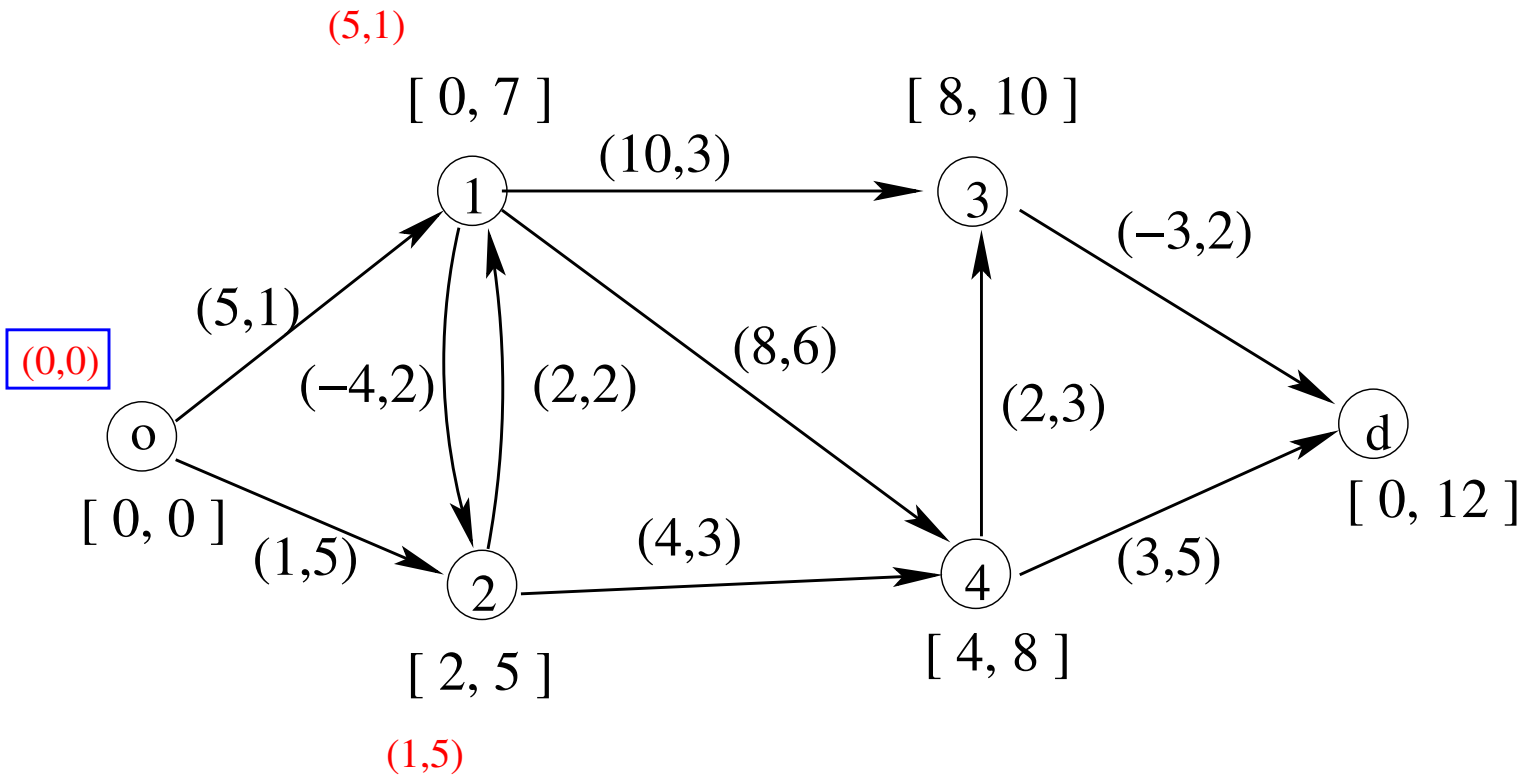
Example without capacity constraint



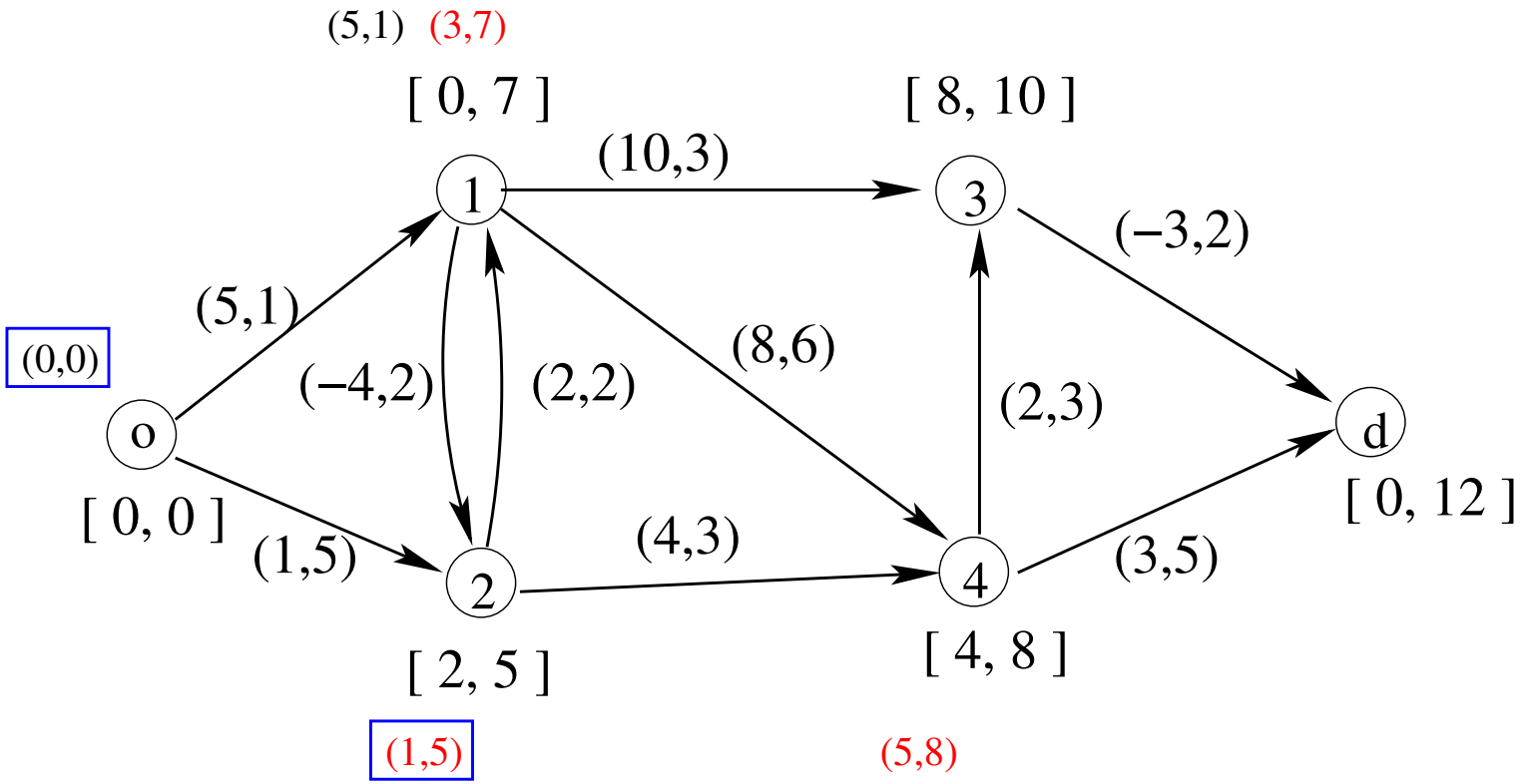
Initial label



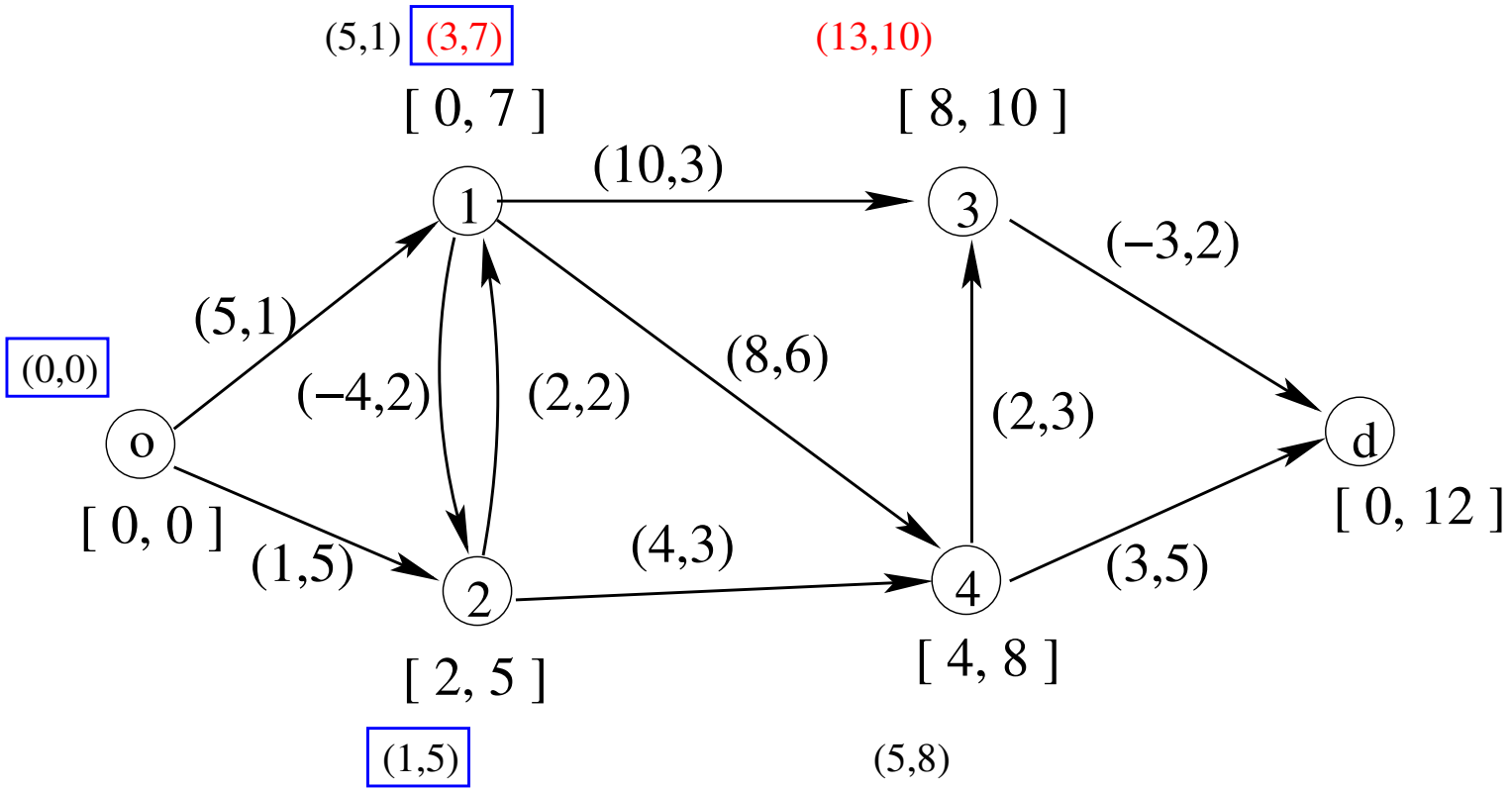
Extension of label (0, 0)



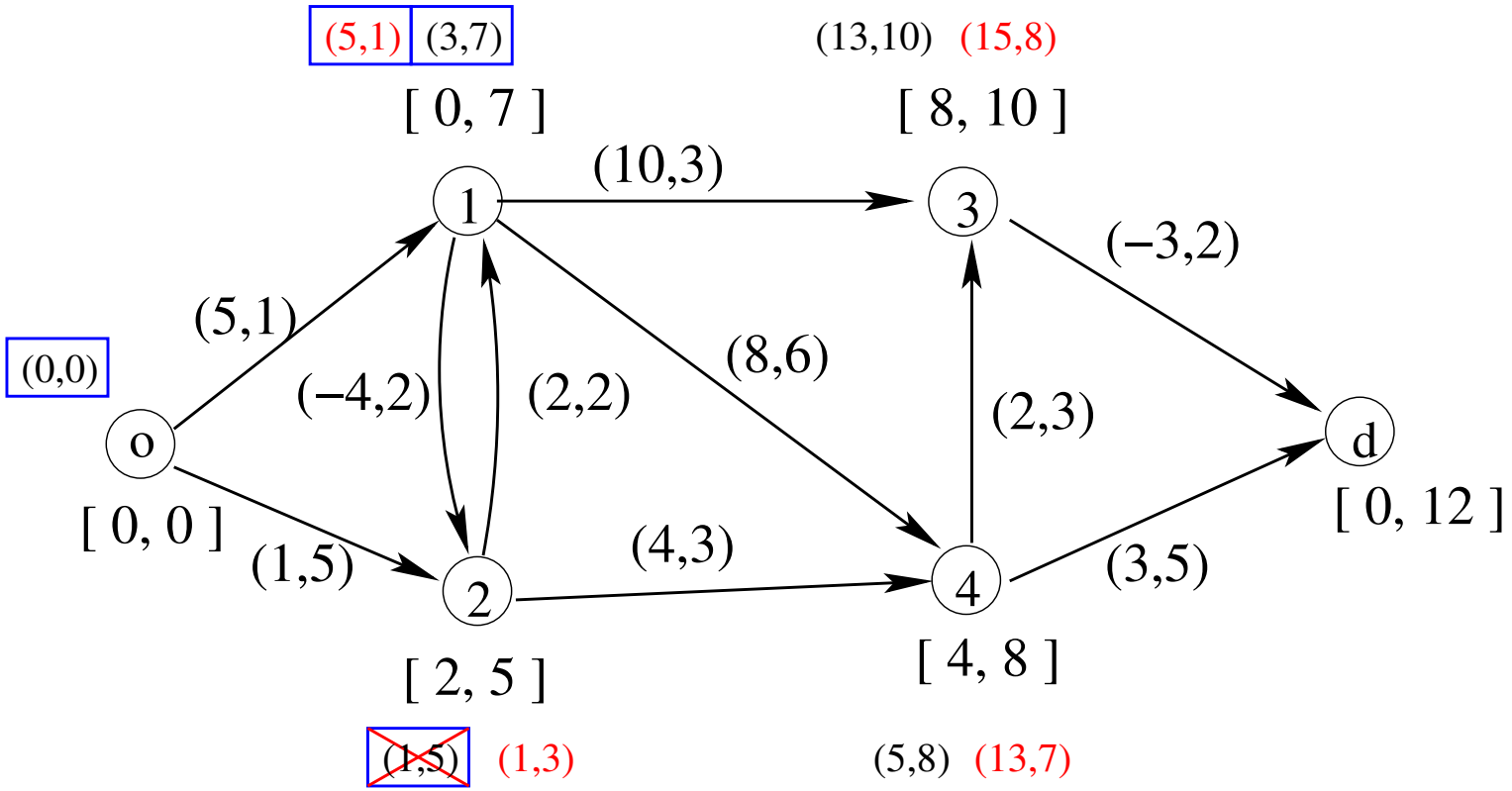
Extension of label $(1, 5)$ (least cost)



