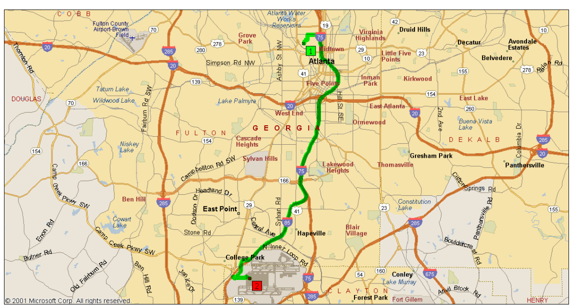
Logistics Systems Design: Transportation Models

1. Introduction
2. Forecasting
3. Transportation Systems
4. Transportation Models
5. Inventory Systems
6. Supply Chain Systems

Vehicle Routing Problems Overview

- Single Origin-Destination Routing
- Multiple Origin-Destination Routing
- Single Vehicle Round-Trip Routing
- Vehicle Routing and Scheduling

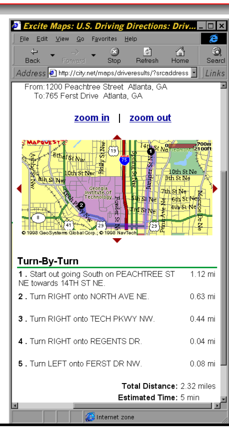
Shortest Path Applications: Route Planning to ATL Airport



Shortest Path Applications: Point-To-Point Instructions

Time	Mile	Instruction	For
Summary: 13.9 miles (23 minutes)			
5:00 PM	0	Depart 765 Ferst Dr NW, Atlanta, GA 30318 on Ferst Dr NW (North)	0.3 mi
5:01 PM	0.3	Turn LEFT (North) onto Dalney St NW	0.2 mi
5:02 PM	0.5	Turn RIGHT (East) onto 10th St NW	0.5 mi
5:03 PM	0.9	Turn RIGHT (South) onto Ramp	0.1 mi
5:03 PM	1.1	Merge onto I-75 [I-85] (South)	6.8 mi
5:12 PM	7.8	Continue (South) on I-85	3.6 mi
5:16 PM	11.5	At I-85 Exit 72, turn off onto Ramp	0.4 mi
5:17 PM	11.9	Continue (West) on Airport Blvd [S Terminal Pkwy]	1.0 mi
5:20 PM	12.9	Continue (South-West) on Airport Circle	0.2 mi
5:21 PM	13.1	Bear RIGHT (East) onto N Terminal Pkwy	0.6 mi
5:22 PM	13.7	Turn RIGHT (East) onto Local road(s)	0.2 mi
5:23 PM	13.9	Arrive Hartsfield-Atlanta International Airport	

Shortest Path Application: Excite Driving Instructions



25-Mar-03 © Marc Goetschalckx

Single Vehicle Origin-Destination Routing

- *Shortest Path Problem (SPP)*
  - *Network nodes = points to be visited*
  - *Network links = connecting the nodes*
- *Dijkstra's optimal algorithm (1959)*
- *100,000 nodes*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Shortest Path Problem (SPP) Variants

- *One Source to One Sink (s to t)*
- *One Source to All Sinks (s to all)*
- *All Pairs*
- *k Shortest Paths (Sensitivity)*
- *All Non-Negative Costs (Label Setting)*
- *General Costs (Label Correcting)*
- *Longest Path in Acyclic Graphs (PERT and CPM)*

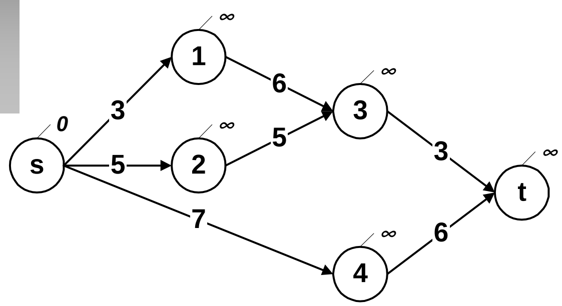
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Labeling Algorithms

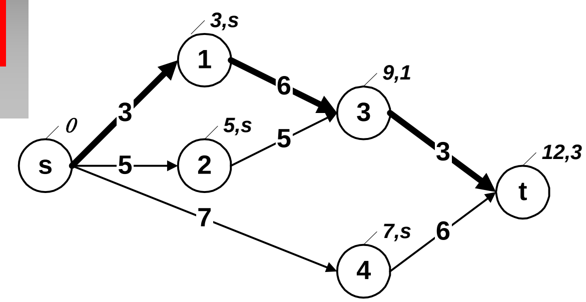
- *Temporary Labels = Upper Bound*
  Permanent Label = Exact Path Length
- *Reduce Labels by Iterative Procedure*
- *Label Setting*
  - *One temporary label becomes permanent per iteration*
- *Label Correcting*
  - *All temporary labels become permanent at the last iteration*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Dijkstra's Algorithm Illustration



Dijkstra's Algorithm Solution



Dijkstra's Algorithm Notation

- *N = Set of all nodes*
- *P = Set of permanently labeled nodes*
- *T = Set of nodes with temporary labels (complement of P, N=P+T)*
- *L(k) = Label of node k*
- c_{ij} *= Arc length from node i to j*
- $\Gamma(k)$ *= all successor nodes of node k (forward star)*
- *pred (k) = predecessor node of node k on the shortest path to node k*

Dijkstra's SPP Algorithm Description

❶ *Set all node labels l(x) = ∞, set l(s) = 0, set all nodes to temporary*

❷ *Find temporary node with minimum label, l(p)=min{l(x)}*

❸ *For all temporary x ε Γ(p) update labels l(x) = min { l(x), l(p) + c(p,x)}*

❹ *Mark node p as permanent*

❺ *If all destinations are permanent stop, else go to step 2*

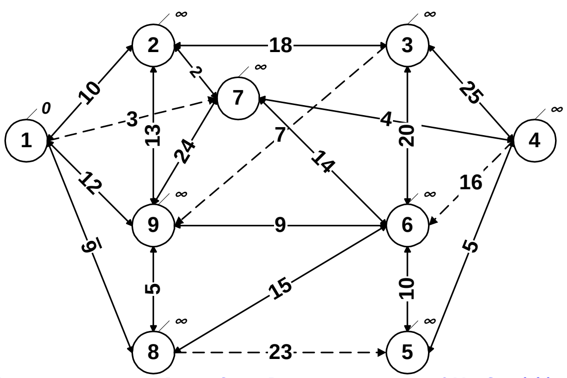
Dijkstra's SPP Formal Algorithm Description

$$
\begin{aligned}
& P = \varnothing,\ T = N,\ l(s) = 0,\ l(i) = \infty \ \ \forall i \in N \\
& \textit{while } |P| < n \ \{ \\
& \quad k \leftarrow l(k) = \min\{l(j) : j \in T\} \\
& \quad P \leftarrow P \cup \{k\},\ T \leftarrow T - \{k\} \\
& \quad \textit{for } \ j \in \Gamma(k) \cap T \\
& \qquad \textit{if } \ l(j) > l(k) + c_{kj} \\
& \qquad\quad l(j) = l(k) + c_{kj},\ \ \textit{pred}(j) = k \\
& \}\ \textit{endwhile}
\end{aligned}
$$

Dijkstra's SPP Algorithm Characteristics

- *Forward dynamic programming*
- *Nonnegative arc "lengths" or "costs"*
- *$O(n^2) + O(m)$ or $O(n^2)$ for fully dense graphs*
- *Directed out-tree rooted at s*
- *Node selection computationally most expensive*
- *100,000 nodes*

SPP Example Network

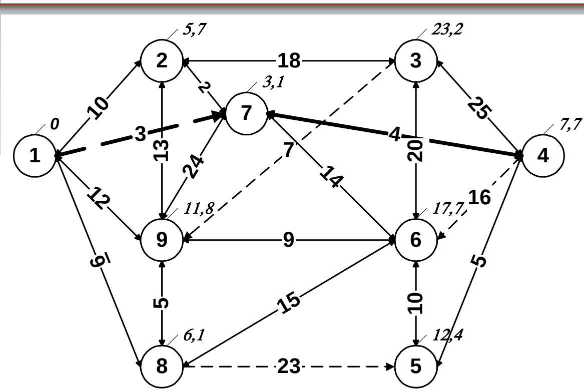


SPP Example Distances

To / **From**

	1	2	3	4	5	6	7	8	9
1		10					3	6	12
2	10		18				2		13
3		18		25		20			7
4			25		5	16	4		
5				5		10			
6			20		10		14	15	9
7		2		4		14			24
8	6				23	15			5
9	12	13				9	24	5	

SPP Example Solution



Network Flow Formulation

$$Min \quad \sum_{i}^{N}\sum_{j}^{N} c_{ij}x_{ij}$$

$$s.t. \quad \sum_{j}^{N} x_{ij} - \sum_{h}^{N} x_{hi} = b_i \qquad \forall i$$

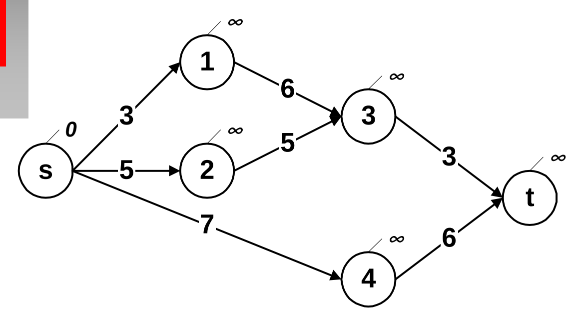
$$0 \le x_{ij} \le u_{ij} \qquad \forall ij$$

Flow Balance: Out - In = External In

Network Formulation Characteristics

- One variable for each arc and commodity (flow)
- One conservation of flow constraint for each node and commodity
- Flows from the outside (sign convention: entering = positive)
- Individual and joint upper (and lower) bounds

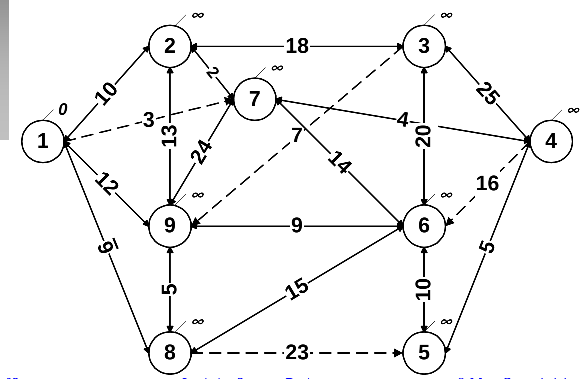
Network Example



Algebraic Network Formulation Example

$$
\begin{aligned}
\min \quad & 3x_{s1} + 5x_{s2} + 7x_{s4} + 6x_{13} + 5x_{23} + 3x_{3t} + 6x_{4t} \\
s.t. \quad & 1 - x_{s1} - x_{s2} - x_{s4} = 0 \\
& x_{s1} - x_{13} = 0 \\
& x_{s2} - x_{23} = 0 \\
& x_{13} + x_{23} - x_{3t} = 0 \\
& x_{s4} - x_{4t} = 0 \\
& x_{3t} + x_{4t} - 1 = 0
\end{aligned}
$$

Shortest Path Example Network



Shortest Path Example Excel Arc Capacities

Microsoft Excel - Shortest Path Network.xls

Arc Capacities

From/To	1	2	3	4	5	6	7	8	9
1		1					1	1	1
2	1		1				1		1
3		1		1		1			1
4			1		1	1	1		
5				1		1			
6			1		1		1	1	1
7		1		1		1			1
8	1				1	1			1
9	1	1				1	1	1	

Shortest Path Example Excel Arc Costs

Microsoft Excel - Shortest Path Network.xls

Arc Cost

From/To	1	2	3	4	5	6	7	8	9
1		10					3	6	12
2	10		18				2		13
3		18		25		20			7
4			25		5	16	4		
5				5		10			
6			20		10		14	15	9
7		2		4		14			24
8	6				23	15			5
9	12	13				9	24	5	

