

An efficient incremental cluster first- route second algorithm for the capacitated vehicle routing problem

Route 2007 Workshop

Olli Bräysy (oli.braysy@jyu.fi)

Pekka Hotokka



UNIVERSITY OF JYVÄSKYLÄ

O. Bräysy

AGORA
HUMAN TECHNOLOGY CENTER

2007-5-14

Scope of the presentation

- In this presentation we focus on the capacitated vehicle routing problem
- A new cluster first-route second heuristic is described
- Some ideas for using the information on the desired solution structures are discussed
- A new variant of the ruin and recreate principle is suggested
- Results on the standard CVRP benchmarks are illustrated

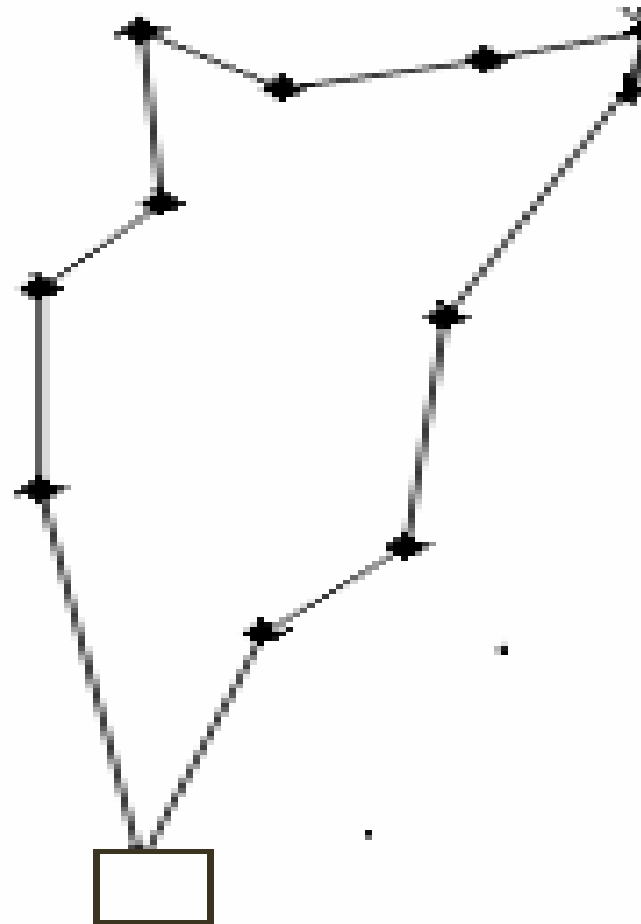
Background

- The goal is to develop a fast and well scalable heuristic
- The basic strategy is to create as good initial solutions as possible
- The search is based on a cheapest insertion heuristic and variants of CROSS and Or-opt exchanges.

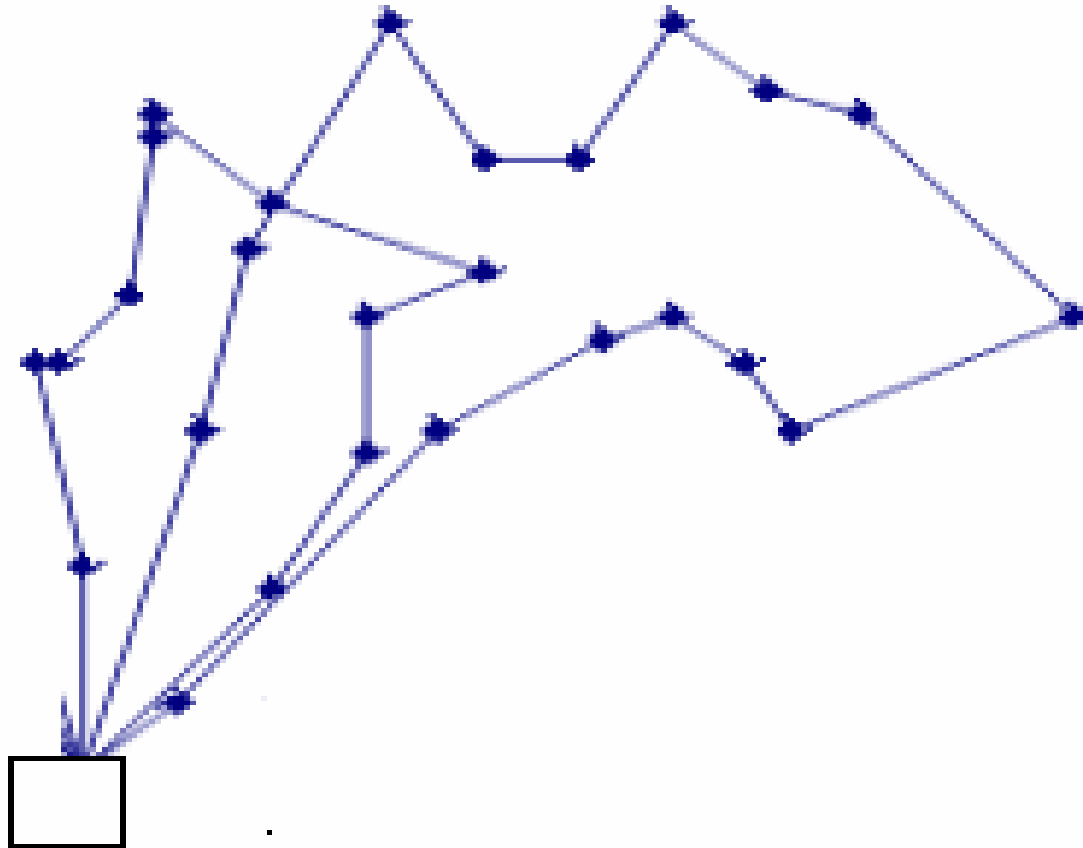
Using the information on the known solution structures

