A new branch-and-cut algorithm for the capacitated vehicle routing problem

Jens Lysgaard
Department of Business Studies
Aarhus School of Business
Aarhus University
Denmark

Adam N. Letchford, Richard W. Eglese
Department of Management Science
Lancaster University
England

(J. Lysgaard, A.N. Letchford & R.W. Eglese:
“A new branch-and-cut algorithm for the capacitated vehicle routing problem”,
Mathematical Programming, 2004, vol. 100, pp. 423-445)

The Capacitated Vehicle Routing Problem (CVRP)
Problem definition

Given:

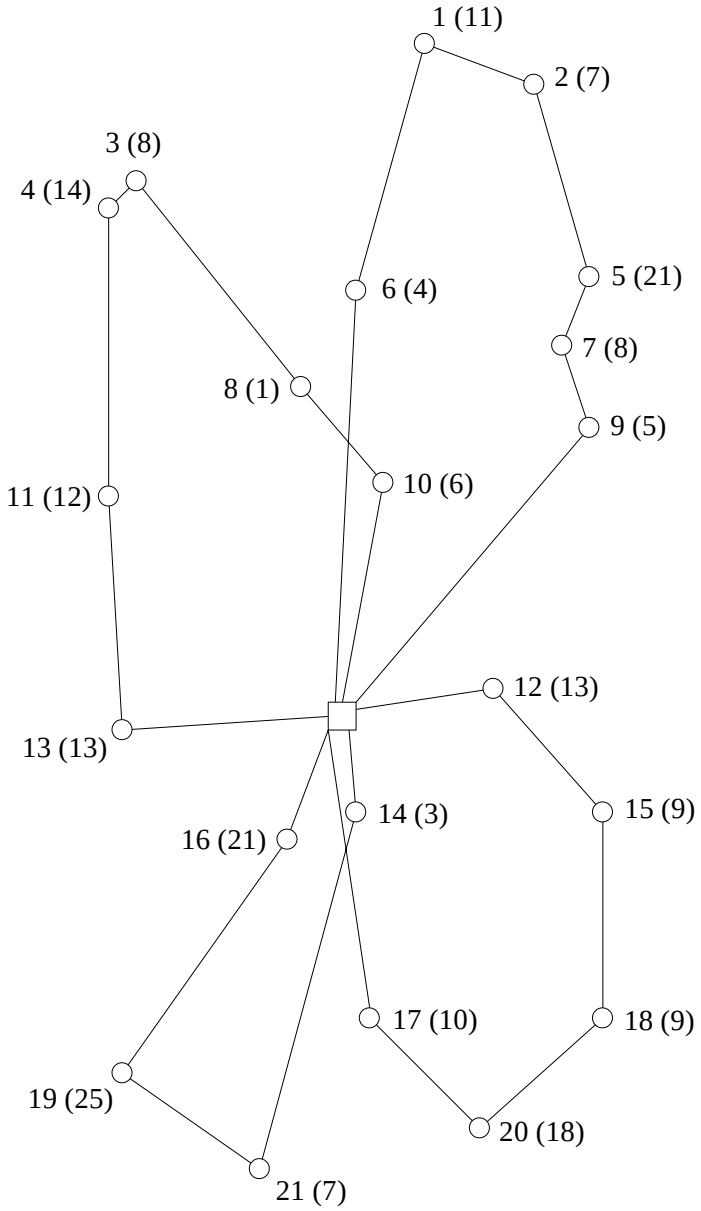
- A depot
- A fleet of identical vehicles with capacity Q
- n customers with demands $q_i \leq Q$, for $i = 1, \ldots, n$, to be delivered from the depot
- Symmetric travel cost c_{ij} between points i and j for all pairs of points

Determine routes of minimum total travel cost, subject to:

- Each customer is serviced exactly once
- The total quantity delivered on each route does not exceed Q
- Each route begins and ends at the depot

Example: E-n22-k4

Q = 60



Integer Linear Programming (ILP) formulation (vehicle flow formulation)

Graph representation:

- $G = (V, E)$
- $V = \{0, 1, \ldots, n\}$
- Customer set $V_C = \{1, \ldots, n\}$, node 0 is the depot

Decision variables x_{ij}:

x_{ij} = the number of times a vehicle travels between nodes i and j

$$\text{min.} \quad \sum_{\{i,j\} \in E} c_{ij} x_{ij} \qquad (1)$$

$$\text{s.t.:} \quad x\left(\delta(i)\right) = 2 \qquad \forall i \in V_C \qquad (2)$$

$$x\left(\delta(S)\right) \geq 2r(S) \qquad \forall S \subseteq V_C : |S| \geq 2 \qquad (3)$$

$$x_{ij} \in \{0,1\} \qquad \forall \{i, j\} \in E(V_C) \qquad (4)$$

$$x_{0j} \in \{0,1,2\} \qquad \forall j \in V_C \qquad (5)$$

Capacity inequalities

1. Capacity inequalities: $x\left(\delta\left(S\right)\right) \geq 2r\left(S\right) \forall S \subseteq V_c : \left|S\right| \geq 2$

   $r(S)$ is the optimum solution to the Bin Packing Problem given by the demands of customers in S and bin capacity Q

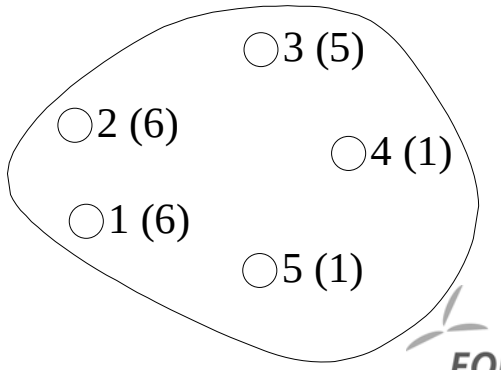
2. Rounded capacity inequalities: $x\left(\delta\left(S\right)\right) \geq 2k\left(S\right) \forall S \subseteq V_c : \left|S\right| \geq 2$

   $k\left(S\right) = \left\lceil \frac{\sum_{i \in S} q_i}{Q} \right\rceil$ is an easily computed lower bound on $r\left(S\right)$

3. *Fractional* capacity inequalities: $x\left(\delta\left(S\right)\right) \geq 2\frac{\sum_{i \in S} q_i}{Q} \forall S \subseteq V_c : \left|S\right| \geq 2$

Ex.:

$Q = 10,\ S = \{1,2,3,4,5\},\ r(S) = 3,\ k(S) = 2,\ \frac{\sum_{i \in S} q_i}{Q} = 1.9$



The Bin Packing Problem

Given:

- n items with weights q_i, for $i = 1, \ldots, n$
- Bins with capacity Q

Compute the minimum number of used bins, subject to:

- Each item is allocated to a bin
- The total weight allocated to each bin does not exceed Q

ILP formulation

$y_i = 1$, if bin i is used, 0 otherwise

$z_{ij} = 1$, if item j is assigned to bin i, 0 otherwise

$$\min \quad \sum_{i=1}^{n} y_i$$

$$s.t.: \quad \sum_{i=1}^{n} z_{ij} = 1 \qquad j = 1, \ldots, n$$

$$\sum_{j=1}^{n} q_j z_{ij} \le Q y_i \qquad i = 1, \ldots, n$$

$$y_i \in \{0,1\} \qquad i = 1, \ldots, n$$

$$z_{ij} \in \{0,1\} \qquad i, j = 1, \ldots, n$$

The cutting plane algorithm

1) Solve the initial LP
2) REPEAT
3) Search for one or more valid inequalities that cut off the current LP solution x^* (= the "separation" / "identification" problem)
4) If at least one violated inequality was found in step 3, then add these inequalities to the LP and resolve the LP
5) UNTIL (stopping criterion is satisfied)

Stopping criteria:

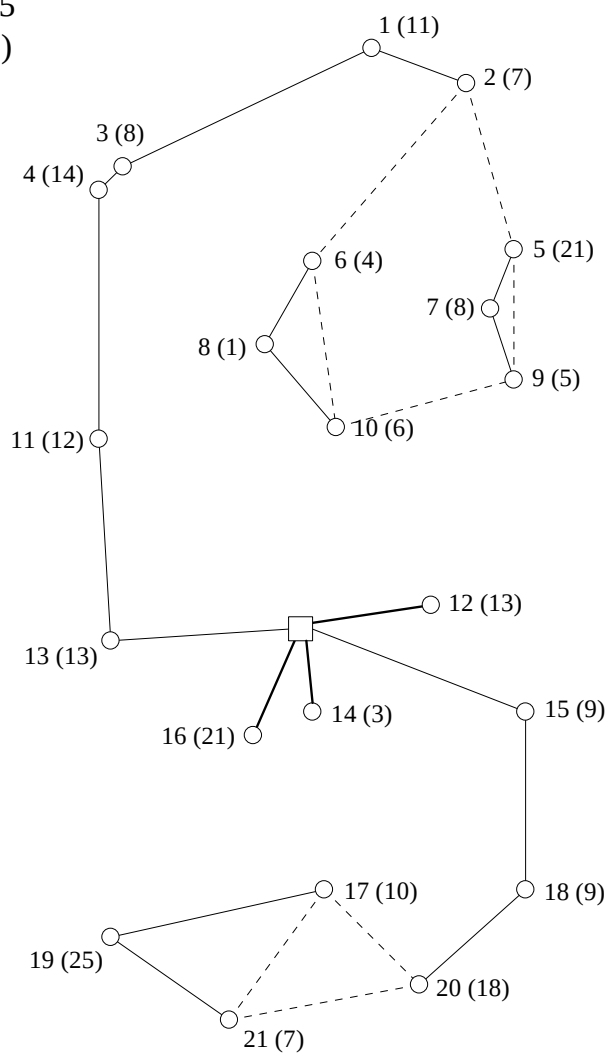
- No inequalities found in step 3
- Tailing off (small change in the objective function value)
- ...

Example: The initial LP solution for E-n22-k4

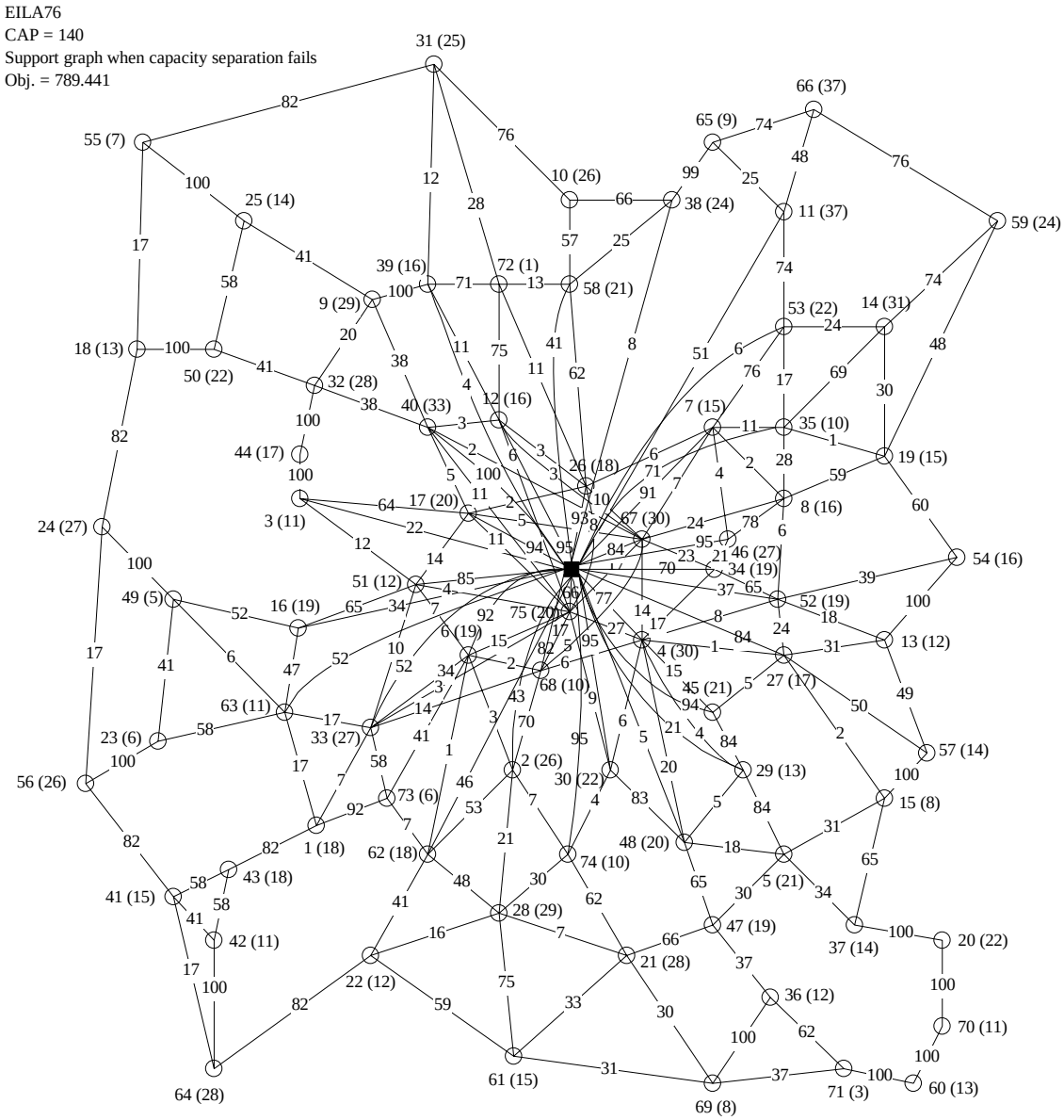
Initial model

Degree equations
Min. 4 routes
Bounds on variables
Obj. = 308.5
(Opt. = 375)

Q = 60



Example: LP solution when separation of rounded capacity inequalities fails for E-n76-k10



Separation routines

We developed separation routines for a number of classes of inequalities:

- rounded capacity inequalities
- homogeneous multistar inequalities
- generalized multistar inequalities
- framed capacity inequalities
- strengthened comb inequalities
- hypotour inequalities

The separation routines were made publicly available in the CVRPSEP package (http://www.hha.dk/~lys/)

Some computational results

Table 10. Results for the B instances.