Shortest Path Example Excel Initial Zero Arc Flows

Microsoft Excel - Shortest Path Network.xls

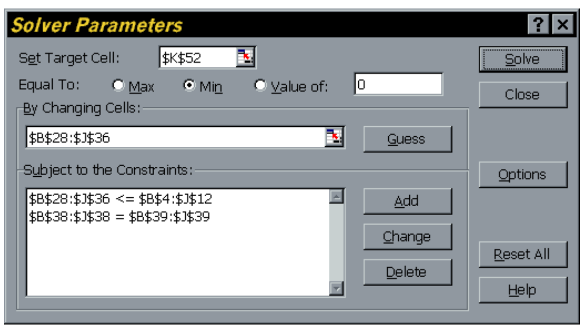
	A	B	C	D	E	F	G	H	I	J	K
26	**Arc Flows**										
27	From/To	1	2	3	4	5	6	7	8	9	Out
28	1										0
29	2										0
30	3										0
31	4										0
32	5										0
33	6										0
34	7										0
35	8										0
36	9										0
37	In	0	0	0	0	0	0	0	0	0	
38	Balance	0	0	0	0	0	0	0	0	0	
39	External	1			-1						

Shortest Path Example: Excel Initial Zero Objective Function

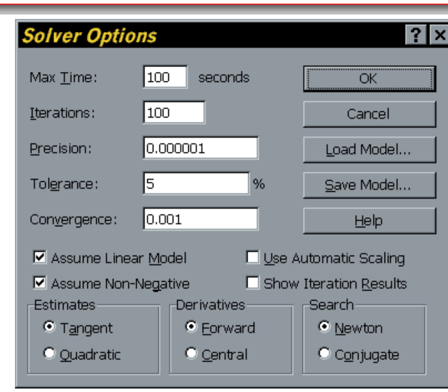
Microsoft Excel - Shortest Path Network.xls

	A	B	C	D	E	F	G	H	I	J	K
41	**Objective**										
42	From/To	1	2	3	4	5	6	7	8	9	Sum
43	1	0	0	0	0	0	0	0	0	0	0
44	2	0	0	0	0	0	0	0	0	0	0
45	3	0	0	0	0	0	0	0	0	0	0
46	4	0	0	0	0	0	0	0	0	0	0
47	5	0	0	0	0	0	0	0	0	0	0
48	6	0	0	0	0	0	0	0	0	0	0
49	7	0	0	0	0	0	0	0	0	0	0
50	8	0	0	0	0	0	0	0	0	0	0
51	9	0	0	0	0	0	0	0	0	0	0
52	Sum	0	0	0	0	0	0	0	0	0	0

Shortest Path Example: Excel Spreadsheet Solver Parameters



Shortest Path Example: Excel Spreadsheet Solver Options



Shortest Path Example Excel Solution Arc Flows

Microsoft Excel - Shortest Path Network.xls

File Edit View Insert Format Tools Data Window Help

	A	B	C	D	E	F	G	H	I	J	K
26	**Arc Flows**										
27	From/To	1	2	3	4	5	6	7	8	9	Out
28	1	0	0	0	0	0	0	1	0	0	1
29	2	0	0	0	0	0	0	0	0	0	0
30	3	0	0	0	0	0	0	0	0	0	0
31	4	0	0	0	0	0	0	0	0	0	0
32	5	0	0	0	0	0	0	0	0	0	0
33	6	0	0	0	0	0	0	0	0	0	0
34	7	0	0	0	1	0	0	0	0	0	1
35	8	0	0	0	0	0	0	0	0	0	0
36	9	0	0	0	0	0	0	0	0	0	0
37	In	0	0	0	1	0	0	1	0	0	
38	Balance	1	0	0	-1	0	0	0	0	0	
39	External	1			-1						

Sheet1 Sheet2 Sheet3

Shortest Path Example: Excel Solution Objective Function

Microsoft Excel - Shortest Path Network.xls

File Edit View Insert Format Tools Data Window Help

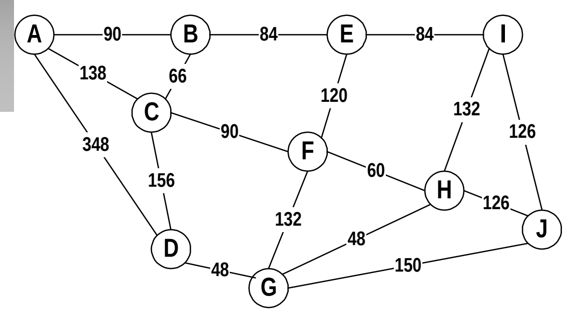
	A	B	C	D	E	F	G	H	I	J	K
41	**Objective**										
42	From/To	1	2	3	4	5	6	7	8	9	Sum
43	1	0	0	0	0	0	0	3	0	0	3
44	2	0	0	0	0	0	0	0	0	0	0
45	3	0	0	0	0	0	0	0	0	0	0
46	4	0	0	0	0	0	0	0	0	0	0
47	5	0	0	0	0	0	0	0	0	0	0
48	6	0	0	0	0	0	0	0	0	0	0
49	7	0	0	0	4	0	0	0	0	0	4
50	8	0	0	0	0	0	0	0	0	0	0
51	9	0	0	0	0	0	0	0	0	0	0
52	Sum	0	0	0	4	0	0	3	0	0	7

Sheet1 Sheet2 Sheet3

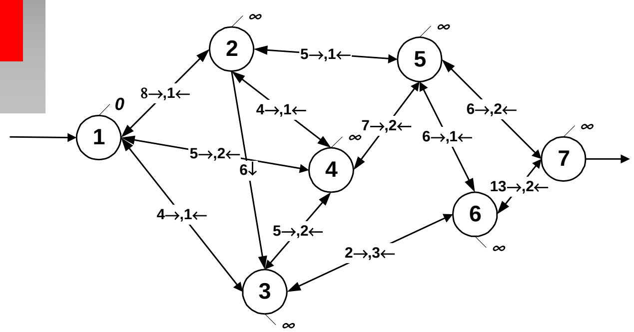
Dijkstra's Algorithm Sparse Graphs

- Heap implementations (binary, Fibonacci, radix,...)
- Running times
  - O(m+n log n) for Fibonacci heap
  - O(m log n) for binary heap
- Intricate implementation

Shortest Path Exercise



Shortest Path Exercise 2

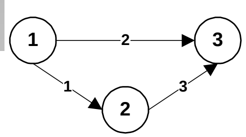


Directed Graph Representation

- Adjacency matrix
- Successor list
- Successor linked list

Directed Graph Adjacency Matrix

- A[i, j] present if arc from node i to node j



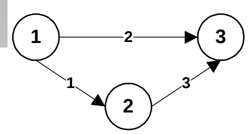
	1	2	3
1		1	2
2			3
3			

Directed Graph Adjacency Matrix Characteristics

- N^2 storage memory
- Constant O(1) lookup, addition, deletion
- Linear o(N) successor and predecessor loops

Directed Graph Successor Array

- Head or first arc array [nodes+1]
  to-node array with every arc [arcs]



Head
1
3
4
4

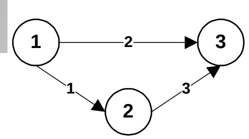
To	Arc
2	1
3	2
3	3

Directed Graph Successor Array Characteristics

- N + 2M memory storage
- Linear o(N) lookup,
  linear o(N) insertion and deletion
  - Plus constant time to move the array elements for insertion and deletion
- Linear o(N) successor loop,
  linear o(M) predecessor loop

Directed Graph Successor Double Linked List

- Head or first arc array [nodes]
  to-node double linked list with every arc [arcs]



Head
1
3
-

Prev	To	Arc	Next
-	2	1	2
1	3	2	3
2	3	3	-

Directed Graph Successor Dbl. Linked List Characteristics

- N + 4M memory storage
- Linear O(N) Lookup,
  Constant O(1) Insertion,
  Linear O(N) Deletion
- Linear O(N) Successor Loop,
  Linear O(M) Predecessor Loop

Double Linked List Deletion

-	1	2

1	2	3

2	3	-

-	1	3

1	3	-

- *k to be deleted element*
  - *If Pred(k) then Succ(Pred(k)) = Succ(k)*
  - *If Succ(k) then Pred(Succ(k)) = Pred(k)*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Double Linked List Insertion

-	1	2

1	2	-

-	1	3

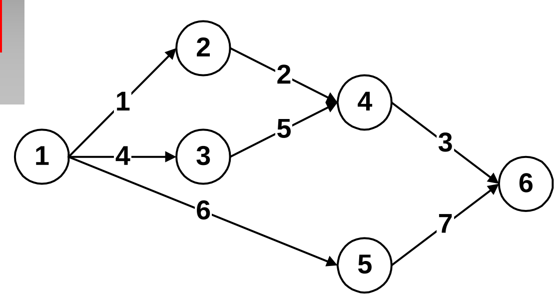
1	3	2

3	2	-

- *k to be inserted element after p*
  - *Succ(k) = Succ(p), Pred(k)=p, Succ(p)=k*
  - *If Succ(k) then Pred(Succ(k))=k*
- *k to be inserted element before s*
  - *Pred(k)=Pred(s), Succ(k)=s, Pred(s)=k*
  - *if Pred(k) then Succ(Pred(k))=k*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Directed Graph Example



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Directed Graph Example Adjacency Matrix

	1	2	3	4	5	6
1		1	4		6	
2				2		
3				5		
4						3
5						7
6						

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Directed Graph Example Successor Array

Head
1
4
5
6
7
8
8

To	Arc
2	1
3	4
5	6
4	2
4	5
6	3
6	7

Directed Graph Example Successor Linked List

Head
1
4
5
6
7
-

List			
-	2	1	2
1	3	4	3
2	5	6	4
3	4	2	5
4	4	5	6
5	6	3	7
6	6	7	-

Arc (3-4) Deletion: Adjacency Matrix

	1	2	3	4	5	6
1		1	4		6	
2				2		
3						
4						3
5						7
6						

Arc (3-4) Deletion: Successor Array

Head
1
4
5
5
6
7
7

To	Arc
2	1
3	4
5	6
4	2
6	3
6	7

Arc (3-4) Deletion Successor Linked List

Head

1
4
-
6
7
-

List

-	2	1	2
1	3	4	3
2	5	6	4
3	4	2	6

4	6	3	7
6	6	7	-

Arc (3-6) Addition (Cost = 8) Adjacency Matrix

	1	2	3	4	5	6
1		1	4		6	
2				2		
3				5		8
4						3
5						7
6						

Arc (3-6) Addition (Cost = 8) Successor Array

Head

1
4
5
7
8
9
9

To Arc

2	1
3	4
5	6
4	2
6	8
4	5
6	3

6	7

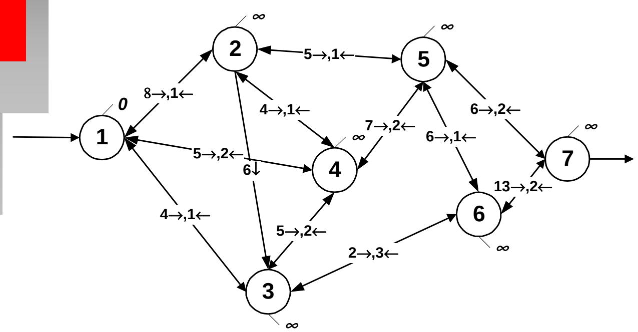
Arc (3-6) Addition (Cost = 8) Successor Linked List

1
4
8
6
7
-

-	2	1	2
1	3	4	3
2	5	6	4
3	4	2	8
8	4	5	6
5	6	3	7
6	6	7	-

4	6	8	5

SPP Exercise 2



Predecessor Loop: Data Independent Pseudo Code

- Perform action on all predecessor nodes of node k

```
p = find_first_predecessor (k)
while (p <> null) {
        perform_action (p)
        p = find_next_predecessor (k, p)
        } // end while
```


Data Structures and Algorithms References

- Horowitz, E., and S. Shani, (1984). Fundamentals of Computer Algorithms. Computer Science Press, Rockville, Maryland.
- Sedgewick, R. (1983). Algorithms. Addison-Wesley, Reading, Massachusetts.
- Aho, A., J. Hopcroft, and J. Ullman, (1983). Data Structures and Algorithms. Addison-Welsey, Reading, Massachusetts.