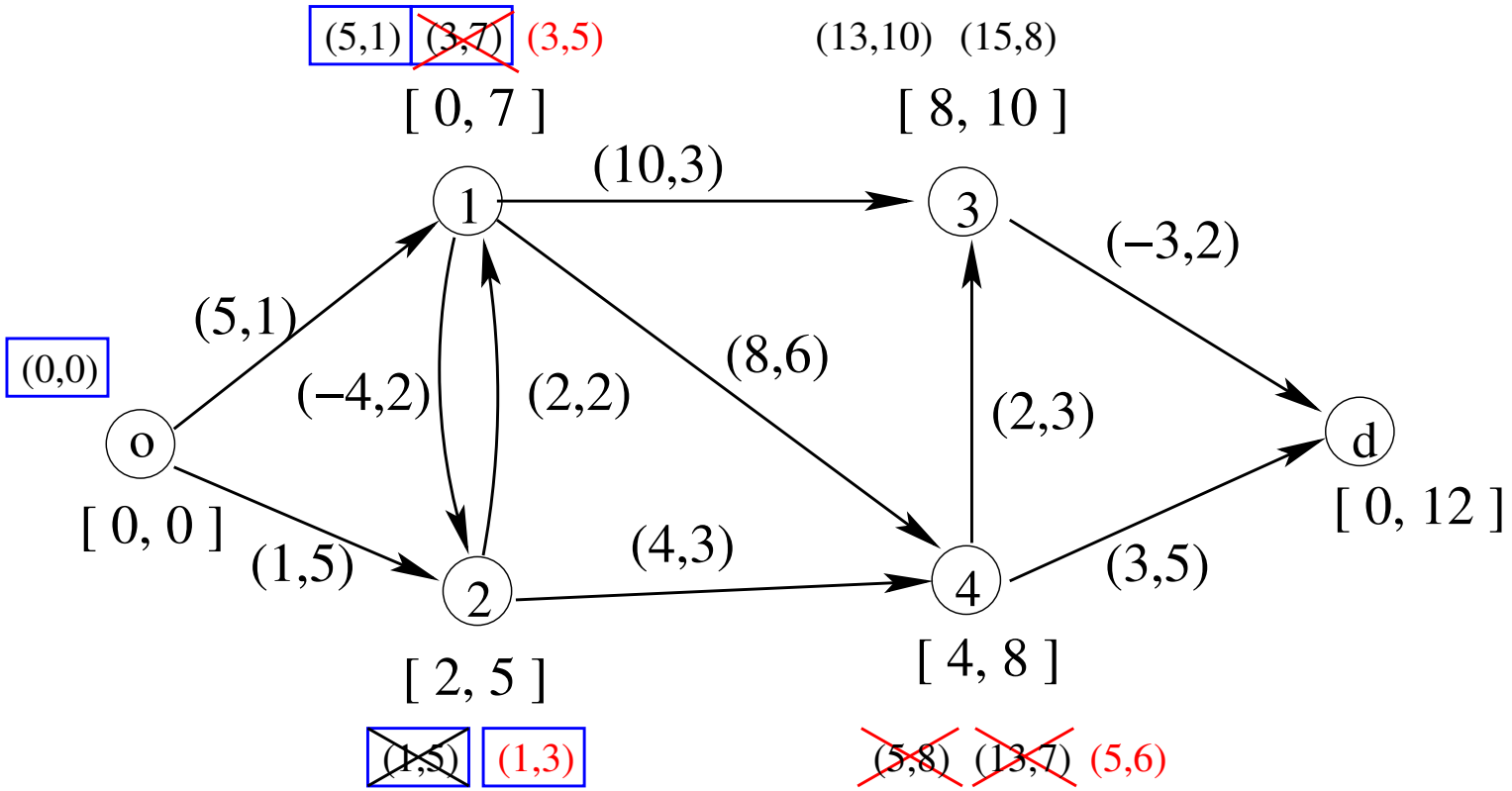
Extension of label (3, 7)



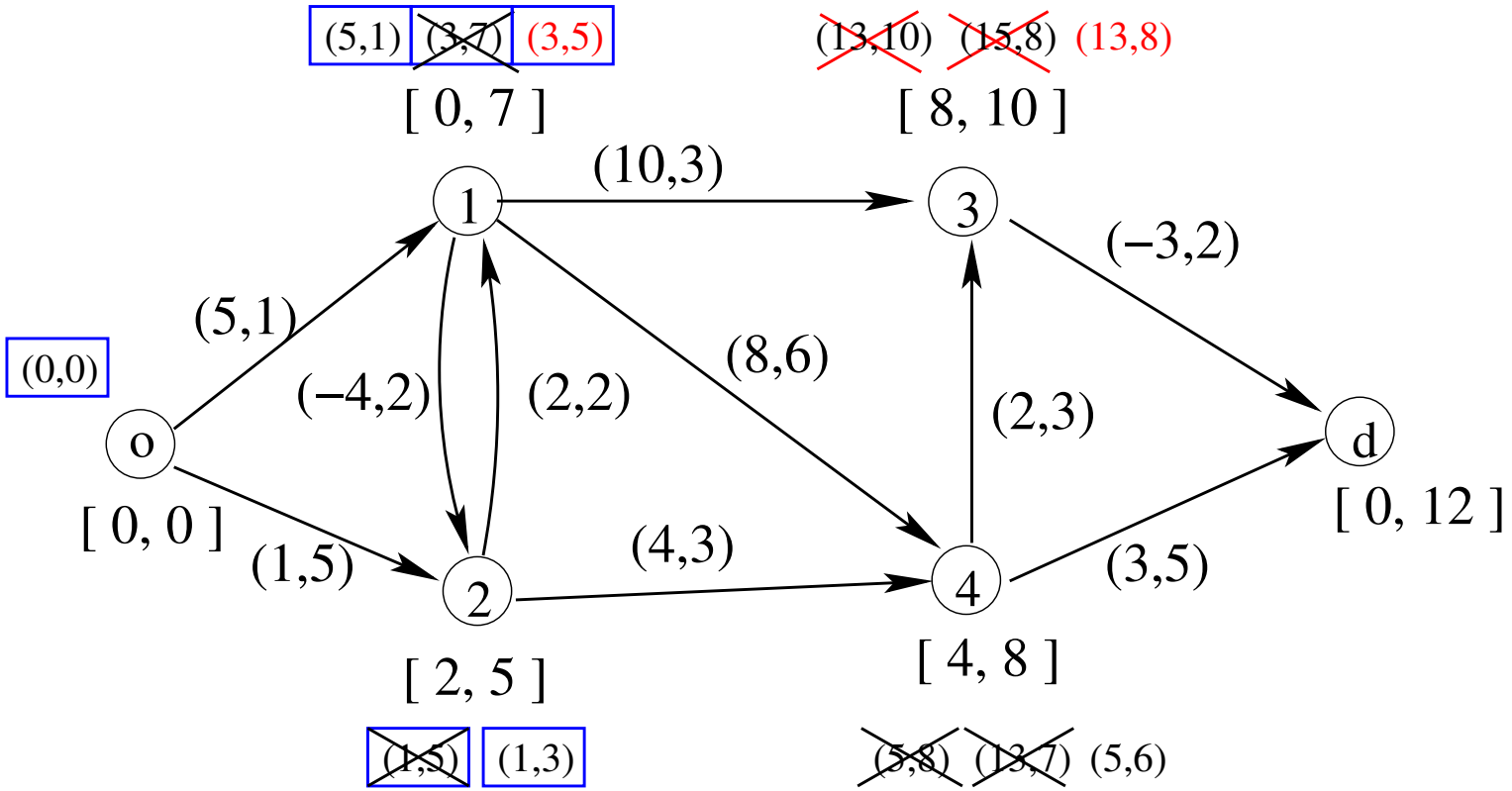
Extension of label (5, 1)



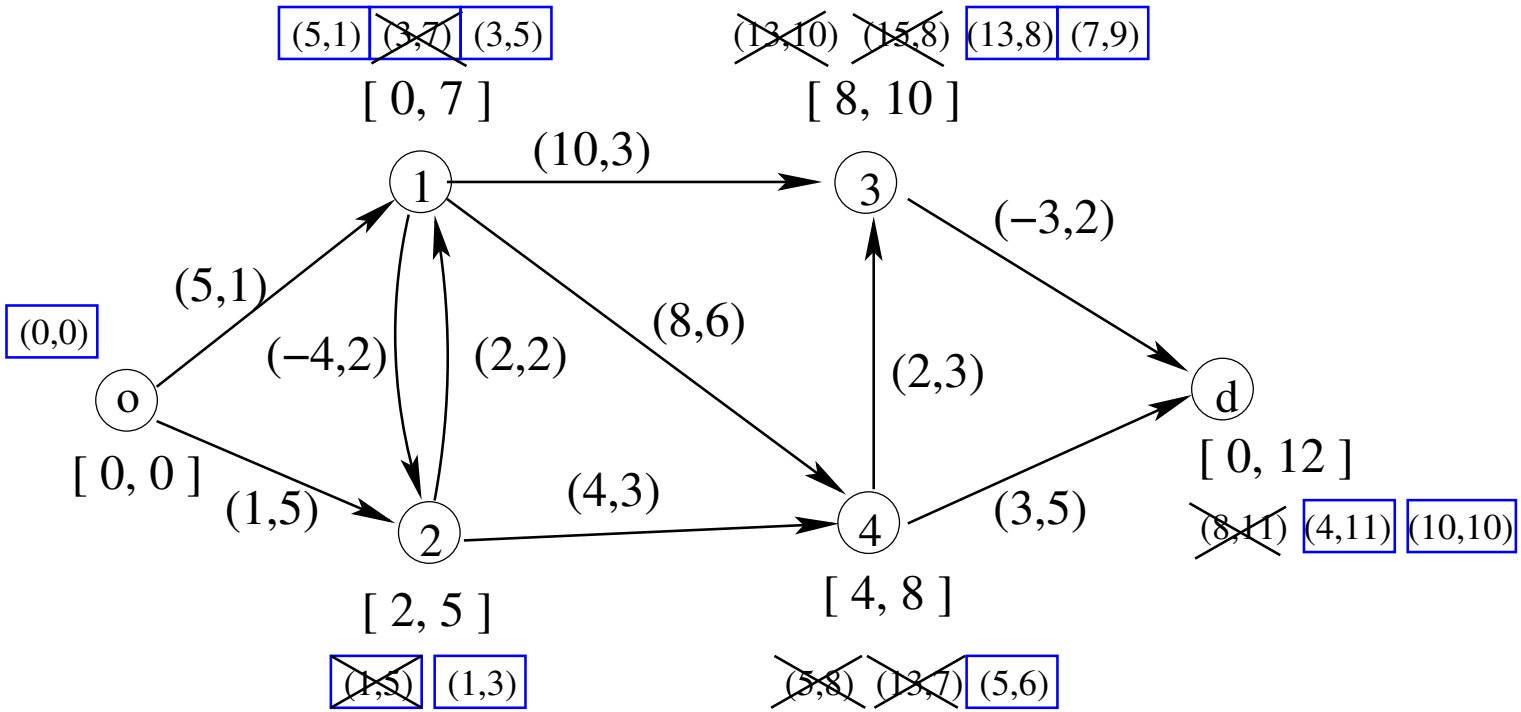
Extension of label (1, 3)



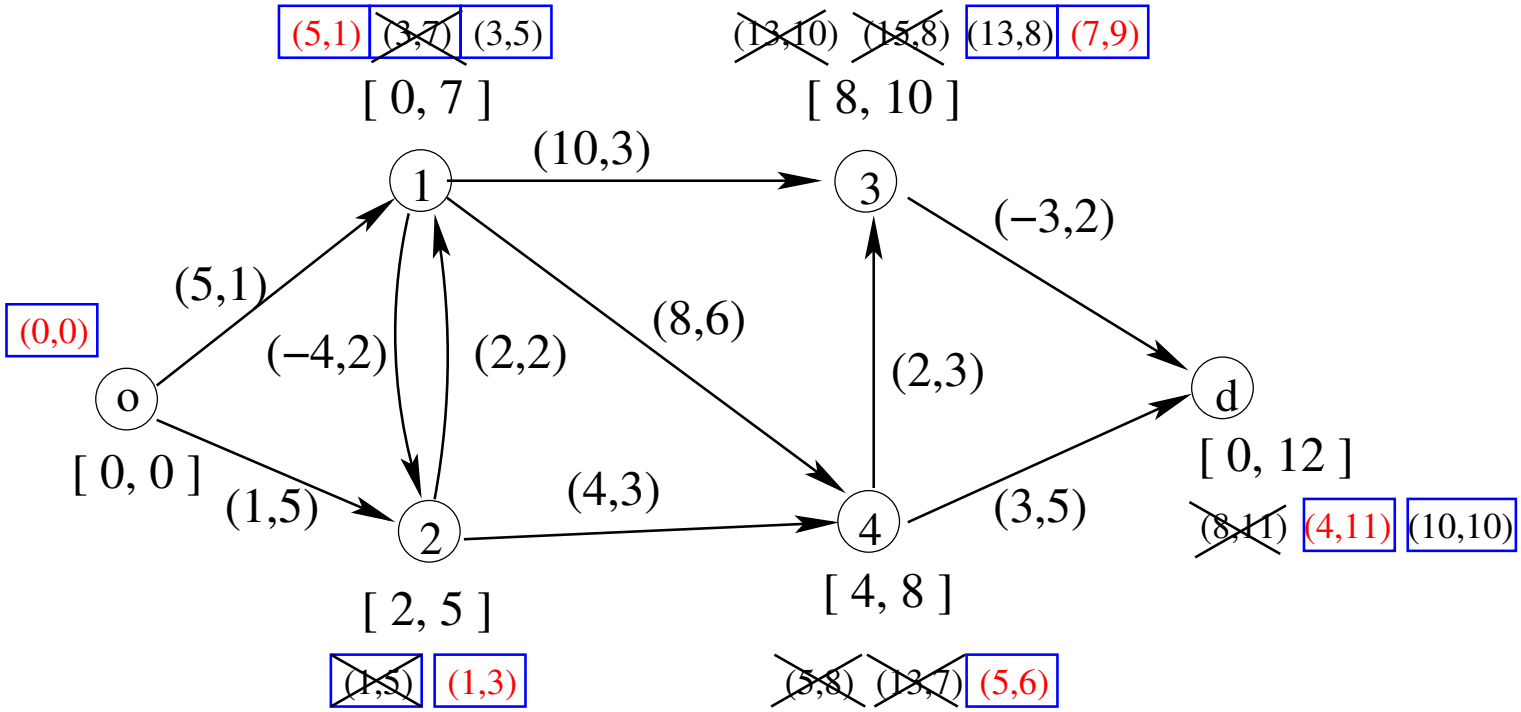
Extension of label (3, 5)



and so on


(5,1) (3,7) (3,5)
[0, 7]
(13,10) (15,8) (13,8) (7,9)
[8, 10]
(10,3)
1
3
(–3,2)
(5,1)
(0,0)
(–4,2)
(2,2)
(8,6)
(2,3)
o
d
[0, 0]
[0, 12]
(1,5)
(4,3)
(3,5)
2
4
(8,11) (4,11) (10,10)
[2, 5]
[4, 8]
(1,5) (1,3)
(5,8) (13,7) (5,6)