Instance	UB	Root node LB	Root node Time	Branch & cut Time (LB)	Branch & cut Tree size
B-n31-k5	672*	672*	8	—	—
B-n34-k5	788*	784.25	12	36	19
B-n35-k5	955*	955*	5	—	—
B-n38-k6	805*	801	10	37	21
B-n39-k5	549*	549*	9	—	—
B-n41-k6	829*	827	9	42	24
B-n43-k6	742*	735.417	14	125	63
B-n44-k7	909*	909*	8	—	—
B-n45-k5	751*	748.68	14	46	21
B-n45-k6	678*	673.801	18	299	159
B-n50-k7	741*	741*	11	—	—
B-n50-k8	1313	1281.139	26	31026 (1312*)	5694
B-n51-k7	1032*	1025.571	7	209	122
B-n52-k7	747*	746	8	25	15
B-n56-k7	707*	705.018	20	46	14
B-n57-k7	1153*	1150.092	33	441	168
B-n57-k9	1598*	1589.23	49	1366	264
B-n63-k10	1496*	1481	31	6513	1752
B-n64-k9	861*	860.5	17	42	13
B-n66-k9	1318	1298.509	80	24424 (1316*)	4614
B-n67-k10	1032*	1024.805	28	3309	935
B-n68-k9	1275	1258.054	65	(1267)	5245
B-n78-k10	1221	1205.55	114	87408 (1221*)	8641

Some computational results (2)

Table 7. Results for the E instances.

Instance	UB	Root node		Branch & cut	
		LB	Time	Time (LB)	Tree size
E-n13-k4	247*	247*	4	—	—
E-n22-k4	375*	375*	2	—	—
E-n23-k3	569*	569*	2	—	—
E-n30-k3	534*	534*	14	—	—
E-n31-k7	379*	377.028	11	28	10
E-n33-k4	835*	834.707	12	16	3
E-n51-k5	521*	519	24	59	17
E-n76-k7	682*	666.408	72	118683	8631
E-n76-k8	735*	717.852	136	(729)	2321
E-n76-k10	830	799.878	158	(816)	2209
E-n76-k14	1021	969.609	181	(986)	2081
E-n101-k8	815*	802.646	222	(811)	1621
E-n101-k14	1071	1026.94	555	(1040)	917

A Branch-and-Cut Algorithm for the Capacitated Open Vehicle Routing Problem

Jens Lysgaard
Department of Business Studies
Aarhus School of Business
Aarhus University
Denmark

Adam N. Letchford, Richard W. Eglese
Department of Management Science
Lancaster University
England

(A.N. Letchford, J. Lysgaard & R.W. Eglese: “A branch-and-cut algorithm for the capacitated open vehicle routing problem”, Journal of the Operational Research Society, 2007, vol. 58, pp. 1642-1651)

Capacitated VRPs: Problem definitions

Closed version (CVRP):

1) A depot (indexed 0)
2) A vehicle fleet: K identical vehicles with capacity Q
3) n customers with demands $q_i \leq Q$, for $i = 1,\ldots,n$, to be delivered from the depot
4) Symmetric travel cost c_{ij} between points i and j for all pairs of points
5) Each customer is serviced exactly once
6) The total quantity delivered on each route does not exceed Q
7) Each route begins and ends at the depot
8) Objective: Determine routes of minimum total travel cost

Open version (COVRP):

Replace 7) by 9):

9) Each route begins at the depot and ends at a customer

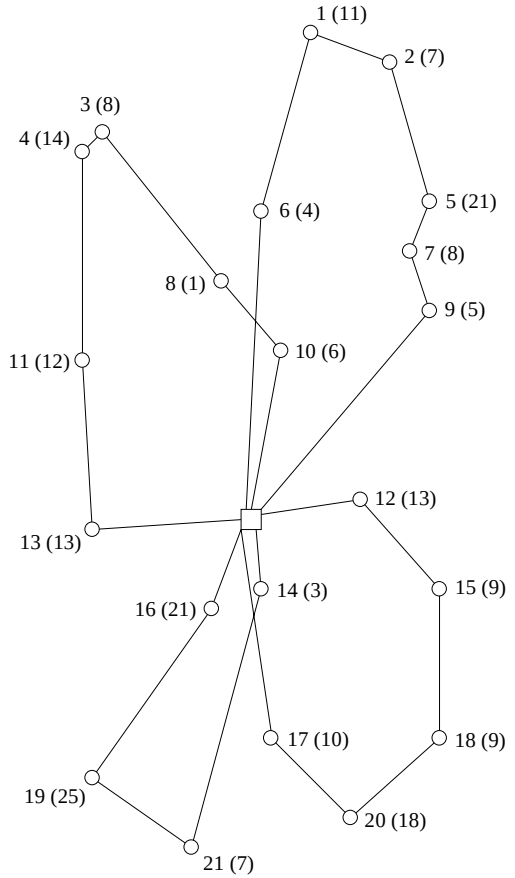
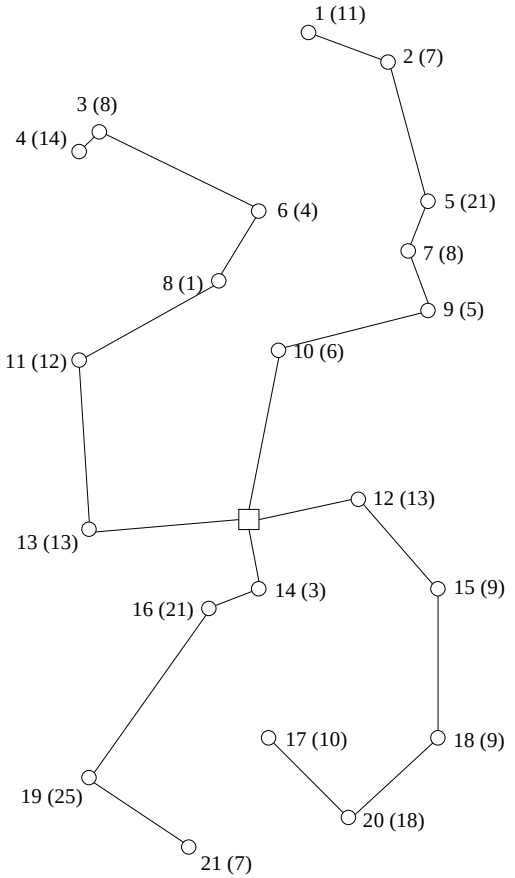
or replace 4) by "partially asymmetric costs" 10a) – 10c):

10a) Symmetric travel cost c_{ij} between customers i and j for all pairs of customers

10b) The "usual" cost from the depot to any customer

10c) A travel cost of <u>zero</u> from any customer to the depot

VRP instance E-n22-k4: Closed and open optimal solutions

Closed	Open
	

Open VRPs: Papers

- Sariklis & Powell: “A heuristic method for the open vehicle routing problem”, JORS, vol. 51, pp. 564-573, 2000.
- Brandão: “A tabu search algorithm for the open vehicle routing problem”, EJOR, vol. 157, pp. 552-564, 2004.
- Fu, Eglese & Li: “A new tabu search heuristic for the open vehicle routing problem”, JORS, vol. 56, pp. 267-274, 2005.
- Tarantilis, Ioannou, Kiranoudis & Prastacos: “Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm”, JORS, vol. 56, pp. 588-596, 2005.
- Li, Golden and Wasil: "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results", C & OR, vol. 34, pp. 2918-2930, 2007.
- Pisinger and Ropke: "A general heuristic for vehicle routing problems", Computers & Operations Research, vol. 34, pp. 2403-2435, 2007.

- All these are concerned with heuristic methods for variants of the open VRP.

Open VRPs: Applications

- Third party logistics
  - Delivery planning with hired vehicles
  - Cost structures which include traveling only to the last customer on a route
- Deliveries in one direction and pickups in the reverse direction
  - Train planning for British Rail freight services through the Channel Tunnel (Fu & Wright, JORS, 1994)
  - School bus routing (Li & Fu, JORS, 2002)

Capacitated Open VRP: Motivation and approach

- Previously no existing exact algorithm for the COVRP

- Planned results from an exact algorithm:
  - Comparison between closed and open version wrt. required computational effort
  - Optimal solutions to test instances for benchmarking of heuristics

Approach:

- Modify a special-purpose algorithm for the (symmetric) Capacitated closed VRP (Lysgaard, Letchford, Eglese: “A new branch-and-cut algorithm for the capacitated vehicle routing problem”, Math.Prog., vol. 100, pp. 423-445, 2004)

Capacitated (closed) VRP
A vehicle flow formulation

x_{ij} denotes the flow of vehicles between points *i* and *j* (the number of times that a vehicle travels along edge $\{i,j\}$):

$$\min \quad \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij}$$

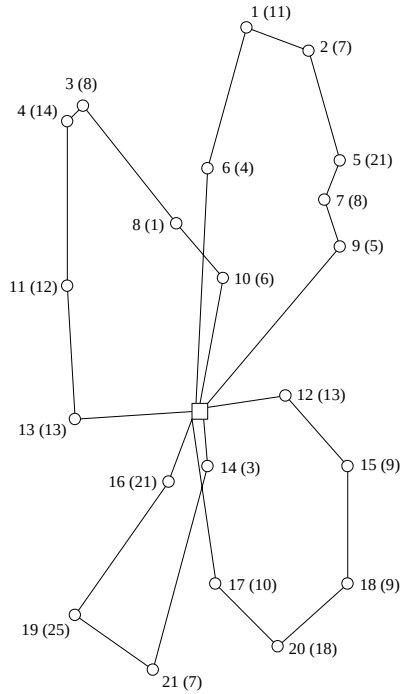
$$s.t. \quad \sum_{j=0}^{n} x_{ij} = 2 \qquad \text{for } i = 1,\ldots,n$$

$$\sum_{i,j \in S,\ i<j} x_{ij} \le |S| - k(S) \qquad \text{for } S \subseteq \{1,\ldots,n\},\ |S| \ge 2$$

$$\sum_{j=1}^{n} x_{0j} = 2K$$

$$x_{ij} \in \{0,1\} \qquad \text{for } i,\ j = 1,\ldots,n$$

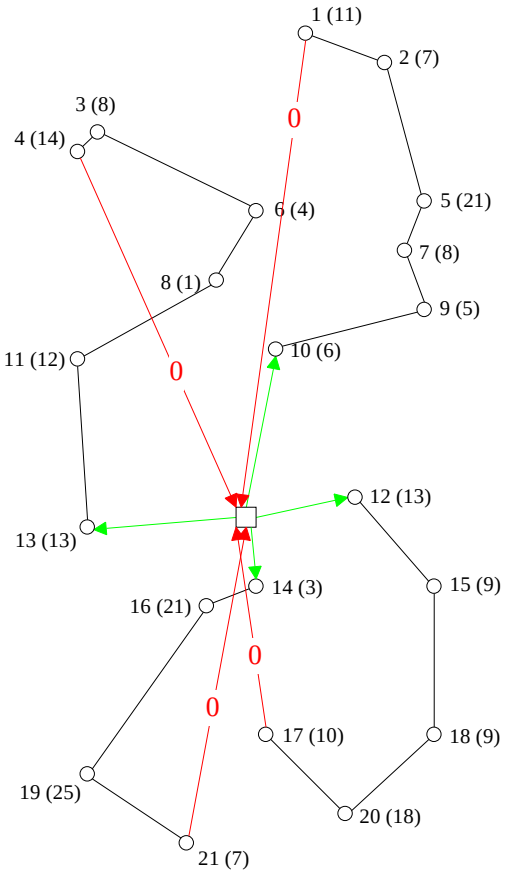
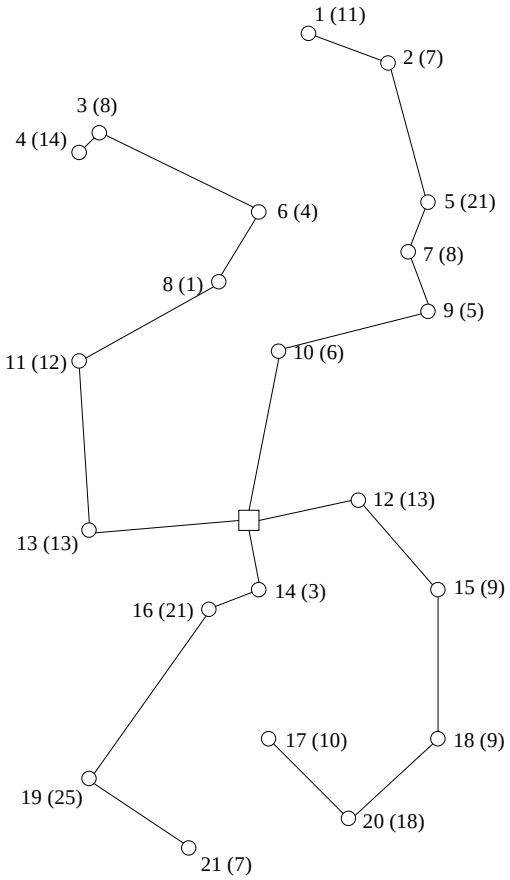
$$x_{0j} \in \{0,1,2\} \qquad \text{for } j = 1,\ldots,n$$