Vehicle Routing Problems Overview

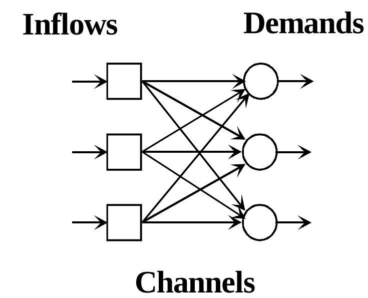
- Single Origin-Destination Routing
- Multiple Origin-Destination Routing
- Single Vehicle Round-Trip Routing
- Vehicle Routing and Scheduling

Multiple Vehicle Origin-Destination Routing Overview

- Problem Description & Variants
- Examples
- Successive Shortest Path Algorithm
- Formulations
- References

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Network Flow Illustration



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Network Components

- Nodes
  - Sources, sinks, intermediate
  - No capacities or cost
  - Conservation of flow
- Channels or Arcs
  - Directed (non-negative flow variables)
  - Costs and capacities

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Multiple Origin-Destination Routing

- Max flow network flow problem
  - Capacity models, public sector
- Min cost network flow problem
  - Economic models, private industry
- Network simplex algorithm is very efficient
- 100,000 channels

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Network Variants

- Transportation
- Transshipment
- Min Cut - Max Flow Network
- Min Cost Network
- Multicommodity Network
- Generalized Network

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Integrality Property

- Solution is naturally integer if
  - Integer external flows
  - Integer capacity bounds
  - Binary conservation of flow coefficients
  - Single commodity
- Desirable for unit load transportation moves
- Does not hold for generalized network and multicommodity networks

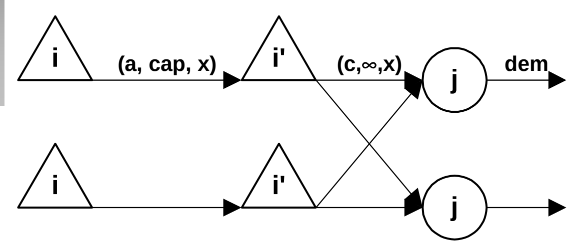
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Tactical Production-Distribution Planning Problem

- Products p
- Customers j, demand dem_{jp}
- Plants i, capacity cap_i
- Marginal production cost a_{ip} and resource consumption req_{ip}
  - generalized network $req \neq 1$
- Transportation cost c_{ijp} and quantity x_{ijp}

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Tactical Production-Distribution Planning Network



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Tactical Production-Distribution Planning Model

$$\begin{aligned} Min \quad & \sum_{i=1}^{F}\sum_{j=1}^{C}\sum_{p=1}^{P}\left(a_{ip}+c_{ijp}\right)x_{ijp} \\ s.t. \quad & \sum_{i=1}^{F} x_{ijp} = dem_{jp} \qquad \forall jp \\ & \sum_{j=1}^{C}\sum_{p=1}^{P} req_{ip}x_{ijp} \le cap_i \qquad \forall i \\ & x_{ijp} \ge 0 \end{aligned}$$

Operator Scheduling Problem

- Parameters and variables
  - Time periods with coverage requirements b_i
  - Operator shifts with costs c_i cover consecutive time periods
  - Number of operators for each shift x_i

Operator Scheduling Linear Programming Formulation

$$\begin{aligned} Min \quad & cx \\ s.t. \quad & Ax \ge b \\ & x \ge 0 \end{aligned}$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix} x \ge \begin{bmatrix} 5 \\ 12 \\ 8 \\ 10 \\ 4 \end{bmatrix}$$

Operator Scheduling Transformation

- Consecutive ones in each column
- Add negative identity matrix (row surplus variables s: Ax-s=b)
- Add "zero" row (node N+1 flow balance constraint)
- Linear row operation
  - For r = N Down To 1
    Row[r+1] = Row[r+1] - Row[r]

Operator Scheduling Network Formulation (Initial)

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} 5 \\ 12 \\ 8 \\ 10 \\ 4 \\ 0 \end{bmatrix}$$

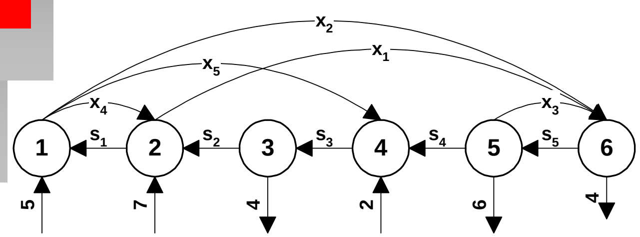
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Operator Scheduling Network Formulation (Final)

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ -4 \\ 2 \\ -6 \\ -4 \end{bmatrix}$$

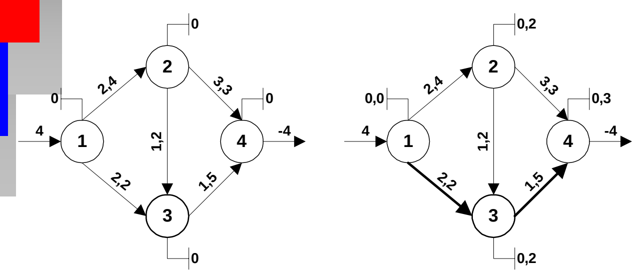
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Operator Scheduling Network



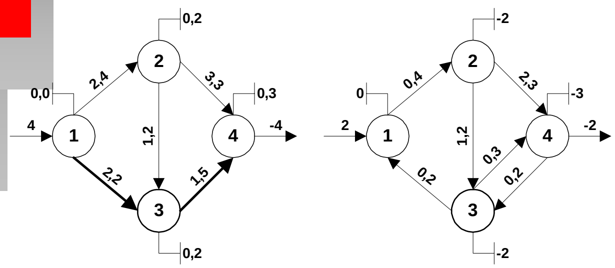
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Illustration (1)



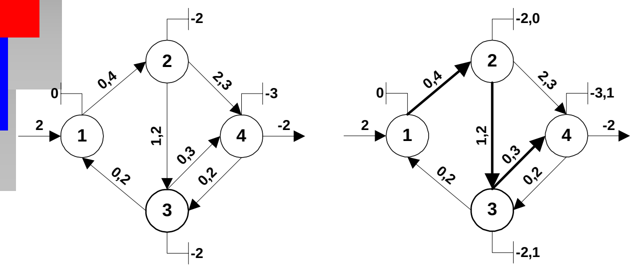
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Illustration (2)



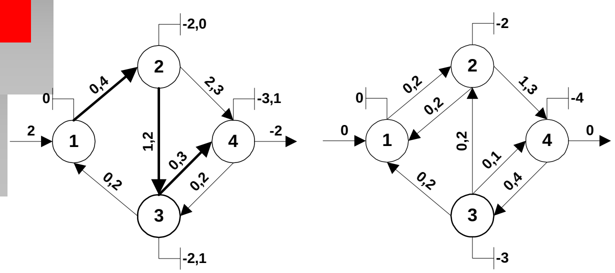
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Illustration (3)



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Illustration (4)



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Algorithm (1)

- Start all flows x = 0,
  all node potentials $\pi = 0$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Algorithm (2)

❷ Construct incremental/residual graph

- Same nodes as original graph
- If $x_{ij} > 0$ then add artificial arc ji
- If $x_{ij} = u_{ij}$ then eliminate arc ij or $d_{ij} = \infty$
- If $x_{ij} < u_{ij}$ then $d_{ij} = c_{ij} - \pi_i + \pi_j$
- If $x_{ij} > 0$ then $d_{ji} = -c_{ij} - \pi_j + \pi_i = - d_{ij}$
- Add super source (sink), arcs to (from) sources (sinks) with 0 cost, capacity = remaining supply (demand)

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Algorithm (3)

❸ Find shortest path from source to sink

- Find shortest path to any sink node k with Dijkstra's algorithm
- If no such path, stop, network flow problem is infeasible

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Algorithm (4)

❹ Compute maximum flow change on the shortest path

- on backflow arcs, $\delta_{ij} = x_{ji}$
- on regular arcs, $\delta_{ij} = u_{ij} - x_{ij}$
- $\delta = \min \{\delta_{ij}\}$

❺ Augment flow on the shortest path

- on backflow arcs, $x_{ji} = x_{ji} - \delta$
- on regular arcs, $x_{ij} = x_{ij} + \delta$
- update remaining supply and demand

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Successive Shortest Paths Algorithm (5)

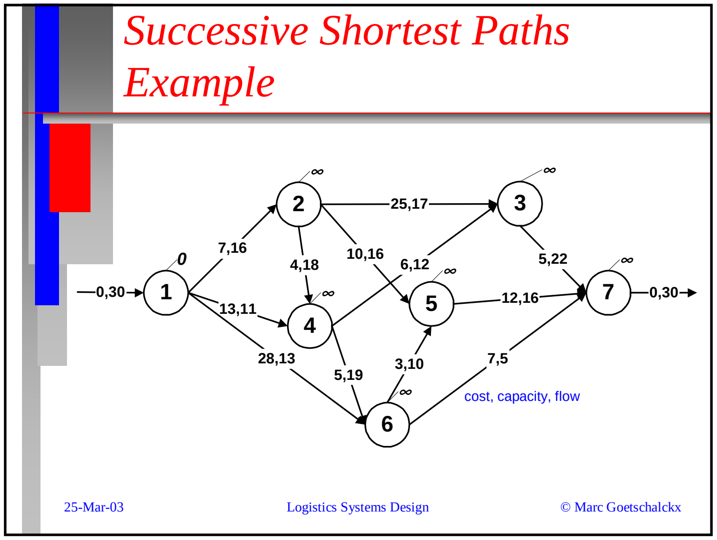
❻ If all remaining demands are zero, stop,network is optimal

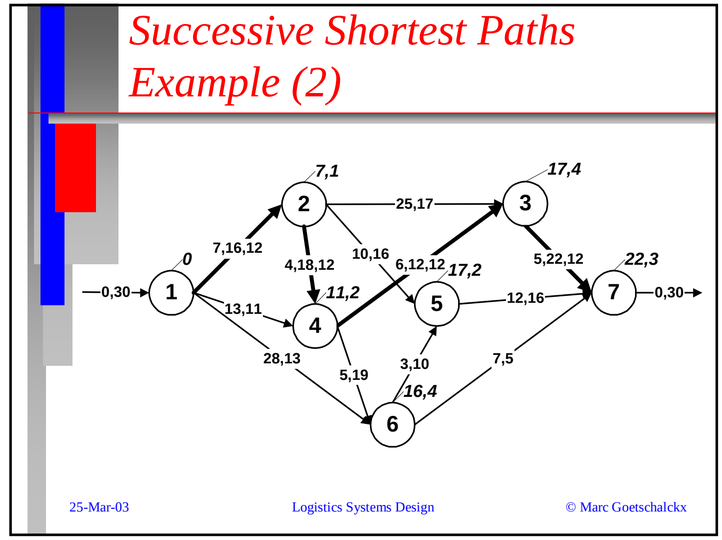
❼ Update node potentials

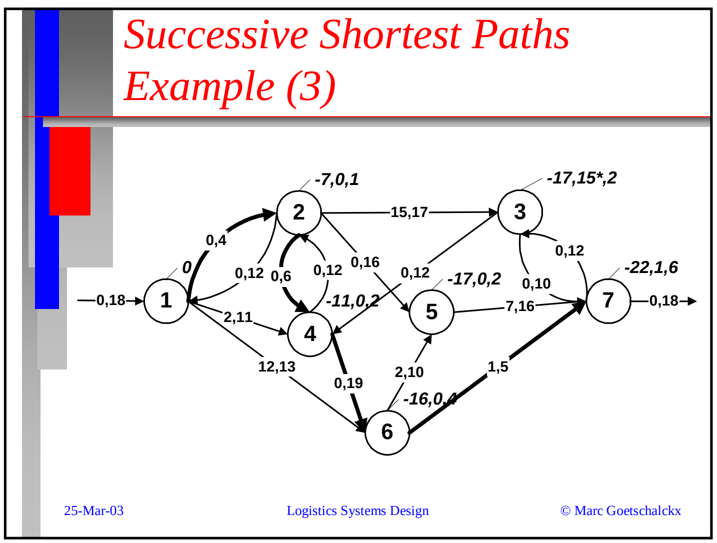
- k = shortest path sink node
- if node i is permanent, $\pi_i = \pi_i - SP_i$
- if node i is temporary, $\pi_i = \pi_i - SP_k$

