A route minimization Heuristic for rich Vehicle Routing Problems

Morten Smedsrud, SINTEF ICT

SPIDER - A Generic VRP Solver

- Designed to be widely applicable
- Based on generic, rich model
- Predictive route planning
- Plan repair, reactive planning
- Dynamic planning with stochastic model

- Framework for VRP research

SPIDER - Generalisations of CVRP

- Heterogeneous fleet
  - Capacities
  - Equipment
  - Arbitrary tour start/end locations
  - Time windows
  - Cost structure
- Linked tours with precedences
- Mixture of order types
- Multiple time windows, soft time windows
- Capacity in multiple dimensions, soft capacity
- Alternative locations, tours and orders
- Arc locations, for arc routing and aggregation of node orders
- Alternative time periods
- Non-Euclidean, asymmetric, dynamic travel times
- Compatibility constraints
- A variety of constraint types and cost components
  - driving time restrictions
  - visual beauty of routing plan, non-overlapping

Limitations and motivation

- Performance with a rich VRP model been lagging behind best known results on synthetic cases.
- Quality of solutions has been lower than best known results.
- Example: Exchange neighborhood. In a delivery case you can simply move a segment from one tour to another, but in a PDP case you also have to consider placement of the complementary tasks, increasing complexity by a whole degree.
- Focus on work been on trying to improve performance in SPIDER and come closer to best known results. Initial focus on tour reduction.

2 phase tour reduction approach

- Alternates between 2 phases, performing a local search with fast simple neighborhood operators in each phase.
- Operators are: 2-opt, 3-opt, or-exchange, exchange, cross.
- 1st phase uses a pure tour reduction objective
- 2nd phase uses traditional minimize number of vehicles first, tour distance second.
- The phases are otherwise identical.

Tour reduction objective

$$o(\sigma) = \left\langle |\sigma|, -\sum |r|^2 + N * st(\sigma), mdl(\sigma) \right\rangle$$

- Based on the tour reduction phase in Bent and Van Hentenryck's paper of 2004 [1].
- Primary objective is number of tours
- Secondary objective is maximizing the sum of the square length of the tours in [1]. This is modified relatively to [1] by subtracting the length of the shortest tour multiplied by number of orders in the case from the square sum.
- Motivation of the modification of secondary objective is so that it is always preferred to make the shortest tour shorter rather than moving an order from a long tour to even longer tour.
- Tertiary objective is minimizing the minimum delay in the plan, defined on next page. First introduced in J. Homberger and H. Gehring's paper of 1999 [2] and first used as objective in [1].

Minimum delay objective

$$mdl(\sigma) = mdl(r,\sigma) \text{ where } |r| = \min_{r' \in \sigma} |r'|$$

$$mdl(r,\sigma) = \sum_{i \in cust(r)} mdl(i,r,\sigma)$$

$$mdl(i,r,\sigma) = \begin{cases} 0 \text{ if } N(relocation,i,\sigma) \neq \emptyset \\ \infty \text{ if } \forall r' \in r : r \neq r' : q(r') + q_i > Q \\ \min_{j \in Customers \setminus cust(r)} mdl(i,j,r,\sigma) \end{cases}$$

$$mdl(i,j,r,\sigma) = \max(\delta_j + c_{ji} - l_i, 0) + \max(\delta_i + c_{ij^+} - z_{j^+}, 0)$$

Observations

- When starting new local search we see that the operators initially find a lot of improvements to the plan but slows down eventually until a new local minimum is reached.
- After switching objectives operators find improvements fast again.
- Running to local minimum then switching between objectives seems to yield ok results in tour reduction.
- Seems to give good diversification, quality of initial solution seems irrelevant.
- Often the local minima after running with tour reduction phase ended with shortest tours consisting of only 1-3 orders, so a tour depletion procedure was implemented

Tour depletion

- Tries to shorten each of the shortest tours by moving orders to the remaining tours.
- For each of the remaining tours it tries to move orders to the other remaining tours to make room for orders from the shortest tour.
- Above can be applied recursively, but computation time increases geometrically so only do one step for now.
- If a move from shortest tour is possible it is performed, and we continue with the new solution.
- If no move is possible, backtracks and tries move to next remaining tour.