The shortest path is retrieved by finding the preceding labels



shortest path = o–1–2–4–3–d
optimal cost = 4
optimal path earliest arrival time at node d = 11

ESPPRC

Basic version

- Also solved by a labeling algorithm
- One additional resource per customer $k \in N$ that indicates whether this customer has been visited or not
- Corresponding resource variables: T^{cust_k}, $k \in N$
- REFs for these resources and arc (i, j):

$$T_j^{cust_k} = \begin{cases} T_i^{cust_k} + 1 & \text{if } j = k \\ T_i^{cust_k} & \text{otherwise} \end{cases} \qquad \forall\, k \in N$$

- Yields labels with $3 + |N|$ dimensions
- No pseudo-polynomial time algorithms

Improved version

- The additional resources indicate whether the customers are reachable or not
- A customer k is unreachable from a label E_i at node i (that is, $U_k(E_i) = 1$) if at least one of the following three conditions holds (assuming triangle inequality for travel times)
  - Customer k has already been visited
  - Load prevents reaching k $(L_i + q_k > Q)$
  - Time prevents reaching k $(T_i + t_{ik} > b_k)$
- REFs become:

$$T_j^{cust_k} = \begin{cases} T_i^{cust_k} + 1 & \text{if } j = k \\ \max\{T_i^{cust_k}, U_k(E_i)\} & \text{otherwise} \end{cases} \qquad \forall\, k \in N$$

- Allows more dominance

Example

- Consider two paths $p_1 = o - 1 - 3$ and $p_2 = o - 2 - 3$
- For the basic version, p_1 and p_2 yield labels E^1 and E^2

label	Z	L	T	T^{cust_1}	T^{cust_2}	T^{cust_3}
E^1	-12	18	100	1	0	1
E^2	-8	25	110	0	1	1

- In this case, E^1 does not dominate E^2

- Assume $[a_1, b_1] = [30, 50]$, $[a_3, b_3] = [75, 150]$, $[a_2, b_2] = [40, 125]$, $t_{3,2} = 20$
- For the improved version, p_1 and p_2 yield labels E^1 and E^2

label	Z	L	T	T^{cust_1}	T^{cust_2}	T^{cust_3}
E^1	-12	18	100	1	0	1
E^2	-8	25	110	1	1	1

- In this case, E^1 dominates E^2

Bidirectional search

- Select a bounded resource (e.g., time)
- Select a middle value M for this resource ($M = (b_d - a_o)/2$)
- From the source node o, generate forward labels
  - Forward extension of a label E_i along arc (i, j) is allowed if $T_i \leq M$
- From the sink node d, generate backward labels using backward REFs
  - Backward extension of a label E_j along arc (i, j) is allowed if $\min\{b_i, T_j - t_{ij}\} > M$
- At each node $i \in \bar{N}$, join forward and backward labels to identify feasible o-d paths
- Among these paths, find the shortest one

Decremental state-space relaxation

Add customer resources gradually, as needed

1. Start without any customer resources
2. Solve the problem with the current set of customer resources
3. If the computed optimal path contains a cycle
   1. Add a customer resource to forbid this cycle
   2. Return to Step 2

Remarks

- In Step 3.1, there are several ways to select the resource
- In a column generation context
  - Add a condition to the test in Step 3: the current optimal value is negative
  - In Step 1, start with the set of customer resources used at the end of the previous column generation iteration

Other relaxations

Partial ESPPRC

- Select a subset S of customers on which to impose elementarity
  - Define a customer resource only for these customers
- Cycles are allowed for the other customers
- Customers in S can be selected dynamically (m = maximal size of S)
  1. Start with $S = \emptyset$
  2. Solve MP by column generation
  3. If $|S| < m$ and the computed MP solution contains a path with a cycle
     1. Add a customer in S and its corresponding resource to forbid this cycle
     2. Return to Step 2

SPPRC with k-cycle elimination

- In a path, a k-cycle occurs when the same node is visited twice with $k-1$ nodes between these two visits
- Example: $p = o - 1 - 2 - 3 - 1 - 4 - 5 - 4 - d$ contains a 2-cycle (for node 4) and a 3-cycle (for node 1)
- SPPRC with k-cycle elimination forbids paths with l-cycles for $l = 2, \ldots, k$
- Yields stronger lower bounds than SPPPRC
- Easy to implement for $k = 2$
- Can be used in decremental state-space relaxation for ESPPRC

SPPRC with neighborhoods (*ng*-paths)

- Associate with each node $i \in N$ a neighborhood $N_i \subset N$ (e.g., node i and its closest nodes)
- Allows a cycle for a node i only if this cycle contains a node j for which $i \notin N_j$ (returning to i from j is a detour)

Example

- Consider the following neighborhoods
  - $N_1 = \{1, 2, 3\}$, $N_2 = \{1, 2, 5\}$, $N_3 = \{1, 3, 4\}$, $N_4 = \{3, 4, 5\}$, $N_5 = \{2, 4, 5\}$
- Path $p = o - 1 - 2 - 3 - 1 - 5 - d$ is not feasible
- Path $p = o - 1 - 3 - 5 - 3 - 4 - d$ is feasible

Treatment in the labeling algorithm

- Use customer resources T^{cust_k}, $k \in N$
- For a path $p = o - i_1 - \ldots - i_p$, T^{cust_k} is equal to 1 if k belongs to p (that is, $k = i_g$ for $g \in \{1, \ldots, p\}$) and $k \in N_{i_h}$ for all $h \in \{g, \ldots, p\}$ and 0 otherwise
- Thus, $T^{cust_k} = 0$ for a node k if k has not been visited or if it does not belong to the neighborhood of a node visited after its last visit
- REFs for arc (i, j)

$$T_j^{cust_k} = \begin{cases} 1 & \text{if } j = k \text{ or } (T_i^{cust_k} = 1 \text{ and } k \in N_j) \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in N$$

- At node i, number of customer resources at 1 is less than or equal to $|N_i|$

Heuristics

- Solving the ESPPRC exactly might be highly time consuming
- Its exact solution is required only for proving the optimality of the current MP solution
- Fast heuristics can be used to generate negative reduced cost columns
  - Heuristic labeling algorithm
  - Local search (meta)heuristics

Heuristic labeling algorithm

- Use a reduced network
  - Omit arcs based on their cost or their reduced cost
- Use a relaxed dominance rule
  - Consider a subset of the customer resources
  - Increases the number of dominated labels
  - All customer resources are considered to check feasibility

Tabu search heuristic

- Each column in the basis of the current MP solution is associated with a zero reduced cost path
- These paths are good initial solutions for a tabu search algorithm

- Tabu search algorithm
  - Iterative method that explores a neighborhood at each iteration to find the best neighbor solution (cost might deteriorate)
  - Neighborhood defined by feasible moves
    - Insert or remove a node from the current path
  - Selected move remains tabu (cannot be undone) for a fixed number of iterations
  - Stop after a fixed number of iterations (switch to next initial solution)

Column generator usage

Column generator characteristics

- Tabu search: very fast (when small number of iterations per initial column), generates a large number of columns in the early CG iterations, efficiency decreases rapidly toward the end
- Heuristic labeling: fast, more efficient than tabu search to find columns but generates less columns in the early CG iterations
- Exact labeling: very slow, highly efficient to find columns

Usage

At each CG iteration

- Use tabu search algorithm
- If it fails, use heuristic labeling algorithm
- If it fails again, use exact labeling algorithm

Solomon's instances

- 56 instances with 100 customers
- 6 classes: R1, C1, RC1, R2, C2, RC2
  - R: random customer locations
  - C: clustered customer locations
  - RC: mixed locations
  - 1: relatively tight time windows
  - 2: large time windows

Linear relaxation results (with or without tabu)

	with tabu					without tabu			
Instance	cpu (s)	nb it	nb tabu	nb hdp	nb edp	cpu (s)	nb it	nb hdp	nb edp
RC104	36	77	57	19	1	35	152	149	3
RC108	26	56	40	14	2	42	136	131	5
R108	26	84	66	17	1	23	102	99	3
R112	30	61	47	12	2	23	61	59	2
RC203	95	117	103	13	1	1131	608	599	9
RC206	68	140	116	23	1	78	449	448	1
RC207	148	133	104	28	1	397	374	371	3
R203	172	214	174	39	1	758	613	601	12
R205	97	170	135	34	1	77	351	349	2
R206	210	183	137	45	1	864	511	503	8
R209	181	204	164	39	1	278	405	403	2
R210	269	184	141	41	2	1420	451	438	13

From Desaulniers et al. (2008)

Lower bounds (easy RC and R instances)

Instance	spprc	spprc 2-ce	spprc 3-ce	pespprc $E_{max} = 20$	pespprc $E_{max} = 30$	espprc	UB
RC101	1567.5	1584.1	*id*	*id*	*id*	*id*	1619.8
RC102	1380.2	1403.6	1404.9	1406.3	*id*	*id*	1457.4
RC103	1170.3	1218.5	1221.7	1224.6	1225.5	*id*	1258.0
RC104	1052.6	1094.3	1097.2	1098.7	1101.1	1101.8	1132.3
RC105	1453.9	1471.2	1471.8	1471.9	*id*	*id*	1513.7
RC106	1249.0	1308.8	1317.2	1318.0	1318.8	*id*	1372.7
RC107	1117.3	1170.7	1181.1	1182.8	1183.4	*id*	1207.8
RC108	1035.9	1063.0	1066.8	1068.3	1072.9	1073.4	1114.2
R101	1631.1	*id*	*id*	*id*	*id*	*id*	1637.7
R102	1466.6	*id*	*id*	*id*	*id*	*id*	1466.6
R103	1203.2	1206.3	1206.5	1206.8	*id*	*id*	1208.7
R104	937.1	949.1	955.3	955.4	956.9	956.9	971.5
R105	1341.2	1346.1	*id*	*id*	*id*	*id*	1355.3
R106	1212.3	1226.4	1226.4	1226.9	*id*	*id*	1234.6
R107	1037.0	1051.8	1052.2	1053.2	*id*	*id*	1064.6
R108	891.7	907.2	911.5	913.3	913.5	*id*	932.1
R109	1097.5	1130.6	1134.3	1134.3	*id*	*id*	1146.9
R110	1021.3	1048.5	1052.5	1055.4	1055.6	*id*	1068.0
R111	1005.9	1032.0	1034.2	1034.7	1034.7	*id*	1048.7
R112	892.5	919.2	923.6	926.0	926.7	926.7	948.6

From Desaulniers et al. (2008)

Lower bounds (difficult RC and R instances)

Instance	spprc	spprc 2-ce	spprc 3-ce	pespprc $E_{max} = 40$	pespprc $E_{max} = 50$	espprc	UB
RC201	1108.9	1240.4	1255.4	1255.9	*id*	*id*	1261.8
RC202	882.6	1004.1	1050.9	1088.1	*id*	*id*	1092.3
RC203	698.1	815.3	-	920.6	922.5	*id*	923.7
RC204	608.6	688.3	-	772.4	777.9	-	783.5
RC205	967.1	1055.5	1117.9	1147.6	*id*	*id*	1154.0
RC206	852.2	952.2	1005.9	1034.6	1037.4	1038.6	1051.1
RC207	769.1	866.3	886.9	931.5	937.8	947.3	962.9
RC208	627.2	704.6	-	762.2	764.4	-	776.1
R201	1080.7	1136.2	1140.0	1140.3	*id*	*id*	1143.2
R202	934.2	1009.8	-	1022.2	*id*	*id*	1029.6
R203	756.7	846.5	-	863.7	866.9	*id*	870.8
R204	640.4	689.8	-	722.1	724.7	-	731.3
R205	838.4	916.6	930.6	937.9	938.9	*id*	949.8
R206	749.1	834.6	-	865.9	866.5	866.9	875.9
R207	668.9	747.6	-	786.3	789.8	790.7	794.0
R208	610.7	661.7	-	-	-	-	701.0
R209	750.4	819.8	833.9	839.5	840.9	841.4	854.8
R210	754.0	849.3	-	884.5	885.7	889.4	900.5
R211	650.8	705.8	-	731.9	734.4	-	746.7

From Desaulniers et al. (2008)

Strong degree constraints (SDC)

Alternative strategy to impose elementarity

- Paths with cycles can be generated
- When the linear relaxation solution contains a cycle starting at customer $i \in N$, add the following strong degree constraint in the MP

$$\sum_{p \in \Omega} w_{pi} \theta_p \geq 1,$$

where $w_{pi} = 1$ if customer i is visited (at least once) in path p and 0 otherwise

- Several constraints can be added at the same time

Modifications to the labeling algorithm if combined with SPPRC

- σ^k: Dual variable for SDC associated with $k \in N$
- Resource T^{cust_k} for every customer i associated with an SDC
- When extending a label along an arc (i, j) with an SDC at j, subtract σ^j from Z_j if $T_i^{cust_j} = 0$ (i.e., j is visited for the first time)
- Replace the dominance condition on the reduced cost by

$$Z^1 + \sum_{k \in N \mid T^{cust_k,1}=1 \;\&\; T^{cust_k,2}=0} \sigma^k \leq Z^2$$

- No direct dominance conditions on the T^{cust_k} resources
- ESSPRC is equivalent to using $\sigma^k = \infty, \forall k \in N$
- Can be adapted to be combined with *ng*-paths

Computational experiments

Algorithm components

CPA: Cyclic paths are allowed but, when cycles occur in a linear relaxation solution, customer resources, neighbors, or SDCs are added to get rid of them

EPO: Only elementary paths are generated (customer resources or neighbors are added if needed at each CG iteration)

DSSR: Customer resources with decremental state-space relaxation

ng-Aug: *ng*-paths with neighborhoods augmented dynamically

SDC: Strong degree constraints added dynamically

CPU times (14 difficult instances)

Instance	EPO-DSSR	CPA-DSSR	EPO-*ng*-Aug	CPA-*ng*-Aug	CPA-SDC	CPA-*ng*-SDC
C203	312	120	193	**55**	217	**55**
C204	9,272	**291**	7,612	1,450	630	1,455
R203	**44**	200	150	199	121	146
R204	*time*†	*time*†	*time*†	1,161	3,033	**753**
R206	804	486	617	348	**192**	253
R207	*time*†	14,040	3,932	2,384	2,022	**585**
R208	*time*†	*time*†	*time*†	*time*†	35,870	**12,398**
R209	**74**	347	192	183	163	128
R210	273	542	365	202	204	**123**
R211	*time*†	*time*†	3,036	1,078	**361**	626
RC203	**46**	352	261	220	423	152
RC204	*time*†	*time*†	7,348	3,651	11,220	**1,607**
RC207	278	328	139	147	164	**117**
RC208	*time*†	*time*†	9,255	6,123	1,875	**1,326**
No. solved	8	9	12	13	14	14
Average	–	–	–	–	4,035	1,409

† Optimization aborted due to the time limit reached (36,000 s).