$k(S) = \left\lceil \dfrac{\sum_{i \in S} q_i}{Q} \right\rceil$: A lower bound on the number of vehicles needed to service *S*

Building on an algorithm for the symmetric closed version

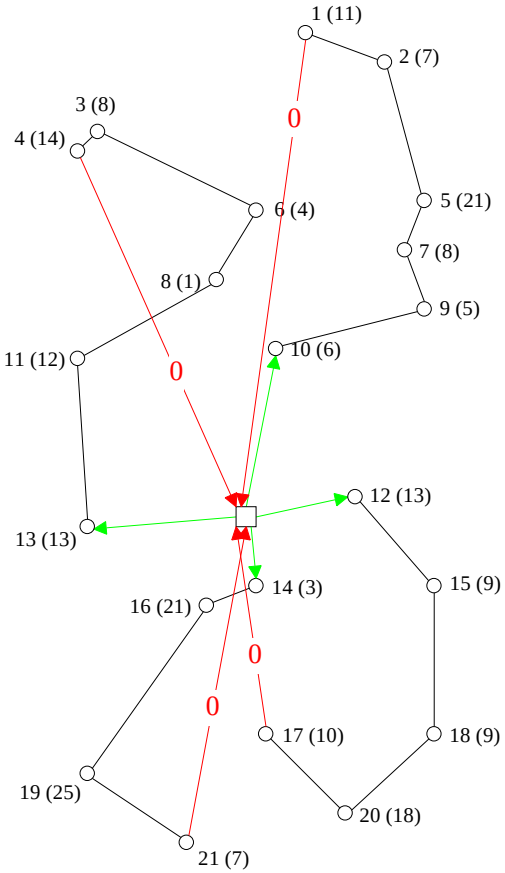
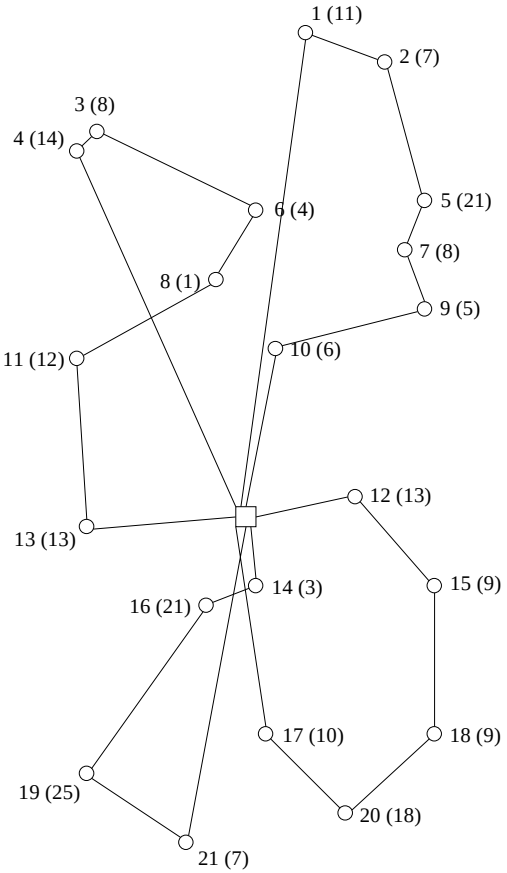
A choice between formulations of the open version when modifying the special-purpose algorithm for the closed version

Partially asymmetric costs	Each route ends at a customer
	

(Degree=2: Computationally attractive)

(Degree=1 or 2)

Relations between open and closed version (1)

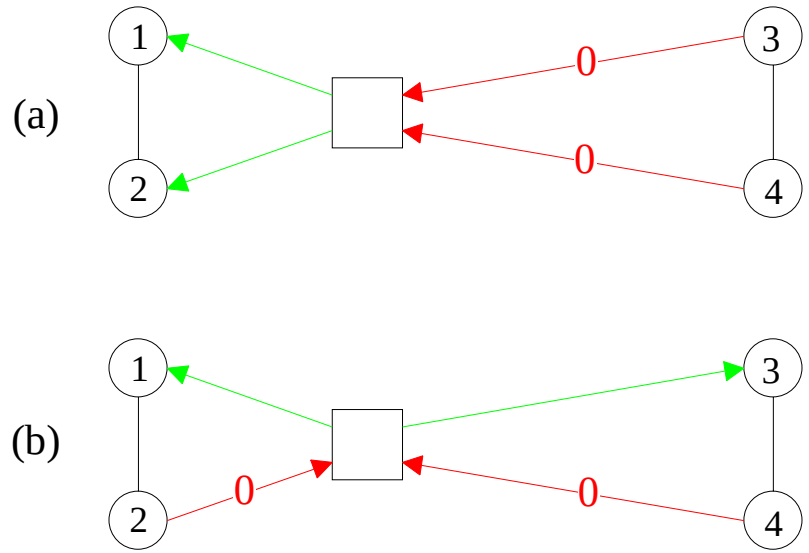
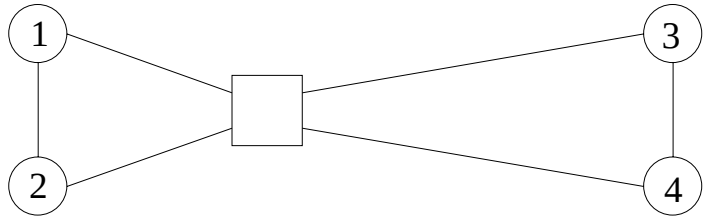
Solution to open version	Solution to closed version
	

Feasible **=>** **Feasible**

So:

Infeasible **<=** **Infeasible**

Relations between open and closed version (2)

Solution to open version	Solution to closed version
	

? **< =** **Feasible**

We need *balancing inequalities* to eliminate (a) and similar infeasible solutions:

For any customer set *S*, green (or red) arcs can account for at most half of the total flow between *S* and the outside of *S*

Our ILP formulation for the Capacitated Open Vehicle Routing Problem

$$\min \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij} + \sum_{i=1}^{n} c_{0i} y_{0i}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} + y_{i0} + y_{0i} = 2 \qquad \text{for } i = 1, \ldots, n \qquad (1)$$

$$\sum_{i,j \in S,\ i<j} x_{ij} \le |S| - k(S) \qquad \text{for } S \subseteq \{1, \ldots, n\},\ |S| \ge 2 \qquad (2)$$

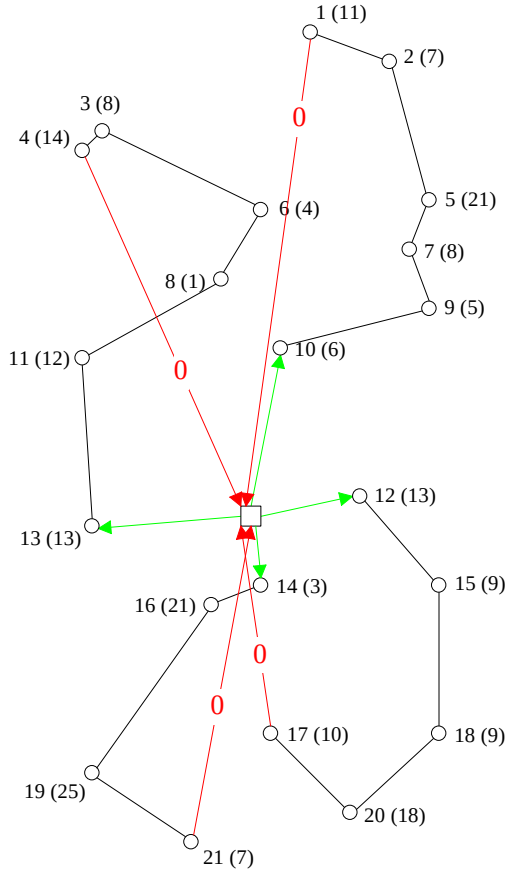
$$\sum_{i \in S} \sum_{j \notin S} x_{ij} + \sum_{i \in S} y_{i0} \ge \sum_{i \in S} y_{0i} \qquad \text{for } S \subseteq \{1, \ldots, n\},\ |S| \ge 2 \qquad (3)$$

$$\sum_{i=1}^{n} y_{0i} = K \qquad (4)$$

$$\sum_{i=1}^{n} y_{i0} = K \qquad (5)$$

$$x_{ij} \in \{0,1\} \qquad \text{for } i, j = 1, \ldots, n \qquad (6)$$

$$y_{0i}, y_{i0} \in \{0,1\} \qquad \text{for } i = 1, \ldots, n \qquad (7)$$



Solve using branch-and-cut…

Branch-and-cut

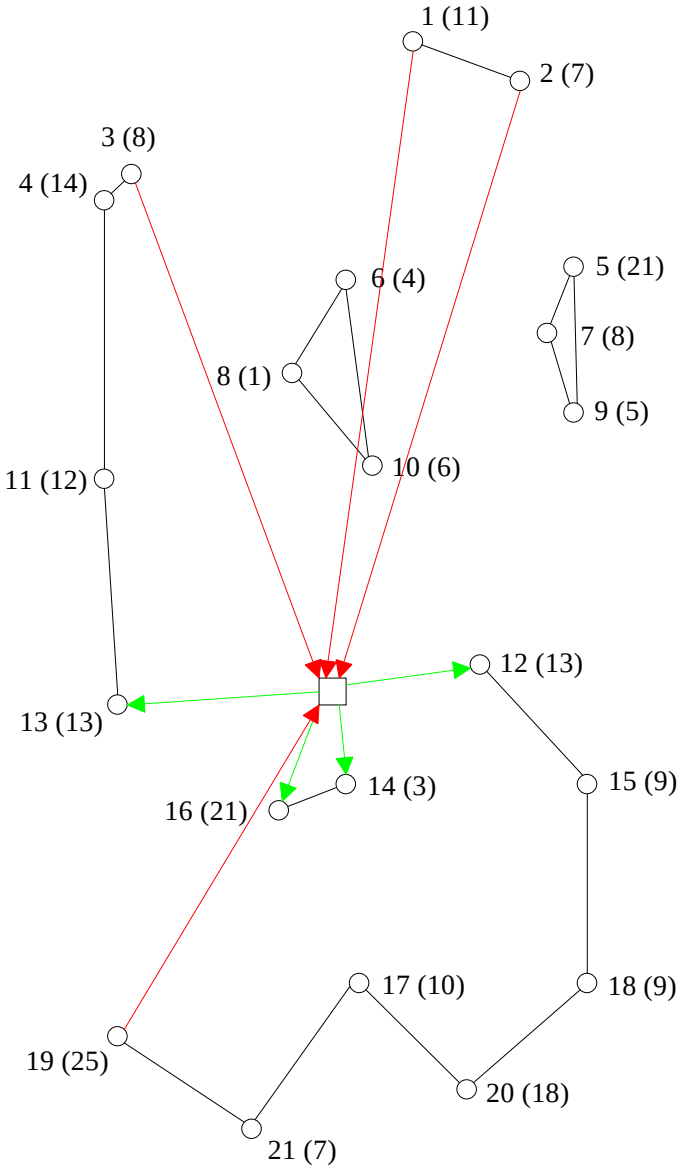
1) Replace integrality constraints with bounds => LP instead of ILP
2) Begin with a small LP containing only degree constraints
3) Iterate through this process:
- Attempt to identify valid cuts which are violated by the current LP solution
- If not cuts are identified, then stop, otherwise add these to the LP, and reoptimize

⇨ **Root LP solution**

4) Apply branch-and-bound, with step 3) at every subproblem

Ingredients:
- Routines for identifying violated cuts of various classes: Capacity, balancing, comb, multistar, framed capacity, and hypotour inequalities
- Branching rules
- Cut pool management
- LP basis reconstruction
- Sparse representation of cuts

E-n22-k4: Initial LP solution



Computational B&C-results
Open vs. closed version

	COVRP		CVRP	
Name	LB	Time	LB	Time
E-n22-k4	252.614*	0.08	375.28*	0.2
E-n23-k3	442.984*	0.3	568.563*	0.05
E-n30-k3	393.512*	1	535.797*	2
E-n33-k4	511.263*	0.6	837.672*	2
E-n51-k5	416.063*	16	524.611*	11
E-n76-k7	530.022*	1103	682.563	3600
E-n76-k8	537.239*	636	733.686	3600
E-n76-k10	559.233	3600	818.655	3600
E-n76-k14	614.442	3600	989.257	3600
E-n101-k8	639.744*	2379	820.552	3600
E-n101-k14	699.985	3600	1049.534	3600
F-n45-k4	463.896*	0.1	723.541*	1
F-n72-k4	176.999*	11	241.974*	4
F-n135-k7	762.894	3600	1162.957*	1587
M-n101-k10	534.239*	89	819.558*	5

(All times in seconds on a PC, 1.6 GHz Intel Pentium M, 512 MB RAM, Windows XP)