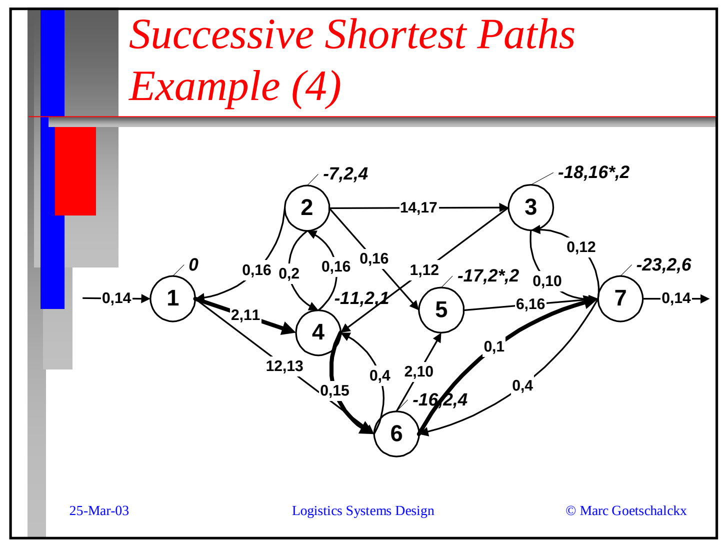
❽ Goto step 2

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

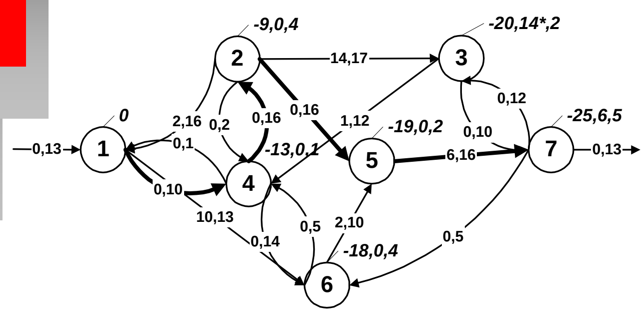

Successive Shortest Paths
Example
cost, capacity, flow
25-Mar-03
Logistics Systems Design
© Marc Goetschalckx


Successive Shortest Paths
Example (2)
25-Mar-03
Logistics Systems Design
© Marc Goetschalckx

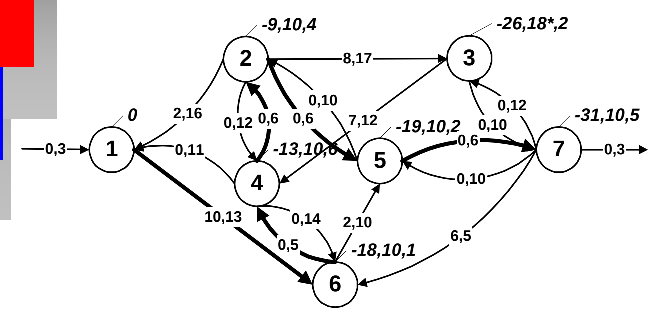

Successive Shortest Paths
Example (3)
25-Mar-03
Logistics Systems Design
© Marc Goetschalckx


Successive Shortest Paths
Example (4)
25-Mar-03
Logistics Systems Design
© Marc Goetschalckx

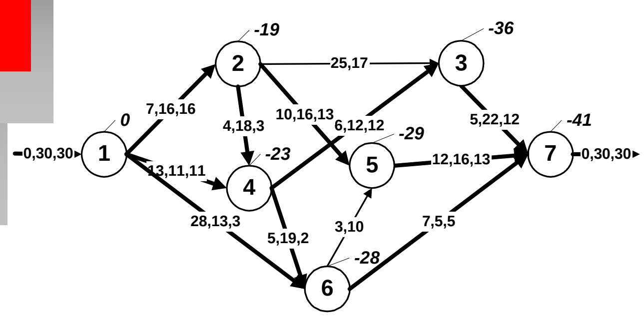
Successive Shortest Paths Example (5)



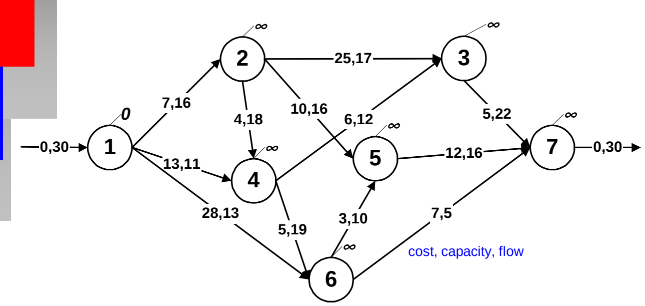
Successive Shortest Paths Example (6)



Successive Shortest Paths Example (7)



Successive Shortest Paths Example



Min Cost Network Example
Excel Arc Capacities

Microsoft Excel - Min Cost Capacitated Netw...

	A	B	C	D	E	F	G	H
1	**Min Cost Network Flow Example**							
2	**Arc Capacities**							
3	From/To	1	2	3	4	5	6	7
4	1		16		11		13	
5	2			17	18	16		
6	3							22
7	4			12			19	
8	5							16
9	6					10		5
10	7							

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Min Cost Network Example
Excel Arc Costs

Microsoft Excel - Min Cost Capacitated Netw...

	A	B	C	D	E	F	G	H
12	**Arc Costs**							
13	From/To	1	2	3	4	5	6	7
14	1		7		13		28	
15	2			25	4	10		
16	3							5
17	4			6			5	
18	5							12
19	6					3		7
20	7							

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Min Cost Network Example
Excel Initial Zero Arcs Flows

Microsoft Excel - Min Cost Capacitated Network.xls

	A	B	C	D	E	F	G	H	I
22	**Arc Flows**								
23	From/To	1	2	3	4	5	6	7	Out
24	1								0
25	2								0
26	3								0
27	4								0
28	5								0
29	6								0
30	7								0
31	In	0	0	0	0	0	0	0	
32	Balance	0	0	0	0	0	0	0	
33	External	30						-30	

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

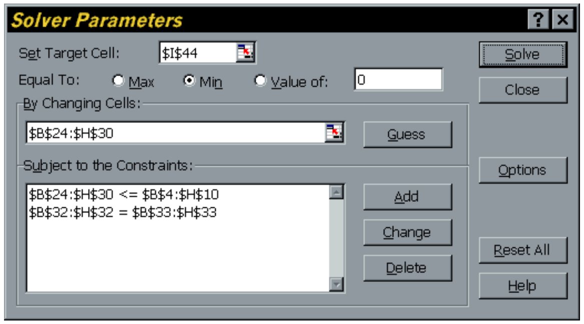
Min Cost Network Example:
Excel Initial Zero Objective

Microsoft Excel - Min Cost Capacitated Network.xls

	A	B	C	D	E	F	G	H	I
35	**Objective**								
36	From/To	1	2	3	4	5	6	7	Sum
37	1	0	0	0	0	0	0	0	0
38	2	0	0	0	0	0	0	0	0
39	3	0	0	0	0	0	0	0	0
40	4	0	0	0	0	0	0	0	0
41	5	0	0	0	0	0	0	0	0
42	6	0	0	0	0	0	0	0	0
43	7	0	0	0	0	0	0	0	0
44	Sum	0	0	0	0	0	0	0	0

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Min Cost Network Example
Excel Solver Parameters



Min Cost Network Example
Excel Optimal Arc Flows

Microsoft Excel - Min Cost Capacitated Network.xls

	A	B	C	D	E	F	G	H	I
22	**Arc Flows**								
23	From/To	1	2	3	4	5	6	7	Out
24	1	0	16	0	11	0	3	0	30
25	2	0	0	0	3	13	0	0	16
26	3	0	0	0	0	0	0	12	12
27	4	0	0	12	0	0	2	0	14
28	5	0	0	0	0	0	0	13	13
29	6	0	0	0	0	0	0	5	5
30	7	0	0	0	0	0	0	0	0
31	In	0	16	12	14	13	5	30	
32	Balance	30	0	0	0	0	0	-30	
33	External	30						-30	

Min Cost Network Example
Excel Optimal Objective

Microsoft Excel - Min Cost Capacitated Network.xls

	A	B	C	D	E	F	G	H	I
35	**Objective**								
36	From/To	1	2	3	4	5	6	7	Sum
37	1	0	112	0	143	0	84	0	339
38	2	0	0	0	12	130	0	0	142
39	3	0	0	0	0	0	0	60	60
40	4	0	0	72	0	0	10	0	82
41	5	0	0	0	0	0	0	156	156
42	6	0	0	0	0	0	0	35	35
43	7	0	0	0	0	0	0	0	0
44	Sum	0	112	72	155	130	94	251	814

Min Cost Network Flow Formulation

$$\text{Min} \quad \sum_{i}^{N}\sum_{j}^{N} c_{ij} x_{ij}$$

$$\text{s.t.} \quad -\sum_{h}^{N} x_{hi} + \sum_{j}^{N} x_{ij} = b_i \qquad \forall i \qquad [\pi_i]$$

$$0 \le x_{ij} \le u_{ij} \qquad \forall ij \qquad [\alpha_{ij}]$$

- b_i external node flow (supply >0, or demand <0)

Residual Network

$arc(i,j)$ with c_{ij} and u_{ij}

if $x_{ij} > 0$ then

$arc(i,j)\; c_{ij}^{\pi} = c_{ij}^{orig} + \pi_j - \pi_i \qquad r_{ij} = u_{ij} - x_{ij}$

$arc(j,i)\; c_{ji}^{\pi} = -c_{ij}^{\pi} \qquad r_{ji} = x_{ij}$

Optimality Conditions

- Shortest path reduced cost

$d_j \le d_i + c_{ij} \qquad \forall (i,j)$

$c_{ij}^{d} = c_{ij} + d_i - d_j \ge 0 \qquad \forall (i,j)$

- Dual variable reduced cost

$c_{ij}^{\pi} = c_{ij} - \pi_i + \pi_j \ge 0 \qquad \forall (i,j)$

Min Cost Network Flow Primal Formulation

$$Min \quad \sum_{i}^{N}\sum_{j}^{N} c_{ij} x_{ij}$$

$$s.t. \quad -\sum_{h}^{N} x_{hi} + \sum_{j}^{N} x_{ij} = b_i \qquad \forall i \qquad [\pi_i]$$

$$-x_{ij} \ge -u_{ij} \qquad \forall ij \qquad [\alpha_{ij}]$$

Min Cost Network Flow Dual Formulation

$$\max \quad \sum_{i}^{N} b_i \pi_i - \sum_{i}^{N}\sum_{j}^{N} u_{ij}\alpha_{ij}$$

$$s.t. \quad \pi_i - \pi_j - \alpha_{ij} \le c_{ij} \qquad \forall (i,j) \qquad [x_{ij}]$$

$$\alpha_{ij} \ge 0$$

$$\pi_i \text{ unrestricted}$$

Transformed Dual Formulation

$$\max \quad \sum_i^N b_i \pi_i - \sum_i^N \sum_j^N u_{ij} \alpha_{ij}$$

$$\begin{aligned} s.t. \quad & \alpha_{ij} + c_{ij}^{\pi} \geq 0 \qquad \forall (i,j) \qquad [x_{ij}] \\ & \alpha_{ij} \geq 0 \\ & \pi_i \ \ unrestricted \end{aligned}$$

Complementary Slackness Conditions

$$(u_{ij} - x_{ij}^*)\alpha_{ij}^* = 0$$

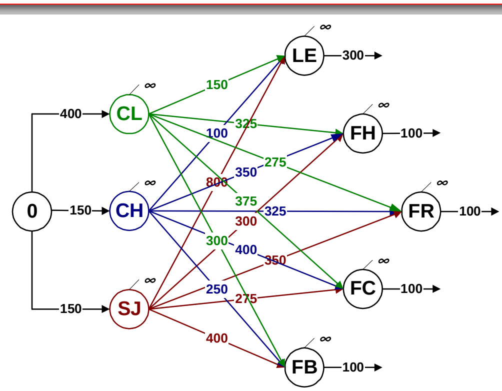
$$(\alpha_{ij}^* + c_{ij}^{\pi})x_{ij}^* = 0$$

$$if \ \ c_{ij}^{\pi} > 0 \ \ then \ \ \alpha_{ij}^* + c_{ij}^{\pi} > 0 \ \ then \ \ x_{ij}^* = 0$$

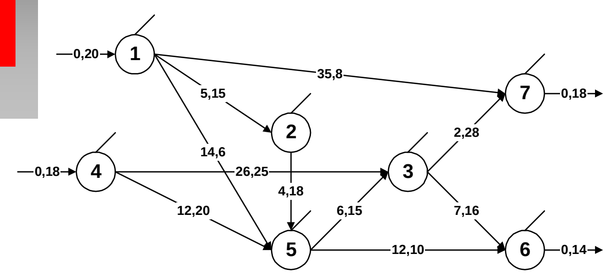
$$if \ \ 0 < x_{ij}^* < u_{ij} \ \ then \ \ \alpha_{ij}^* = 0 \ \ then \ \ c_{ij}^{\pi} = 0$$

$$if \ \ c_{ij}^{\pi} < 0 \ \ then \ \ \alpha_{ij}^* > 0 \ \ then \ \ x_{ij}^* = u_{ij}$$

Ballou Ch7-4 Network Exercise



Network Exercise 2



Network Flow References

- Ahuja, R., T. Magnanti, and J. Orlin. 1993. Network Flows. Prentice Hall, Englewood Cliffs, New Jersey.
- Christofides, N. 1975. Graph Theory: An Algorithmic Approach. Academic Press, New York.
- Evans, J. and E. Minieka. (2nd Ed.) 1992. Optimization Algorithms for Networks and Graphs. Marcel Dekker , New York, New York.
- Kennington, J. and R. Helgason. 1980. Algorithms for Network Programming. John Wiles & Sons, New York, New York.

