A branch-and-price algorithm for the multi-depot heterogeneous-fleet vehicle routing problem with time windows

Andrea Bettinelli, Alberto Ceselli, Giovanni Righini

Dipartimento di Tecnologie dell'Informazione - Università degli Studi di Milano

Odysseus 2009, Cesme

May 26, 2009

Problem description

Demand:

- a set $\mathcal{N}$ of customers
- a demand $q_i \ \forall i \in \mathcal{N}$
- a service time $s_i \ \forall i \in \mathcal{N}$
- a time window $[a_i, b_i] \ \forall i \in \mathcal{N}$

Resources:

- a set $\mathcal{H}$ of depots
- a set $\mathcal{K}$ of vehicle types
- a fleet of u_k vehicles for each type $k \in \mathcal{K}$
- a capacity w_k for each vehicle of type $k \in \mathcal{K}$
- a fixed cost f_k for each vehicle of type $k \in \mathcal{K}$
- a transportation network $\mathcal{G} = (\mathcal{N} \cup \mathcal{H}, \mathcal{A})$
- a traveling time $t_{ij} \ \forall (i,j) \in \mathcal{A}$
- a traveling cost $d_{ij} \ \forall (i,j) \in \mathcal{A}$

Constraints and objective

Constraints:

No more than g_h routes can depart from each depot $h \in \mathcal{H}$.

No more than u_k vehicles can be used for each type $k \in \mathcal{K}$.

No route can have a duration longer than a given upper limit D.

The arrival time T_i at each customer i must fall in the range $[a_i, b_i]$.

Split deliveries are not allowed.

Objective: minimize the overall cost (fixed costs + routing costs).

Literature review

Exact

VRPTW:

- Salani (2006)

MDVRP, HVRP, MDHVRP:

- Baldacci et al. (2008)

Heuristics

HVRPTW:

- Liu and Shen (1999)
- Dullaert et al. (2002)
- Belfiore and Fàvero (2007)

MDVRPTW:

- Polacek et al. (2004).

MDHVRPTW:

- Dondo and Cerdá (2007).

No exact algorithm is known for the MDHVRPTW.

Set covering formulation

$$
\begin{aligned}
\text{minimize} \quad & \sum_{k\in\mathcal{K}} \sum_{h\in\mathcal{H}} \sum_{r\in\Omega_{hk}} c_r x_r \\
\text{subject to} \quad & \sum_{k\in\mathcal{K}} \sum_{h\in\mathcal{H}} \sum_{r\in\Omega_{hk}} a_{ir} x_r \geq 1 && \forall i \in \mathcal{N} && (1) \\
& \sum_{h\in\mathcal{H}} \sum_{r\in\Omega_{hk}} x_r \leq u_k && \forall k \in \mathcal{K} && (2) \\
& \sum_{k\in\mathcal{K}} \sum_{r\in\Omega_{hk}} x_r \leq g_h && \forall h \in \mathcal{H} && (3) \\
& x_r \in \{0,1\} && \forall k \in \mathcal{K}\ \forall h \in \mathcal{H}\ \forall r \in \Omega_{hk}.
\end{aligned}
$$

- Ω_{hk} is the set of feasible routes of type $k \in \mathcal{K}$ from depot $h \in \mathcal{H}$;
- c_r is the cost of route r;
- a_{ir} is the number of times route r visits customer $i \in \mathcal{N}$.

Reduced costs

The reduced cost of a route $r \in \Omega_{hk}$ is:

$$\overline{c}_r = c_r - \sum_{i \in \mathcal{N}} a_{ir} \lambda_i - \mu_k - \gamma_h$$

where $\lambda \geq 0$, $\mu \leq 0$ and $\gamma \leq 0$ are dual variables of constraints (1), (2) and (3).

Pricing subproblem:
Elementary Shortest Path Problem with Resource Constraints (capacity and time) on a graph with negative arc costs.

Strongly NP-hard (Dror, 1994).

Pricing algorithms

We use three different pricing algorithms:

- Greedy algorithm (resembling nearest neighbor heuristic)
- Heuristic dynamic programming
- Exact dynamic programming

Each of them is executed only if the previous ones fail.

Pricing must be repeated $\forall k \in \mathcal{K}$ and $\forall h \in \mathcal{H}$.

Exact dynamic programming

The RCESPP is solved with Dynamic Programming (DP):

- bi-directional DP (Righini and Salani, 2006)
- decremental state space relaxation (Righini and Salani, 2008).

States

A label associated with vertex $i \in \mathcal{N}$ is a tuple $(\mathcal{S}, \phi, \tau, z, i)$, where

- $\mathcal{S}$ is the set of vertices visited along the path,
- ϕ is the amount of capacity consumed up to i,
- τ is the time at which the service at vertex i begins,
- z is the cost of the path,
- i is the last reached vertex.