- By analyzing the the best-known solutions to CVRP benchmarks and real-life problems, one can conclude that in most of the cases the routes are organized according to three basic structures or their combinations.
- We suggest to use this information to guide the search

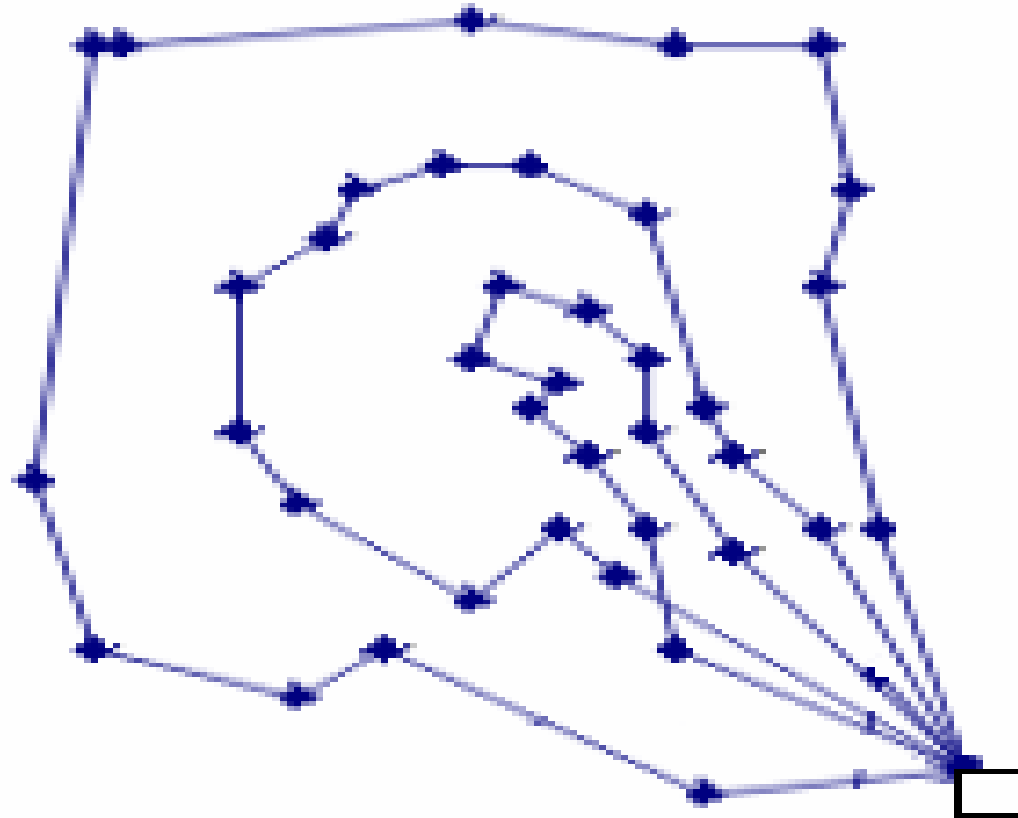
Structure 1



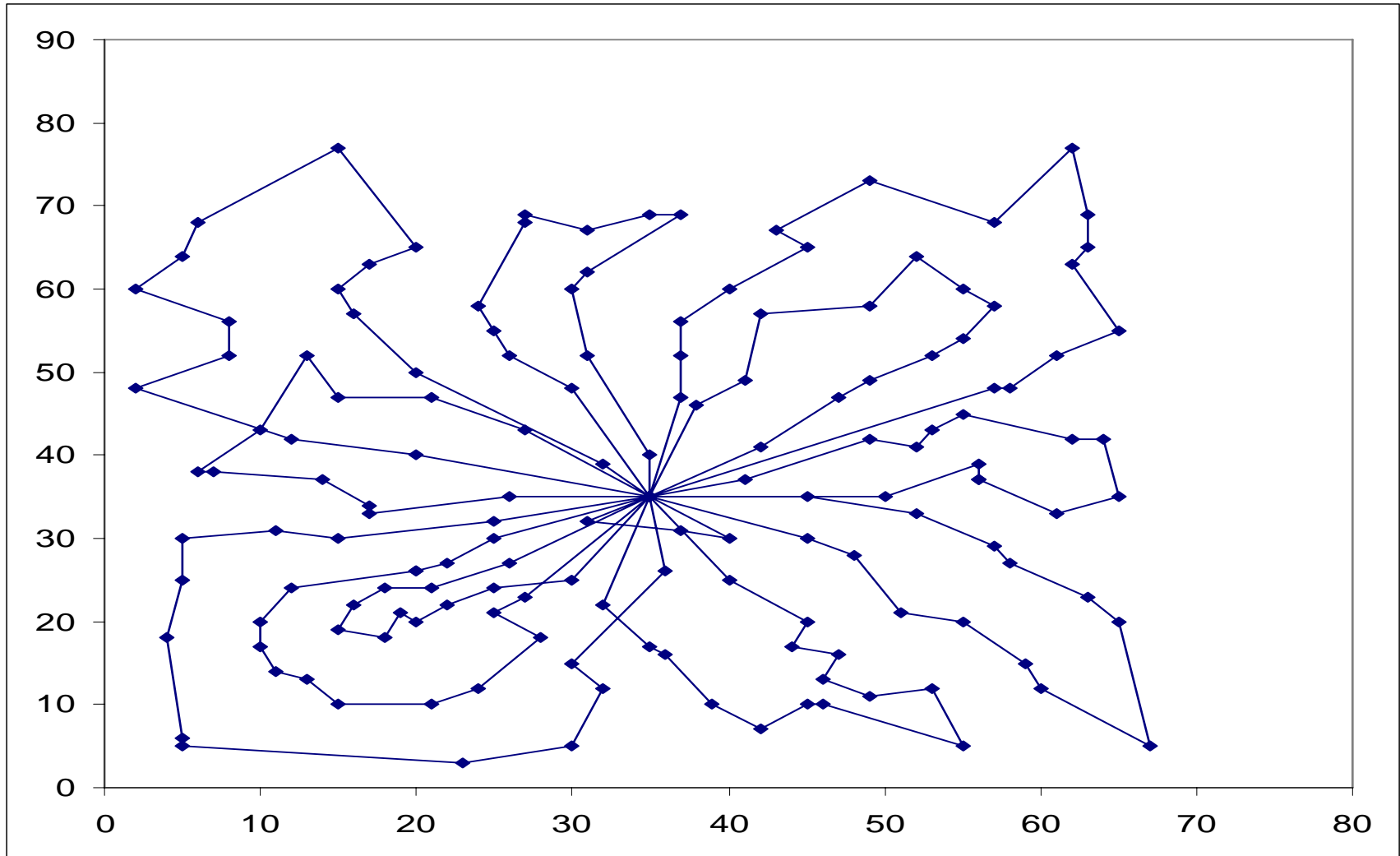
Structure 2



Structure 3



An example



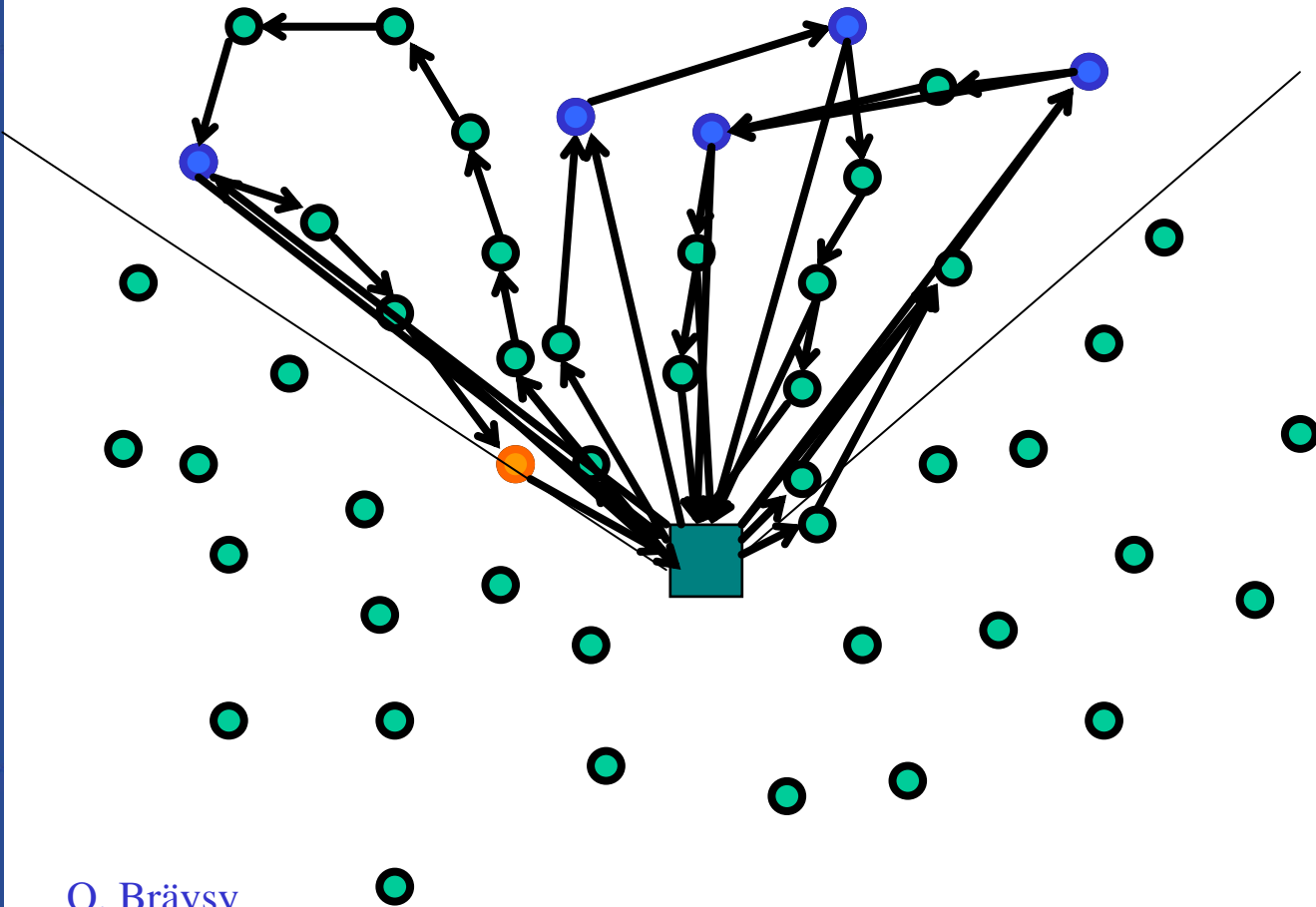
Our approach

- We suggest a multi-seed constructive method to force the different structures in the initial solution
- Routes are constructed according to a sweep-type multi-route clusters to limit and focus the search
- Moreover, the insertions are done in two phases, considering customers closest to depot last
- A special capacity maximization scheme is also applied

Scenarios

- We consider in each case three-route sectors
- Within each sector six different scenarios are tested and the best is selected
- The scenarios are:
 - 1: I/1 – I/1 – I/1
 - 2: II/2 – I/1
 - 3: I/1 – II/2
 - 4: III/2 – I/1
 - 5: I/1 – III/2
 - 6: III/3

An illustration



The main rules