Vehicle Routing Problems Overview

- Single Origin-Destination Routing
- Multiple Origin-Destination Routing
- Single Vehicle Round-Trip Routing
- Vehicle Routing and Scheduling

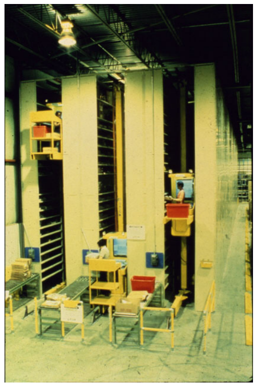
Single Round-Trip Vehicle Routing

- Traveling Salesman Problem (TSP)
- Specialized branch and bound algorithms
- 2000 nodes (million nodes)
- Many heuristic algorithms

TSP Applications

- Traveling salesman
- Shortest Hamiltonian cycle
- Knight’s tour
- Person-aboard order picking
- Running domestic errands
- Sequencing jobs in a paint booth

Person-Aboard Order Picking: Outside View

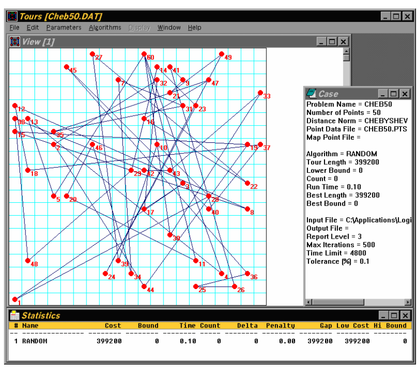


25-Mar-03 © Marc Goetschalckx

Person-Aboard Order Picking: Inside View

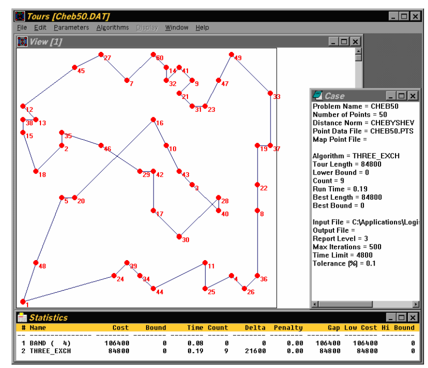
25-Mar-03 © Marc Goetschalckx

Random Sequence Tour



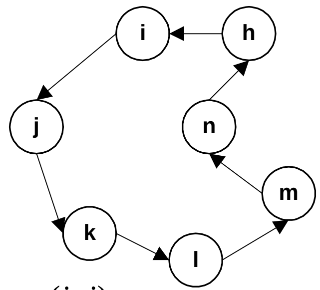
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Local Improvement (3 Opt)



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Asymmetric Traveling Salesman Problem Illustration



c_{ij} *cost of taking arc* (i,j)

$x_{ij} \in \{0,1\}$ *to take arc* (i,j) *or not*

Asymmetric Traveling Salesman Problem Formulation

$$\begin{aligned}
&Min \quad \sum_{i=1}^{N}\sum_{j=1}^{M} c_{ij}x_{ij} \\
&s.t. \quad \sum_{i=1}^{N} x_{ij} = 1 \qquad \forall j \\
&\qquad \sum_{j=1}^{N} x_{ij} = 1 \qquad \forall i \\
&\qquad \sum_{i\in S}\sum_{j\in S} x_{ij} \le |S|-1 \qquad \forall S \subset N
\end{aligned}$$

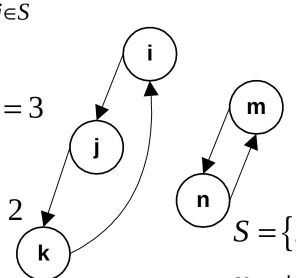
Subtour Elimination Constraints

$$\sum_{i\in S}\sum_{j\in S} x_{ij} \le |S|-1 \qquad \forall S \subset N$$

$S = \{i, j, k\}, |S| = 3$

$x_{ij} + x_{jk} + x_{ki} +$

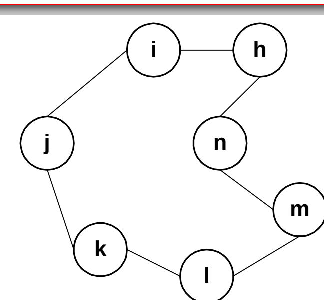
$x_{ji} + x_{kj} + x_{ik} \le 2$



$S = \{m, n\}, |S| = 2$

$x_{mn} + x_{nm} \le 1$

Symmetric Traveling Salesman Problem Illustration



c_{ij} *cost of taking edge* (i,j)

$x_{ij} \in \{0,1\}$ *to take edge* (i,j) *or not*

Symmetric Traveling Salesman Problem Formulation

$$\text{Min} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{j-1} x_{ij} + \sum_{k=j+1}^{N} x_{jk} = 2 \qquad \forall j$$

$$\sum_{i \in S} \sum_{j \in S, j>i} x_{ij} \le |S|-1 \qquad \forall S \subset N$$

Simple TSP Heuristics

- Create an initial tour
  - convex hull, sweep, nearest neighbor
- Insert remaining free points
  - nearest, cheapest, farthest insertion
- Improve existing tour
  - two, three, or Or (chain) exchanges

TSP Example Point Coordinates

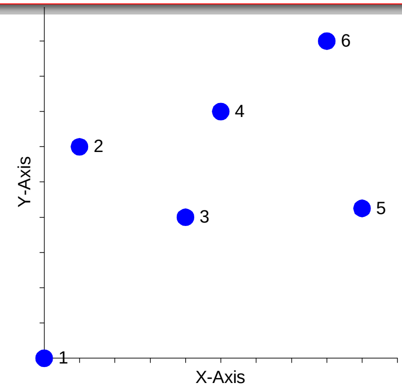
#	x	y
1	0	0
2	100	600
3	400	400
4	500	700
5	900	400
6	800	900

TSP Example: Euclidean Distances

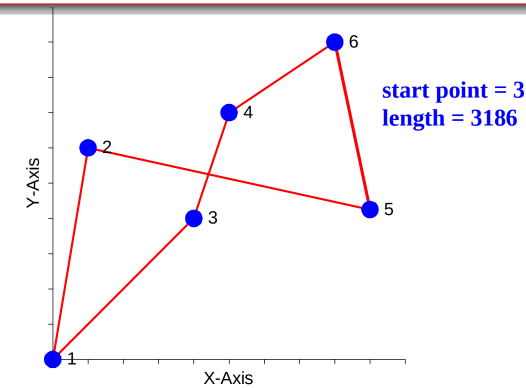
$$d_{ij}^{E} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2}$$

	1	2	3	4	5	6
1	0	608	566	860	985	1204
2	608	0	361	412	825	762
3	566	361	0	316	500	640
4	860	412	316	0	500	361
5	985	825	500	500	0	510
6	1204	762	640	361	510	0

TSP Example: Point Locations



TSP Example: Nearest Neighbor



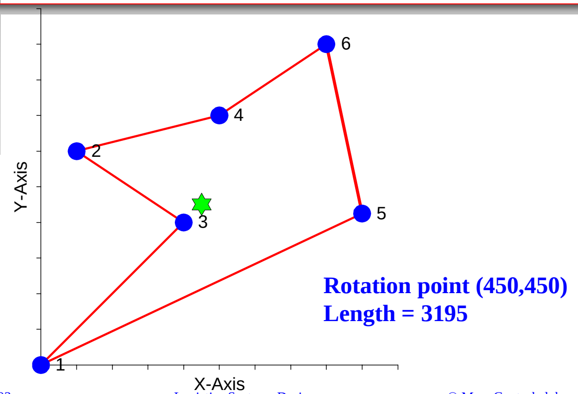
TSP Example: Nearest Neighbor Computations

- Start point 3
- Min {566, 361, 316, 500, 640}=316 (4)
- Min {860, 412, 500, 361} = 361 (6)
- Min {1204, 762, 510} = 510 (5)
- Min {985, 825} = 825 (2)
- Min {608} = 608 (1)
- Total tour length = 3186

TSP Example: Sweep

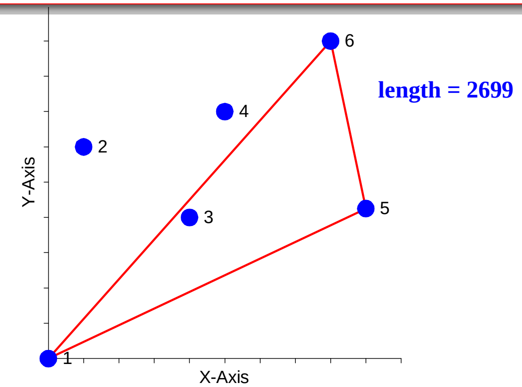
- Find a centrally located rotation point
  - Center or average coordinates
- Rotate a ray and add points to the tour in the sequence they are traversed by the ray

TSP Example: Sweep



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Quad Initial Tour



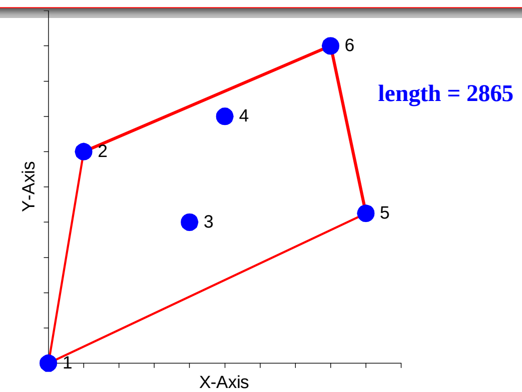
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Quad Calculations

- *Leftmost 0 (1), bottom 0 (1), rightmost 900 (5), top 900 (6)*
- *In this case quadrilateral degenerates to triangle, tour {1-5-6-1}*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Convex Hull Tour Skeleton



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

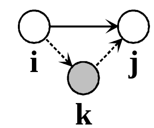
Insertion Heuristics

- Two decisions
  - Which free point to insert
  - Where to insert (which link to break)
  - Which decision to make first
- Many variants
  - Nearest Addition, Nearest Insertion Cheapest Insertion, Priciest Insertion Farthest Insertion, Min. Ratio, Max. Angle, Optimal Insertion...