Extensions

Forward extension from i to j:

$$\mathcal{S}' = \mathcal{S} \cup \{j\}$$

$$\phi' = \phi + q_j$$

$$\tau' = \max\{\tau + s_i + t_{ij}, a_j\}$$

$$z' = z - \lambda_i/2 + d_{ij} - \lambda_j/2$$

where $\lambda_{depot} = -f_k - \mu_k - \gamma_h$.

Backward extensions follow symmetrical rules.

Bi-directional DP

Feasibility:
Elementary path constraints: $j \notin \mathcal{S}$
Capacity constraints: $\phi' \leq w_k$
Time windows: $\tau' \leq b_j$ if $j \neq$ depot
Duration constraints: $\tau' \leq D$.

Stopping rule:
Extensions are stopped when half of a critical resource has been consumed.
Choosing time as the critical resource, only states with $\tau \leq D$ are generated.

Bi-directional DP

Forward and backward paths $(\mathcal{S}^{fw}, \phi^{fw}, \tau^{fw}, z^{fw}, i)$ and $(\mathcal{S}^{bw}, \phi^{bw}, \tau^{bw}, z^{bw}, i)$ are joined to produce routes.

Feasibility conditions:

- $\mathcal{S}^{fw} \cap \mathcal{S}^{bw} = \emptyset$
- $\phi^{fw} + \phi^{bw} \leq w_k$
- $\tau^{fw} + s_i + t_{ij} + s_j + \tau^{bw} \leq D$

The (reduced) cost of the resulting route is $z^{fw} - \lambda_i/2 + d_{ij} - \lambda_j/2 + z^{bw}$.

Bounding

In order to associate a lower bound with DP states, we pre-compute LB_{iq}, that is a lower bound to the routing cost of any path

- from i to the depot,
- using no more than q units of capacity,
- not necessarily elementary,
- disregarding time window constraints,
- taking into account the prizes λ.

A forward state $(\mathcal{S}, \phi, \tau, z, i)$ can be fathomed whenever

$$z + LB_{iq} + f_k/2 - \mu_k/2 - \gamma_h/2 \geq 0$$

with $q = w_k - \phi$.

Dominance

$(\mathcal{S}', \phi', \tau', z', i)$ dominates $(\mathcal{S}'', \phi'', \tau'', z'', j)$ only if

$$\begin{aligned}
\mathcal{S}' &\subseteq \mathcal{S}'' \\
\phi' &\leq \phi'' \\
\tau' &\leq \tau'' \\
z' &\leq z'' \\
i &= j
\end{aligned}$$

and at least one of the inequalities is strict.

Heuristic dominance

Condition $\mathcal{S}' \subseteq \mathcal{S}''$ is replaced by $R' \leq R''$, where R is the optimal value of a fractional knapsack problem:

$$\begin{aligned} \max \ & \sum_{i \notin \mathcal{S}} \lambda_i y_i \\ \text{s.t.} \ & \sum_{i \notin \mathcal{S}} q_i y_i \leq w_k - \phi \\ & y_i \in [0, 1] \qquad \forall i \notin \mathcal{S}. \end{aligned}$$

Decremental State Space Relaxation

The elementary path constraint is only checked on a subset $\overline{\mathcal{S}}$ of $\mathcal{S}$.

If the solution turns out to be non-elementary, one or more vertices visited more than once are inserted into $\overline{\mathcal{S}}$ and the DP algorithm is executed again.

Aggregated pricing

We solve the pricing problem only for the vehicle type with larger capacity.
Capacity constraint are checked during the “join” phase.
This weakens the bound used to fathom states,
but it reduces the number of executions of the DP algorithm.

We also consider all depots at one time,
by keeping a different time resource τ_h for each of them.
Dominance occurs only only if $\tau'_h \leq \tau''_h \; \forall h \in \mathcal{H}$.

Hence states can be feasible for some depots and infeasible for others.

2-path inequalities

We search for subsets $\mathcal{P}$ of nodes requiring at least 2 vehicles, but such that they are visited by a smaller number of vehicles in the fractional solution of the Linear Restricted Master Problem.

$$\sum_{k \in \mathcal{K}} \sum_{h \in \mathcal{H}} \sum_{r \in \Omega_{hk}} \alpha_r x_r \geq 2$$

where α_r is the number of arcs (i, j) in route r with $i \in \mathcal{P}$ and $j \notin \mathcal{P}$.

This yields dual variables $\sigma_{\mathcal{P}} \geq 0$ to be subtracted from d_{ij} for all arcs (i, j) such that $i \in \mathcal{P}$ and $j \notin \mathcal{P}$.

The pricing problem does not change.