Results for all A, B, E, F, M, and P instances in the paper

In general, the open version appears to be a little easier to solve

Computational B&C-results for the COVRP
Assessment of heuristics

Name	Our LB	Brandão	Fu et al.	Li et al.	Pisinger & Ropke
E-n51-k5	416.1*	416.1*	416.1*	416.1*	416.1*
E-n76-k10	559.2	574.5	567.1	567.1	567.1
E-n101-k8	639.7*	641.6	641.9	639.7*	641.8
M-n101-k10	534.2*	535.1	534.7	534.2*	534.2*
M-n121-k7	657.1	683.4	717.2	682.5	682.1
M-n151-k12	730.2	740.8	738.9	733.1	733.1
M-n200-k16	848.5	953.4	---	925.0	896.1
M-n200-k17	847.6	---	879	---	---
F-n72-k4	177.0*	177.4	177.0*	177.0*	177.0*
F-n135-k7	762.9	781.2	777.1	769.7	770.2

A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands

Jens Lysgaard
Department of Business Studies
Aarhus School of Business
Aarhus University
Denmark

Christian H. Christiansen

(C.H. Christiansen & J. Lysgaard:
"A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands", Operations Research Letters, 2007, vol. 35, pp. 773-781)

CVRP vs. CVRPSD: Problem definitions

Capacitated Vehicle Routing Problem (CVRP)	Capacitated Vehicle Routing Problem with Stochastic Demands (CVRPSD)
Given: • A depot • Symmetric travel costs • All routes begin and end at the depot • All routes are planned before any vehicles leave the depot • A fleet of identical vehicles with capacity Q • n customers with deterministic demands $q_i \leq Q$, for $i = 1,\ldots,n$, to be delivered from the depot	Given: • A depot • Symmetric travel costs • All routes begin and end at the depot • All routes are planned before any vehicles leave the depot • A fleet of identical vehicles with capacity Q • n customers with stochastic demands $q_1, q_2, \ldots, q_n$ to be delivered from the depot • Each individual customer's demand becomes known when arriving at the customer
Determine: • Routes of minimum total travel cost	Determine: • Routes of minimum expected total travel cost
Subject to: • Each customer is serviced exactly once • The total demand on each route does not exceed Q	Subject to: • Each customer is serviced exactly once • The total expected demand on each route does not exceed Q

CVRPSD: Failures and strategies

- At the time of planning, the individual customer's demand is given by a probability distribution
- We assume that each individual customer's demand becomes known when arriving at the customer
- We assume that the customers' demands are independent
- It may happen that the actual demand on a route exceeds the vehicle capacity (although the total expected demand on any route does not exceed the vehicle capacity)
  - If this happens, a failure is said to occur
  - A strategy is needed to deal with failures. In particular, it is necessary to know in advance what to do when a failure occurs, if the total expected cost of a route should be calculated

Recourse strategy

- We formulate the CVRPSD as a two stage stochastic program with fixed recourse
- If a failure occurs, the vehicle
  - is depleted at the point of failure,
  - returns to the depot to replenish,
  - and continues the originally planned route from the point of failure.
- The expected cost of a route is composed of two parts:
  - The deterministic cost of following the route, which must be done irrespective of actual demands
  - Expected failure costs, i.e., the total extra distance expected to be traveled due to failure(s) along the route
- The complicating part is the calculation of the expected failure costs

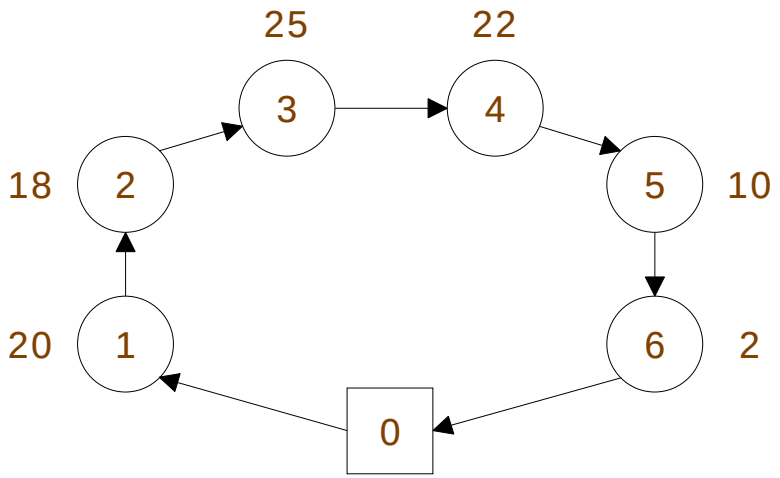
Recourse action: Example

Planned route with expected demands

Vehicle capacity = 100

Total expected demand on this route is 20+18+…+2 = 97

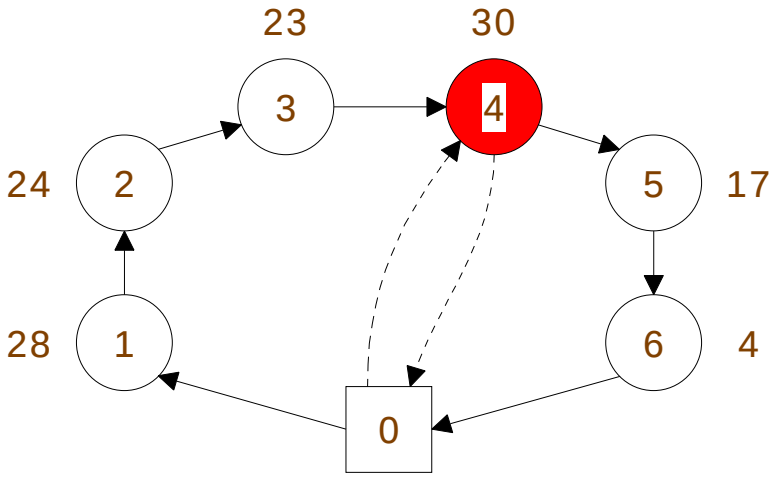
In general, the total expected demand on a route does not exceed the vehicle capacity



Route with actual demands

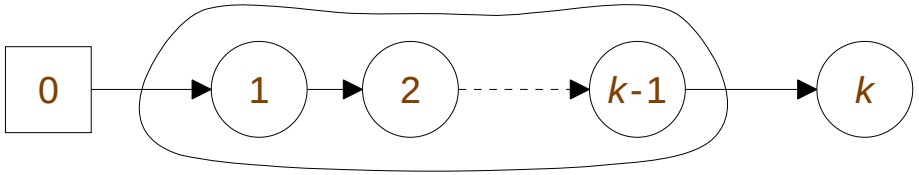
Total actual demand of the first four customers = 28+24+23+30 = 105 > 100, so a failure occurs at customer 4

The recourse action is to deplete the vehicle at the point of failure, return to the depot to replenish, and continue the planned route from the point of failure. The cost of this failure is 2 × d(0,4).



Expected route costs

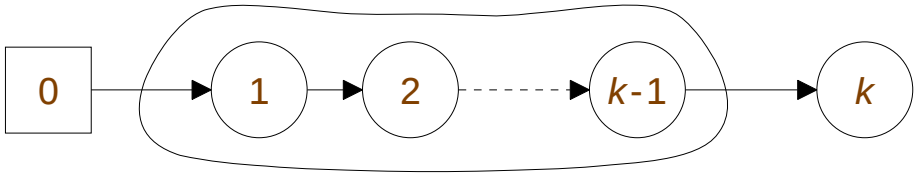
- Given an elementary path (0,…,*k*-1) from the depot (0) to any customer *k*-1, we wish to compute the expected cost of extending the path to customer *k*:



- The expected cost increase is composed of two contributions:
  - The distance traveled from *k*-1 to *k*
  - The increase in expected failure cost; this is given by:

    (The expected number of failures at customer *k*) × 2d(0,*k*)

Expected failure costs



- If at most one failure was possible at each individual customer, then the expected number of failures at customer *k* would be:

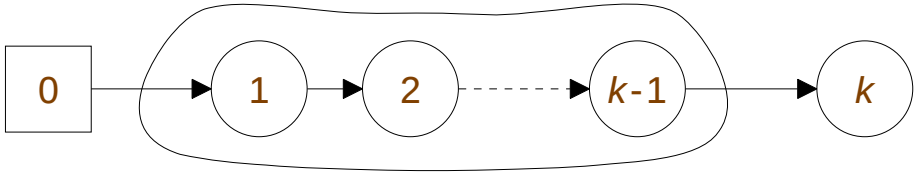
$$P\left(\sum_{i=1}^{k-1} q_i \le Q < \sum_{i=1}^{k} q_i\right) = P\left(\sum_{i=1}^{k-1} q_i \le Q\right) - P\left(\sum_{i=1}^{k} q_i \le Q\right)$$

- However, in general, more than one failure is possible, resulting in the following expected number of failures at customer *k*:

$$\sum_{u=1}^{\infty}\left[P\left(\sum_{i=1}^{k-1} q_i \le uQ\right) - P\left(\sum_{i=1}^{k} q_i \le uQ\right)\right]$$

  This is composed of a contribution from the first failure ($u=1$), the second failure ($u=2$), etc., at customer *k*

Demand properties



- For the purpose of keeping the calculations tractable, we make the following assumption on the distribution of the demands:
  - The demands of the individual customers are independent
  - The demand distributions are such that the sum of two or more independent demand variables follow the same type of distribution as the individual variables (with different parameters)
    - For example, the sum of two independent Poisson distributed variables follows a Poisson distribution
    - The same holds for, e.g., the Normal distribution

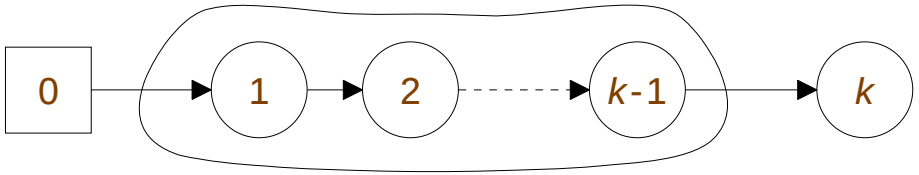
- Using this property, it is more manageable to compute the required probabilities, e.g.,

$$P\left(\sum_{i=1}^{k} q_i \leq Q\right)$$

may be viewed as being concerned with only a single stochastic variable, namely

$$\sum_{i=1}^{k} q_i$$

Demand properties (2)



- It follows that when we consider the extension of the path (0,…,k-1) to customer k, then we don't need to know how the total expected demand along (0,…,k-1) is divided among the customers on the path. This is an important observation for our algorithm.

- In our experiments we assume that all demands are Poisson distributed.

- As such, when computing shortest paths by dynamic programming, we can use labels (q,j) to represent all paths (0,…,j) on which the total expected demands equals q.

A Set Partitioning Formulation

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \in \{0,1\} \quad \forall r \in R \qquad (3)$$

where:

- R is the set of feasible routes
- $x_r = 1$, if route r is used, 0 otherwise
- c_r is the expected cost of route r
- $a_{ir} = 1$, if customer i is serviced on route j, 0 otherwise

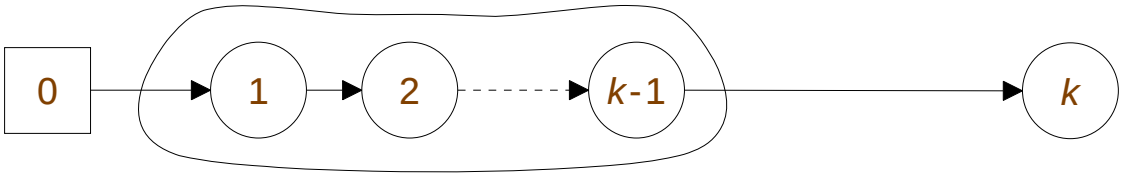
All complications resulting from stochastic demands are embedded in the calculation of the expected costs c_r