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Cheapest Insertion

- Insert free point with smallest cost increase (insertion penalty)

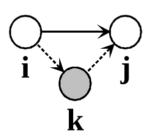


$$\min_k \left\{ \min_{i,j} \left\{ \delta_{ijk} = c_{ik} + c_{kj} - c_{ij} \right\} \right\}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Priciest Insertion

- Insert free point with largest minimum cost increase



$$\max_k \left\{ \min_{i,j} \left\{ \delta_{ijk} = c_{ik} + c_{kj} - c_{ij} \right\} \right\}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Nearest Insertion

- Find free point closest to a point on the tour

$$\min_{k \notin T,\, j \in T} \left\{ c_{kj} \right\} = \min_{k \notin T} \left\{ \min_{j \in T} c_{kj} \right\}$$

- Insert on best link of the tour for point k

$$\min_{(i,j) \in T} \left\{ \delta_{ijk} = c_{ik} + c_{kj} - c_{ij} \right\}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Farthest Insertion

- Find free point with maximum distance to closest point on the tour

$$\max_{k \notin T} \left\{ \min_{j \in T} c_{kj} \right\}$$

- Insert on best link of the tour for point k

$$\min_{(i,j) \in T} \left\{ \delta_{ijk} = c_{ik} + c_{kj} - c_{ij} \right\}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Nearest Addition

- Find free point closest to a point on the tour

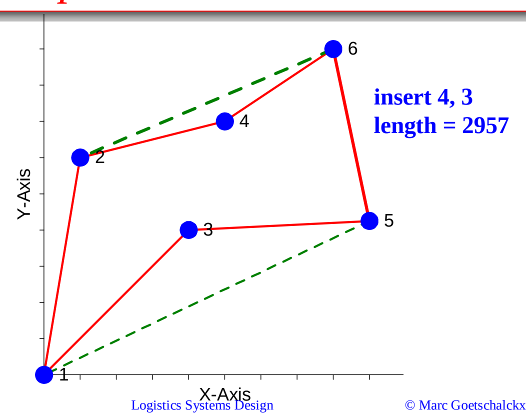
$$\min_{k \notin T,\, j \in T} \left\{ c_{kj} \right\}$$

- Insert on best of two links out of point j for point k

$$\min \left\{ \begin{array}{l} \delta_{ijk} = c_{ik} + c_{kj} - c_{ij}, \\ \delta_{jkm} = c_{jk} + c_{km} - c_{jm} \end{array} \right\}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Cheapest Insertion



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Cheapest Insertion Computations

- Point 3 & 4
  - Min {566+500-985=81 (1-5), 500+640-510=630 (5-6), 640+361-762=239 (6-2), 361+566-608=319 (2-1)} = 81 (1-5)
  - Min {860+500-985=375 (1-5), 500+361-510=351 (5-6), 361+412-762=11 (2-6), 412+860-608=644 (2-1)} = 11 (2-6)*
- Point 3
  - Min {81 (1-5), 595 (6-4), 265 (4-2)} = 81 (1-5)

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Priciest Insertion Computations

- Point 3 & 4
  - Min {566+500-985=81 (1-5), 500+640-510=630 (5-6), 640+361-762=239 (6-2), 361+566-608=319 (2-1)} = 81 (1-5) *
  - Min {860+500-985=375 (1-5), 500+361-510=351 (5-6), 361+412-762=11 (6-2), 412+860-608=644 (2-1)} = 11 (2-6)
- Point 4
  - Min { 610 (1-3), 316 (3-5), 11 (6-2)} = 11 (6-2)

TSP Example: Nearest Insertion

- Points 3 & 4
  - [3] Min {566 (3-1), 361 (3-2), 500 (3-5), 640 (3-6)} = 361 (3-2)* -- tie
  - [4] Min {860 (4-1), 412 (4-2), 500 (4-5), 361 (4-6)} = 361 (4-6)
  - [3] Min {566+500-985=81 (1-5), 500+640-510=630 (5-6), 640+361-762=239 (6-2), 361+566-608=319 (2-1)} = 81 (1-5)

TSP Example: Nearest Insertion Continued

- Point 4
  - [4] Min {361 (4-6), 316 (4-3)} = 316 (4-3)
  - [4] Min {860+316-566=610 (1-3), 316+500-500=316 (3-5), 500+361-510=351 (5-6), 361+412-762=11 (2-6), 412+860-608=644 (2-1)} = 11 (2-6)*

TSP Example: Farthest Insertion Computations

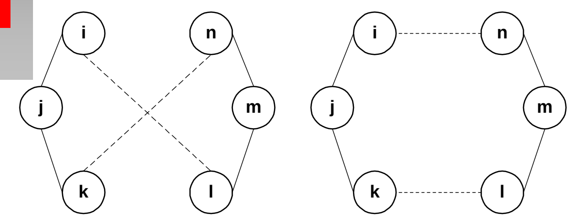
- Point 3 & 4
  - [3] Min {566 (3-1), 361 (3-2), 500 (3-5), 640 (3-6)} = 361 (3-2)* -- tie
  - [4] Min {860 (4-1), 412 (4-2), 500 (4-5), 361 (4-6)} = 361 (4-6)
  - [3] Min {566+500-985=81 (1-5), 500+640-510=630 (5-6), 640+361-762=239 (6-2), 361+566-608=319 (2-1)} = 81 (1-5)

TSP Example: Farthest Insertion Calculations Continued

- Point 4
  - [4] Min {361 (4-6), 316 (4-3)} = 316 (4-3)
  - [4] Min {860+316-566=610 (1-3), 316+500-500=316 (3-5), 500+361-510=351 (5-6), 361+412-762=11 (2-6), 412+860-608=644 (2-1)} = 11 (2-6)*

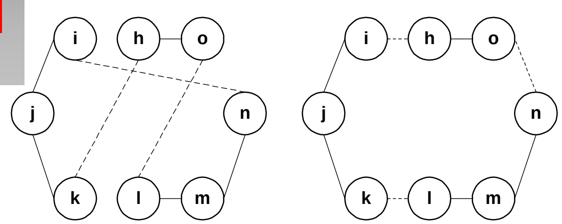
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Two Exchange Improvement Illustration



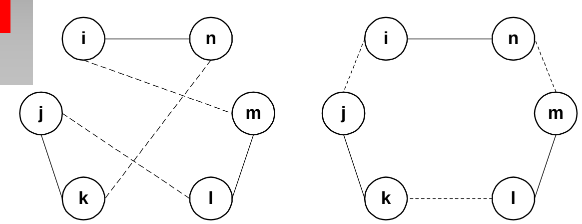
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Three Exchange Improvement Illustration



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Or (2 Chain) Exchange Improvement Illustration



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Exchange Improvement Variants

- First descent exchange
- Steepest descent exchange
- Simulated annealing
- Tabu search

Simulated Annealing

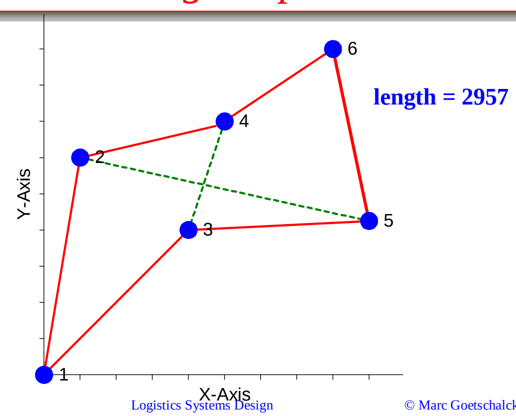
- Evaluate random exchange Δ
- Execute exchange with probability

$$\text{if} \quad \Delta < 0 \quad P[Exch] = 1$$

$$\text{if} \quad \Delta \geq 0 \quad P[Exch] = e^{-\Delta/T}$$

- T search control parameter (temperature) systematically decreasing

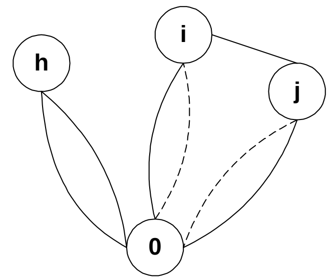
TSP Example: Two Exchange Improvement



TSP Example: Two Exchange Computations

- Original tour length = 3186
- Crossing edges (3-4) and (2-5) in geometric TSP
- Exchange edges (3-4) and (2-5) with (2-4) and (3-5)
- Savings = 316 + 825 - 412 - 500 = 229
- Improved tour length = 2957

Clarke and Wright Savings Illustration



$$s_{ij} = c_{i0} + c_{j0} - c_{ij}$$

Clarke and Wright Savings

- *Select base point {0} (somewhere on perimeter or corner)*
- *Compute savings of combining tours*

$$\max_j \{ s_{ij} = c_{i0} + c_{0j} - c_{ij} \}$$

- *Construct tour primitive {0ij0} (max savings)*
- *Append point with largest savings to either end of tour*

TSP Example: Clarke and Wright Savings Computations

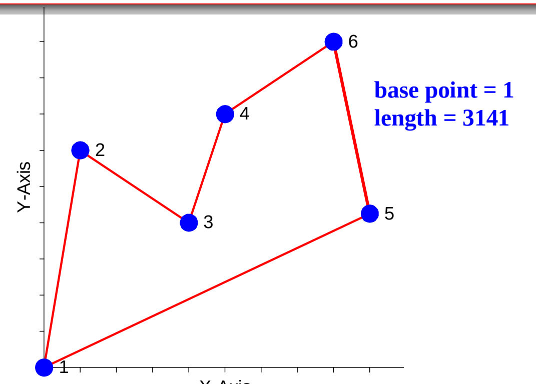
- *Initial route primitive (base point 1):*
  - *(1-2-3-1) [2,3] = 608+566-361=813*
  - *(1-2-4-1) [2,4] = 608+860-412=1056*
  - *(1-4-6-1) [4,6] = 860+1204-361=1703 **
  - *(1-5-6-1) [5,6] = 985+1204-510=1679*

	3	4	5	6
2	813	1056	768	1050
3		1110	1051	1130
4			1345	1703
5				1679

TSP Example: Clarke and Wright Savings Computations

- *Tour primitive (1-6-4-1)*
  - *Max {1050 (6-2-1), 1130 (6-3-1), 1679 (6-5-1), 1056 (4-2-1), 1110 (4-3-1), 1345 (4-5-1)} = 1679 (6-5-1)*
- *Tour primitive (1-4-6-5-1)*
  - *Max {1056 (1-2-4), 1110 (1-3-4), 768 (1-2-5), 1051 (1-3-5)} = 1110 (1-3-4)*
- *Tour primitive (1-3-4-6-5-1)*
  - *Max { 813 (1-2-3), 768 (1-2-5)} = 813 (1-2-3)*

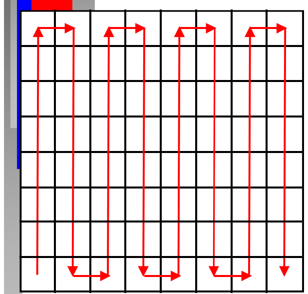
TSP Example: Clarke & Wright Savings



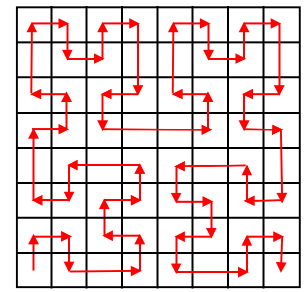
base point = 1
length = 3141

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Spacefilling Curve Examples



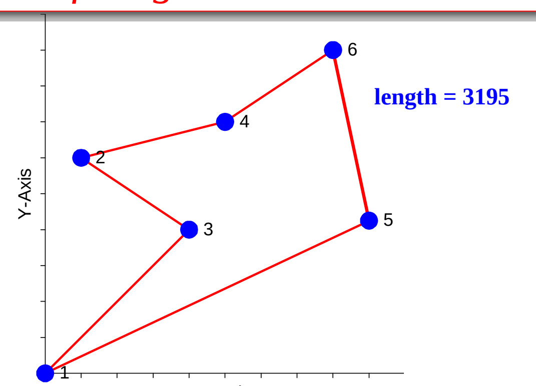
Serpentine Curve



Hilbert Curve (19)

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Spacefilling Curve



length = 3195

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Christofides' Heuristic

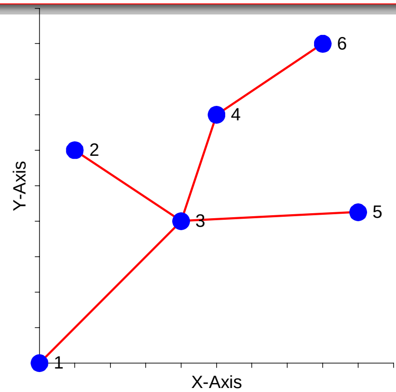
❶ *Construct minimum spanning tree*
❷ *Construct minimum matching of odd degree nodes*
❸ *Find maximum savings shortcut for nodes with degree four or higher*

$$\min_{k|nd_k>2}\left\{\delta_{ijk}=c_{ik}+c_{kj}-c_{ij}\right\}$$

❋ *Worst case performance ratio = 1.5*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Christofides: Minimum Spanning Tree



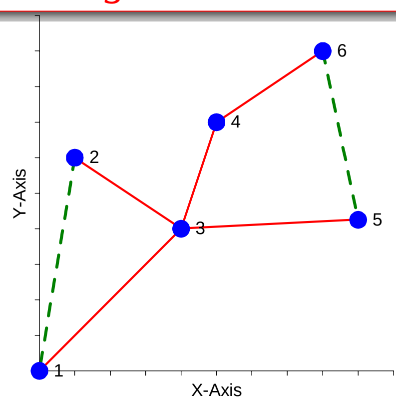
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Christofides: Minimum Matching and Max Shortcuts