- We consider several neighboring starting points for sectoring
- The customers are renumbered according to polar angles
- The routes are constructed sequentially with a limit for insertion cost
- After removing (15%) of the closest customers to depot, the capacity of the routes/vehicles is adjusted accordingly
- In the end, the customers close to depot are inserted with a parallel insertion strategy
- Customers located distant from depot are favoured
- IOPT is applied once within the construction
- The capacity maximization is done with ICROSS and only in the end with the customers closest to depot

Seed selection

- The seed selection is based on capacity accumulation within each sector
- Narrow (10% or 20% of the sector capacity) subsectors are defined according to given percentages and the seed is the farthest unrouted customer from depot within the subsector
- In each scenario, a 10% sector from both edges is defined.
- With nested scenarios, 3 seeds are used.
- The rest are defined to make sure desired overlapping

Improvement phase

- The improvement phase is based on ICROSS and IOPT exchanges (with segment length 3)
- Only nearby route-pairs are considered in ICROSS
- To escape local minimum, two new ruin and recreate based approaches are applied in turn, for a given number of iterations

A ring removal

- First two circles with random distances from depot are defined (we favour zero distance for the first circle)
- A given percentage of customers are randomly removed within the defined ring so that the smaller the demand the higher the probability for removal
- Then ICROSS and IOPT are applied to the partial solution until local minimum
- The removed customers are then reinserted using a parallel insertion strategy
- Finally, the ICROSS and IOPT are applied again to local minimum

Sector removal

- The sector removal sweeps the routes through once.
- The start angle as well as the sector sizes are determined randomly in each case
- Here a larger number of points are removed within the defined sector, favouring points with smaller demand
- Then, as in ring removal, local searches are applied to partial solution, followed by parallel reinsertion and another local search iteration.
- The two improvement schemes are applied in turn until a given time limit

