Tournées de véhicules avec contraintes de clustering

ROADEF 2014
February 27, 2014 (Bordeaux, France)

M. Sevaux[1] C. Expósito[2]
A. Rossi[1]

[1]marc.sevaux@univ-ubs.fr,
andre.rossi@univ-ubs.fr
Université de Bretagne-Sud (France)

[2]cexposit@ull.es
University of La Laguna (Spain)

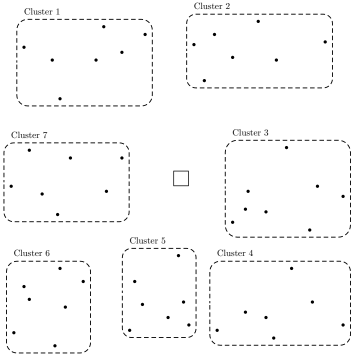
Contents

Introduction

- Parcel delivery and courier services companies:
  - TNT, MyUS, MRW, etc
- Picking up of parcels which have to be redistributed to a given final customer
- All the collected parcels are brought together in a central depot:
  - Unloading
  - Sorting
  - Loading
- The number of final customers supplied by these companies is generally large (hundreds or thousands)
- It is required to divide the service area for customer deliveries into manageable units, called **clusters** (*Districting Problem*)

Clustered Capacitated Vehicle Routing Problem

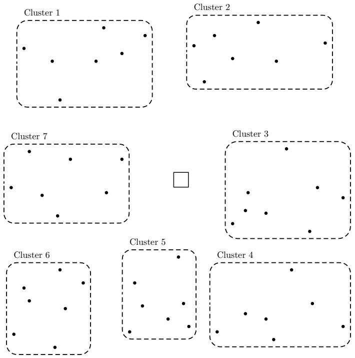
- Customers: $N = \{1, ..., n\}$
- Each customer $i \in N$ has a service demand $d_i > 0$
- Unlimited number of delivery trucks with load capacity k
- Each customer $i \in N$ belongs to a cluster $r_i \in R = \{1, ..., m\}$
- The travel cost between the points i and j is denoted by c_{ij}
- **Objective:** Minimize the total travel cost of the routes



Clustered Capacitated Vehicle Routing Problem

Assumptions:

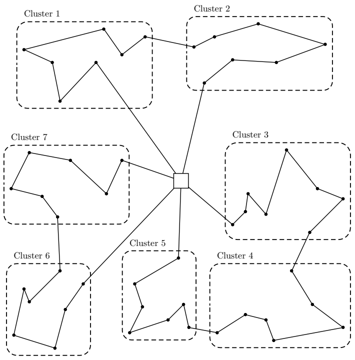
- Each cluster has to be visited by a single delivery truck
- All the customers belonging to the same cluster must be served one after another
- Once a truck starts to serve a given cluster, it has to serve all the customers belonging to it
- A single delivery truck can serve several clusters according to its load capacity



Clustered Capacitated Vehicle Routing Problem

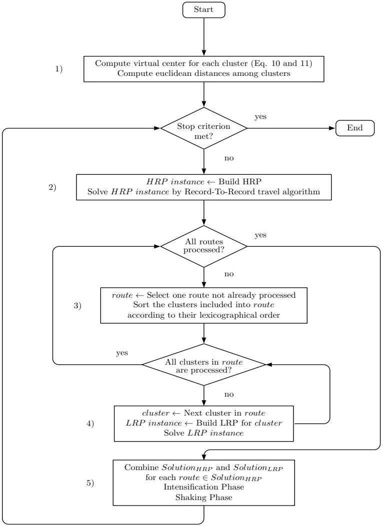
Assumptions:

- Each cluster has to be visited by a single delivery truck
- All the customers belonging to the same cluster must be served one after another
- Once a truck starts to serve a given cluster, it has to serve all the customers belonging to it
- A single delivery truck can serve several clusters according to its load capacity