- *Minimum matching on {1, 2, 5, 6}*
  - *Min {(12)+(56)=1118, (15)+(26)=1747, (16)+(25)=2029} = 1118 (12)+(56)*
- *Maximum shortcuts for node {3}*
  - *Max {265 (2-3-4), 81 (1-3-5), 36 (2-3-5), 22 (1-3-4)} = 265 (24)*

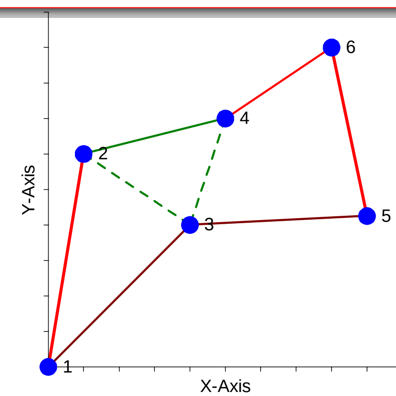
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Christofides: Minimum Spanning Tree and Matching



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Christofides: Max Savings Shortcut



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Symmetric Traveling Salesman Problem Formulation

$$Min \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} x_{ij}$$

$$s.t. \quad \sum_{i=1}^{j-1} x_{ij} + \sum_{k=j+1}^{N} x_{jk} = 2 \qquad \forall j \quad [\lambda_j]$$

$$\sum_{i \in S} \sum_{j \in S, j>i} x_{ij} \le |S|-1 \qquad \forall S \subset N$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Lagrangean Relaxation of 1-Tree (Held & Karp)

❶ Initialize node degree penalties λ to zero

❷ Compute adjusted distances

$$d_{ij}^{\lambda} = d_{ij} + \lambda_i + \lambda_j$$

❸ Construct minimum spanning tree on {N} - base point

❹ Connect base point with two cheapest edges to spanning tree

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Lagrangean Relaxation of 1-Tree cont.

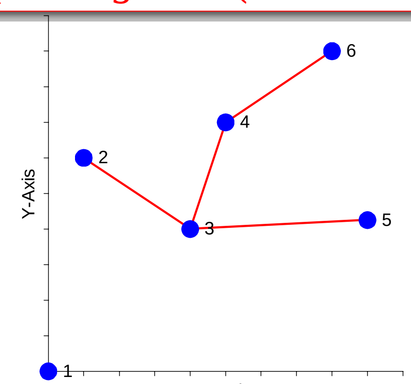
❺ If all node degrees = 2, stop else update node degree penalties (subgradient method)

$$\begin{array}{lll} if \;\; nd_i = 2 & \lambda_i & unchanged \\ if \;\; nd_i < 2 & \lambda_i & decreased \\ if \;\; nd_i \ge 2 & \lambda_i & increased \end{array}$$

❻ Go to step 2

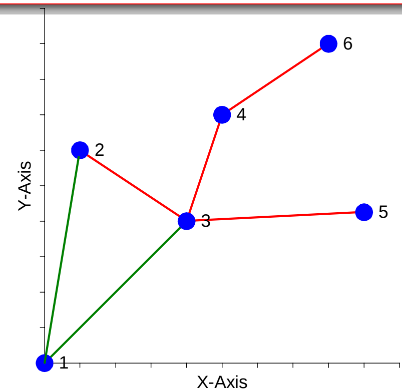
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

1-Tree: First Minimum Spanning Tree (Base Point = 1)



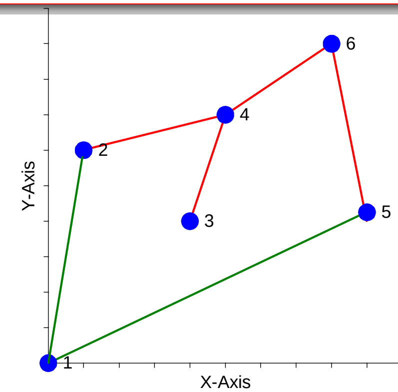
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

1-Tree: First 1-Tree (Base Point = 1)



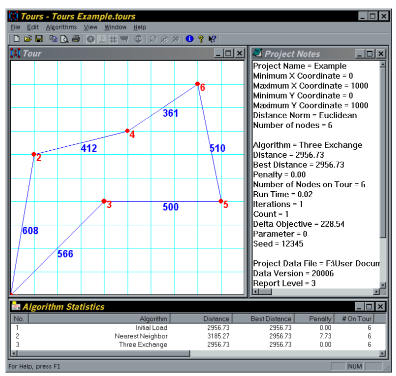
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

1-Tree: Second 1-Tree (Base Point = 1)



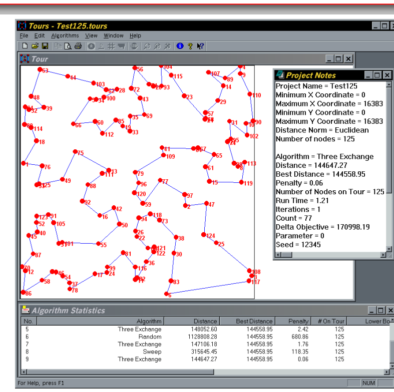
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

TSP Example: Tours Illustration



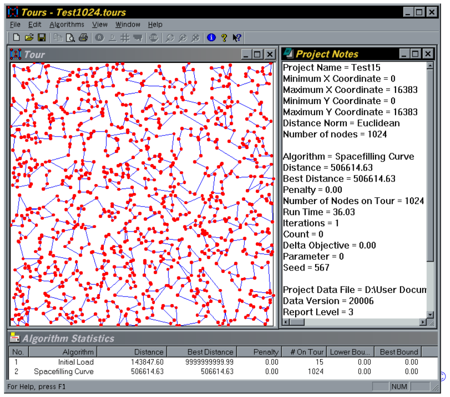
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Tours Illustration: Test 125



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Tours Illustration: Test 1024



Traveling Salesman Problem References

- *Lawler, E., J. Lenstra, A. Rinnooy Kan, and D. Shmoys. 1985. "The Traveling Salesman Problem." John Wiley & Sons, New York, New York.*

Vehicle Routing Problems Overview

- *Single Origin-Destination Routing*
- *Multiple Origin-Destination Routing*
- *Single Vehicle Round-Trip Routing*
- *Vehicle Routing and Scheduling*

Vehicle Routing and Scheduling Overview

- *Problem definition*
- *Variants*
- *Conclusions*

Vehicle Routing Problem Definition

- Goal is to efficiently use a fleet of vehicles
- Given a number of stops to pick up or deliver passenger or goods
- Under a variety of constraints
  - Vehicle capacity
  - Delivery time restrictions
  - Precedence constraints

Vehicle Routing Decisions

- Which customers served by which vehicle (allocation or clustering)
- Sequence of stops for each vehicle (sequencing)
- Number of vehicles (fleet planning)

Vehicle Routing Variants

- Traveling salesman problem
- Pure vehicle routing
- Linehaul-backhaul
- Vehicle routing with time windows
- Vehicle routing and scheduling
- Mixed pickup and delivery

Vehicle Routing Algorithms Classification

- Route generating versus route selecting
- Basic methodology

Vehicle Routing Algorithms Classification by Required Input

- Route generating
  - Large variety
  - Tightly capacitated
- Route selecting
  - Set partitioning algorithm (SPP)
  - Complex costs and constraints

Vehicle Routing Algorithms Classification by Methodology

- VRP problem based
  - Sweep, savings, exchange, nearest neighbor
- Optimization based
  - GAP, SPP, k-trees
  - Simulated annealing, tabu search
- Artificial intelligence based
  - Genetic search

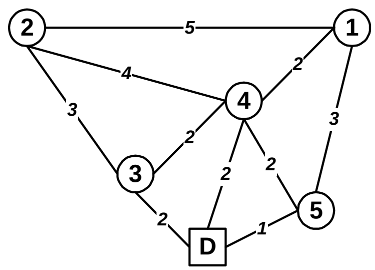
Pure Vehicle Routing – VRP Overview

- Problem definition
- Mathematical formulation
- Optimal algorithms
- Heuristic algorithms
- Conclusions

VRP Problem Definition

- Single depot
- N customers (x_i, y_i, dem_i)
  - Known location and demand
- K vehicles (cap_k)
  - Known and equal size
- Minimum travel cost objective
- Travel distance norm

Vehicle Routing Example: Distance Data



Truck capacity = 15

Vehicle Routing Example: Demand Data

Customer	Demand	Distance to Depot
1	3	4
2	6	5
3	4	2
4	7	2
5	6	1

Truck capacity = 15

VRP Route Generating Algorithms

- Route construction
  - Clarke and Wright, nearest neighbor
- Route improvement
  - Exchange improvements
- Two phase algorithms
  - Cluster first, route second
    - Sweep A, Fisher and Jaikumar (GAP)
  - Route first, cluster second
    - Sweep B, great tour

Nearest Neighbor

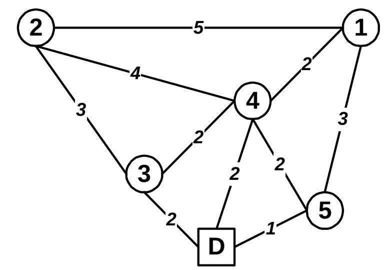
1. Start a new route with the depot
2. Find nearest unvisited customer
3. If customer demand is less than remaining vehicle capacity, append customer, otherwise go to 5
4. If all customers are visited stop
5. If maximum number vehicles is used stop, else go to 1

Sweep Algorithm Variants

- Cluster first, route second sweep (variant A)
  - Ray determines clusters
  - TSP construction routines to route each cluster
- Route first sweep (variant B)
  - Ray determines routes
  - TSP improvement routines for each initial route

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Vehicle Routing Example: Distance Data



Truck capacity = 15

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Vehicle Routing Example: Demand Data

Customer	Demand	Distance to Depot
1	3	4
2	6	5
3	4	2
4	7	2
5	6	1

Truck capacity = 15

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Number of Routes

- Maximum number of routes usually parameter (for example max = 3)
- Minimum number of routes

$$\min \# routes = \left\lceil \frac{\sum_{i=1}^{N} q_i}{Cap} \right\rceil = \left\lceil \frac{26}{15} \right\rceil = \lceil 1.73 \rceil = 2$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Nearest Neighbor

- Customer 5, distance 1, truck load 6
- Customer 4, distance 2, truck load 13
- Customer 1, distance 2, load 16 > 15
  route length = 5, new route
- Customer 3, distance 2, truck load 4
- Customer 2, distance 3, truck load 10
- Customer 1, distance 5, truck load 13
  route length = 14, total length = 19

Cluster-First Sweep (Variant A)

- Start direction = east

$$D-5[6]-1[9]-D \qquad L_1=8$$

$$D-4[7]-2[13]-D \qquad L_2=11$$

$$D-3[4]-D \qquad L_3=4$$

$$L=\sum_r L_r = 8+11+4=23$$

Note initial groupings are clusters only without point sequence

Cluster-First Sweep (Variant A)

- Start direction = north

$$D-2[6]-3[10]-D \qquad L_1=10$$

$$D-5[6]-1[9]-D \qquad L_2=8$$

$$D-4[7]-D \qquad L_3=4$$

$$L=\sum_r L_r = 10+8+4=22$$

Note initial groupings are clusters only without point sequence

Route-First Sweep (Variant B)

- Start direction = east

$$D-5-1-4-2-3-D \qquad (TSP)$$

$$D-5[6]-1[9]-D \qquad L_1=8$$

$$D-4[7]-2[13]-D \qquad L_2=11$$

$$D-3[4]-D \qquad L_3=4$$

$$L=\sum_r L_r = 8+11+4=23$$

Clarke and Wright Savings

- Maximum number of routes = K
- Serial variant
  - One route at-a-time
  - Simpler implementation
- Parallel variant
  - No more than K routes at-a-time
  - More complex programming

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Clarke and Wright Savings Initial Savings (Common)

- Compute savings for every feasible pair of points

$$q_1 + q_2 = 3 + 6 = 9 \leq 15$$
$$s_{12} = d_{10} + d_{02} - d_{12} = 4 + 5 - 5 = 4$$

$$q_1 + q_4 = 3 + 7 = 10 \leq 15$$
$$s_{14} = d_{40} + d_{01} - d_{14} = 4 + 2 - 2 = 4$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Clarke and Wright Savings Initial Savings Matrix