Computational results

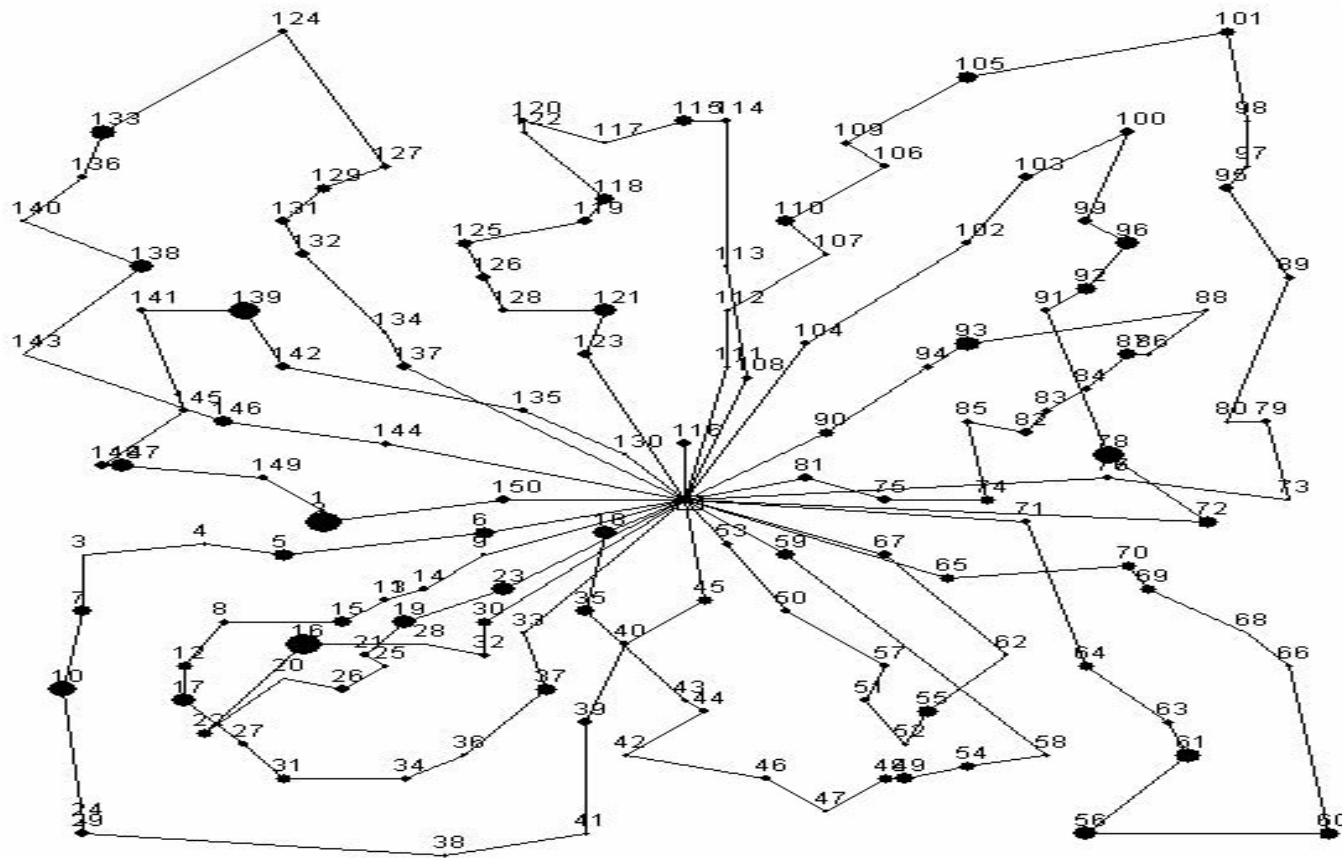
- The algorithm was coded with Java, and tested with AMD 3800+ X2 computer with 1GB memory
- A simple graphical user interface is also implemented
- The tests were done with the standard CVRP benchmarks by Christofides et al. (1979) and Golden et al. (1998)
- Problems with duration limit ignored