Two-Level Solution Approach

- Decomposition of the CCVRP into routing subproblems
- The subproblems can be addressed by specific-purpose solution methods
- Dependencies between clusters allow to obtain reasonable travel costs between them
- This approach can be extended to manage those environments where there is no information about clusters



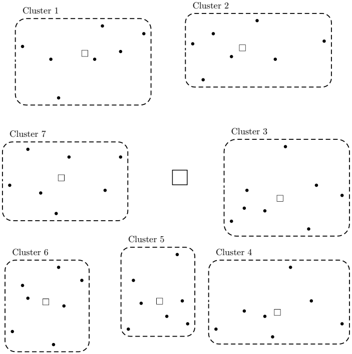
High-Level Routing Problem

- Defining the routes to serve the clusters (inter-cluster)
- For each cluster $r \in R$, a *virtual center* (x_r, y_r) is computed:

$$x_r = \frac{\sum_{i \in C_r} d_i \times x_i}{\sum_{i \in C_r} d_i}$$

$$y_r = \frac{\sum_{i \in C_r} d_i \times y_i}{\sum_{i \in C_r} d_i}$$

$$d_r = \sum_{i \in C_r} d_i$$



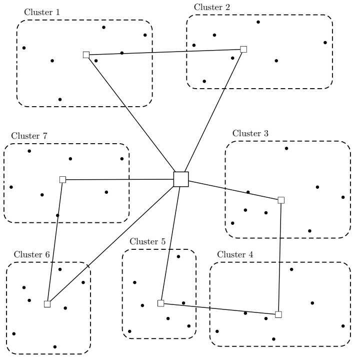
High-Level Routing Problem

- Defining the routes to serve the clusters (inter-cluster)
- For each cluster $r \in R$, a *virtual center* (x_r, y_r) is computed:

$$x_r = \frac{\sum_{i \in C_r} d_i \times x_i}{\sum_{i \in C_r} d_i}$$

$$y_r = \frac{\sum_{i \in C_r} d_i \times y_i}{\sum_{i \in C_r} d_i}$$

$$d_r = \sum_{i \in C_r} d_i$$



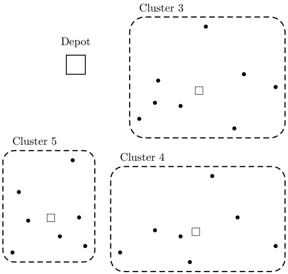
High-Level Routing Problem

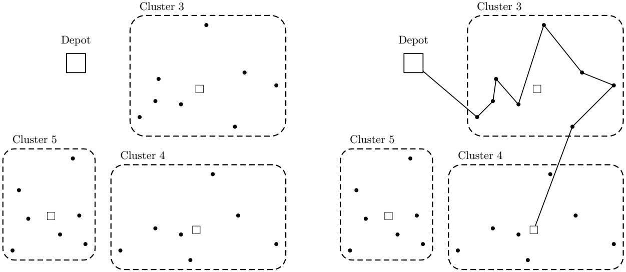
Record-to-Record Algorithm (Golden *et al.*):

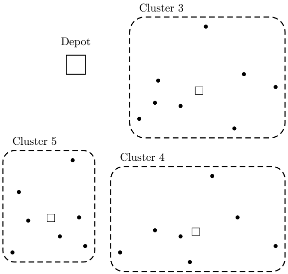
- An initial solution is generated by the Clarke and Wright algorithm
- It is based upon performing exchanges of points belonging to the same or different routes
- Non-improving movements are allowed in order to avoid stagnation and reach further promising regions of the search space
- After a given number of iterations without any improvement, a restarting procedure is carried out. It consists in perturbing some points from the best solution in such a way that they are moved toward new positions within the routes
- Improvements proposed by Li *et al.*