From Contardo and Desaulniers (2013)

Branching on the MP variables

- In a typical branch-and-bound algorithm, we can branch directly on the model variables
- Here, this would correspond to imposing the decisions:

$$\theta_p = 0 \qquad \text{and} \qquad \theta_p = 1$$

- Usually inefficient
  - Branch $\theta_p = 1$ is strong (good increase of LB)
  - Branch $\theta_p = 0$ is very weak (close to no improvement of LB)
- Difficult implementation
  - Both decisions imposed in the MP
  - Decision $\theta_p = 0$ destroys the subproblem structure because p must not be generated again

Branching on the arc-flow formulation variables

Principle

If you can define branching rules on the variables of the arc-flow formulation, then you can easily rewrite them in terms of the MP variables or impose them directly in the subproblem

Different rules on

1. Number of vehicles used
2. Time variables
3. Aggregated arc-flow variables

1 and 3 are the most popular; other rules exist

Number of vehicles used

- In the arc-flow model, the number of vehicles used in a solution is given by $V = \sum_{k \in K} \sum_{(o,j) \in A} X_{oj}^k$
- Applying the DW decomposition principle and aggregating the vehicles, we get

$$V = \sum_{k \in K} \sum_{(o,j) \in A} X_{oj}^k = \sum_{k \in K} \sum_{p \in \Omega^k} \theta_p^k = \sum_{p \in \Omega} \theta_p$$

- If V takes a fractional value $\tilde{V}$, then one can impose the rules

$$\sum_{p \in \Omega} \theta_p \leq \lfloor \tilde{V} \rfloor \qquad \text{and} \qquad \sum_{p \in \Omega} \theta_p \geq \lceil \tilde{V} \rceil$$

Treatment

- These decisions are global (they cannot be verified one route at a time) and imposed in the MP by adding the corresponding constraints
- Let μ be the dual variable associated with such a decision
- Because $\sum_{p \in \Omega} \theta_p = \sum_{k \in K} \sum_{(o,j) \in A} X_{oj}^k$, we see that μ can be subtracted from the arc (reduced) cost of every arc $(o,j) \in A$

Remarks

- Not necessarily sufficient to yield an integer solution
- Branch $\sum_{p \in \Omega} \theta_p \leq \lfloor \tilde{V} \rfloor$ may yield a hard-to-solve subproblem

Time variables

- In a fractional solution of the MP, different paths with a fractional value may visit the same node at different times
- Branching on time variables eliminate some of these paths

- Select a time $\tau \in [a_i, b_i[$ such that the visiting time is less than or equal to τ for some paths and greater than τ for others
- Impose the decisions

$$T_i \in [a_i, \tau] \qquad \text{and} \qquad T_i \in [\tau + 1, b_i]$$

Treatment

- These decisions are local (they can be verified one route at a time) and imposed in the subproblem by modifying the corresponding time window
- The columns in the current restricted MP that cannot respect the imposed decision must be removed from it

Remarks

- Not necessarily sufficient to yield an integer solution
- Choosing the best candidate (node i and value τ) is not easy

Aggregated arc-flow variables

- In the arc-flow model, the aggregated flow on an arc (i, j) in a solution is given by $X_{ij} = \sum_{k \in K} X_{ij}^k$
- It must take value 0 or 1 in an integer solution
- If X_{ij} takes a fractional value, then one can impose the rules

$$X_{ij} = 0 \qquad \text{and} \qquad X_{ij} = 1$$

Treatment

- These decisions are local and imposed in the subproblem by modifying network G
  - For $X_{ij} = 0$, remove arc (i, j)
  - For $X_{ij} = 1$, remove all arcs (i, j') with $j' \neq j$ and all arcs (i', j) with $i' \neq i$
- The columns in the current restricted MP that do not respect the imposed decision must be removed from it

Cutting planes

Two types of cutting planes have been successfully used

- k-path cuts
- subset row cuts

k-path inequalities

- Generalization of the capacity inequalities for the CVRP
- $S \subseteq N$: subset of customers
- $\delta^-(S) \subset A$: subset of arcs (i,j) entering S ($i \notin S$ and $j \in S$)
- $k(S)$: lower bound on the number of vehicles required to service the customers in S
- Corresponding k-path inequality

$$\sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k \geq k(S)$$

- Never violated by a solution of the path-flow model if $k(S) = 1$

Computation of $k(S)$

- For a given set S, lower bounds are given by
  1. lb_1 = Optimal value of a bin packing problem with items of length q_i, $i \in S$, and bins of length Q (NP-hard)
  2. $lb_2 = \left\lceil \sum_{i \in S} q_i / Q \right\rceil$ (easy to compute but weaker)
  3. lb_3 = Optimal value of a vehicle scheduling problem that minimizes the number of vehicles required to service the customers in S considering only the time windows (NP-hard)
  4. $lb_4 = 2$ if the traveling salesman problem with time windows (TSPTW) for S is infeasible and 1 otherwise (still NP-hard but easier to compute)
- In practice, $k(S)$ is set to $\max\{lb_2, lb_4\}$
- When time windows are more constraining than capacity, $lb_2 \leq lb_4$ and $k(S) = lb_4 \quad \Rightarrow \quad$ only 2-path cuts can be found

Example

- Consider three customers 1, 2 and 3 with $[a_1, b_1] = [10, 15]$, $[a_2, b_2] = [20, 25]$, $[a_3, b_3] = [30, 40]$, $t_{12} = 15$, $t_{23} = 20$, $t_{13} = 25$
- Arcs $(1, 2)$, $(2, 3)$ and $(1, 3)$ exist
- To use arc $(2, 3)$, start of service time at 2 must equal 20
- Assume a fractional solution with three paths p_1, p_2 and p_3 such that
  - p_1 contains arc $(1, 2)$ and $y_{p_1} = 0.5$
  - p_2 contains arc $(2, 3)$ and $y_{p_2} = 0.5$
  - p_3 contains arc $(1, 3)$ and $y_{p_3} = 0.5$
- For $S = \{1, 2, 3\}$, $k(S) = 2$ and the corresponding 2-path inequality is violated by this solution

Separation

- No efficient algorithm yet
- Enumerate some sets S
- For each set
  - Compute lb_2
  - If $\sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k < lb_2$, then a cut is found for $k(S) = lb_2$
  - Else if $1 < \sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k < 2$, then solve a TSPTW to compute lb_4
    - If $\sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k < lb_4$, then a cut is found for $k(S) = lb_4$

Treatment

- Applying DW decomposition to $\sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k \geq k(S)$ yields

$$\sum_{p \in \Omega} \sum_{(i,j) \in \delta^-(S)} v_{pij} \theta_p \geq k(S)$$

  where $v_{pij} = 1$ if path p uses arc (i,j) and 0 otherwise
- This cut is added to the MP
- In the subproblem, its dual variable is subtracted from the arc reduced cost of all arcs in $\delta^-(S)$
- Weak version of the cuts because a same path p contributes more than once if it enters S more than once $(\sum_{(i,j) \in \delta^-(S)} v_{pij} > 1)$

Subset row inequalities

- Defined directly on the MP variables θ_p
- Chvátal-Gomory inequalities of rank 1
- S: subset of customers (rows)

$$\sum_{p \in \Omega} \lfloor \frac{1}{k} \sum_{i \in S} v_{pi} \rfloor \theta_p \leq \lfloor \frac{|S|}{k} \rfloor, \qquad \forall\, S \subseteq N,\ 2 \leq k < |S|$$

- Applied for subsets of three customers ($|S| = 3,\ k = 2$)
  $\Rightarrow$ clique cuts defined for three customers

$$\sum_{p \in \Omega} l_{pS} \theta_p \leq 1, \qquad \forall\, S \subseteq N, |S| = 3$$

where $l_{pS} = 1$ if path p visits at least two customers in S

Example

- The fractional solution of the previous example violates the subset row inequality defined for $S = \{1, 2, 3\}$ because $l_{p_1 S} \geq 2$, $l_{p_2 S} \geq 2$ and $l_{p_3 S} \geq 2$
- It would also violate it even if $k(S) = 1$, for instance, if $t_{23} = 15$

Separation

- NP-hard separation problem
- For $|S| = 3$, complete enumeration of all subsets S

Treatment if ESPPRC

- $\mathcal{SR}$: set of subsets S with a subset row cut
- Add the cuts for $S \in \mathcal{SR}$ in MP (non-positive dual variables σ^S)
- In the ESPPRC subproblem (option 1)
  - Add one unrestricted resource for each cut
    - R^S = number of customers visited in subset S modulo 2
  - When extending a label along an arc (i, j), subtract σ^S from Z_j if $R_i^S = 1$ and $R_j^S = 0$ for all $S \in \mathcal{SR}$
  - In the dominance rule
    $Z_1 \leq Z_2$ and $R_1^S \leq R_2^S$ for all $S \in \mathcal{SR}$

Treatment if ESPPRC (cont'd)

- In the ESPPRC subproblem (option 2)
  - Add one unrestricted resource for each cut
    - $\tilde{R}^S$ = number of customers visited in subset S (no modulo)
  - When extending a label along an arc (i,j), subtract σ^S from Z_j if $\tilde{R}_i^S = 1$ and $\tilde{R}_j^S = 2$ for all $S \in \mathcal{SR}$
  - In the dominance rule, replace all previous conditions by

$$Z_1 - \sum_{S \in \mathcal{SR} : \tilde{R}_1^S = 1 \ \& \ \tilde{R}_2^S = 0} \sigma^S \leq Z_2$$

  - May allow to eliminate more labels in the case $(R_1^S, R_2^S) = (1,0)$ which may correspond to the cases $(\tilde{R}_1^S, \tilde{R}_2^S) = (1,0)$ or $(1,2)$

Remarks

- Handling subset row cuts is costly
  - Limit the number of these cuts generated at once (cuts for different customers)
  - Generate 2-path cuts first

Integer solution results (easy RC and R instances)

	Desaulniers et al. (2008)									Jepsen et al. (2008)	
	pespprc $E_{max} = 30$ 2-pc, src			espprc 2-pc, src			espprc src			espprc src	
Instance	cpu (s)	nb cuts	nb BB	cpu (s)	nb cuts	nb BB	cpu (s)	nb cuts	nb BB	cpu (s)	nb BB
RC101	18	148	1	17	148	1	19	87	1	12	1
RC102	119	212	1	124	225	1	120	193	3	77	1
RC103	341	239	5	295	247	5	541	262	5	2706	3
RC104	3707	396	17	2874	345	17	11773	437	21	65807	7
RC105	17	58	1	14	57	1	33	79	1	27	1
RC106	3045	846	63	3268	1017	83	3916	755	71	15892	37
RC107	92	117	1	135	158	1	161	158	1	154	1
RC108	306	205	1	279	138	1	635	228	1	3365	1
R101	8	16	15	8	16	15	8	19	15	2	3
R102	3	0	1	3	0	1	*id*	*id*	*id*	4	1
R103	32	72	1	21	76	1	20	53	1	24	1
R104	2612	460	11	2260	406	11	3103	391	11	32343	3
R105	26	144	3	25	144	3	36	144	3	43	5
R106	135	169	3	122	162	3	87	144	3	75	1
R107	281	193	3	359	205	5	416	227	5	1310	3
R108	921	345	1	875	325	1	891	296	1	5912	1
R109	1383	596	55	1406	655	69	1127	588	65	1432	19
R110	484	227	9	459	219	9	426	219	5	1068	3
R111	5145	968	95	5411	881	101	5738	736	111	83931	39
R112	24409	727	31	19313	605	21	16073	574	19	202804	9

From Desaulniers et al. (2008)

Integer solution results (difficult RC and R instances)

	Desaulniers *et al.*(2008)						Jepsen *et al.*(2008)	
	pespprc $E_{max} = 50$ src			espprc src			espprc src	
Instance	cpu (s)	nb cuts	nb BB	cpu (s)	nb cuts	nb BB	cpu (s)	nb BB
RC201	143	53	3	92	55	3	229	3
RC202	246	34	1	89	39	1	313	1
RC203	842	62	1	324	47	1	14917	1
RC204	-	-	-	-	-	-	-	-
RC205	183	33	1	111	32	1	221	1
RC206	541	68	1	344	73	1	340	1
RC207	**464209**	**290**	**5**	**91405**	**210**	**5**	-	-
RC208	-	-	-	-	-	-	-	-
R201	94	32	1	78	52	1	139	1
R202	1365	156	19	1663	152	17	8282	13
R203	1102	72	1	641	78	1	54187	1
R204	-	-	-	-	-	-	-	-
R205	**21708**	**530**	**17**	**6904**	**345**	**9**	-	-
R206	**26940**	**245**	**1**	**60608**	**171**	**1**	-	-
R207	**4214**	**81**	**1**	**11228**	**24**	**1**	-	-
R208	-	-	-	-	-	-	-	-
R209	10270	307	3	22514	248	3	78560	3
R210	-	-	-	**400904**	**266**	**5**	-	-
R211	-	-	-	-	-	-	-	-

From Desaulniers et al. (2008)

Method of Baldacci et al. (2010)

1. Compute a tight upper bound UB using a good heuristic
2. Compute a dual solution (π, σ) to the linear relaxation of (20)–(22) using:
   - Column generation where the MP is solved by Lagrangean relaxation (three dual ascent heuristics, one based on ng-paths)
   - Subset row inequalities to tighten this relaxation
3. Using dynamic programming, enumerate all path variables θ_p whose reduced cost $\bar{c}_p$ w.r.t. (π, σ) does not exceed $UB - LB$, where LB is the cost of solution (π, σ)
4. Using a commercial MIP solver, solve path-flow formulation (20)–(22) restricted to this subset of variables

Details in Roberto's talk on Friday

Integer solution results

class	nb instances	nb solved BMR	nb solved DLH	nb solved JPPS	avg cputime (s) BMR	avg cputime (s) DLH	avg cputime (s) JPPS
C1	9	9	9	9	15	18	468
RC1	8	8	8	8	230	2150	11004
R1	12	12	12	12	162	2327	27412
ALL1	29	29	29	29	135	1562	14524
C2	8	8	8	7	71	2093	2795
RC2	8	8	6	5	1035	15394	3204
R2	11	10	8	4	44191	63068	35292
ALL2	27	26	22	16	17337	27893	11047
Solved by DHL					3064	27893	-
Solved by JPPS					1025	1637	11047

BMR: Baldacci et al. (2010); DLH: Desaulniers et al. (2008);
JPPS: Jepsen et al. (2008)

R208 closed by Røpke (CG workshop, 2012) using SPPRC, no subset row inequalities and a clever strong branching

From Baldacci et al. (2010)

References for the VRPTW

First branch-and-price algorithm with SPPRC
Desrochers, M., J. Desrosiers, and M.M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

2-path inequalities
Kohl, N., J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-Path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.

First branch-and-price algorithm with ESPPRC
Feillet, D., P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

Subset row inequalities
Jepsen, M., B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.

Tabu search column generator
Desaulniers, G., F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *Transportation Science*, 42(3):387–404, 2008.

Set partitioning with variable elimination
Baldacci, R., E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7(3):229–268, 2010.

Strong degree constraints
Contardo, C. and G. Desaulniers. *Reaching the elementary lower bound in the vehicle routing problem with time windows*. Les Cahiers du GERAD G-2013-50, HEC Montréal, 2013.