The subproblem

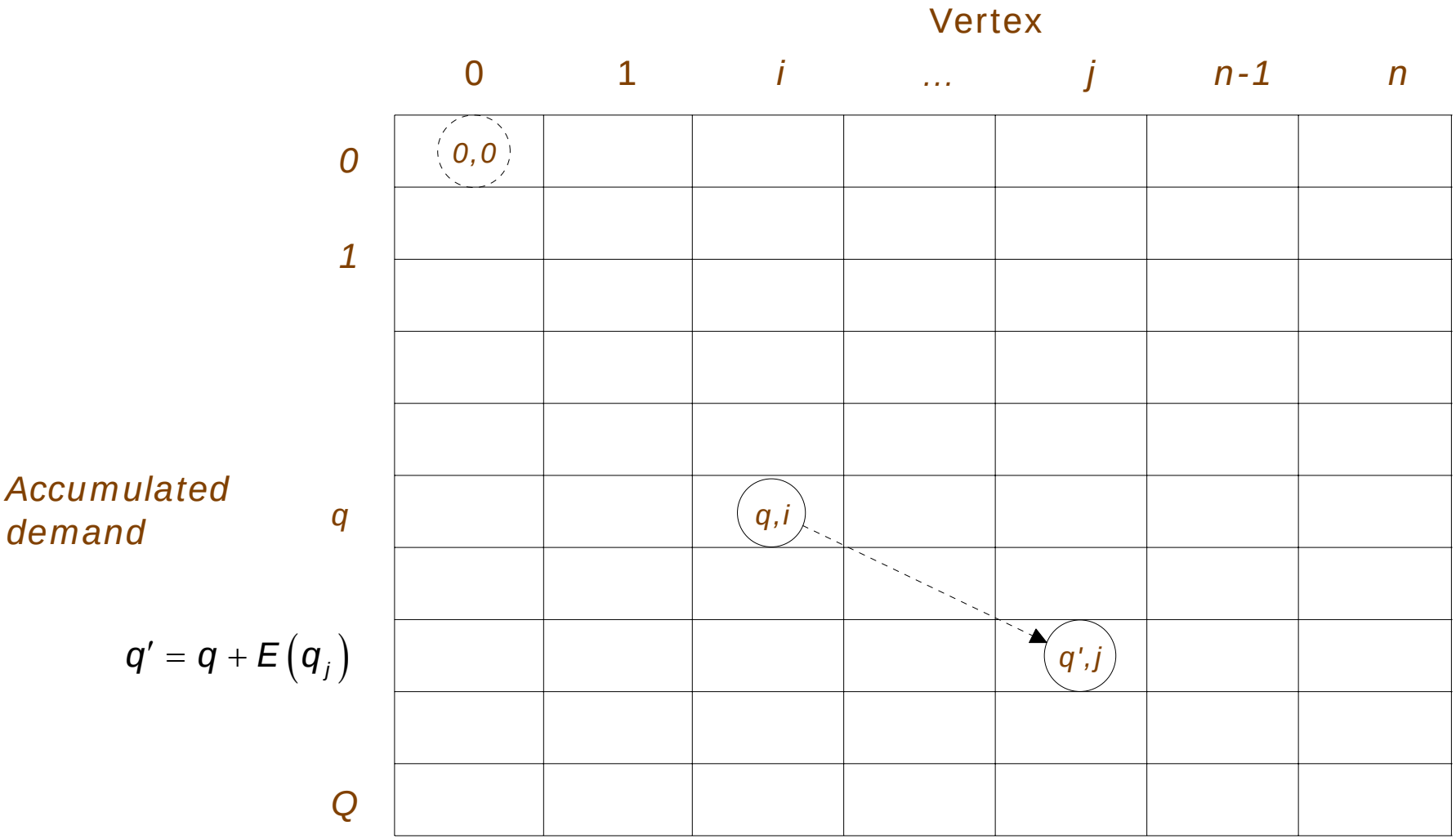
The subproblem can be represented in a network, where the reduced cost of a route equals the length of the route in the network:

The cost of arc (k-1,k) is:

$$d_{k-1,k} + \left(\text{The expected number of failures at customer } k\right) \times 2d\left(0,k\right) - \pi_k$$



Dynamic programming for solving the subproblem



- We can determine the cost of the arc shown, without knowing the path leading up to (q,i)

Branching

- Branching on variables in the master problem makes the subproblem more difficult
- Instead, we branch on the underlying information:
  - Branching on flow variables (e.g., require or probihit customer *j* as successor of customer *i*)
  - Branching on resources, which in this case amounts to branching on the accumulated expected demand up to and including customer *i* on any path (0,…,*i*)

Computational experience

- Stochastic vs. deterministic demands:
  - Up to 10% difference in cost
  - Extra routes may be formed in the stochastic case

- Column generation
  - Solved instances with up to 60 customers and 16 routes
  - Works better when capacity restrictions are tight

- Resource-based branching
  - Promising branching strategy
  - Searches fewer nodes than does the branching strategy based on arc flows

The Set Partitioning Model for Vehicle Routing

Jens Lysgaard
Department of Business Studies
Aarhus School of Business, University of Aarhus
Denmark

(May 2010)

The Vehicle Routing Problem (VRP)

Given:

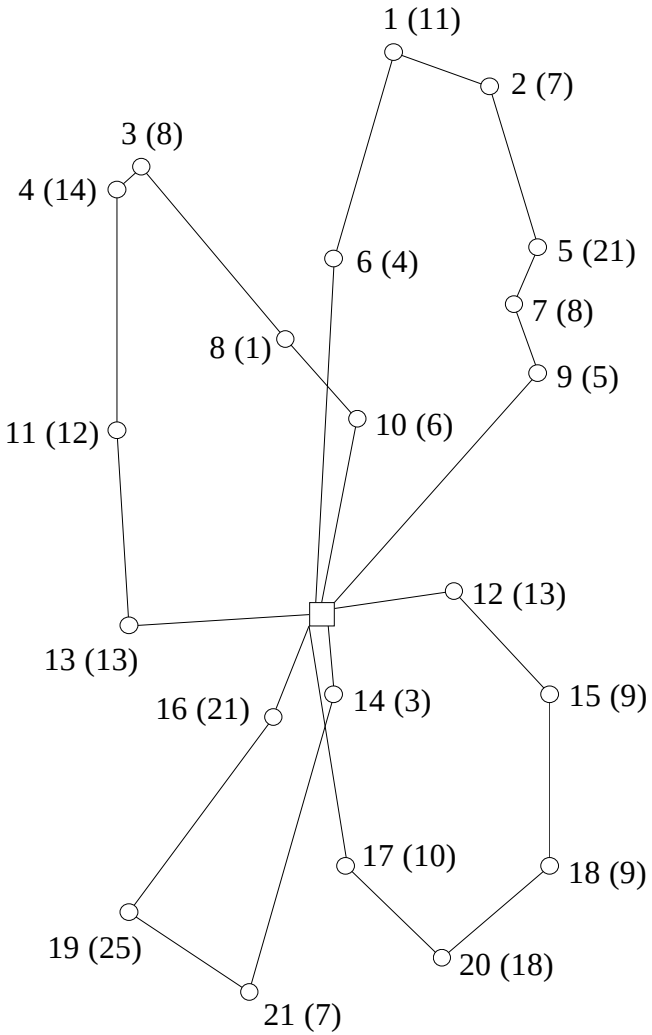
- A number of customers
- The demand of each customer

- A depot, from where all demands must be delivered
- A fleet of identical vehicles at the depot
- Each vehicle has a given capacity

- The cost of travel between each pair of points

Problem:

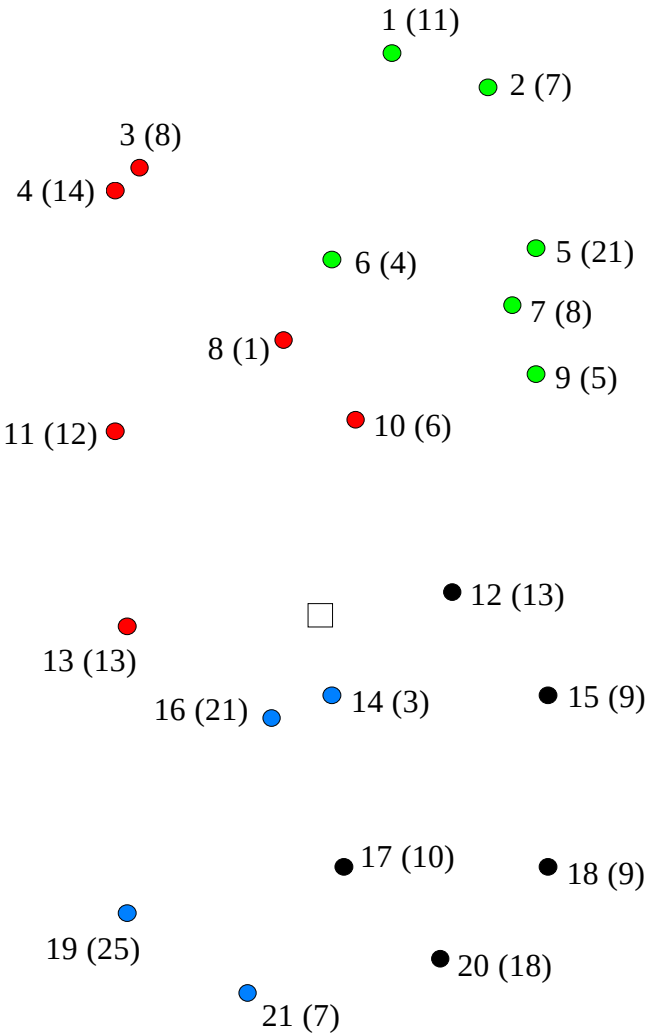
- Find a collection of routes, each beginning and ending at the depot, such that:
  - Each customer is served exactly once
  - The capacity of any vehicle is not exceeded
  - The total travel cost is minimized



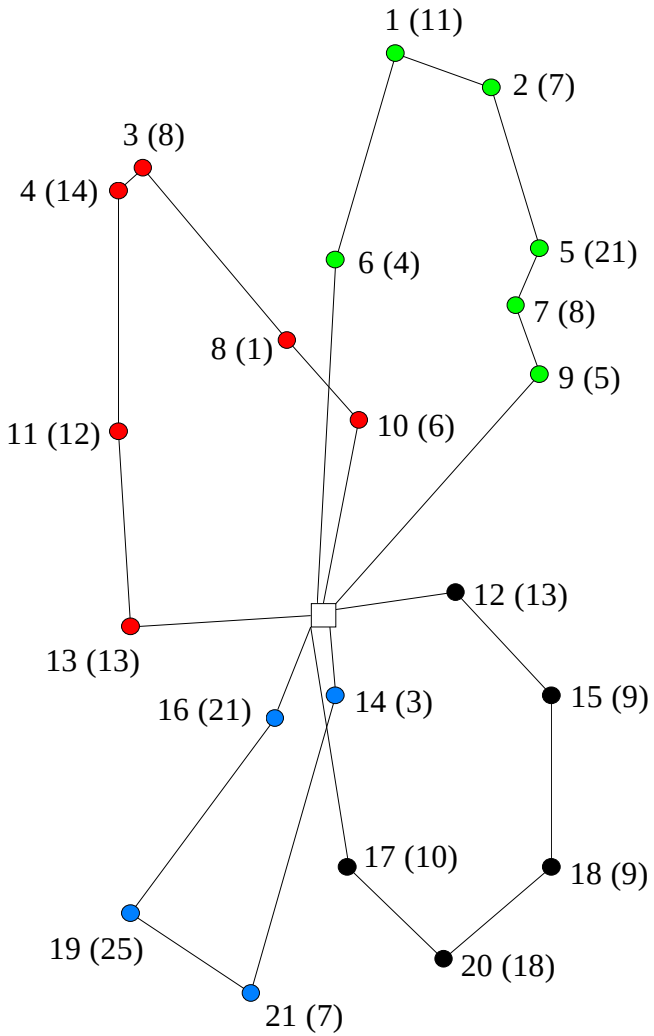
The Vehicle Routing Problem (VRP)

Creating a solution involves two decisions:

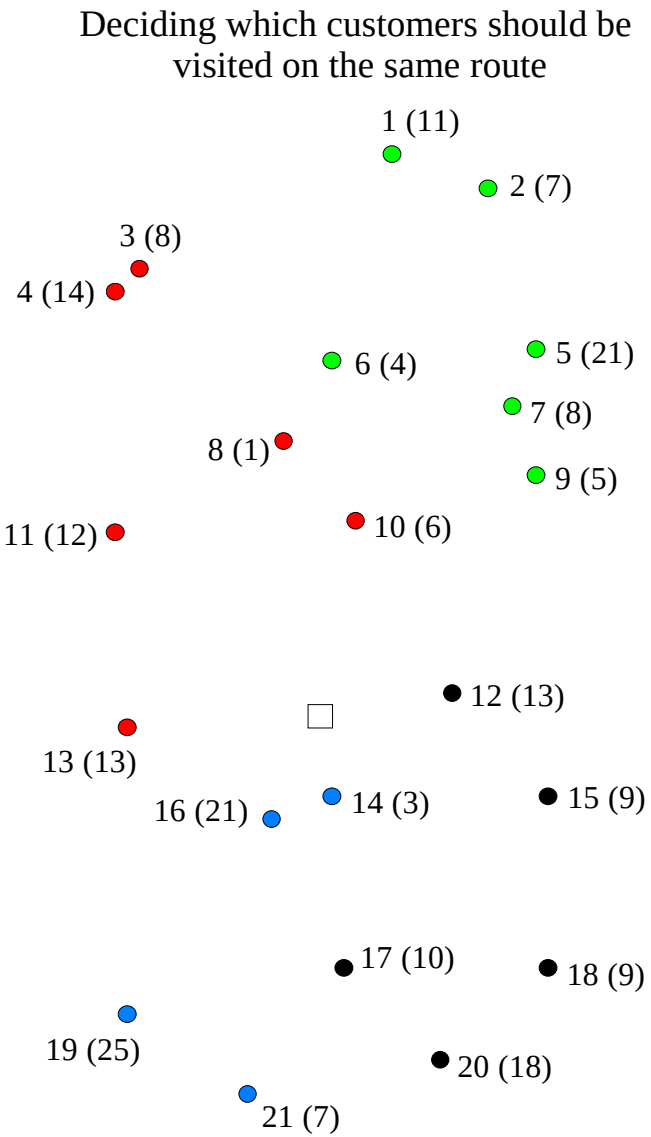
Deciding which customers should be visited on the same route



Deciding the order of visit within each route



The Set Partitioning Problem



The Set Partitioning Problem (SPP):

Determine a partition of the set of all customers into *clusters* (from a predefined list of clusters) of minimum total cost, i.e.:

1) Every customer must be placed in exactly one cluster;
2) The total cost of the clusters must be the minimum possible.