Sensitivity analysis

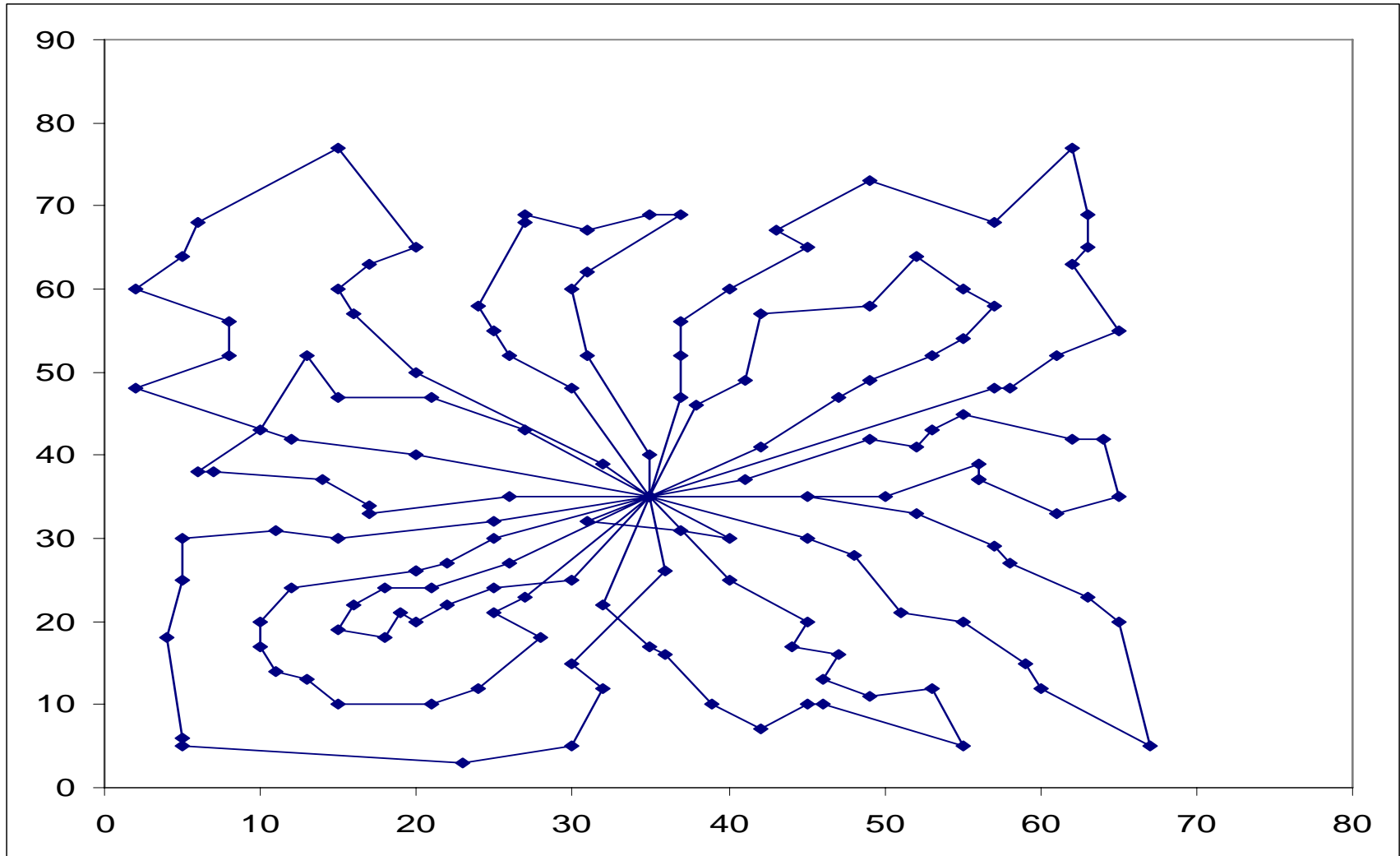
- Applying remove-insert scheme together with local search improves solution quality by 3.3% but increases CPU time by 90%
- Multi-seed construction 1.5% better than one seed
- 2-phase construction 1.6% better than 1-phase
- Capacity maximization improves solution quality by 0.39% at the cost of 20% CPU increase
- Applying local search after removal improves solution quality by 1.8%
- Sector removal appears 0.16% better than ring removal
- Applying both removal strategies together improves solution quality by 0.09% but increases CPU by 30-40%
- Very low values for lambda (<0.5) appear best
- The removed rings should be narrow (5-15% of max depot distance) and start close to depot (5-10%).