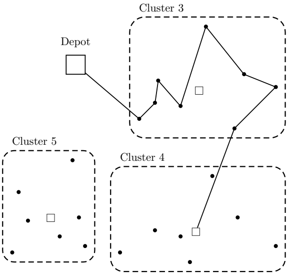
Low-Level Routing Problem

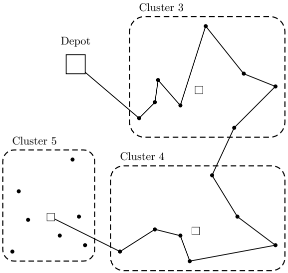
- Defining the visiting order of the customers belonging to each cluster, $C_r = \{1, ..., w\}$ (intra-cluster)
- Traditional responsibility of the driver:
  - He/she is acquainted with the region to serve
  - Non-robust solutions or suboptimal performance
  - High dependence
- It can be modeled as the Shortest Hamiltonian Path Problem (SHPP):
  - $G = (V, E)$
  - V is the set of customers belonging to the cluster, C_r
  - $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$
  - The weight of each $(v_i, v_j) \in E$ is c_{ij}
- Pool of solutions aimed at storing the hamiltonian paths and providing them (reduction of the computational time)

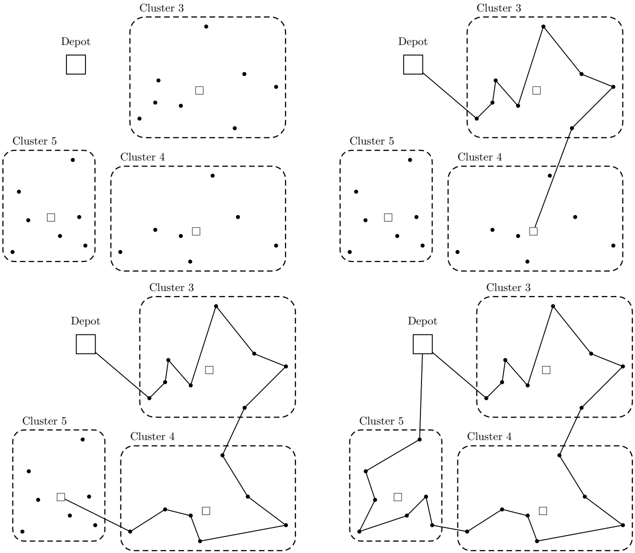

Cluster 3
Depot
Cluster 5
Cluster 4


Cluster 3
Depot
Cluster 5
Cluster 4
Cluster 3
Depot
Cluster 5
Cluster 4


Cluster 3
Depot
Cluster 5
Cluster 4


Cluster 3
Depot
Cluster 5
Cluster 4


Cluster 3
Depot
Cluster 5
Cluster 4


Depot
Cluster 3
Cluster 5
Cluster 4
Depot
Cluster 3
Cluster 5
Cluster 4
Depot
Cluster 3
Cluster 5
Cluster 4
Depot
Cluster 3
Cluster 5
Cluster 4

Low-Level Routing Problem

Mixed Integer Linear Programming:

$$\text{Minimize} \sum_{i=1}^{w} \sum_{j=1}^{i-1} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j=1}^{i-1} x_{ij} + \sum_{j=i+1}^{w} x_{ji} = 2 - y_i, \qquad \forall i \in C_r \tag{2}$$

$$\sum_{i=1}^{w} y_i = 2 \tag{3}$$

$$\sum_{(i,j) \in S \times S, i > j} x_{ij} \leq |S| - 1, \qquad \forall S \subset E \tag{4}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in C_r, \forall j \in \{1 \ldots i-1\} \tag{5}$$

$$y_i \in \{0, 1\}, \qquad \forall i \in C_r \tag{6}$$

Low-Level Routing Problem

Christofides' algorithm:

- Exact method based upon a decision-tree search
- The objective is to find the Shortest Spanning Tree (SST) in G:
  - The degree of no vertex exceeds two, but for the starting and ending nodes, whose degree is one
- The root of the decision tree is composed of G
- At each step, the SST is found (Kruskal)
- If the obtained SST is a hamiltonian path, the search finishes
- Otherwise, for each edge incident to those nodes that avoid being the pursued path, a new decision node is generated