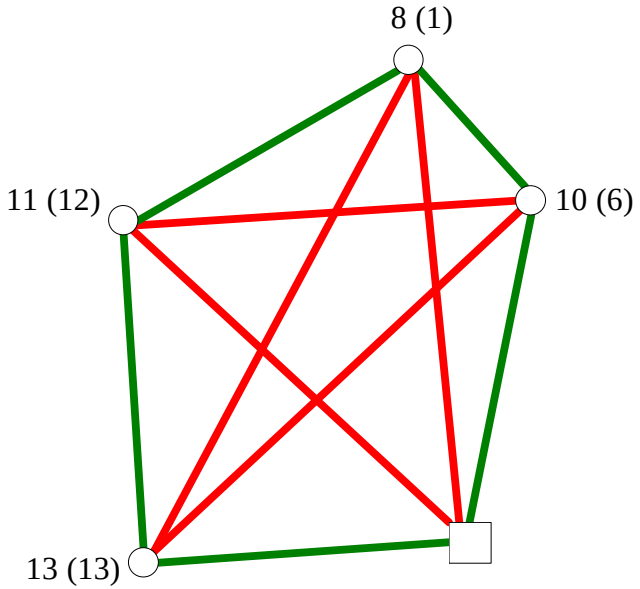
(customers in the same cluster get the same colour; customers in different clusters get different colours)

Required problem structure for using the SPP:

a) It must be possible to assign a specific cost to every possible cluster;
b) Every customer must be visited exactly once.

Required problem structure for using The Set Partitioning Problem

a) It must be possible to assign a specific cost to every possible cluster

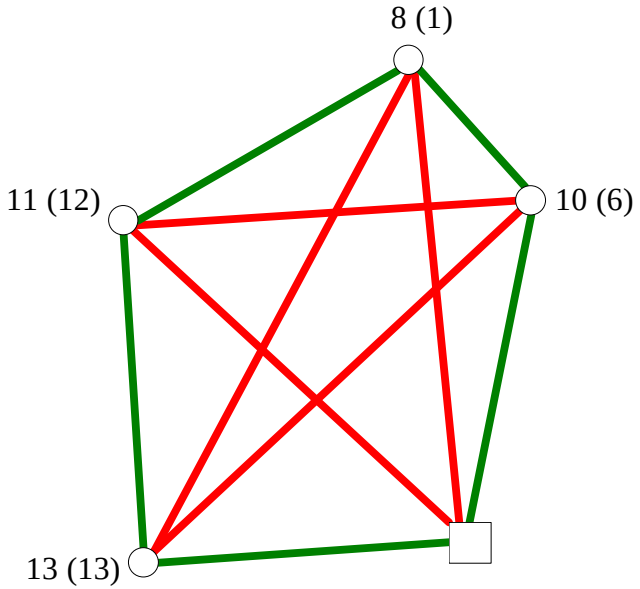


Cost of {8,10,11,13} is the total length of the green lines, if the measure of cost is the total distance driven

(use the cost of the minimum-cost route, starting and ending at the depot, for every set of customers)

Required problem structure for using The Set Partitioning Problem

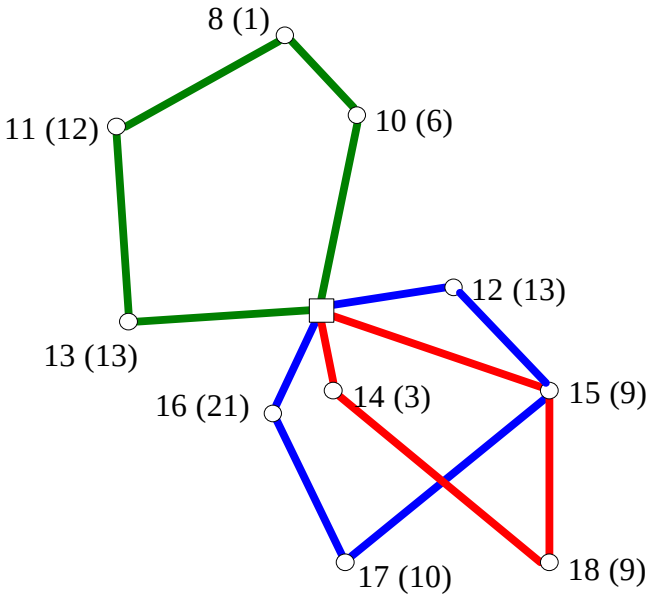
a) It must be possible to assign a specific cost to every possible cluster



Cost of {8,10,11,13} is the total length of the green lines, if the measure of cost is the total distance driven

(use the cost of the minimum-cost route, starting and ending at the depot, for every set of customers)

b) Every customer must be visited exactly once



The red and the blue route cannot both be used

(split deliveries must not be allowed)

A theoretical approach to solving the VRP using the SPP

1. Make a list of all feasible clusters
2. Compute the cost of each feasible cluster
3. Solve the corresponding Set Partitioning Problem, i.e.:

   Choose a collection of feasible clusters, such that every customer is contained in exactly one cluster, and such that the total cost of the chosen clusters is minimum

Theoretical approach: A small example

Suppose there are only three customers, and that every cluster of customers is feasible, except the one with all three customers

		Cluster number					
		1	2	3	4	5	6
Customer	1	1			1	1	
	2		1		1		1
	3			1		1	1

Cluster cost	c_1	c_2	c_3	c_4	c_5	c_6
Use this cluster (1=use, 0=don't use)?	x_1	x_2	x_3	x_4	x_5	x_6

Integer Linear Programming formulation (Set Partitioning Problem):

$$\min \quad c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_5 + c_6x_6$$

$$s.t. \quad x_1 + x_4 + x_5 = 1 \quad \text{(place customer 1 in one cluster)}$$

$$x_2 + x_4 + x_6 = 1 \quad \text{(place customer 2 in one cluster)}$$

$$x_3 + x_5 + x_6 = 1 \quad \text{(place customer 3 in one cluster)}$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \in \{0,1\}$$

A theoretical approach to solving the VRP using the SPP, reviewed

1. Make a list of all feasible clusters
2. Compute the cost of each feasible cluster
3. Solve the corresponding Set Partitioning Problem, i.e.:

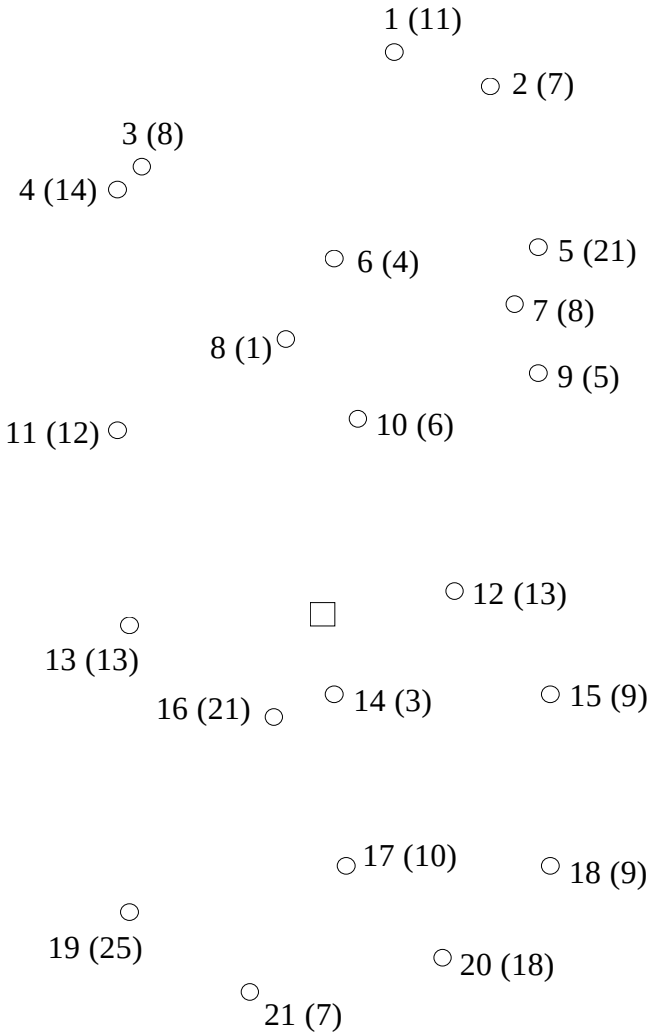
   Choose a collection of feasible clusters, such that every customer is contained in exactly one cluster, and such that the total cost of the chosen clusters is minimum

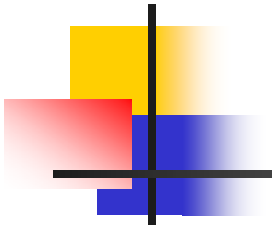
1. Difficult/impossible in practice
2. Requires the solution of a TSP for each cluster
3. Very difficult

A list of all feasible clusters? Too many feasible clusters!

Number of customers	Number of clusters
1	1
2	3
3	7
4	15
5	31
6	63
7	127
8	255
9	511
10	1023
11	2047
12	4095
13	8191
14	16383
15	32767
16	65535
17	131071
18	262143
19	524287
20	1048575
21	**2097151**
22	4194303
23	8388607
24	16777215
25	33554431

Out of 2097151 clusters, 68292 are feasible in this example (*feasible*: the demand in the cluster does not exceed the vehicle capacity)





The solution to the ”too many clusters” problem

Allow variables to take fractional values (=> LP model + cost allocation using dual prices)

Consider only a few clusters *explicitly* at a time:
Explicit clusters: Those that are included in the LP model
Implicit clusters: Existing clusters, which are currently not included in the LP model

a) Solve the LP model using only explicit clusters
b) [*Column generation*] Use the allocated customer costs (dual prices) in the LP solution to find an implicit cluster, which might improve the LP solution, if it is made explicit. If none exists, stop (we have the optimal LP solution). Otherwise make it explicit and go to a).

The column generation step can be formulated as an *optimization problem*:

Find, among all implicit clusters, the cluster which has the largest potential for improving the current LP solution. If the largest potential is zero (or negative), we are done.
(*potential*: (Sum of customer costs in the cluster) minus (the cost of the cluster) = the negative of the reduced cost in minimization LP)

The efficiency of the column generation step is generally very important.

The solution to the ”too many clusters” problem

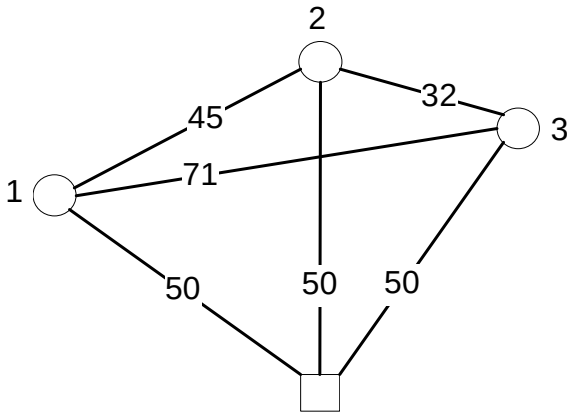
Allow variables to take fractional values (=> LP model + cost allocation using dual prices)

Consider only a few clusters *explicitly* at a time:
Explicit clusters: Those that are included in the LP model
Implicit clusters: Existing clusters, which are currently not included in the LP model

a) Solve the LP model using only explicit clusters
b) [*Column generation*] Use the allocated customer costs (dual prices) in the LP solution to find an implicit cluster, which might improve the LP solution, if it is made explicit. If none exists, stop (we have the optimal LP solution). Otherwise make it explicit and go to a).

		Cluster number					
		1	2	3	4	5	6
Customer	1	1			1	1	
	2		1		1		1
	3			1		1	1
Cluster cost		100	100	100	145	171	132
Use?		0	0	1	1	0	0

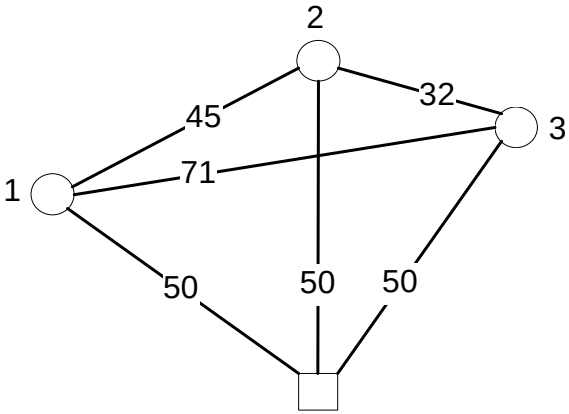
Customer cost
45
100
100
Total=245



Column generation

		Cluster number						Customer cost
		1	2	3	4	5	6	
Customer	1	1			1	1		45
	2		1		1		1	100
	3			1		1	1	100
Cluster cost		100	100	100	145	171	132	
Use?		0	0	1	1	0	0	Total=245

Make cluster 6 explicit!

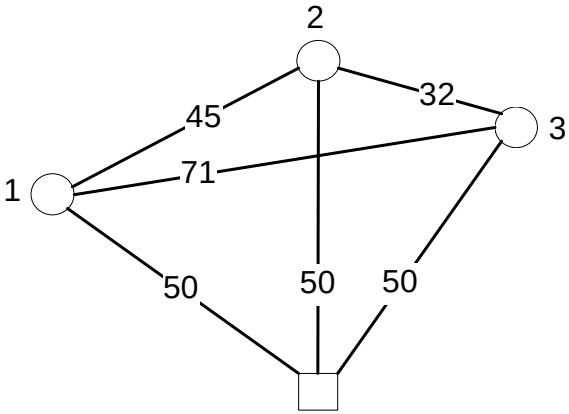


		Cluster number						Customer cost
		1	2	3	4	5	6	
Customer	1	1			1	1		100
	2		1		1		1	45
	3			1		1	1	87
Cluster cost		100	100	100	145	171	132	
Use?		1	0	0	0	0	1	Total=232

Column generation

		Cluster number						Customer cost
		1	2	3	4	5	6	
Customer	1	1			1	1		100
	2		1		1		1	45
	3			1		1	1	87
Cluster cost		100	100	100	145	171	132	Total=232
Use?		1	0	0	0	0	1	

Make cluster 5 explicit!