Part II

Split delivery VRPTW

Outline of Part II: SDVRPTW

Outline

Split delivery VRPTW (SDVRPTW)

Definition

Same as the VRPTW except

- several vehicles can visit each customer
- demand can be split (split delivery)
- demand can be greater than vehicle capacity

Savings

Splitting deliveries can reduce costs by up to 50%, especially when demands are a little over half the vehicle capacity and demand variance is small (on average, very few customers per route)

Properties

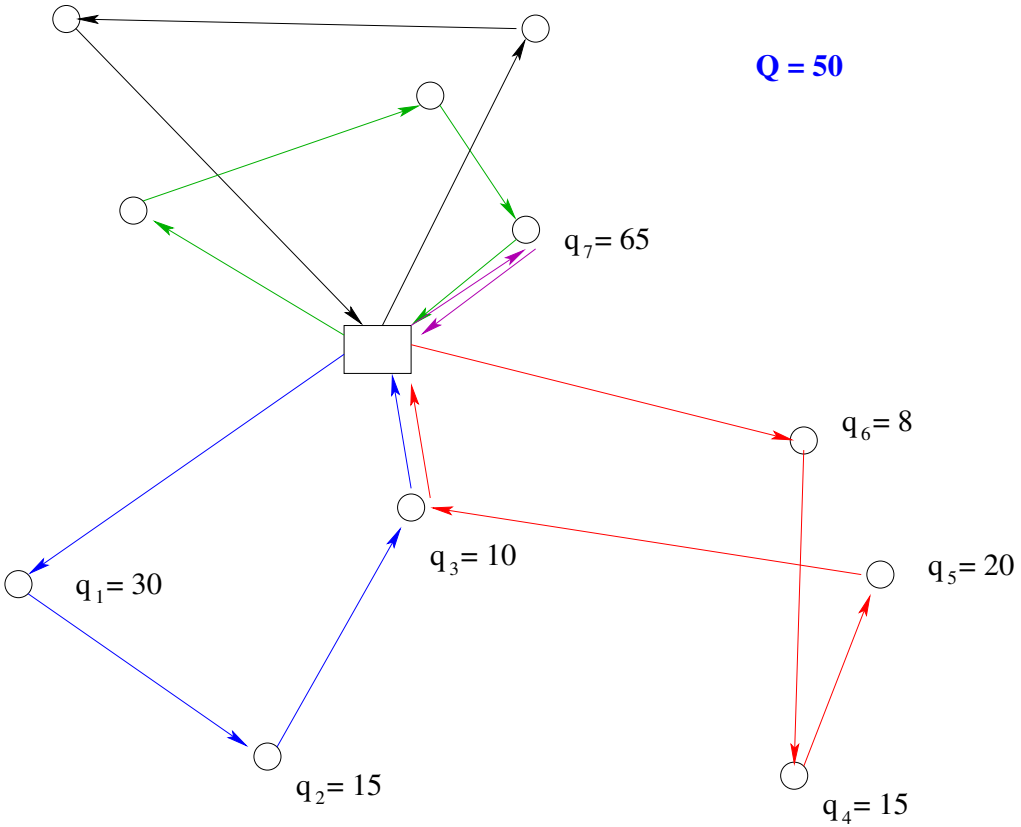
Assumption

Costs and travel times fulfill the triangle inequality

There exists an optimal solution such that

1. No two routes have more than one split customer in common
2. Every arc (i, j) linking two customers i and j is used at most once
3. At most one of every pair of arcs (i, j) and (j, i) (if both exist) is used, where i and j are customers
4. Each route visits no customer more than once
   - Consider elementary routes only

Example of a solution



Remark

Dedicated trips cannot be identified a priori

Outline

Notation

Notation used for the arc-flow formulation of the VRPTW plus the following

- $m_i = \lceil q_i/Q \rceil$: minimum number of vehicles to service customer i
- $\bar{q}_i = \min\{q_i, Q\}$: maximum quantity delivered by one vehicle at customer i
- δ_i^k: nonnegative variable specifying the quantity delivered by vehicle k to customer i

Formulation (1)

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} X_{ij}^k \qquad (32) \quad \text{total cost}$$

$$\text{s.t.:} \quad \sum_{k \in K} \delta_i^k \geq q_i, \qquad \forall\, i \in N \qquad (33) \quad \text{customer demand}$$

$$\sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k \geq m_i, \qquad \forall\, i \in N \qquad (34) \quad \text{min number of visits}$$

$$\sum_{(o,j) \in A} X_{oj}^k = \sum_{(i,d) \in A} X_{id}^k = 1, \qquad \forall\, k \in K \qquad (35) \quad \text{from and to depot}$$

$$\sum_{(i,j) \in A} X_{ij}^k - \sum_{(i,j) \in A} X_{ji}^k = 0, \qquad \forall\, k \in K, i \in N \qquad (36) \quad \text{flow conservation}$$

Formulation (2)

$$\sum_{i \in N} \delta_i^k \leq Q, \qquad \forall\, k \in K \qquad (37)$$ vehicle capacity

$$0 \leq \delta_i^k \leq \bar{q}_i \sum_{(i,j) \in A} X_{ij}^k, \qquad \forall\, k \in K, i \in N \qquad (38)$$ quantity bounds

$$X_{ij}^k (T_i^k + t_{ij} - T_j^k) \leq 0, \qquad \forall\, k \in K, (i,j) \in A \quad (39)$$ time computation

$$a_i \leq T_i^k \leq b_i, \qquad \forall\, k \in K, i \in \bar{N} \qquad (40)$$ time windows

$$X_{ij}^k \in \{0,1\}, \qquad \forall\, k \in K, (i,j) \in A \quad (41)$$ binary requirements

Remark

When $m_i = 1$ ($q_i \leq Q$), constraint (34) is redundant with constraints (33) and (38) and can be omitted

Outline

Dantzig-Wolfe decomposition

Block diagonal structure

- Constraints (35)–(41) are separable by vehicle
- They define the feasible region of an elementary shortest path problem with time windows and vehicle capacity, combined with the linear relaxation of a knapsack problem
- No extreme rays
- Every extreme point corresponds to a combination of a feasible o-d path in G and an extreme delivery pattern

Extreme delivery pattern

- For a vehicle k, fixing the X_{ij}^k variables in (35)–(41) leaves the following constraints

$$\sum_{i \in N} \delta_i^k \leq Q \tag{42}$$

$$0 \leq \delta_i^k \leq \bar{q}_i \sum_{(i,j) \in A} X_{ij}^k, \qquad \forall\, i \in N \tag{43}$$

- They define the feasible region of the linear relaxation of a knapsack problem
- Its extreme points are called extreme delivery patterns
- In such an extreme delivery pattern
  - $\delta_i^k = 0$ for all non-visited customers
  - $\delta_i^k = \bar{q}_i$ for some (possibly no) visited customers
  - $0 < \delta_i^k < \bar{q}_i$ for at most one visited customer
  - $\delta_i^k = 0$ for some (possibly no) visited customers

Notation

Ω^k: set of feasible paths in G for vehicle k

c_p^k: cost of feasible path $p \in \Omega^k$

v_{pi}^k: binary parameter equal to 1 if path $p \in \Omega^k$ visits customer i

v_{pij}^k: binary parameter equal to 1 if path $p \in \Omega^k$ uses arc (i, j)

W_p^k: set of extreme delivery patterns for path $p \in \Omega^k$

δ_{wi}^k: quantity delivered to customer i in delivery pattern $w \in W_p^k$

θ_{pw}^k: nonnegative variable indicating the proportion of path $p \in \Omega^k$ and delivery pattern $w \in W_p^k$ used in the solution

Disaggregated path-flow formulation (1)

$$\min \sum_{k \in K} \sum_{p \in \Omega^k} \sum_{w \in W_p^k} c_p^k \theta_{pw}^k \qquad (44) \quad \text{total cost}$$

$$\text{s.t.:} \sum_{k \in K} \sum_{p \in \Omega^k} \sum_{w \in W_p^k} \delta_{wi}^k \theta_{pw}^k \geq q_i, \quad \forall\, i \in N \qquad (45) \quad \text{customer demand}$$

$$\sum_{k \in K} \sum_{p \in \Omega^k} \sum_{w \in W_p^k} v_{pi}^k \theta_{pw}^k \geq m_i, \quad \forall\, i \in N \qquad (46) \quad \text{min number of visits}$$

$$\sum_{p \in \Omega^k} \sum_{w \in W_p^k} \theta_{pw}^k = 1, \quad \forall\, k \in K \qquad (47) \quad \text{convexity}$$

Disaggregated path-flow formulation (2)

$$\theta^k_{pw} \geq 0, \qquad \forall\, k \in K, p \in \Omega^k, w \in W^k_p \quad (48) \qquad \text{nonnegativity}$$

$$X^k_{ij} = \sum_{p \in \Omega^k} \sum_{w \in W^k_p} v^k_{pij} \theta^k_{pw}, \qquad \forall\, k \in K, (i,j) \in A \quad (49) \qquad X\text{-}\theta \text{ link}$$

$$X^k_{ij} \in \{0,1\}, \qquad \forall\, k \in K, (i,j) \in A \quad (50) \qquad \text{binary requirements}$$

Identical vehicles/subproblems

Allow to aggregate the variables as

$$\theta_{pw} = \sum_{k \in K} \theta_{pw}^{k}$$

in objective (44) and constraints (45)–(46) and to drop index k

Integrality on paths

Constraints (47)–(50) only stipulate that a single path must be assigned to each vehicle, allowing the introduction of the variables

$$\theta_{p} = \sum_{w \in W_p} \theta_{pw}$$

and integrality requirements on these variables

Aggregated path-flow formulation

$$\min \sum_{p \in \Omega} \sum_{w \in W_p} c_p \theta_{pw} \qquad (51) \qquad \text{total cost}$$

$$\text{s.t.:} \sum_{p \in \Omega} \sum_{w \in W_p} \delta_{wi} \theta_{pw} \geq q_i, \quad \forall\, i \in N \qquad (52) \qquad \text{customer demand}$$

$$\sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw} \geq m_i, \quad \forall\, i \in N \qquad (53) \qquad \text{min number of visits}$$

$$\theta_{pw} \geq 0, \quad \forall\, p \in \Omega, w \in W_p \qquad (54) \qquad \text{nonnegativity}$$

$$\theta_p = \sum_{w \in W_p} \theta_{pw}, \qquad \forall\, p \in \Omega \qquad (55) \quad \text{pattern combination}$$

$$\theta_p \in \{0,1\}, \qquad \forall\, p \in \Omega^{mult} \qquad (56) \quad \text{binary requirements}$$

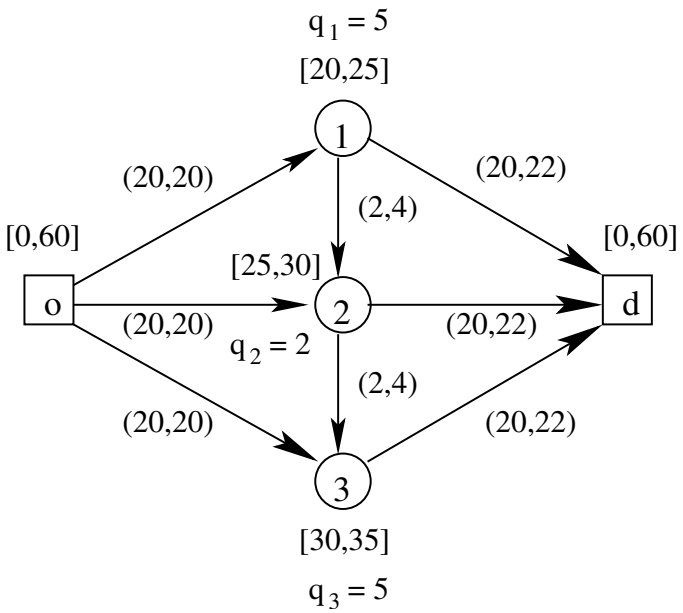
$$\theta_p \text{ integer}, \qquad \forall\, p \in \Omega^{sing} \qquad (57) \quad \text{integrality requirements}$$

- Ω^{mult}: set of paths visiting multiple customers
- Ω^{sing}: set of paths visiting a single customer
- Aggregated convexity constraint removed because $|K|$ is sufficiently large

Remarks

- Binary requirements (56) ensue from Property 2
- Convex combinations of delivery patterns in (55) can yield paths including multiple split deliveries
- Very large number of constraints (55), but they are not required in the linear relaxation

A small example with 3 customers and $Q = 4$



Optimal solution is (delivered quantities in parenthesis)

- o-1(4)-d with a flow of 1, o-3(4)-d with a flow of 1
- o-1(2)-2(2)-3(0)-d with a flow of 0.5
- o-1(0)-2(2)-3(2)-d with a flow of 0.5

Outline

Master problem (MP)

- The integer master problem corresponds to the path-flow formulation (51)–(57)
- The master problem is the linear relaxation of this formulation and is solved by column generation
  - Corresponds to (51)–(54)
  - Constraints (55) are not required

Dual variables and reduced cost

Dual variables

π_i, $i \in N$: dual variables associated with constraints (52)

α_i, $i \in N$: dual variables associated with constraints (53)

$\alpha_o = 0$: defined for notational conciseness

Reduced cost $\bar{c}_{pw}$ of variable θ_{pw}:

$$\bar{c}_{pw} = c_p - \sum_{i \in N} \delta_{wi} \pi_i - \sum_{i \in N} v_{pi} \alpha_i = \sum_{(i,j) \in p} (c_{ij} - \alpha_i) - \sum_{i \in N} \delta_{wi} \pi_i$$

$$= \sum_{(i,j) \in p} \bar{c}_{ij} - \sum_{i \in N} \delta_{wi} \pi_i$$

$\bar{c}_{ij} = c_{ij} - \alpha_i$: arc "reduced" cost

Subproblem formulation

$$\min \quad \sum_{(i,j)\in A} \bar{c}_{ij} X_{ij} - \sum_{i\in N} \delta_i \pi_i \tag{58}$$

$$\text{s.t.} \quad \sum_{(o,j)\in A} X_{oj} = \sum_{(i,d)\in A} X_{id} = 1, \tag{59}$$

$$\sum_{(i,j)\in A} X_{ij} - \sum_{(i,j)\in A} X_{ji} = 0, \qquad \forall\, i \in N \tag{60}$$

$$\sum_{i\in N} \delta_i \leq Q, \tag{61}$$

$$0 \leq \delta_i \leq \bar{q}_i \sum_{(i,j)\in A} X_{ij}, \qquad \forall\, i \in N \tag{62}$$

$$X_{ij}(T_i + t_{ij} - T_j) \leq 0, \qquad \forall\, (i,j) \in A \tag{63}$$

$$a_i \leq T_i \leq b_i, \qquad \forall\, i \in \bar{N} \tag{64}$$

$$X_{ij} \in \{0,1\}, \qquad \forall\, (i,j) \in A \tag{65}$$

Remarks

- Index k was dropped
- Constraints (59)–(65) correspond to constraints (35)–(41)
- Objective (58) seeks to minimize path reduced cost
- Constraints (63)–(64) are resource constraints
  - Constraints (61) can be rewritten as

$$X_{ij}(L_i + \delta_i - L_j) \leq 0, \qquad \forall\, (i,j) \in A \tag{66}$$
$$0 \leq L_i \leq Q, \qquad \forall\, i \in \bar{N} \tag{67}$$

    where δ_i are variables
- For fixed X_{ij} values, constraints (61)–(62) are constraints defining a knapsack problem
- The subproblem is an ESPPRC combined with the linear relaxation of a knapsack problem

Label-setting algorithm

Algorithm for the VRPTW not applicable because

- Delivered quantities are decision variables
- Reduced cost and load resource are functions of these quantities