Low-Level Routing Problem

Lin-Kernighan Heuristic:

- The TSP is stated as the optimization problem of finding that tour starting from a given node that visits all the nodes in a graph exactly once
- The SHPP is the TSP for which the edge that joins the starting and ending nodes of the path is set
- Techniques for the TSP can be adapted to the SHPP
- The LK Heuristic explores the most promising neighbours within the k-opt neighbourhood
- Implementation of Helsgaun: `www.akira.ruc.dk/~keld/research/LKH/`

Two-Level Solution Approach

Intensification Phase:

- It pursues to explore more thoroughly the current region
- Each pair of clusters is exchanged by the two-point operator
- The first improving neighbour solution is chosen

Shaking Phase:

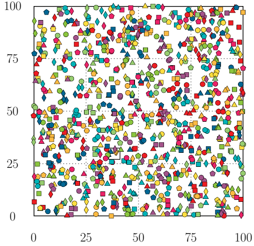
- It pursues to diversify the search by finding unexplored regions
- It perturbs the landscape of the HRP in the hope of finding high-quality solutions
- It adds random noise to the distances between those pairs of clusters visited consecutively:

$$\bar{c_{rr'}} = c_{rr'} + \phi,$$

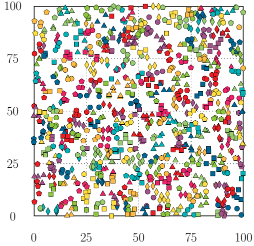
where $\phi \in [-\mu, \mu]$ is random noise. μ controls the shaking degree

Computational Experiments

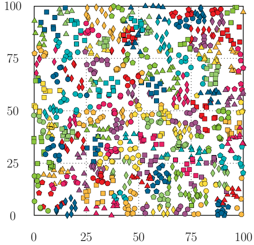
- PC equipped with Ubuntu 12.04, a processor Intel Core 2 Duo 3.16 GHz and 4 GB of RAM
- C++ language and g++ 4.7.2
- MILP has been executed with IBM ILOG CPLEX Optimizer 12.4
- Record-to-Record travel algorithm adapted to solve the CCVRP (RTR-CCVRP)
- Benchmark suite proposed by Golden et al. for the classic VRP. It is composed of 20 problem instances:
  - Customers from 200 up to 483
  - Truck capacity ranges from 200 up to 1000
  - Customers spatially distributed in a two-dimensional area
- We need to provide cluster information to instances of the VRP and generating new ones with the characteristics of the CCVRP