Windowed VRP solver

File Settings Help

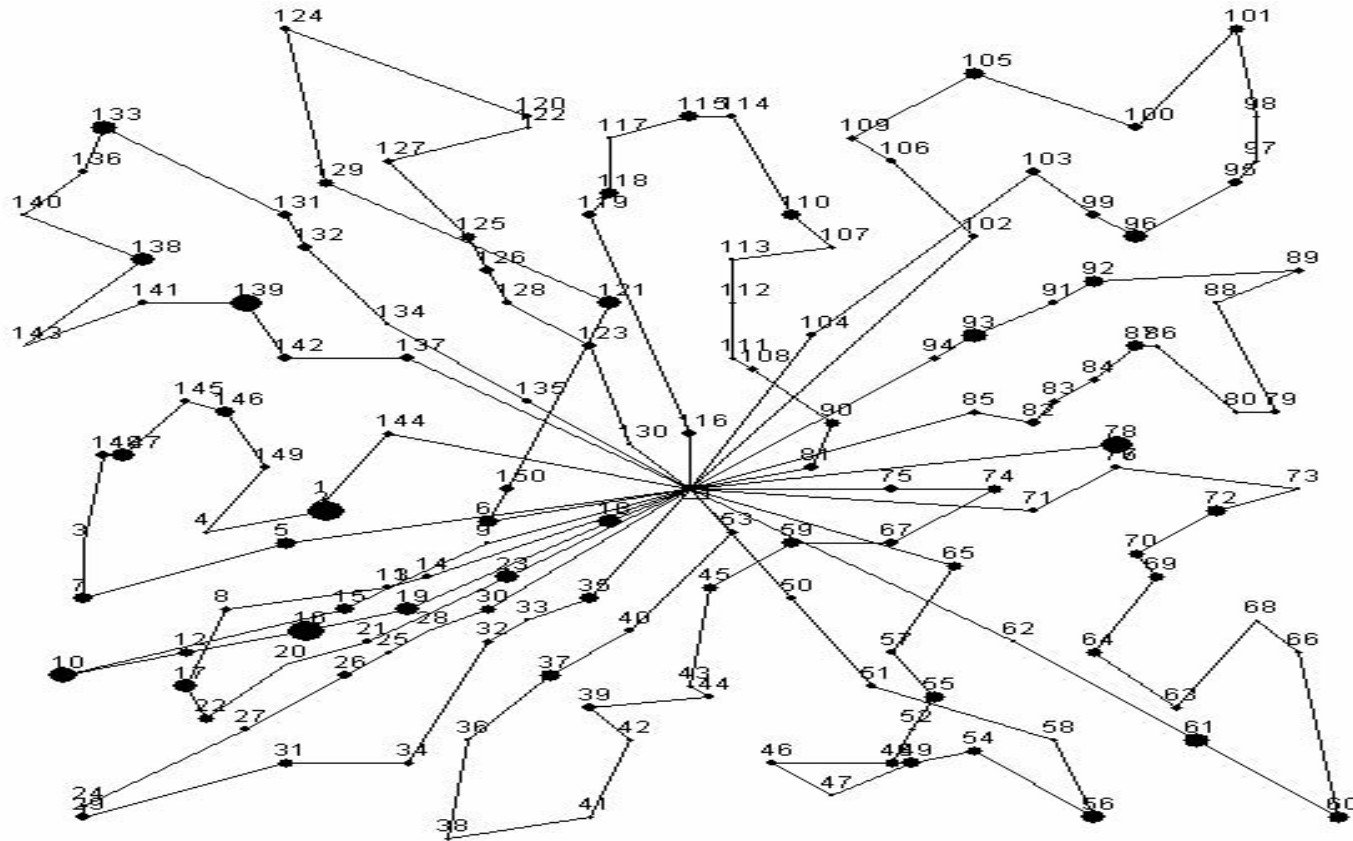


An example



Windowed VRP solver

File Settings Help



Käynnistä

2 Microsoft ...

bin

Komentorivi - ...

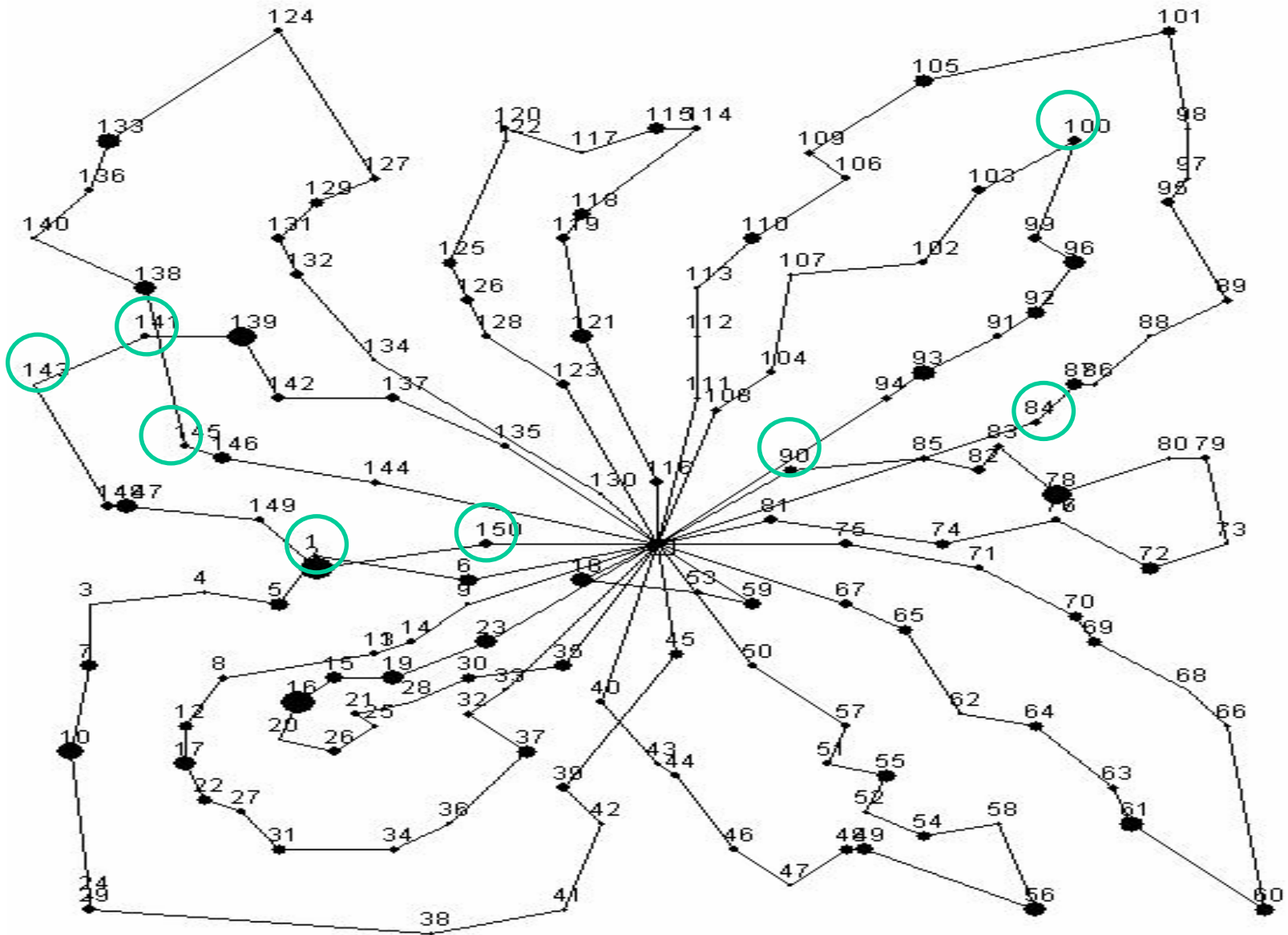
Microsoft Pow...