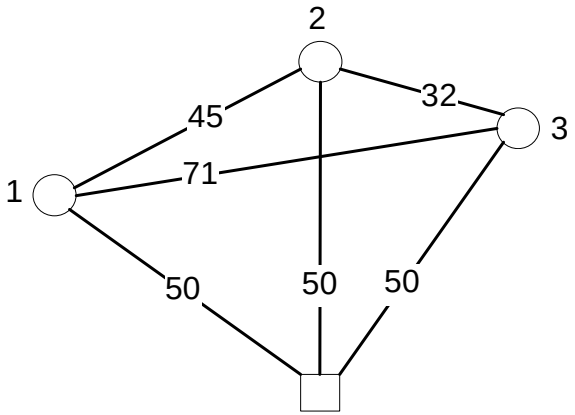
		Cluster number						Customer cost
		1	2	3	4	5	6	
Customer	1	1			1	1		92
	2		1		1		1	53
	3			1		1	1	79
Cluster cost		100	100	100	145	171	132	Total=224
Use?		0	0	0	0.5	0.5	0.5	

Optimal LP solution!

Actions after the optimal LP solution is found

Optimal LP solution:

		Cluster number						Customer cost
		1	2	3	4	5	6	
Customer	1	1			1	1		92
	2		1		1		1	53
	3			1		1	1	79
Cluster cost		100	100	100	145	171	132	Total=224
Use?		0	0	0	0.5	0.5	0.5	



Possible continuations:

a) Try to obtain the optimal *integer* solution

b) Use the optimal LP solution as a lower bound on the optimal integer solution, e.g., to validate a heuristic solution

Note:

1) Integer solutions may be found during the iterations until the LP optimum

2) Which procedure to use for *column generation*, i.e., to find an attractive implicit cluster (or to prove that none exists), is an open question

Jens Lysgaard / May 2010

The Set Partitioning formulation of the CVRP

The CVRP (Capacitated Vehicle Routing Problem) can be formulated as a Set Partitioning Problem (SPP), i.e., it can be written in the following form:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \in \{0,1\} \quad \forall r \in R \qquad (3)$$

where:

- R is the set of feasible routes
- $x_r = 1$, if route r is used, 0 otherwise
- c_r is the cost of route r
- $a_{ir} = 1$, if customer i is serviced on route j, 0 otherwise

A route is feasible, if (and only if) 1) it starts and ends at the depot, 2) visits each customer at most once, and 3) the total demand on the route does not exceed the vehicle capacity

Jens Lysgaard / May 2010

The LP relaxation of SPP

SPP:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \in \{0,1\} \quad \forall r \in R \qquad (3)$$

LP relaxation:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \geq 0 \quad \forall r \in R \qquad (4)$$

The LP model given by (1), (2), and (4) typically contains a large number of variables. Instead of including them all explicitly, the model can be solved by successively including variables (columns) in the model. The included variables are found by *column generation*.

Jens Lysgaard / May 2010

The LP relaxation of SPP

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \geq 0 \qquad \forall r \in R \qquad (4)$$

The reduced cost $\bar{c}_r$ of the variable x_r can be computed as follows:

$$\bar{c}_r = c_r - \sum_{i=1}^{n} a_{ir} \pi_i$$

where π_i is the dual price (shadow price) for the *i*'th constraint, which expresses that customer *i* must be visited on exactly one route.

In an optimal LP solution, all reduced costs are non-negative:

- If the reduced cost is non-negative for all those variables that have not been generated, the current LP solution is optimal.
- Otherwise, i.e., if one or more variables – among those that have not yet been included in the LP model – have a negative reduced cost, then one or more of these variables are included in the LP, which is then reoptimized. This is done until there are no more variables with negative reduced cost.

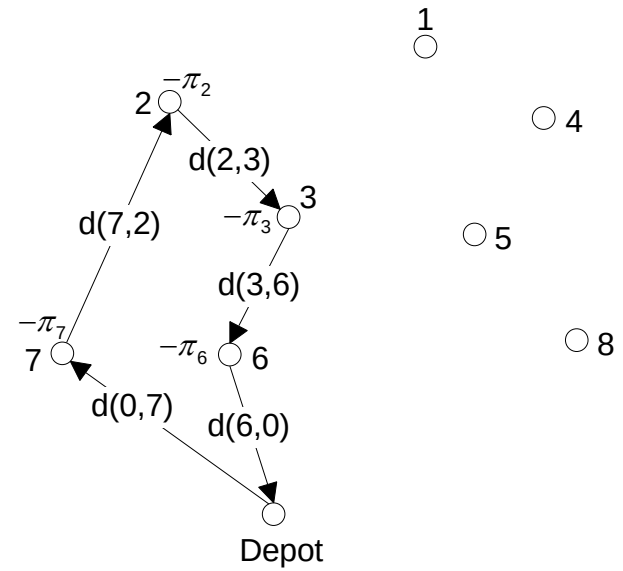
Jens Lysgaard / May 2010

The LP relaxation of SPP
The column generation problem

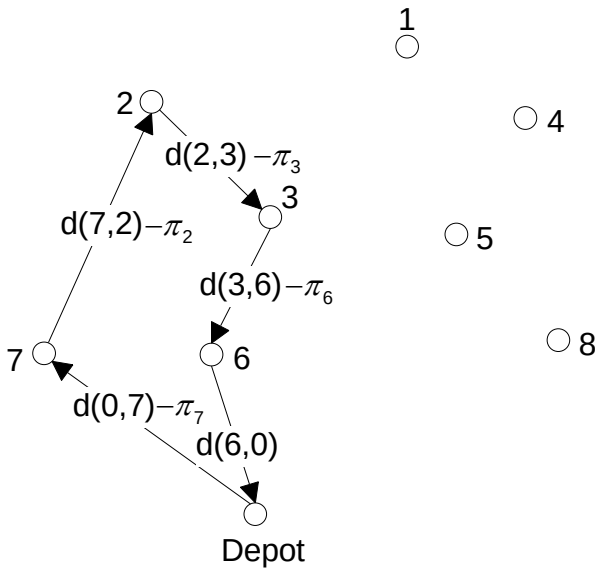
"Given dual prices $\pi_1,\ldots,\pi_n$, find a feasible route with minimum reduced cost."

This is also known as *the subproblem*.

The subproblem can be represented in a network, where the reduced cost of a route equals the length of the route in the network:



(the dual price corresponds to a prize of visiting the customer)



(red. cost = the total length of the route in this network)

Jens Lysgaard / May 2010

The LP relaxation of SPP
Solving the subproblem

1) Construct a network with the points 0,1,…,n, where point 0 represents the depot
2) Let the length of the arc from i to j equal $d_{ij} - \pi_j$, where d_{ij} is the cost of traveling directly from i to j, and where $\pi_0 = 0$
3) Find, in this network, the shortest route from 0 to 0, subject to the following:
   a) The route can visit each customer at most once
   b) The total demand on the route can not exceed the vehicle capacity

In practical computations it may be more efficient to permit routes on which some customers are visited more than once. This leads to the following modified LP, where a_{ir} is replaced by α_{ir}:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} \alpha_{ir} x_r = 1 \quad i = 1,\ldots,n \qquad (5)$$

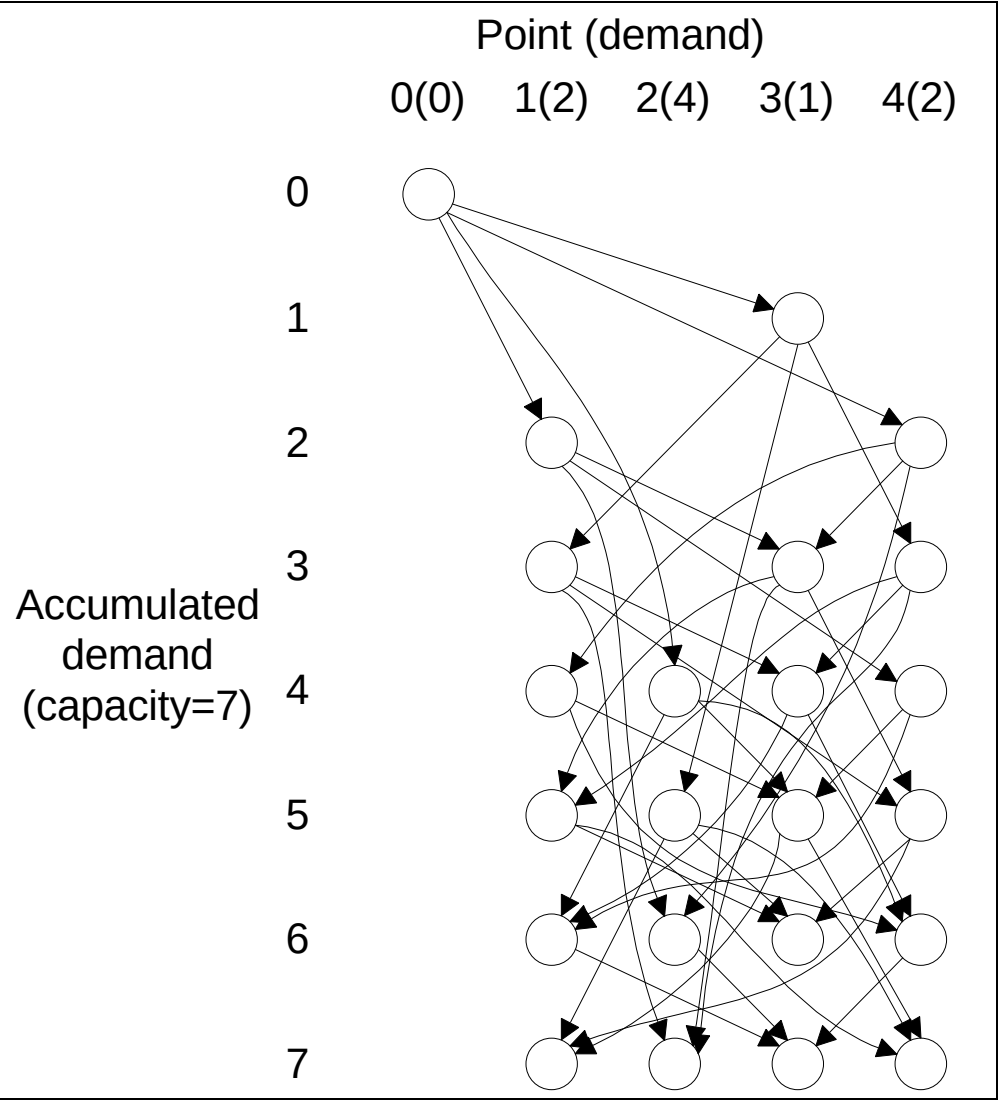
$$x_r \geq 0 \qquad \forall r \in R \qquad (4)$$

where α_{ir} denotes the number of visits at customer i on route r. The subproblem for this LP is as the above, except that it does not contain constraint 3.a) and it is thereby easier to solve. Multiple (k) visits at the same customer count as visiting k copies of that particular customer. (Routes with $\alpha_{ir} \geq 2$ are eliminated by branching)

Jens Lysgaard / May 2010

The LP relaxation of SPP
Solving the subproblem by dynamic programming

- The network has $nQ+1$ points, where Q is the vehicle capacity
- Point (t, j) represents paths from the depot to j with a total demand of t
- The network contains an arcs from (t, i) to $(t+q_j, j)$ with cost $d_{ij}-\pi_j$, for all combinations of (t, i, j) satisfying $t+q_j \le Q$
- L_{tj} denotes the length of the shortest path from $(0,0)$ to point (t, j). As such, $L_{ij}+d_{j0}$ is the reduced cost for the minimum reduced cost route from depot to depot, with a total demand of t and with j as the last customer
- Because the network is acyclic, shortest paths can be computed also with negative arc costs



Jens Lysgaard / May 2010

The LP relaxation of SPP
Duality

The LP relaxation of SPP:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{r \in R} a_{ir} x_r = 1 \quad i = 1, \ldots, n \qquad (2)$$

$$x_r \geq 0 \qquad \forall r \in R \qquad (4)$$

$$\bar{c}_r = c_r - \sum_{i=1}^{n} a_{ir} \pi_i$$

The dual problem:

$$\max \quad \sum_{i=1}^{n} \pi_i \qquad (D.1)$$

$$s.t.: \quad \sum_{i=1}^{n} a_{ir} \pi_i \leq c_r \quad \forall r \in R \qquad (D.2)$$

Adding a column (variable) in the LP relaxation of SPP corresponds to adding a constraint in the dual problem