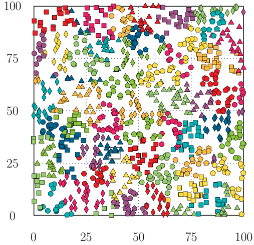
(a) $\rho = 10\%$



(b) $\rho = 25\%$



(c) $\rho = 50\%$



(d) $\rho = 100\%$

Computational Experiments

Small size instances

Instance Name	MILP		RTR-CCVRP			Two-Level		
	$f(x)$	$t\ (s)$	$f(x)$	$t\ (s)$	$Gap\ (\%)$	$f(x)$	$t\ (s)$	$Gap\ (\%)$
a-n10-c2.map	1778.19	0.12	**1778.19**	0.14	0.00	**1778.19**	0.15	0.00
b-n10-c2.map	1614.06	0.35	**1614.06**	0.12	0.00	**1614.06**	0.16	0.00
c-n10-c2.map	1782.26	0.41	1843.13	0.14	3.42	**1782.26**	0.16	0.00
d-n10-c2.map	2096.11	0.15	2138.75	0.10	2.03	**2096.11**	0.15	0.00
e-n10-c2.map	2016.57	0.03	2265.76	0.11	12.35	**2016.57**	0.17	0.00
a-n15-c4.map	1947.30	48.35	**1947.30**	0.21	0.00	**1947.30**	0.11	0.00
b-n15-c4.map	2602.56	77.97	**2602.56**	0.25	0.00	**2602.56**	0.11	0.00
c-n15-c4.map	1872.39	9.96	**1872.39**	0.27	0.00	**1872.39**	0.15	0.00
d-n15-c4.map	2937.79	893.98	3006.47	0.22	2.34	**2937.79**	0.15	0.00
e-n15-c4.map	2222.36	33.51	**2222.36**	0.31	0.00	**2222.36**	0.17	0.00
a-n20-c5.map	-	-	2759.13	0.49	-	2759.13	0.19	-
b-n20-c5.map	-	-	3001.94	0.45	-	3001.94	0.18	-
c-n20-c5.map	-	-	3028.83	0.45	-	3028.83	0.16	-
d-n20-c5.map	2239.09	792.58	**2239.09**	0.44	0.00	**2239.09**	0.19	0.00
e-n20-c5.map	-	-	3343.34	0.40	-	3343.34	0.15	-
a-n30-c6.map	-	-	3426.77	1.02	-	3426.77	0.18	-
b-n30-c6.map	-	-	3116.84	1.12	-	3116.84	0.17	-
c-n30-c6.map	-	-	3571.20	0.94	-	3571.20	0.20	-
d-n30-c6.map	-	-	3896.27	1.03	-	3887.14	0.17	-
e-n30-c6.map	-	-	3547.03	1.18	-	3547.03	0.18	-

Computational Experiments

$\rho = 10\%$

Instance			RTR-CCVRP		Two-Level	
Index	$n+1$	Q	*f(x)*	*t (s)*	*f(x)*	*Gap (%)*
1	240	550	6425.3742	62.91	**5801.6713**	-9.71
2	320	700	10195.8045	81.40	**9649.6744**	-5.36
3	400	900	13795.7911	155.14	**13249.2200**	-3.96
4	480	1000	19599.8398	242.09	**18966.9169**	-3.23
5	200	900	**9392.9511**	92.89	9479.7377	0.92
6	280	900	11982.9887	41.04	**11601.7726**	-3.18
7	360	900	13900.7414	143.81	**13243.1317**	-4.73
8	440	900	14445.3590	144.10	**13756.5063**	-4.77
9	255	1000	731.2050	64.92	**717.1626**	-1.92
10	323	1000	930.5563	62.11	**914.7267**	-1.70
11	399	1000	1164.3196	160.76	**1146.5675**	-1.52
12	483	1000	1415.2416	245.94	**1386.4790**	-2.03
13	252	1000	1056.5625	63.34	**1047.5666**	-0.85
14	320	1000	1356.0048	81.44	**1340.1580**	-1.17
15	396	1000	1729.8406	133.83	**1700.2771**	-1.71
16	480	1000	2147.7285	153.54	**2097.4678**	-2.34
17	240	200	889.3679	75.82	**867.0320**	-2.51
18	300	200	1132.8670	123.29	**1104.8649**	-2.47
19	360	200	1591.8057	136.47	**1522.8257**	-4.33
20	420	200	2098.7685	148.56	**2019.5521**	-3.77
# Best solutions				1		19
Gap to best solutions (%)				4.00		0.08
(s)				120.67		60.00