New algorithm for the SDVRPTW

- When extending a label along an arc, up to three labels can be created: zero, split, and full deliveries
- New binary resource to limit the number of split deliveries
- Reduced cost is a linear function of the total quantity delivered
- Dominance procedure must compare such functions

Label E

- Z: reduced cost excluding cost of split delivery (if any)
- L: total quantity delivered in full deliveries
- T: start of service time
- T^{cust_k}: customer k reachable or not ($\forall\, k \in N$)
- S: split delivery done or not
- Δ: maximum quantity that can be delivered in the split delivery (if any)
- Π: unit dual price for the split delivery (if any)

No resource windows for Δ and Π

Resource extension functions

For a zero delivery: Along arc (i,j) from a label E_i

- Created label E_j is given by

$$
\begin{aligned}
Z_j &= Z_i + \bar{c}_{ij} \\
L_j &= L_i \\
T_j &= \max\{T_i + t_{ij}, a_j\} \\
T_j^{cust_k} &= \begin{cases} T_i^{cust_k} + 1 & \text{if } j = k \\ \max\{T_i^{cust_k}, U_k(E_i)\} & \text{otherwise} \end{cases} \qquad \forall\, k \in N \\
S_j &= S_i \\
\Delta_j &= \Delta_i \\
\Pi_j &= \Pi_i
\end{aligned}
$$

- Feasible extension if $T_j \leq b_j$ and $T^{cust_j} \leq 1$

For a full delivery: Along arc (i,j) from a label E_i $(j \neq d)$

- Created label E_j is given by

$$
\begin{aligned}
Z_j &= Z_i + \bar{c}_{ij} - \pi_j \, \bar{q}_j \\
L_j &= L_i + \bar{q}_j \\
T_j &= \max\{T_i + t_{ij}, a_j\} \\
T_j^{cust_k} &= \begin{cases} T_i^{cust_k} + 1 & \text{if } j = k \\ \max\{T_i^{cust_k}, U_k(E_i)\} & \text{otherwise} \end{cases} \qquad \forall\, k \in N \\
S_j &= S_i \\
\Delta_j &= \min\{\Delta_i, Q - L_i - \bar{q}_j\} \\
\Pi_j &= \Pi_i
\end{aligned}
$$

- Feasible extension if $T_j \leq b_j$, $L_j \leq Q$ and $T^{cust_j} \leq 1$

For a split delivery: Along an arc (i,j) from label E_i $(j \neq d)$

- Created label E_j is given by

$$
\begin{aligned}
Z_j &= Z_i + \bar{c}_{ij} \\
L_j &= L_i \\
T_j &= \max\{T_i + t_{ij}, a_j\} \\
T_j^{cust_k} &= \begin{cases} T_i^{cust_k} + 1 & \text{if } j = k \\ \max\{T_i^{cust_k}, U_k(E_i)\} & \text{otherwise} \end{cases} \qquad \forall\, k \in N \\
S_j &= S_i + 1 \\
\Delta_j &= \min\{\bar{q}_j, Q - L_i\} \\
\Pi_j &= \pi_j
\end{aligned}
$$

- Feasible extension if $T_j \leq b_j$, $\Delta_j > 0$, $S_j \leq 1$ and $T^{cust_j} \leq 1$

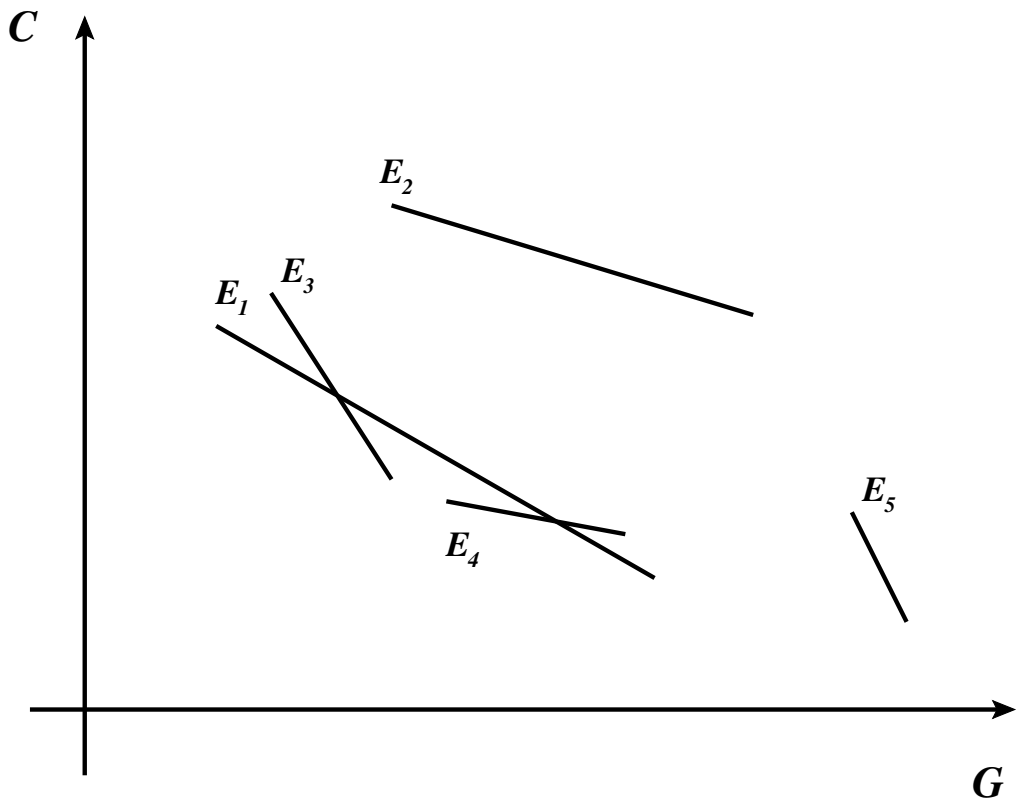
Dominance rule

E_1 dominates E_2 if

- E_1 and E_2 are associated with the same node
- All feasible extensions of E_2 are also feasible for E_1
- Cost of every feasible extension of E_2 is greater than or equal to the cost of a similar extension of E_1

Cost of a label $E = (Z, L, T, T^{cust_k}, S, \Delta, \Pi)$ is

$$C(G) = Z - \Pi(G - L) \quad \text{for } G \in [L, L + \Delta]$$



E_1 can dominate E_2, but not the other labels

E_1 can dominate E_2 if

- $L_1 \leq L_2$
- $T_1 \leq T_2$
- $T_1^{cust_k} \leq T_2^{cust_k}, \forall\, k \in N$
- $S_1 \leq S_2$
- $Z_1 - \Delta_1 \Pi_1 \leq Z_2 - \Delta_2 \Pi_2$
- $Z_1 - (L_2 - L_1)\Pi_1 \leq Z_2$
- $Z_1 - (L_2 + \Delta_2 - L_1)\Pi_1 \leq Z_2 - \Delta_2 \Pi_2$

Remarks

- If $\pi_j = 0$, apply only zero deliveries at j
- Bidirectional search and decremental state-space relaxation can be applied
- Heuristic labeling algorithm can be devised
- Tabu search column generator can be devised (see next slide)

Tabu search column generator

Similar to the one for the VRPTW

For each path/delivery pattern in the current MP basis (zero reduced cost)

- Reoptimize delivery pattern (linear relaxation of knapsack problem)
- Execute a tabu search algorithm
  - Insert or remove a node from the current path
  - For each neighbor, update delivery pattern
- Perform a very limited number of iterations
- Use a tabu list

Column generator usage

As for the VRPTW

- At each column generation
  - Tabu search column generator
  - If no columns found, heuristic labeling algorithm
  - If no columns found, exact labeling algorithm

Instances

Modified Solomon's VRPTW instances

- Allow split deliveries
- Solomon: 56 instances with 100 customers (6 classes)
- 2×56 other instances taking the first 25 and 50 customers
- $Q = 30, 50, 100$
- Total of 504 instances

Linear relaxation results (100-series)

			$Q = 30$				$Q = 50$				$Q = 100$			
			w/o tabu		with tabu		w/o tabu		with tabu		w/o tabu		with tabu	
n	class	Nb inst	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)
25	R1	12	12	<1	12	<1	12	1	12	<1	12	1	12	<1
	C1	9	9	<1	9	<1	9	1	9	<1	9	2	9	2
	RC1	8	8	<1	8	<1	8	<1	8	<1	8	1	8	<1
50	R1	12	12	2	12	1	12	7	12	4	12	15	12	8
	C1	9	9	2	9	1	9	5	9	3	9	21	9	9
	RC1	8	8	1	8	<1	8	2	8	2	8	10	8	6
100	R1	12	12	31	12	20	12	118	12	78	12	597	12	384
	C1	9	9	9	9	7	9	33	9	25	9	130	9	70
	RC1	8	8	11	8	10	8	48	8	39	8	325	8	215

Maximum CPU time = 1 hour, 2.8GHz PC

From Archetti et al. (2011)

Linear relaxation results (200-series)

			$Q = 30$				$Q = 50$				$Q = 100$			
			w/o tabu		with tabu		w/o tabu		with tabu		w/o tabu		with tabu	
n	class	Nb inst	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)	Nb opt.	time (s)
25	R2	11	11	2	11	<1	11	9	11	1	11	75	11	2
	C2	8	8	2	8	<1	8	2	8	1	8	9	8	4
	RC2	8	8	1	8	<1	8	1	8	<1	8	14	8	1
50	R2	11	11	51	11	4	11	107	11	15	11	896	11	64
	C2	8	8	7	8	2	8	17	8	9	8	161	8	46
	RC2	8	8	9	8	1	8	8	8	5	8	245	8	24
100	R2	11	8	1044	11	54	6	1045	11	245	1	189	7	869
	C2	8	8	44	8	17	8	127	8	91	8	812	8	382
	RC2	8	8	264	8	36	8	418	8	156	3	1037	8	1130

Maximum CPU time = 1 hour, 2.8GHz PC

From Archetti et al. (2011)

Branching

Different rules

- Number of vehicles used (as for the VRPTW)
- Number of vehicles visiting a customer
- Aggregated arc-flow variables (as for the VRPTW)
- Aggregated flow on two consecutive arcs

Number of vehicles visiting a customer

- In the arc-flow model, the number of vehicles visiting customer i in a solution is given by $V_i = \sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k$
- Applying the DW decomposition principle and aggregating the vehicles, we get

$$V_i = \sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k = \sum_{k \in K} \sum_{p \in \Omega^k} \sum_{w \in W_p} v_{pi}^k \theta_{pw}^k = \sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw}$$

- If V_i takes a fractional value $\tilde{V}_i$, then one can impose the rules

$$\sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw} \leq \lfloor \tilde{V}_i \rfloor \quad \text{and} \quad \sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw} \geq \lceil \tilde{V}_i \rceil$$

Treatment

- These decisions are global and imposed in the MP by adding the corresponding constraints
- Let ν_i be the dual variable associated with such a decision
- Because $\sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw} = \sum_{k \in K} \sum_{(i,j) \in A} X_{ij}^k$, we see that ν_i must be subtracted from the (reduced) arc cost of every arc $(i,j) \in A$
- On the other hand, these decisions can be combined with constraints (53)

Remarks

- Not necessarily sufficient to yield an integer solution

Aggregated flow on two consecutive arcs

- In the arc-flow model, the aggregated flow on two consecutive arcs (i,j) and (j,l) with $i,j,l \in N$ (variable ζ_{ijl}) in a solution is determined through the constraints

$$X_{ij} + X_{jl} \leq \zeta_{ijl} + 1 \leq \frac{X_{ij} + X_{jl}}{2} + 1 \tag{68}$$

$$\zeta_{ijl} \in \{0,1\} \tag{69}$$

  assuming $X_{ij} = \sum_{k \in K} X_{ij}^k$ and $X_{jl} = \sum_{k \in K} X_{jl}^k \in \{0,1\}$ from Property 2

- If ζ_{ijl} takes a fractional value, then one can impose the rules

$$\zeta_{ijl} = 0 \qquad \text{and} \qquad \zeta_{ijl} = 1$$

Remark

- Even if all aggregated arc-flow variables X_{ij} are integer, such a variable can take a fractional value
- Example with four paths (and different delivery patterns)
  - $p_1 = o - 1 - 3 - 4 - d$
  - $p_2 = o - 2 - 3 - 4 - d$
  - $p_3 = o - 1 - 3 - 5 - d$
  - $p_4 = o - 2 - 3 - 5 - d$
  - If path flow $= 0.5$ for each path, aggregated arc-flow variables are all integer

Treatment

- These decisions are local and imposed in the subproblem by adding one or two resources for each decision (see inter-task constraints in Part III)
- The columns in the current restricted MP that do not respect the imposed decision must be removed from it

Remarks

- Sufficient to yield an integer solution when combined with decisions on aggregated arc-flow variables

Cutting planes

Several types of cutting planes have been proposed

- Arc-flow cuts
- Weak and strong k-path cuts
- Subset row cuts
- Strong minimum number of vehicles cuts

Arc-flow inequalities

- Derived from Property 2 (not valid inequalities but preserve at least one optimal solution)
- Maximum flow of 1 on arcs (i, j) and (j, i) for $i, j \in N$
- If $(i, j) \in A$ and $(j, i) \in A$, the corresponding arc-flow inequality is

$$\sum_{k \in K} (X_{ij}^k + X_{ji}^k) \leq 1$$

- If $(i, j) \in A$ but $(j, i) \notin A$, the corresponding arc-flow inequality is

$$\sum_{k \in K} X_{ij}^k \leq 1$$

Separation

Complete enumeration

Treatment

- Applying DW decomposition yields

$$\sum_{p\in\Omega}\sum_{w\in W_p}(v_{pij}+v_{pji})\theta_{pw}\leq 1 \quad \text{or} \quad \sum_{p\in\Omega}\sum_{w\in W_p}v_{pij}\theta_{pw}\leq 1$$

- This cut is added to the MP
- In the subproblem, its dual variable is subtracted from the arc reduced cost of arcs (i,j) and (j,i)

Remark

Not very efficient, but not costly at all

Weak k-path inequalities

Same as for the VRPTW

- $S \subseteq N$: subset of customers
- $\delta^-(S) \subset A$: subset of arcs (i,j) entering S ($i \neq S$ and $j \in S$)
- $k(S) = \max\{lb_2, lb_4\}$: lower bound on the number of vehicles required to service the customers in S
  - $lb_2 = \left\lceil \sum_{i \in S} q_i / Q \right\rceil$
  - $lb_4 = 2$ if the traveling salesman problem with time windows (TSPTW) for S is infeasible and 1 otherwise
- For set S, the k-path inequality is

$$\sum_{k \in K} \sum_{(i,j) \in \delta^-(S)} X_{ij}^k \geq k(S) \quad \Rightarrow \quad \sum_{p \in \Omega} \sum_{w \in W^p} \sum_{(i,j) \in \delta^-(S)} v_{pij} \theta_{pw} \geq k(S)$$