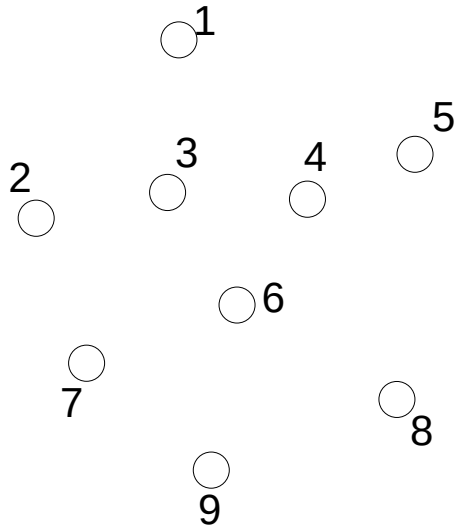
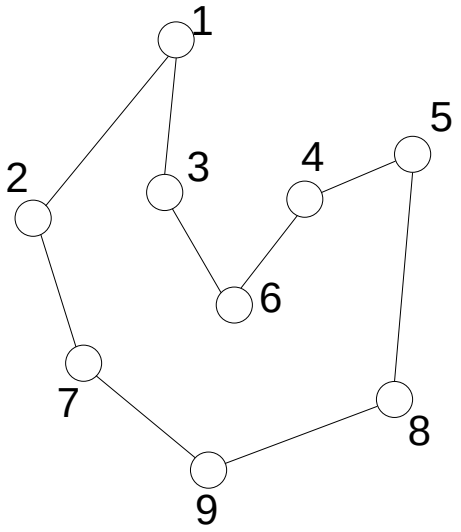
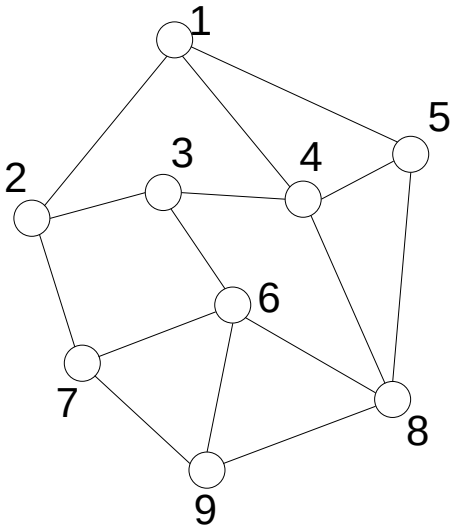
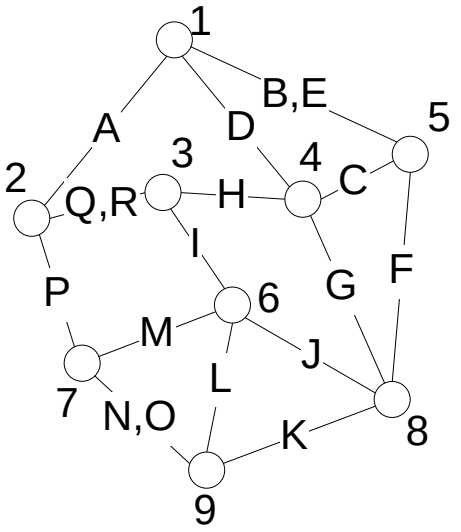
A Branch-and-Price Algorithm for the Capacitated Arc Routing Problem with Stochastic Demands

Jens Lysgaard & Sanne Wøhlk
Department of Business Studies
Aarhus School of Business
Aarhus University
Denmark

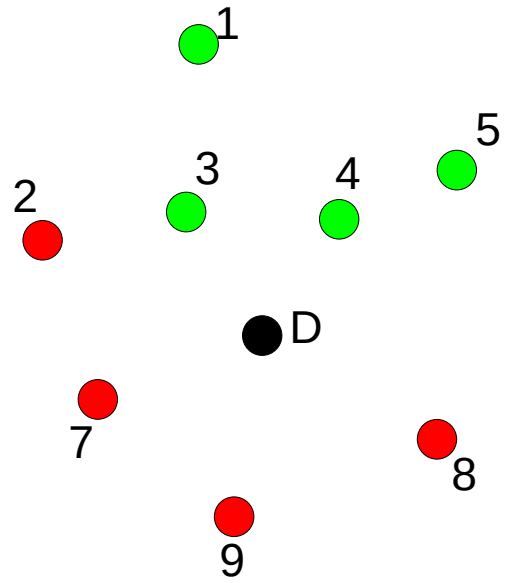
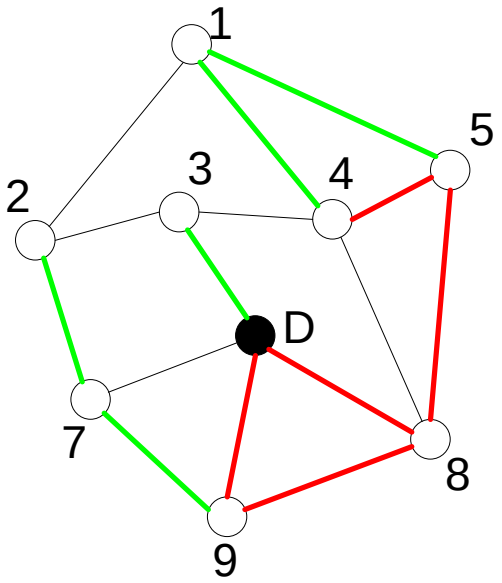
Christian H. Christiansen

(C.H. Christiansen, J. Lysgaard & S. Wøhlk:
"A Branch-and-Price Algorithm for the Capacitated Arc Routing Problem with Stochastic Demands", Operations Research Letters, 2009, vol. 37, pp. 392-398)

Node vs. Arc Routing Problems: Planning a single route

Node Routing: Traveling Salesman Problem (TSP)		**Arc Routing: Chinese Postman Problem (CPP)**	
Given an undirected complete graph, determine the shortest route visiting each point exactly once (end the route where it started)		Given an undirected graph, determine the shortest route which traverses each edge at least once (end the route where it started)	
Input (+ distances)	**A feasible solution**	**Input (+distances)**	**A feasible solution**
			

Node vs. Arc Routing Problems: Planning of multiple routes

Node Routing: Capacitated Vehicle Routing Problem (CVRP)	Arc Routing: Capacitated Arc Routing Problem (CARP)
• One depot • Demand at each customer • Capacitated vehicles • Assign customers to vehicles + determine route for each vehicle	• One depot • Demand on some (or all) edges • Capacitated vehicles • Assign edges to vehicles + determine route for each vehicle
	

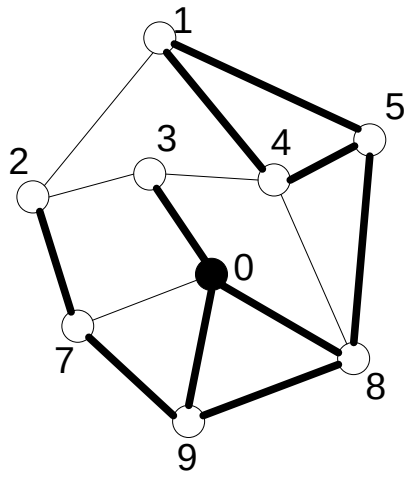
CVRPSD vs. CARPSD: Problem definitions

Capacitated Vehicle Routing Problem with Stochastic Demands (CVRPSD)	**Capacitated Arc Routing Problem with Stochastic Demands (CARPSD)**
Given: • A depot • Symmetric travel costs • All routes begin and end at the depot • All routes are planned before any vehicles leave the depot • A fleet of identical vehicles with capacity Q • n customers with stochastic demands $q_1, q_2, \ldots, q_n$ to be delivered from the depot • Each individual customer's demand becomes known when arriving at the customer	Given: • A depot • Symmetric travel costs • All routes begin and end at the depot • All routes are planned before any vehicles leave the depot • A fleet of identical vehicles with capacity Q • Some edges (service edges) need to be serviced; each service edge has a stochastic demand • Each individual edge's demand becomes known gradually as the vehicle traverses the edge
Determine: • Routes of minimum expected total travel cost	Determine: • Routes of minimum expected total travel cost
Subject to: • Each customer is serviced exactly once The total expected demand on each route does not exceed Q	Subject to: • Each service edge is serviced exactly once • The total expected demand on each route does not exceed Q

CARPSD: Initial transformation

Input:

Graph $G^0 = (V, E^0)$ with vertex set $V = \{0, \ldots, n\}$ and edge set E^0. Vertex 0 represents the depot. The set of service edges is denoted E_s; edges in $E^0 \setminus E_s$ have no demand and may or may not be traversed.



- From the input graph we construct a multigraph $G = (V, E)$ as follows:
  - The service edges are unchanged ($E_s \subseteq E$).
  - In addition, we add a *travel edge* between each pair of vertices. The cost of each travel edge $e = \{i, j\} \in E_t$ is equal to the length of the shortest path in G^0 between i and j.
  - Traversing travel edge $\{i, j\}$ in G corresponds to traversing the corresponding shortest path in G^0.
  - As a result, G contains a travel edge and possibly also a service edge between each pair of vertices.

CARPSD: Failures and strategies

- At the time of planning, the individual edge's demand is given by a probability distribution
- We assume that each individual edge's demand becomes known gradually as the edge is traversed
- We assume that the edges' demands are independent
- It may happen that the actual demand on a route exceeds the vehicle capacity (although the total expected demand on any route does not exceed the vehicle capacity)
  - If this happens, a failure is said to occur
  - A strategy is needed to deal with failures. In particular, it is necessary to know in advance what to do when a failure occurs, if the total expected cost of a route should be calculated

Recourse strategy

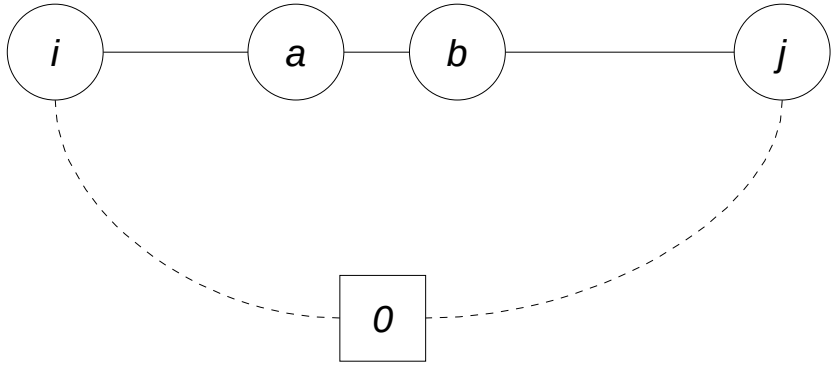
- We formulate the CARPSD as a two stage stochastic program with fixed recourse
- If a failure occurs, the vehicle
  - is depleted at the point of failure,
  - returns to the depot to replenish,
  - and continues the originally planned route from the point of failure.
- The expected cost of a route is composed of two parts:
  - The deterministic cost of following the route, which must be done irrespective of actual demands
  - Expected failure costs, i.e., the total extra distance expected to be traveled due to failure(s) along the route
- The complicating part is the calculation of the expected failure costs

Recourse strategy (2)

- A failure may occur anywhere on an edge, not only at the endpoints
- In case of a failure on edge $\{i,j\}$, the vehicle follows the shortest of the two paths – via i or j – back to the depot to replenish. As such, we allow U-turns in the case of failures
- In order to deal with an infinite number of potential failure points, we divide each edge into an appropriately large number of segments. Moreover, we can compute a lower and upper bound on the cost of any failure on the individual segment, by considering the potential lengths of the shortest path back to the depot

Bounds on failure costs

- Suppose that we consider failures on segment $\{a,b\}$ on edge $\{i,j\}$:



- A lower bound on the failure cost is:

$$L_{\{a,b\}} = 2\min\left\{d_{0,i} + d_{i,a}; d_{0,j} + d_{j,b}\right\}$$

- An upper bound on the failure cost is $L_{\{a,b\}} + 2d_{a,b}$

- As such, we can make the calculations arbitrarily precise by setting the number of segments on the edge to a sufficiently large number, leading to sufficiently short segments

A Set Partitioning Formulation

$$\min \quad \sum_{r \in R} c_r x_r \qquad (1)$$

$$s.t.: \quad \sum_{e \in R} a_{er} x_r = 1 \quad \forall e \in E_s \qquad (2)$$

$$x_r \in \{0,1\} \quad \forall r \in R \qquad (3)$$

where:

- R is the set of feasible routes
- $x_r = 1$, if route r is used, 0 otherwise
- c_r is the expected cost of route r
- $a_{er} = 1$, if edge e is serviced on route j, 0 otherwise

All complications resulting from stochastic demands are embedded in the calculation of the expected costs c_r

The subproblem

- At least one optimal solution is characterized by only having routes that do not contain consecutive travel edges
- As such, we let our pricing procedure disregard all paths containing consecutive travel edges. This makes our pricing procedure run in $O(n^2Q)$ time
- We assume that the demand on each edge follows a Poisson distribution
- The subproblem can be solved using dynamic programming in a way similar to that for the CVRPSD

Branching

- Branching on variables in the master problem makes the subproblem more difficult
- Instead, we branch on resources, which in this case amounts to branching on the accumulated expected demand up to and including a specific edge

Computational experience

- Test instances are obtained by modifying test instances for the CARP
  - For each service edge, the expected demand in the CARPSD is set equal to the demand of that edge in the CARP

- Largest instance solved contained 40 vertices and 69 service edges

- Our pricing procedure has proven most successful on instances that are characterized by tight capacity constraints. This is well in line with the general experience regarding Branch-and-Price algorithms for the CVRP and CVRPSD