Vehicle routing problems

- **TSP** *travelling salesman problem*
- **CVRP** *capacitated vehicle routing problem* (1959)
- **VRPTW** *vehicle routing problem with time-windows* (70-80)
- ... 

- NP-hard, combinatorial optimization
- Explosion of computing time when problem size increases

Real-life VRP solver requirements

- Real-life routing
- Get near-optimal solutions
- Low computing times
- Scale to huge problem instances
DATA
OPTIM

Routing

DATA

Vroom
Vehicle Routing Open-source Optimization Machine

Matrix

Profiles

OSRM
Open Source Routing Machine

Julien Coupey
July 29th 2018
Timeline

Highlights:

- **v0.1** Solve TSP with OSRM integration
- **v0.3** Handle “open” trips, improve results on asymmetric problems
- **v1.0** Stable API, multi-threading, switch to OSRM v5.*
- **v1.1** Support for libosrm
- **v1.2** Multiple vehicles, skills, multi-dimensional capacities, user-defined matrices, exploration level
**TSP**

Christofides heuristic (1976) on symmetrized problem

**CVRP**

Dedicated clustering heuristic using spanning trees
Local search

Apply modification operators to the heuristic solution
Solving approach

Local search

Relocate

Exchange

Or-Opt

Cross-exchange

2-Opt

Reversed 2-Opt
Local search implementation for CVRP

▶ Basic local search step
1. Evaluate validity/gain for operators on all pair of routes
2. Perform “best” move
3. Re-evaluate only what’s necessary until no more improvement is found

▶ Get out of local minimum or deadlock
1. Remove the “worst” jobs for all routes
2. Refill the routes and reapply a local search step

▶ Exploration level ranging from 0 to 5
TSPLIB benchmark description

- 78 TSP instances
- Sizes ranging from 50 to 18,511 points
- Average size $\approx 1,170$ points

Hardware

CPU: Intel Xeon E5-1620 @ 3.50GHz, 4c/8t
Results

- Median computing time: 28 ms
- Average gap to optimal solution: +3.0%
- Worst gap to optimal solution: +7.6%

Examples

<table>
<thead>
<tr>
<th>Instance</th>
<th>Size</th>
<th>Computing time</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>kroA100</td>
<td>100</td>
<td>5 ms</td>
<td>+0.0%</td>
</tr>
<tr>
<td>kroB200</td>
<td>200</td>
<td>9 ms</td>
<td>+1.7%</td>
</tr>
<tr>
<td>d493</td>
<td>493</td>
<td>48 ms</td>
<td>+3.8%</td>
</tr>
<tr>
<td>u1060</td>
<td>1,060</td>
<td>328 ms</td>
<td>+3.0%</td>
</tr>
<tr>
<td>u2152</td>
<td>2,152</td>
<td>1610 ms</td>
<td>+4.9%</td>
</tr>
<tr>
<td>rI5915</td>
<td>5,915</td>
<td>25.8 s</td>
<td>+2.4%</td>
</tr>
<tr>
<td>usa13509</td>
<td>13,509</td>
<td>≃14 m</td>
<td>+3.0%</td>
</tr>
<tr>
<td>d18512</td>
<td>18,512</td>
<td>≃35 m</td>
<td>+2.9%</td>
</tr>
</tbody>
</table>
CVRPLIB benchmark description

- 189 CVRP instances
- Sizes ranging from 15 to 1,000 jobs
- Average size ≃ 240 jobs, number of vehicles ranging from 2 to 207
- Average capacity tightness \( \left( \frac{\sum \text{job amounts}}{\sum \text{vehicle capacity}} \right) \): 0.95
## Global indicators

<table>
<thead>
<tr>
<th>Exploration level</th>
<th>0 (fastest)</th>
<th>5 (best)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median computing time</td>
<td>87ms</td>
<td>1019ms</td>
</tr>
<tr>
<td>Longest computing time</td>
<td>9.1s</td>
<td>254.3s</td>
</tr>
<tr>
<td>Jobs assigned</td>
<td>99.67%</td>
<td>99.88%</td>
</tr>
<tr>
<td>Solutions with all jobs</td>
<td>162 (85.7%)</td>
<td>171 (90.5%)</td>
</tr>
<tr>
<td>Best known solutions</td>
<td>8 (4.2%)</td>
<td>30 (15.9%)</td>
</tr>
</tbody>
</table>

## Gaps to best known solutions

Only reported for instances with all jobs assigned.

<table>
<thead>
<tr>
<th>Exploration level</th>
<th>0 (fastest)</th>
<th>5 (best)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum gap</td>
<td>+0.00%</td>
<td>+0.00%</td>
</tr>
<tr>
<td>Median gap</td>
<td>+3.95%</td>
<td>+1.35%</td>
</tr>
<tr>
<td>Average gap</td>
<td>+4.81%</td>
<td>+2.23%</td>
</tr>
<tr>
<td>Worst gap</td>
<td>+21.45%</td>
<td>+12.95%</td>
</tr>
</tbody>
</table>
https://github.com/VROOM-Project
https://github.com/VROOM-Project
https://github.com/VROOM-Project
**Usage**

**Solving**

Command-line

$ vroom -i input.json -o output.json -t 4 -x 5
$ vroom -i input.json -g -a router.project-osrm.org -p 80

http request

$ curl --header "Content-Type:application/json"
   --data @input.json http://solver.vroom-project.org
Why use VROOM?

- Open
  - Based on OpenStreetMap data and tooling
  - BSD-licensed
- Efficient
  - Very good solutions
  - Very fast
  - Scale to huge problem sizes

Work in progress

Timing constraints are scheduled for v1.3.
http://map.vroom-project.org/

https://github.com/VROOM-Project/vroom/wiki

@VroomProject

Thank you for your attention!