Algorithm

```
Solution solution = problem.getInitialSolution();
Solution bestSolution = solution;
while (time < timeLimit)
{
	problem.setObjective(routeReductionObjective);
	solution.localSearch(lsTimeLimit);
	solution.tourDepletion();
	solution.localSearch(lsTimeLimit);
	solution.tourDepletion();
	problem.setObjective(distanceReductionObjective);
	solution.localSearch(lsTimeLimit);
	solution.tourDepletion();
	if (solution.objective < bestSolution.objective)
		bestSolution = solution;
}

return bestSolution;
```

Results

Test	Tours	Distance	Test	Tours	Distance	Test	Tours	Distance
C1_4_1	40	7152.06	C2_4_1	12	4116.33	RC1_4_1	36	10530.2
C1_4_2	37	7451.33	C2_4_2	12	3931.6	RC1_4_2	36	9083.9
C1_4_3	36	7617.65	C2_4_3	12	3807.45	RC1_4_3	36	7994.56
C1_4_4	36	7006.08	C2_4_4	12	3686.32	RC1_4_4	36	7598.94
C1_4_5	40	7152.06	C2_4_5	12	3945.29	RC1_4_5	36	9863.32
C1_4_6	40	7153.45	C2_4_6	12	3875.94	RC1_4_6	36	8925.06
C1_4_7	40	7149.43	C2_4_7	12	3914.22	RC1_4_7	36	9041.1
C1_4_8	38	7444.08	C2_4_8	12	3813.23	RC1_4_8	36	8469.47
C1_4_9	37	7158.36	C2_4_9	12	3894.84	RC1_4_9	36	8587.73
C1_410	36	8112.77	C2_410	12	3706.55	RC1_410	36	7871.65
	380	73397.27		120	38691.77		360	87965.93
R1_4_1	40	10528.3	R2_4_1	8	9479.48	RC2_4_1	12	6611.34
R1_4_2	36	10150.1	R2_4_2	8	7745.79	RC2_4_2	10	6124.63
R1_4_3	36	8398.81	R2_4_3	8	6186.31	RC2_4_3	8	5126.14
R1_4_4	36	7671.36	R2_4_4	8	4614.28	RC2_4_4	8	3842.05
R1_4_5	36	10503.5	R2_4_5	8	7415.72	RC2_4_5	9	6245.48
R1_4_6	36	9241.39	R2_4_6	8	6479.14	RC2_4_6	8	6375.98
R1_4_7	36	8067.68	R2_4_7	8	5506.08	RC2_4_7	8	5675.12
R1_4_8	36	7594.22	R2_4_8	8	4331.44	RC2_4_8	8	5043.74
R1_4_9	36	10016.7	R2_4_9	8	6766.41	RC2_4_9	8	4788.56
R1_410	36	8594.15	R2_410	8	6235.97	RC2_410	8	4453.03
	364	90766.21		80	64760.62		87	54286.07

Results continued

- Total number vehicles 1391, a lot closer to the best known to us result of 1388 (PRESCOTT-GAGNON et al. 2008) [3].
- Total distance 409868. Compared to 390771 in [3].
- Maximum amount of computational time in tour reduction phase before finding plan with fewest vehicles: 2h22min
- Average time before finding the plan with fewest vehicles: 17.5min

Conclusion

- This is work in progress, hoping to find ways to improve these results.
- Managed to get much closer to world records in number of tours inside the generic SPIDER framework.
- Route distance need improving still.

Future work

- Tests on different problem sizes
- Generalizing the procedure to account for more aspects of SPIDER's rich model.
- Travel distance minimizing.

References

[1] R. Bent and P. Van Hentenryck. A two stage hybrid local search for the vehicle routing problem with time windows. TRANSPORTATION SCIENCE Vol. 38, No. 4, November 2004, pp. 515–530.

[2] J. Homberger and H. Gehring. Two evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows. INFOR, 37:297-318, 1999

[3] A Branch-and-Price-Based Large Neighborhood Search Algorithm for the Vehicle Routing Problem with Time Windows. E. Prescott-Gagnon, G. Desaulniers, L.-M. Rousseau, submitted to Networks.