Solving the Green Vehicle Routing Problem

Juho Andelmin  Enrico Bartolini


• Andelmin, J., Bartolini, E. A Multi-Start Local Search Heuristic for the Green Vehicle Routing Problem Based on a Multigraph Reformulation [Submitted 09/2016 to Computers and Operations Research – Request for revision 03/2017]

1 RWTH Aachen University
School of Business and Economics
Green Vehicle Routing Problem (G-VRP)

- A fleet of vehicles based at a **depot** is to serve a set of customers
  - Customers have known service times
  - Vehicles have limited fuel capacity
  - Vehicles can visit refueling stations to refuel

- **Objective:** Design a set of **vehicle routes** so that
  - Every customer is served
  - Duration of each route $\leq T$
  - Sum of route costs is minimized
Simple example: 9 customers, electric vehicles

- Vehicle speed: 90 km/h
- Service time: 5 min
- Charging delay: 20 min
- Max route duration: 12 h
Optimal solution with driving range $= \infty$

Optimal cost
694.71 km

- Vehicle speed: 90 km/h
- Service time: 5 min
- Charging delay: 20 min
- Max route duration: 12 h
Optimal solution with driving range = 200 km

Optimal cost 823.26 km

- Vehicle speed: 90 km/h
- Service time: 5 min
- Charging delay: 20 min
- Max route duration: 12 h
Optimal solution with driving range = 160 km

- Optimal cost: 1148.08 km

- Vehicle speed: 90 km/h
- Service time: 5 min
- Charging delay: 20 min
- Max route duration: 12 h
Refuel paths

- **Refuel path**: a simple path between two customers that visits a subset of refueling stations

![Refuel path diagram](image)

- Many refuel paths are dominated

- Example:
  - **Green path** $i \to c \to j$ is dominated by
  - **orange one** $i \to b \to j$
We model the G-VRP on a multigraph $\mathcal{G}$ with one arc for each non-dominated refuel path.
Three phases

1) Iteratively construct new solutions
2) Store vehicle routes forming these solutions in a pool $\mathcal{R}$
3) Find a set of routes in $\mathcal{R}$ that gives least cost solution

- Example operators used in phase 1

- Clarke and Wright Merge

- Customer relocate

- Depot
- Customer
Exact algorithm

- Set partitioning formulation (SP)
  - Each possible vehicle route serves a subset of customers
  - Find least cost set of routes serving each customer exactly once

\[
\text{(SP) } \min \sum_{l \in \mathcal{R}} c_l x_l \\
\text{s.t. } \sum_{l \in \mathcal{R}} a_{il} x_l = 1 \quad \forall i \in N \\
x_l \in \{0,1\} \quad \forall l \in \mathcal{R}
\]

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(c_l)</td>
<td>cost of route (l)</td>
</tr>
<tr>
<td>(x_l)</td>
<td>0-1 variable equal to 1 if route (l) is in solution</td>
</tr>
<tr>
<td>(a_{il})</td>
<td>0-1 coefficient equal to 1 if route (l) serves customer (i)</td>
</tr>
<tr>
<td>(\mathcal{R})</td>
<td>index set of all possible vehicle routes</td>
</tr>
<tr>
<td>(N)</td>
<td>set of customers</td>
</tr>
</tbody>
</table>

Phase 1:
- Compute lower bound \(\text{LB}\) by solving Linear Programming relaxation of \(\text{SP}\)
- Compute upper bound \(\text{UB}\) with the MSLS heuristic

Phase 2:
- Enumerate all routes \(\mathcal{R}^*\) having reduced cost \(\leq \text{UB} - \text{LB}\)
- Solve \(\text{SP}\) using only the routes in \(\mathcal{R}^*\) \(\rightarrow\) optimal solution
- If all routes \(\mathcal{R}^*\) cannot be enumerated optimality not guaranteed
Computational results

- **Benchmark problems:**
  - 56 instances with 20-500 customers and 3-28 stations

- **Heuristic:** best new solutions to instances with 111-500 customers
  - Compared to 7 state-of-the-art heuristics

- **Exact algorithm:**
  - Instances up to 111 customers 28 stations solved to optimality
  - Best exact from literature solves up to 20 customer instances

**Instance name example:**
75c_21s: 75 customers 21 stations

\[
\%LB = \left( \frac{UB - LB}{UB} \right) \times 100\%
\]

<table>
<thead>
<tr>
<th>Inst.</th>
<th>Opt</th>
<th>%LB</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>75c_21s</td>
<td>*</td>
<td>0.00</td>
<td>5217</td>
</tr>
<tr>
<td>75c_28s</td>
<td>*</td>
<td>0.00</td>
<td>5582</td>
</tr>
<tr>
<td>100c_21s</td>
<td>*</td>
<td>0.00</td>
<td>5206</td>
</tr>
<tr>
<td>100c_28s</td>
<td>*</td>
<td>0.00</td>
<td>6531</td>
</tr>
<tr>
<td>111c_28s</td>
<td>*</td>
<td>0.13</td>
<td>11265</td>
</tr>
<tr>
<td>200c_21s</td>
<td></td>
<td>0.60</td>
<td>28135</td>
</tr>
<tr>
<td>250c_21s</td>
<td></td>
<td>0.57</td>
<td>27939</td>
</tr>
<tr>
<td>300c_21s</td>
<td></td>
<td>0.84</td>
<td>25903</td>
</tr>
</tbody>
</table>
Optimal solution to 111c_28s
Optimal solution to Distance-constrained VRP instance
Heuristic solution to VRP with satellite facilities instance
References