Computational Experiments

$\rho = 50\%$

Instance			RTR-CCVRP		Two-Level	
Index	$n+1$	Q	*f(x)*	*t (s)*	*f(x)*	*Gap (%)*
1	240	550	6821.9436	42.61	**6719.1711**	-1.51
2	320	700	9952.8343	85.04	**9904.4014**	-0.49
3	400	900	**13294.9777**	168.46	13303.3085	0.06
4	480	1000	**17813.0729**	349.49	17935.5843	0.69
5	200	900	9237.7946	67.17	**8790.4445**	-4.84
6	280	900	10891.8214	127.60	**10714.3367**	-1.63
7	360	900	12983.9693	148.19	**12862.8973**	-0.93
8	440	900	14052.9702	211.49	**13924.7896**	-0.91
9	255	1000	708.8568	90.22	**703.0738**	-0.82
10	323	1000	**897.7009**	87.63	898.1876	0.05
11	399	1000	1116.5932	137.92	**1112.3526**	-0.38
12	483	1000	1321.8978	175.71	**1319.9841**	-0.14
13	252	1000	1090.3918	46.95	**1080.8394**	-0.88
14	320	1000	1382.4219	88.34	**1363.9887**	-1.33
15	396	1000	1695.4373	107.26	**1685.6071**	-0.58
16	480	1000	2057.4539	166.64	**2030.5982**	-1.31
17	240	200	930.6724	50.42	**910.7285**	-2.14
18	300	200	1233.4388	70.05	**1217.7125**	-1.27
19	360	200	1671.2650	81.61	**1631.2084**	-2.40
20	420	200	2325.6720	153.82	**2325.4717**	-0.01
# Best solutions				3		17
Gap to best solutions (%)				1.07		0.12
Average CPU times (*s*)				122.83		60.00

Computational Experiments

$\rho = 100\%$

Instance			RTR-CCVRP		Two-Level	
Index	$n+1$	Q	*f(x)*	*t (s)*	*f(x)*	*Gap (%)*
1	240	550	6315.9475	131.88	**6293.0355**	-0.36
2	320	700	9902.9831	229.25	**9879.5855**	-0.24
3	400	900	12386.5523	308.78	**12361.0907**	-0.21
4	480	1000	16200.4848	579.02	**16130.3901**	-0.43
5	200	900	8417.5264	84.76	**8394.1106**	-0.28
6	280	900	10905.5934	201.42	**10777.3284**	-1.18
7	360	900	11460.9133	217.33	**11346.1129**	-1.00
8	440	900	13243.8320	252.03	**13188.9350**	-0.41
9	255	1000	705.7461	82.24	**705.1943**	-0.08
10	323	1000	841.3554	128.08	**837.5160**	-0.46
11	399	1000	1058.3821	176.19	**1054.1331**	-0.40
12	483	1000	1303.9874	267.67	**1297.3100**	-0.51
13	252	1000	**996.3603**	48.23	**996.3603**	0.00
14	320	1000	1223.4956	73.49	**1223.0864**	-0.03
15	396	1000	1533.2332	224.09	**1531.2902**	-0.13
16	480	1000	1879.8945	216.40	**1874.6909**	-0.28
17	240	200	859.4310	56.69	**844.2719**	-1.76
18	300	200	1215.6799	116.59	**1212.9717**	-0.22
19	360	200	1667.8706	181.25	**1667.4536**	-0.03
20	420	200	2133.3389	137.11	**2128.5967**	-0.22
# Best solutions				1		20
Gap to best solutions (%)				0.49		0.00
Average CPU times (*s*)				185.63		60.00

Conclusions and Further Research

- Organizing the customers into clusters allows to perform a cluster-based distribution
- A two-level solution approach based upon an a priori decomposition into two routing subproblems
- The exact solution methods aimed at solving the low-level problems are suitable whenever the number of customers is small
- The framework includes an intensification phase aimed at finding local optimum solutions
- Feedback process based upon perturbing the landscape of the high-level routing problem in order to diversify the search

- Integrating the Districting Problem into the developed approach
- Addressing variants of the VRP with clustering constraints

Tournées de véhicules avec contraintes de clustering

ROADEF 2014

February 27, 2014 (Bordeaux, France)

M. Sevaux[1] C. Expósito[2]
A. Rossi[1]

[1]marc.sevaux@univ-ubs.fr,
andre.rossi@univ-ubs.fr
Université de Bretagne-Sud (France)

[2]cexposit@ull.es
University of La Laguna (Spain)