Separation

- Limited enumeration based on the support graph of the fractional solution
- Three heuristic algorithms (based only on vehicle capacity)
  - Extended shrinking heuristic
    - Shrink into a single pseudo-node two (pseudo-)nodes linked by the (pseudo-)arc with the highest flow (pseudo-nodes define subsets S)
  - Connected component heuristic
    - Start with S containing the nodes of a connected component of the support graph and remove one node at a time
  - Modified connected component heuristic
    - At each iteration (recursive algorithm), remove all customers from a path
    - Paths are sorted in decreasing order of $\theta_{pw} - \sum_{i \in S \cap p} \frac{q_i}{Q}$

Strong k-path inequalities

- Weak cuts

$$\sum_{p\in\Omega}\sum_{w\in W_p}\sum_{(i,j)\in\delta^-(S)} v_{pij}\theta_{pw} \geq k(S), \qquad \forall\, S \subseteq N$$

  - $\sum_{(i,j)\in\delta^-(S)} v_{pij} = 1$ if arc (i,j) enters subset S
- Strong cuts

$$\sum_{p\in\Omega}\sum_{w\in W_p} v_{pS}\theta_{pw} \geq k(S), \qquad \forall\, S \subseteq N$$

  - $v_{pS} = 1$ if path p enters subset S (once or more)

Cannot be treated directly by the labeling algorithm

- Add one binary resource for each subset S
  - $R^{SP,S} = 1$ if partial path enters subset S
- Subtract the dual variable $\sigma^{SP,S}$ when first entering S
- Dominance rule
  - Add $R_1^{SP,S} \leq R_2^{SP,S}$ for all S or
  - Restrict the tests on the cost components by replacing Z_1 with

$$Z_1 + \sum_{S\,:\,R_1^{SP,S} > R_2^{SP,S}} \sigma^{SP,S}$$

Subset row inequalities

For the VRPTW

- Applied considering subsets of three rows only
- Correspond in this case to clique cuts defined for three customers
- $$\sum_{p \in \Omega} \sum_{w \in W_p} l_{pS} \theta_{pw} \leq 1, \qquad \forall\, S \subset N, |S| = 3$$
  - $l_{pS} = 1$ if path p visits at least two customers in subset S

For the SDVRPTW

- Customers can be visited more than once
- Each customer receives at most one full delivery
- $$\sum_{p \in \Omega} \sum_{w \in W_p} l_{pwS} \theta_{pw} \leq 1, \qquad \forall S \in N, |S| = 3$$
  - $l_{pwS} = 1$ if path p visits at least two customers in S to perform full deliveries according to pattern w

Cannot be treated directly by the labeling algorithm

- Add one resource for each subset S
  - $R^{SR,S}$ = number of customers visited in subset S
- Subtract the dual variable $\sigma^{SR,S}$ only once when two customers (full deliveries) have been visited
- Dominance rule
  - Restrict the tests on the cost components by replacing Z_1 with

$$Z_1 - \sum_{S\,:\,R_1^{SR,S}=1\ \&\ R_2^{SR,S}=0} \sigma^{SR,S}$$

Strong minimum number of vehicles inequalities

For every customer i such that $q_i \leq Q$

- Implicit weak constraint

$$\sum_{p \in \Omega} \sum_{w \in W_p} v_{pi} \theta_{pw} \geq 1$$

- Strong constraint (added a priori)

$$\sum_{p \in \Omega} \sum_{w \in W_p} (2a^F_{pwi} + a^{SZ}_{pwi}) \theta_{pw} \geq 2$$

  - $a^F_{pwi} = 1$ if full delivery at i in pattern w for path p
  - $a^{SZ}_{pwi} = 1$ if split or zero delivery at i in pattern w for path p

- Treated directly in the labeling algorithm

For every subset S such that $\sum_{i \in S} q_i \leq Q$

- Strong cut (generated as cuts for $|S| \in \{2, 3\}$)

$$\sum_{p \in \Omega} \sum_{w \in W_p} (2a^F_{pwS} + a^{SZ}_{pwS})\theta_{pw} \geq 2$$

- $a^F_{pwS} = 1$ if all customers in S are visited in path p and receive full deliveries in pattern w
- $a^{SZ}_{pwS} = 1$ if (at least one customer in S is visited in p) and (not all customers in S are visited in p or at least one visited customer in S receives a split or zero delivery in pattern w)

Cannot be treated directly by the labeling algorithm

- Add one resource for each subset S
  - $R^{SMV,S}$ = number of customers (full deliveries) visited in subset S
- Subtract the dual variable $\sigma^{SMV,S}$ when first entering in S
- Subtract it again if all customers receive a full delivery
- Dominance rule
  - Restrict the tests on the cost components by replacing Z_1 with

$$Z_1 + \sum_{\substack{S : R_1^{SMV,S} \neq 0 \ \& \ R_2^{SMV,S} \neq |S| \\ \& \ ((R_2^{SMV,S} = 0 \ \& R_1^{SMV,S} < |S|-1) \mid R_1^{SMV,S} < R_2^{SMV,S})}} \sigma^{SMV,S}$$

Lower bounds for various inequalities

inst	n	Q	UB	WP	WP +SP	WP +SMV	WP +SR	WP +NSA	ALL
C105	25	30	821.1	803.2	803.5	803.2	803.4	**821.1**	**821.1**
C206	25	30	909.9	907.3	907.4	907.4	907.3	909.8	**909.9**
R111	25	30	747.5	739.2	739.7	741.5	739.2	739.2	741.5
R202	25	30	744.9	739.1	739.0	741.1	739.1	739.1	741.1
Avg. gap closed (%)					2	17	0	49	66
C103	50	50	1012.3	1008.5	1008.8	1008.7	1009.0	1011.2	**1012.3**
C205	50	50	1157.1	1154.8	1154.9	1155.2	1155.3	1155.5	1156.6
R102	50	50	1114.2	1102.9	1102.9	1104.0	1102.9	1106.1	1108.7
RC102	25	50	933.5	909.3	909.9	909.5	909.3	**933.5**	**933.5**
Avg. gap closed (%)					4	8	9	57	82
C103	25	100	291.5	287.8	288.7	288.8	288.1	291.2	**291.5**
R109	50	100	804.2	789.3	790.9	799.0	792.1	789.3	800.8
R205	50	100	758.9	753.4	755.9	**758.9**	755.1	754.9	**758.9**
R206	25	100	404.1	393.5	394.1	398.5	394.8	399.1	**404.1**
Avg. gap closed (%)					22	59	20	43	94
Averages over all instances									
Gap closed (%)					9	28	10	50	81
Number cuts				33	27+17	33+30	33+27	48	48+26
Total time (s)				12	23	43	25	16	38

UB: optimal value; *WP*: weak path cuts (enumeration separation algorithm only);
SP: strong path cuts; *SMV*: strong minimum number of vehicles cuts;
SR: subset row cuts; *NSA*: new separation algorithms; *ALL*: all cuts.

From Archetti et al. (2011)

Integer solution results (25 customers)

			Nb	numbers			gap (%)		numbers		times (s)		
n	class	Q	inst	solved	veh	splits	lp	lp+cuts	cuts	nodes	lp	cuts	total
25	R1	30	12	12	12.0	3.8	1.9	0.2	50.0	77.9	<1	4	48
25	R1	50	12	12	7.2	1.1	1.2	<0.1	22.7	1.2	<1	<1	5
25	R1	100	12	12	4.9	0.1	0.4	0	10.7	1.0	<1	<1	2
25	C1	30	9	9	16.0	5.2	2.8	0	96.2	6.1	<1	2	9
25	C1	50	9	9	10.0	2.1	1.7	0	27.4	1.0	<1	<1	5
25	C1	100	9	9	5.0	0	1.9	<0.1	26.3	1.2	2	<1	15
25	RC1	30	8	8	17.8	7.1	2.4	<0.1	81.2	4.5	<1	2	5
25	RC1	50	8	8	10.6	1.8	6.9	0	86.8	1.0	<1	<1	5
25	RC1	100	8	8	6.0	0.4	0.8	0	12.6	1.0	<1	0	2
25	R2	30	11	11	12.0	3.9	2.3	0.2	61.5	218.2	<1	5	165
25	R2	50	11	11	7.0	1.1	1.3	<0.1	26.5	1.2	2	2	15
25	R2	100	11	11	3.8	0.1	1.6	0	38.9	1.0	3	3	24
25	C2	30	8	8	16.0	6.4	1.4	0	57.8	2.9	<1	<1	9
25	C2	50	8	8	10.0	2.5	1.3	<0.1	25.0	5.2	1	<1	11
25	C2	100	8	8	5.0	1.0	0.7	<0.1	12.0	1.2	4	<1	19
25	RC2	30	8	8	18.0	6.6	2.4	0	79.5	3.8	<1	2	9
25	RC2	50	8	8	10.8	1.8	6.9	<0.1	91.4	1.2	<1	1	12
25	RC2	100	8	8	6.0	0.4	0.8	0	12.5	1.0	1	0	4

Maximum CPU time = 1 hour, 2.8GHz PC

From Archetti et al. (2011)

Integer solution results (50 and 100 customers)

			Nb	numbers			gap (%)		numbers		times (s)		
n	class	Q	inst	solved	veh	splits	lp	lp+cuts	cuts	nodes	lp	cuts	total
50	R1	50	12	2	15.0	4.0	1.9	0.3	84.0	314.0	1	16	533
50	R1	100	12	6	9.7	0.8	1.1	0.2	129.0	7.0	3	37	553
50	C1	30	9	3	29.0	10.7	1.7	<0.1	214.0	56.7	3	25	219
50	C1	50	9	9	18.0	4.3	1.5	<0.1	113.6	10.8	5	12	114
50	C1	100	9	8	8.8	1.0	4.7	<0.1	173.6	2.2	5	43	353
50	RC1	30	8	8	33.0	8.9	1.6	0	287.9	1.0	<1	9	50
50	RC1	50	8	8	20.0	4.4	0.5	0	29.0	1.0	2	<1	11
50	RC1	100	8	8	10.0	1.0	0.8	0	28.6	1.0	7	<1	22
50	R2	100	11	1	8.0	1.0	0.7	0	54.0	1.0	23	44	134
50	C2	50	8	7	18.0	7.1	1.2	<0.1	120.0	35.6	9	22	395
50	C2	100	8	2	9.0	3.0	1.4	0.2	165.0	7.0	19	66	1314
50	RC2	30	8	8	33.0	9.2	1.6	0	333.5	1.0	2	8	161
50	RC2	50	8	8	20.0	4.8	0.5	0	30.9	1.0	5	<1	30
50	RC2	100	8	8	10.0	0.9	0.9	0	27.8	1.0	29	<1	94
100	R1	100	12	1	20.0	0	0.1	0	16.0	1.0	2	<1	5
100	C1	100	9	5	19.0	2.4	3.6	<0.1	319.8	1.8	31	99	1667
100	C2	100	8	2	19.0	5.5	1.7	0	139.0	1.0	152	14	1407

Maximum CPU time = 1 hour, 2.8GHz PC

From Archetti et al. (2011)

References for the SDVRPTW

Exact branch-and-price algorithm with quantity decisions in the MP

Gendreau, M., P. Dejax, D. Feillet, C. Gueguen. Vehicle routing with time windows and split deliveries, Technical Report 2006-851, Laboratoire Informatique d'Avignon, France, 2006.

Exact branch-and-price algorithms with quantity decisions in the subproblem

Desaulniers, G. Branch-and-price-and-cut for the split delivery vehicle routing problem with time windows, *Operations Research* 58(1):179–192, 2010.

Archetti, C., M. Bouchard, G. Desaulniers. Enhanced branch-and-price-and-cut for vehicle routing with split deliveries and time windows, *Transportation Science* 45(3):285–298, 2011.

Part III

Other SPPRCs

Outline of Part III: Other SPPRCs

Outline

Generic SPPRC

- SPPRC (with or without elementary requirements) appears as a subproblem in various vehicle routing and crew scheduling problems
- Resources can be used to model a wide variety of real-life constraints
- Various objective functions and REFs

Let's see a generic SPPRC model (not necessarily the most complex one)

Notation

- R: set of resources, including an unrestricted one for cost ($r = cost$)
- $[a_i^r, b_i^r]$: resource window at node i for resource r
- T_i^r: variable for resource r at node i
- $\mathbf{T}_i$: resource variable vector at node i
- $f_{ij}^r(\mathbf{T}_i)$: REF for resource r along arc (i, j)
- X_{ij}: arc flow variable along arc (i, j)

Formulation

$$\min \quad T_d^{cost} \tag{70}$$ total cost

s.t.:

$$\sum_{(o,j)\in A} X_{oj} = \sum_{(i,d)\in A} X_{id} = 1, \tag{71}$$ from and to depot

$$\sum_{(i,j)\in A} X_{ij} - \sum_{(i,j)\in A} X_{ji} = 0, \quad \forall\, i \in N \tag{72}$$ flow conservation

$$X_{ij}(f_{ij}^r(\mathbf{T}_i) - T_j^r) \leq 0, \quad \forall\, r \in R, (i,j) \in A \tag{73}$$ resource computation

$$a_i^r \leq T_i^r \leq b_i^r, \quad \forall\, r \in R, i \in \bar{N} \tag{74}$$ resource window

$$X_{ij} \in \{0,1\}, \quad \forall\, (i,j) \in A \tag{75}$$ binary requirements

Labeling algorithm

Remarks

- Assuming that all REFs are non-decreasing (as for the VRPTW), the labeling algorithm described in Part I can be applied for solving this generic SPPRC
  - Under this assumption, the proposed dominance rule is valid

$$T_i^{r,1} \leq T_i^{r,2}, \quad \forall r \in R$$

- When not all REFs are non-decreasing (as for the SDVRPTW), an ad hoc labeling algorithm (with an ad hoc dominance rule) can be devised in certain cases
- Other solution methods such as Lagrangian relaxation, branch-and-cut and constraint programming can be used

Outline

SPPRC with driver constraints

- Vehicle routes must be assigned to drivers
- Drivers are subject to various safety and collective agreement regulations
  - Maximum/minimum driving time per duty and per week
  - Maximum consecutive driving time (without break or rest)
  - Minimum rest time between two duties
  - Minimum break time between two consecutive work stretches in the same duty
- In certain cases, routes must be designed to respect these regulations

$\Rightarrow$ SPPRC with driver constraints

Maximum driving time per duty

Rule

The accumulated driving time in a duty (since last rest) cannot exceed D^{max}, the maximum driving time per duty

Network

- Network similar to the one for the VRPTW
- When time permits, an arc can include a rest
  - In this case, the arc is seen as a sequence of arcs (drive, rest, drive)
  - d_{ij}: driving time on arc (i, j)

Treatment

- Add a resource $r = maxD$ to compute the accumulated driving time since last rest
- Resource variables T_i^{maxD}
- Resource window $[0, D^{max}]$ on all nodes
- Non-decreasing REFs

$$f_{ij}^{maxD}(T_i^{maxD}) = \begin{cases} 0 & \text{if } (i,j) \text{ is a rest arc} \\ T_i^{maxD} + d_{ij} & \text{otherwise} \end{cases}$$

Possible extension of maximum driving time per duty

Rule

The maximum driving time per duty can be extended to $D^{max} + \bar{D}$ (with $\bar{D} < D^{max}$) at most M^{ext} times per week

Treatment

- Same network and resource $r = maxD$
- Add two resources
- Resource $r = extW$: number of extensions used in the week
  - Resource variables T_i^{extW}
  - Resource window $[0, M^{ext}]$ on all nodes
- Resource $r = extD$: number of extensions used in a day
  - Resource variables T_i^{extD}
  - Resource window $[0, 1]$ on all nodes