Branching strategies

- Branching on the number of vehicles:
  - Compute $\sum_{h \in \mathcal{H}} \sum_{r \in \Omega_{hk}} x_r \ \forall k \in \mathcal{K}$.
  - Choose $k \in \mathcal{K}$ for which the fractional part of the above quantity is closest to 1/2.
  - Perform the usual binary branching.

  The pricing problem is not affected.
- Branching on arcs:
  - Select the node i which is split among the largest number of routes.
  - Forbid half of its outgoing arcs in each “child” sub-problem.

  Arc weights are set to ∞.

Data-sets

- 168 HVRPTW instances from Shen and Liu, derived from Solomon's VRPTW data-sets (56 instances) with 3 different fixed costs for each instance class;
- 4 MDVRPTW instances from Cordeau et al. with 4 to 6 depots and 48 to 144 customers.

Data-set Liu and Shen 1

File	CG iterations	Cuts	LB	LB_C	UB	Gap	Time
R1a	153.58	0.33	3997.07	3997.26	4194.02	4.69%	75.78
R1b	263.5	0.25	1817.93	1818.08	1913.68	5.00%	195.54
R1c	273.33	0.25	1507.56	1507.85	1587.04	4.99%	131.98
R1	230.14	0.28	2440.85	2441.06	2564.91	4.83%	134.43
C1a	217.33	0	6748.58	6748.58	7268.95	7.16%	15.31
C1b	310.67	0	2255.65	2255.65	2423.46	6.92%	281.19
C1c	328.22	0	1590.39	1590.39	1656.1	3.97%	565.06
C1	285.41	0	3606.2	3606.2	3782	4.65%	287.19
RC1a	210.13	2.88	4815.69	4819.78	5015.69	3.91%	425.1
RC1b	281.5	5	2056.25	2062.52	2192.92	5.95%	144.42
RC1c	289.75	6.25	1690.33	1697.2	1802.7	5.85%	69.78
RC1	260.46	4.71	2854.09	2859.83	3003.77	4.79%	213.1

Tabella: Lower bounds, data-set 1, aggregated results.

Data-set Liu and Shen 2

File	CG iterations	Cuts	LB	LB_C	UB	Gap	Time
R201b	1806	0	1633.54	1633.54	1776.18	8.03%	1171.26
R201c	948	0	1408.81	1408.81	1550.14	9.12%	377.06
C201a	5832	0	5210.94	5210.94	5741.14	9.24%	2172.41
C201b	1553	0	1590.94	1590.94	1737.93	8.46%	102.58
C205b	2125	0	1588.32	1588.32	1761.36	9.82%	262.1
C206b	2473	0	1587.67	1587.67	1758.98	9.74%	707.15
C201c	890	0	1131.26	1131.26	1221.14	7.36%	43.54
C205c	1751	0	1128.08	1128.08	1225.45	7.95%	151.94
C206c	2112	0	1127.91	1127.91	1230.23	8.32%	392.28
C208c	1985	0	1127.29	1127.29	1237.32	8.89%	901.9
RC201a	298	0	4255.48	4255.48	4458.86	4.56%	1022.6
RC201b	388	0	1904.63	1904.63	1998.57	4.70%	475.44
RC201c	396	0	1596.17	1596.17	1702.05	6.22%	189.69

Tabella: Lower bounds, data-set 2, aggregated results.

Data-set Cordeau

instance	customers	depots	duration	capacity	best known	LB	gap	time
pr01	48	4	500	200	1074.12	1074.12	0.00%	2.21
pr02	96	4	480	195	1762.21	1740.87	1.21%	434.47
pr07	72	6	500	200	1418.22	1414.79	0.24%	23.88
pr08	144	6	475	190	2096.73	-	-	-

Tabella: Lower bounds, Cordeau data-set.

Comments

- Data-set 1: CG could compute a valid lower bound at the root node for all instances but one within 1 hour.
- The average gap is around 5% but upper bounds are not guaranteed to be optimal.
- 2-path inequalities are useless.
- Data-set 2: CG could compute a valid lower bound for 13 instances.
- Data-set Cordeau: the gap is less than 2%.

Branch-and-Price

- Tests were done on reduced instances with 25 and 50 customers.
- Branch-and-Price solved:
  - 79 instances in Data-set 1 (87) and 57 in Data-set 2 (81) with 25 customers.
  - 35 instances in Data-set 1 (87) and 9 in Data-set 2 (81) with 50 customers.
  - 5 instances with 100 customers.
  - 2 instances in Data-set Cordeau (4).

Approximation

Heuristic results

Heuristic solutions can be obtained by solving the RMP with the columns generated.

We used CPLEX 11.0 as an ILP solver with two different time limits: 1 hour and 10 minutes.

Comparing the solutions with those obtained by Liu and Shen and by Belfiore and Favero (one depot only) for some instances this method obtained results up to 28% better, in other cases up to 30% worse.