Results for the Christofides et al. instances

Problem	Previous best known	Rochat & Taillard	Toth & Vigo	Reimann et al.	Tarantilis	Mester & Bräysy	Our method
1 (50)	524.61	524.61	524.61	524.61	524.61	524.61	524.61
2 (75)	835.26	835.26	838.60	840.61	835.26	835.26	835.26
3 (100)	826.14	826.14	828.56	828.21	826.14	826.14	826.14
4 (150)	1028.42	1028.42	1033.21	1037.57	1028.42	1028.42	1030.48
5 (199)	1291.29	1291.45	1318.25	1306.91	1311.48	1291.29	1300.21
11 (120)	1042.11	1042.11	1042.87	1043.46	1042.11	1042.11	1042.11
12 (100)	819.56	819.56	919.56	819.56	819.56	819.56	819.56
Computer	N/A	SGI*	Pentium	Pentium	Pentium	Pentium	AMD
		100	200	900	400	IV 2800	3800+
CPU min	N/A	N/A	3.8	3.8	6.6	2.8	0.02

Results for the Golden et al. problems

Problem	Previous best known	Reimann et al.	Tarantilis	Kytöjoki et al.	Pisinger & Röpke	Mester & Bräysy	Our method
9 (255)	580.60	586.87	585.43	620.67	585.14	583.39	596.63
10 (323)	738.92	750.77	746.56	784.77	748.89	741.56	762.21
11 (399)	917.17	927.27	923.17	986.80	922.70	918.45	944.76
12 (483)	1107.19	1140.87	1130.40	1209.02	1119.06	1107.19	1145.82
13 (252)	857.19	865.07	865.01	925.81	864.68	859.11	874.66
14 (320)	1080.55	1093.77	1086.07	1155.19	1095.40	1081.31	1100.81
15 (396)	1340.24	1358.21	1353.91	1461.49	1359.94	1345.23	1373.25
16 (480)	1622.69	1635.16	1634.74	1742.86	1639.11	1622.69	1657.95
17 (240)	707.76	708.76	708.74	726.01	708.90	707.79	720.80
18 (300)	995.39	998.83	1006.90	1077.53	1002.42	998.73	1029.76
19 (360)	1366.14	1367.20	1371.01	1444.51	1374.24	1366.86	1420.98
20 (420)	1820.09	1822.94	1837.67	1938.12	1830.80	1820.09	1901.34
Computer	N/A	Pentium 900	Pentium 400	Athlon64 3000+	Pentium IV 3000	Pentium IV 2800	Athlon 3800+
CPU min	N/A	49.3	45.5	0.02	10.8	24.4	0.21



Conclusions

- We have presented ongoing work on a cluster first-route second CVRP heuristic
- The main new ideas include forcing and comparing given solution structures using a multi-seed and two-phase construction heuristic and combination of ruin and recreate principle with local search
- The first results on standard benchmarks are promising
- The next step is to include strategies for escaping local minimum and duration and time window constraint checks.