Treatment (cont'd)

- Non-decreasing REFs

$$f_{ij}^{maxD}(T_i^{maxD}) = \begin{cases} 0 & \text{if } (i,j) \text{ is a rest arc} \\ T_i^{maxD} + d_{ij} & \text{if } (i,j) \text{ is a drive arc and} \\ & \quad T_i^{maxD} + d_{ij} \leq D^{max} \\ T_i^{maxD} + d_{ij} - \bar{D} & \text{otherwise} \end{cases}$$

$$f_{ij}^{extW}(T_i^{maxD}, T_i^{extW}) = \begin{cases} T_i^{extW} & \text{if } (i,j) \text{ is a rest arc} \\ T_i^{extW} & \text{if } (i,j) \text{ is a drive arc and} \\ & \quad T_i^{maxD} + d_{ij} \leq D^{max} \\ T_i^{extW} + 1 & \text{otherwise} \end{cases}$$

$$f_{ij}^{extD}(T_i^{maxD}, T_i^{extD}) = \begin{cases} 0 & \text{if } (i,j) \text{ is a rest arc} \\ T_i^{extD} & \text{if } (i,j) \text{ is a drive arc and} \\ & \quad T_i^{maxD} + d_{ij} \leq D^{max} \\ T_i^{extD} + 1 & \text{otherwise} \end{cases}$$

Minimum driving time per duty

Rule

The accumulated driving time in a duty must be greater than or equal to D^{min}, the minimum driving time per duty

Treatment

- Same network as above (with drive and rest arcs)
- Add a resource $r = minD$ to compute the negative of the accumulated driving time since last rest
  - Resource variables T_i^{minD}
  - Resource windows
    - $[0, 0]$ at the source node o
    - $[-\infty, -D^{min}]$ at the sink node d
    - $[-\infty, -D^{min}]$ on all nodes representing the start of a rest
    - $[-\infty, 0]$ on all other nodes

Treatment (cont'd)

- Non-decreasing REFs

$$f_{ij}^{minD}(T_i^{minD}) = \begin{cases} 0 & \text{if } (i,j) \text{ is a rest arc} \\ \max\{a_j^{minD}, T_i^{minD} - d_{ij}\} & \text{otherwise} \end{cases}$$

Remarks

- More efficient if $-\infty$ is replaced by $-D^{min}$ in the resource windows (allows more dominance)
- Difficult to consider both maximum and minimum rules simultaneously
  - No possible dominance between labels E_1 and E_2 unless $T_1^{minD} = T_2^{minD}$ (if $T_1^{minD} < T_2^{minD}$, then $T_1^{maxD} > T_2^{maxD}$)

Outline

SPPRC with simultaneous pickups and deliveries

- Same as VRPTW except
  - N^P: set of pickup customers
  - N^D: set of delivery customers
  - Delivered merchandise originates from the depot
  - Picked up merchandise returns to the depot
  - Delivered and picked up merchandise can be in a vehicle at the same time (shared capacity)
  - Example: Deliver full bottles, pick up empty ones

$\Rightarrow$ SPPRC with simultaneous pickups and deliveries

Treatment

- Same network as for the VRPTW but with pickup nodes and delivery nodes
- Consider a load resource $r = pick$ but only for pickups
  - Resource window $[0, Q]$ on all nodes
  - Resource variables T_i^{pick}
  - Non-decreasing REFs

$$f_{ij}^{pick}(T_i^{pick}) = \begin{cases} T_i^{pick} + q_j & \text{if } j \in N^P \\ T_i^{pick} & \text{otherwise} \end{cases}$$

Treatment (cont'd)

- Add a maximum load resource $r = maxL$ indicating the maximum load reached along the partial path
  - Resource window $[0, Q]$ on all nodes
  - Resource variables T_i^{maxL}
  - Non-decreasing REFs

$$f_{ij}^{maxL}(T_i^{pick}, T_i^{maxL}) = \begin{cases} T_i^{maxL} & \text{if } j = d \\ T_i^{maxL} + q_j & \text{if } j \in N^D \\ \max\{T_i^{maxL}, T_i^{pick} + q_j\} & \text{otherwise} \end{cases}$$

Outline

SPPRC with pickups and deliveries

- Same as VRPTW except
  - N^P: set of pickup customers
  - N^D: set of delivery customers
  - Pickup and delivery customers are paired, that is, the merchandise picked up at a pickup customer must be delivered to a specific delivery customer
  - Delivered and picked up merchandise can be in a vehicle at the same time (shared capacity)

$\Rightarrow$ SPPRC with pickups and deliveries

Treatment

- Same network as for the VRPTW but with pickup nodes and delivery nodes
- Consider a load resource $r = load$
  - Resource variables T_i^{load}
  - Resource window $[0, Q]$ on all nodes
  - Non-decreasing REFs

$$f_{ij}^{load}(T_i^{load}) = \begin{cases} T_i^{load} + q_j & \text{if } j \in N^P \\ T_i^{load} & \text{if } j = d \\ T_i^{load} - q_j & \text{otherwise} \end{cases}$$

Treatment (cont'd)

- For each pair (u, v) of pickup and delivery customers, add a pair of resources: $r = pd1_{uv}$ and $r = pd2_{uv}$
  - Binary resource $pd1_{uv}$ indicates whether or not pickup u is in the vehicle
  - Resource $pd2_{uv}$ is the negative of resource $pd1_{uv}$
- For resource $pd1_{uv}$
  - Resource variables $T_i^{pd1_{uv}}$
  - Resource windows
    - $[0, 0]$ at source node o and sink node d
    - $[0, 1]$ at all other nodes
  - Non-decreasing REFs

$$f_{ij}^{pd1_{uv}}(T_i^{pd1_{uv}}) = \begin{cases} T_i^{pd1_{uv}} + 1 & \text{if } j = u \\ T_i^{pd1_{uv}} - 1 & \text{if } j = v \\ T_i^{pd1_{uv}} & \text{otherwise} \end{cases}$$

Treatment (cont'd)

- For resource $pd2_{uv}$
  - Resource variables $T_i^{pd2_{uv}}$
  - Resource windows
    - $[0, 0]$ at source node o and sink node d
    - $[-1, 0]$ at all other nodes
  - Non-decreasing REFs

$$f_{ij}^{pd2_{uv}}(T_i^{pd2_{uv}}) = \begin{cases} T_i^{pd2_{uv}} - 1 & \text{if } j = u \\ T_i^{pd2_{uv}} + 1 & \text{if } j = v \\ T_i^{pd2_{uv}} & \text{otherwise} \end{cases}$$

Remarks

- Resource $pd1_{uv}$ ensures
  - Pickup u is not performed more than once before delivery v
  - Delivery v is performed after pickup u
- Resource $pd2_{uv}$ ensures
  - Delivery v is not performed more than once after pickup u
  - Pickup u is performed before delivery v
- A label can dominate another only if both labels have the same pickups on board

Outline

SPPRC with inter-task constraints

Task

- In a vehicle routing or crew scheduling problem, a task must be accomplished exactly once
- Examples:
  - Customers that must be visited once (as in the VRPTW)
  - Bus trips (one bus or one driver)
  - Flights (one aircraft or one crew)
- A task is associated with a set partitioning constraint in MP

Network types

Two network types are often used to model problems with tasks

- Connection networks
- Time-space networks

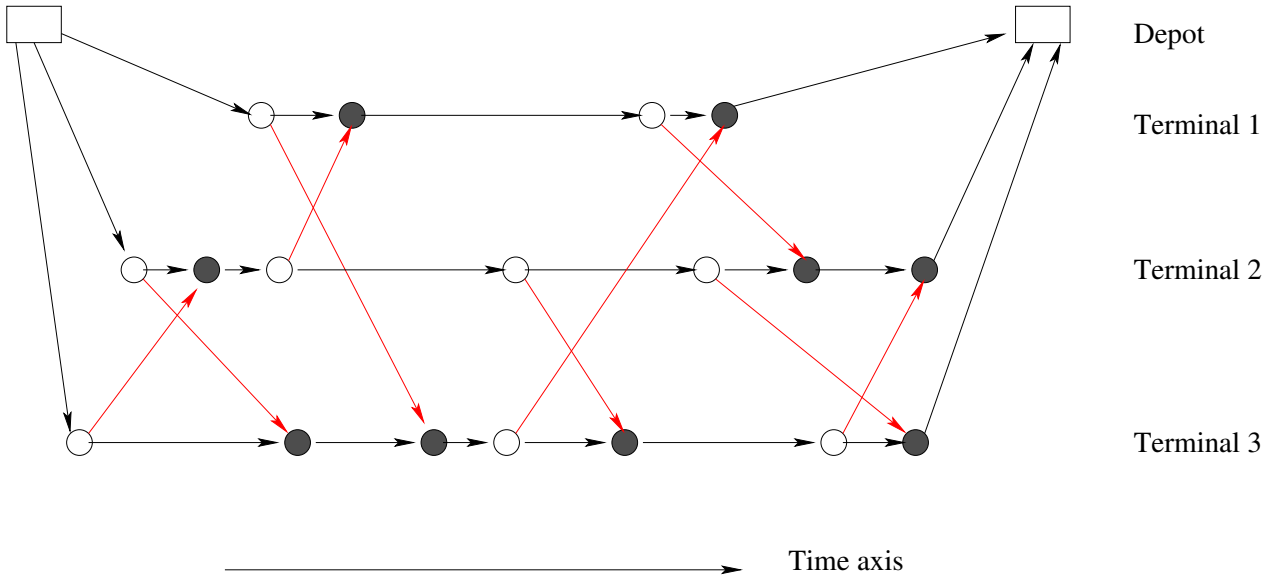
Connection networks (as for the VRPTW)

- Each task is represented by a node
- An arc represents a connection (possibly involving a break, a rest or a repositioning) between two tasks

Time-space network

- Each task is represented by an arc
- A node is associated with a time and a location (e.g., flight arrival node)
- An arc represents a move in time (e.g., wait, rest) or a move in space and time (task or repositioning)

Example of a time-space network



Remarks

- In general, the size of a time-space network is less than the size of a corresponding connection network
- Connection networks are more flexible to model connections with complex rules or costs

SPPRC with inter-task constraints

- Several vehicle routing and crew scheduling problems are modeled using tasks
- W: set of tasks
- In this case, to obtain an integer solution in a branch-and-price algorithm, inter-task branching decisions (or follow-on decisions) can be imposed
  1. Task v must be performed immediately after task u by the same vehicle (imposed inter-task)
  2. Task v must not be performed immediately after task u by the same vehicle (forbidden inter-task)
- In a connection network, these decisions are easily handled by removing arcs
- In a time-space network, these decisions yield an SPPRC with inter-task constraints

Treatment

Imposed inter-task (u, v) in a time-space network

- Add one binary resource ($r = imp1_{uv}$) indicating whether or not task u is the last performed
- Resource variables $T_i^{imp1_{uv}}$
- Resource windows
  - $[0, 0]$ at source node o and sink node d
  - $[0, 1]$ at all other nodes
- Non-decreasing REFs

$$f_{ij}^{imp1_{uv}}(T_i^{imp1_{uv}}) = \begin{cases} T_i^{imp1_{uv}} + 1 & \text{if } (i,j) = u \\ T_i^{imp1_{uv}} - 1 & \text{if } (i,j) = v \\ \infty & \text{if } (i,j) = w \in W \setminus \{v\} \\ & \quad \text{and } T_i^{imp1_{uv}} = 1 \\ T_i^{imp1_{uv}} & \text{otherwise} \end{cases}$$

Imposed inter-task (u, v) (cont'd)

- One resource ($r = imp2_{uv}$) that is the negative of resource $imp1_{uv}$
- Resource variables $T_i^{imp2_{uv}}$
- Resource windows
  - $[0, 0]$ at source node o and sink node d
  - $[-1, 0]$ at all other nodes
- Non-decreasing REFs

$$f_{ij}^{imp2_{uv}}(T_i^{imp2_{uv}}) = \begin{cases} T_i^{imp2_{uv}} - 1 & \text{if } (i,j) = u \\ T_i^{imp2_{uv}} + 1 & \text{if } (i,j) = v \\ \infty & \text{if } (i,j) = w \in W \setminus \{v\} \\ & \quad \text{and } T_i^{imp2_{uv}} = -1 \\ T_i^{imp2_{uv}} & \text{otherwise} \end{cases}$$

Remarks for imposed inter-tasks

- Resource $imp1_{uv}$ ensures
  - Task u is not performed more than once before task v
  - Task v is performed after task u
  - No other task w can be performed immediately after task u
- Resource $imp2_{uv}$ ensures
  - Task u is performed before task v
  - Task v is not performed more than once after task u
- A label can dominate another only if both labels have the same value for $T_i^{imp1_{uv}}$

Forbidden inter-task (u, v) in a time-space network

- Add one binary resource ($r = fbd_{uv}$)
- Resource variables $T_i^{fbd_{uv}}$
- Resource windows
  - $[0, 0]$ at source node o
  - $[0, 1]$ at all other nodes
- Non-decreasing REFs

$$f_{ij}^{fbd_{uv}}(T_i^{fbd_{uv}}) = \begin{cases} 1 & \text{if } (i,j) = u \\ T_i^{fbd_{uv}} - 1 & \text{if } (i,j) = w \in W \setminus \{v\} \\ \infty & \text{if } (i,j) = v \\ & \quad \text{and } T_i^{fbd_{uv}} = 1 \\ T_i^{fbd_{uv}} & \text{otherwise} \end{cases}$$

Remarks for forbidden inter-tasks

- A label with $T_i^{fbd_{uv}} = 0$ can dominate any other label
- A label with $T_i^{fbd_{uv}} = 1$ can dominate another label only if this label also has $T_i^{fbd_{uv}} = 1$

Remarks for both types of inter-tasks

- Memory requirements can be reduced by using a single informative resource for all inter-task constraints (last performed task) together with ad hoc REFs and dominance rules
- Treatment can be generalized to
  - Mixed networks with tasks on nodes and on arcs
  - More than one task per node or per arc
  - Same task on several nodes or arcs

References for the SPPRC

One of the first labeling algorithms for the SPP with time windows
Desrochers, M., F. Soumis. A generalized permanent labelling algorithm for the shortest path problem with time windows, *Information Systems and Operations Research*, 26(3):191–212, 1988.

Surveys on the SPPRC
Desrosiers, J., Y. Dumas, M.M. Solomon, F. Soumis. Time constrained routing and scheduling, in M. Ball, T. Magnanti, C. Monma, G. Nemhauser, eds., *Handbooks in Operations Research and Management Science, Vol. 8, Network Routing*, chap. 2, Elsevier, Amsterdam, Netherlands, 35–139, 1995.

Desaulniers, G., J. Desrosiers, I. Ioachim, M.M. Solomon, F. Soumis D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems, in T. Crainic, G. Laporte, eds., *Fleet Management and Logistics*, chap. 3, Kluwer, Boston, MA, 57–93, 1998.

Irnich, S., G. Desaulniers. Shortest path problems with resource constraints, in G. Desaulniers, J. Desrosiers, M.M. Solomon, eds., *Column Generation*, chap. 2, Springer, New-York, NY, 33–65, 2005.