	1	2	3	4	5
1		4	2	4	2
2	4		4	3	0
3	2	4		2	0
4	4	3	2		1
5	2	0	0	1	

Selected pair (1-2) with ties broken by first encountered maximum savings

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Serial Clarke and Wright Savings: Second Iteration

- Copy corresponding savings rows and columns, eliminating infeasible combinations
- (4-1-2),(1-2-4) = 3+6+7=16 > 15

	(1-2)	3	4	5
(1-2)		4		0
3	2			
4				
5	2			

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Serial Clarke and Wright Savings: Third Iteration

- Capacity infeasibilities eliminate all positive savings
- No feasible or profitable extension of the route, so start a new route

	(1-2-3)	4	5
(1-2-3)			
4			
5			

Serial Clarke and Wright Savings: Third Iteration

- Eliminate all visited point rows and columns from the savings matrix and restart

	4	5
4		1
5	1	

$$L = L_{(1-2-3)} + L_{(4-5)} = 14 + 5 = 19$$

Parallel Clarke and Wright Savings: Second Iteration

- Maximum two routes
- Copy corresponding rows and columns, eliminating infeasible combinations

	(1-2)	3	4	5
(1-2)		4		0
3	2		2	0
4		2		1
5	2	0	1	

Parallel Clarke and Wright Savings: Third Iteration

- All partial routes remain "extendable"

	(1-2-3)	4	5
(1-2-3)			
4			1
5		1	

$$L = L_{(1-2-3)} + L_{(4-5)} = 14 + 5 = 19$$

Improvement Algorithms

- Intra-route improvements (TSP)
  - Always feasible
  - 2 exchange, chain exchange, 3 exchange
- Inter-route improvements
  - Test and make only feasible exchanges
  - Move (one point to another route)
  - Swap (exchange two points between two routes)

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

VRP GAP Formulation: Clustering

$$\begin{aligned}
\min \quad & \sum_k f_k(y_{ik}) \\
s.t. \quad & \sum_{i=1}^{N} dem_i y_{ik} \le cap_k & \forall k \\
& \sum_k y_{0k} = K \\
& \sum_k y_{ik} = 1 & i = 1..N \\
& y_{ik} \in \{0,1\}
\end{aligned}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

VRP GAP Formulation: Routing

$$\begin{aligned}
f(y_{ik}) = \quad \min \quad & \sum_i \sum_j c_{ij} x_{ijk} \\
s.t. \quad & \sum_i x_{ijk} = y_{jk} & \forall jk \\
& \sum_j x_{ijk} = y_{ik} & \forall ik \\
& \sum_{i \in S} \sum_{j \in S} x_{ijk} \le |S| - 1 & S \subseteq N(y_{ik}) \\
& x_{ijk} \in \{0,1\}
\end{aligned}$$

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

VRP GAP Formulation: Linearized Clustering

$$f(y_{ik}) = \sum_i d_{ik} y_{ik}$$

$$d_{ik} = c_{0i} + c_{is_k} - c_{s_k 0}$$

- Heuristic approximation
- s_k seed (customer) for route k
- Many variants for d_{ik} and seed selection

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

GAP Characteristics

- Alternative generating algorithm
- Predominant importance of capacity constraints
- Mathematical programming based algorithm (requires solver)
- Strongly capacitated or strongly combinatorial problems

Set Partitioning Problem (SPP) Definition

- Every column j = feasible alternative action
- Every row i = service request
- Minimize overall cost while servicing all requests

Set Partitioning Notation

- a_{ij} = 1 if alternative server j satisfies customer request i
- c_j = cost of alternative j
- p_i = cost estimate for servicing customer request i
- x_j = 1 if server j is selected (enabled)

Set Partitioning Formulation

$$\min \sum_{j=1}^{N} c_j x_j$$

$$s.t. \sum_{j=1}^{N} a_{ij} x_j = 1 \qquad i = 1..M$$

$$x_j \in \{0,1\}$$

Set Partitioning Characteristics

- Alternative selecting algorithm
- Accurate costs and feasibility constraints
- Optimal solution for “small” problem sizes (IP solver)
- Efficient column generation and pricing
- Complex problems

Set Partitioning Algorithm

1. Start with a feasible partition J^+
2. Determine row prices by allocating column prices “equitable” such that

$$c_j^+ = \sum_{i=1}^{M} a_{ij} p_i$$

VRP example: $p_i = \dfrac{dem_i \cdot d_{0i}}{\sum_{i \in J_i} dem_i \cdot d_{0i}} c_j^+$

Set Partitioning Algorithm (2)

3. Generate and evaluate new column j, if

$$c_j - \sum_{i=1}^{M} a_{ij} p_i \le 0$$

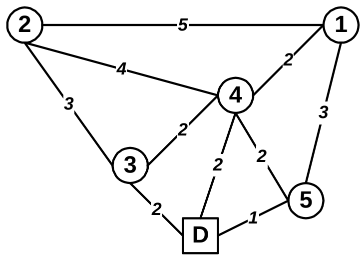
add column j to the partitioning problem

4. If enough new columns are added, solve the partitioning problem else go to step 3

Set Partitioning Algorithm (3)

5. If all columns have been evaluated or solution is within tolerance, stop else go to step 2

Set Partitioning Example: Distance Data



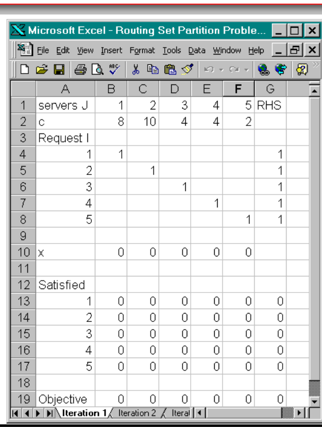
Truck Capacity = 15

Set Partitioning Example: Demand Data

Customer	Demand	Distance to Depot	Distance * Demand
1	3	4	12
2	6	5	30
3	4	2	8
4	7	2	14
5	6	1	6

Truck Capacity = 15

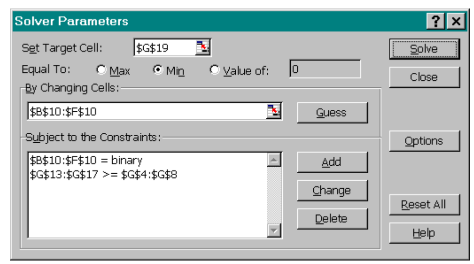
Set Partition Example: One-Stop Routes Excel Data


Microsoft Excel - Routing Set Partition Proble...

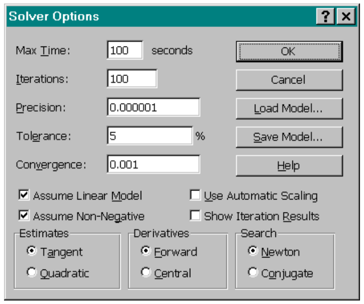
	A	B	C	D	E	F	G
1	servers J	1	2	3	4	5	RHS
2	c	8	10	4	4	2	
3	Request I						
4	1	1					1
5	2		1				1
6	3			1			1
7	4				1		1
8	5					1	1
9							
10	x	0	0	0	0	0	
11							
12	Satisfied						
13	1	0	0	0	0	0	0
14	2	0	0	0	0	0	0
15	3	0	0	0	0	0	0
16	4	0	0	0	0	0	0
17	5	0	0	0	0	0	0
18							
19	Objective	0	0	0	0	0	0

Iteration 1 / Iteration 2 / Iteral

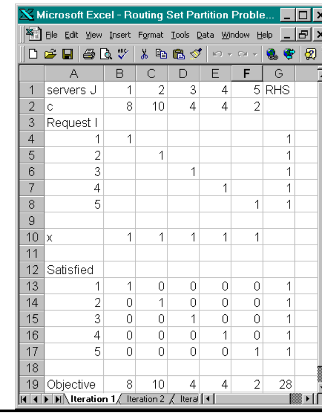
Set Partition Example: One-Stop Routes Excel Solver



Set Partition Example: One-Stop Routes Excel Solver Options



Set Partition Example: One-Stop Routes Excel Solution

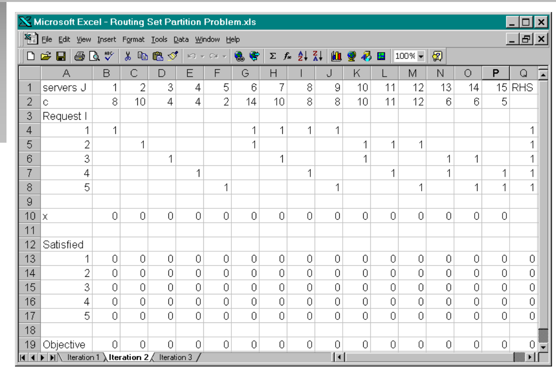


	A	B	C	D	E	F	G
1	servers J	1	2	3	4	5	RHS
2	c	8	10	4	4	2	
3	Request I						
4	1	1					1
5	2		1				1
6	3			1			1
7	4				1		1
8	5					1	1
9							
10	x	1	1	1	1	1	
11							
12	Satisfied						
13	1	1	0	0	0	0	1
14	2	0	1	0	0	0	1
15	3	0	0	1	0	0	1
16	4	0	0	0	1	0	1
17	5	0	0	0	0	1	1
18							
19	Objective	8	10	4	4	2	28

Set Partitioning Example: Two Customer Routes

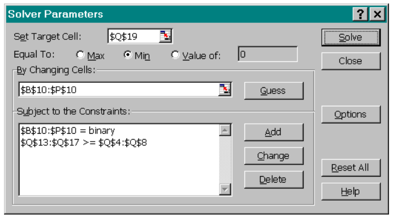
Route	Customer One	Customer Two	Length	Savings
6	1	2	14	4
7	1	3	10	2
8	1	4	8	4
9	1	5	8	2
10	2	3	10	4
11	2	4	11	3
12	2	5	12	0
13	3	4	6	2
14	3	5	6	0
15	4	5	5	1

Set Partition Example: Two Customer Routes Excel Data

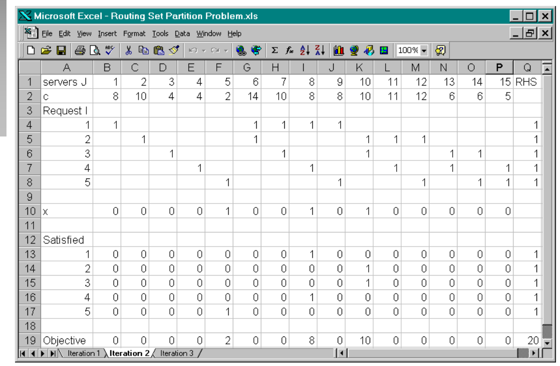


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	servers J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	RHS
2	c	8	10	4	4	2	14	10	8	8	10	11	12	6	6	5	
3	Request I																
4	1	1					1	1	1	1							1
5	2		1				1				1	1	1				1
6	3			1				1			1			1	1		1
7	4				1				1			1		1		1	1
8	5					1				1			1		1	1	1
9																	
10	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11																	
12	Satisfied																
13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18																	
19	Objective	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Set Partition Example: Two Customer Routes Excel Solver



Set Partition Example: Two Customer Routes Excel Solution


Microsoft Excel - Routing Set Partition Problem.xls

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	servers J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	RHS
2	c	8	10	4	4	2	14	10	8	8	10	11	12	6	6	5	
3	Request I																
4	1	1					1	1	1	1							1
5	2		1				1				1	1	1				1
6	3			1				1			1			1	1		1
7	4				1				1			1		1		1	1
8	5					1				1			1		1	1	1
9																	
10	x	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	
11																	
12	Satisfied																
13	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
14	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
15	3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
16	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
17	5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
18																	
19	Objective	0	0	0	0	2	0	0	8	0	10	0	0	0	0	0	20

Iteration 1 Iteration 2 Iteration 3

Set Partitioning Example: Two Customer Row Prices

$$j = 5 \quad I_5 = \{5\} \quad c_5 = p_5 = 2$$

$$j = 8 \qquad I_8 = \{1,4\} \qquad j = 10 \qquad I_{10} = \{2,3\}$$

$$c_j = 8 \qquad p_1 + p_4 = 8 \qquad c_j = 10 \qquad p_2 + p_3 = 10$$

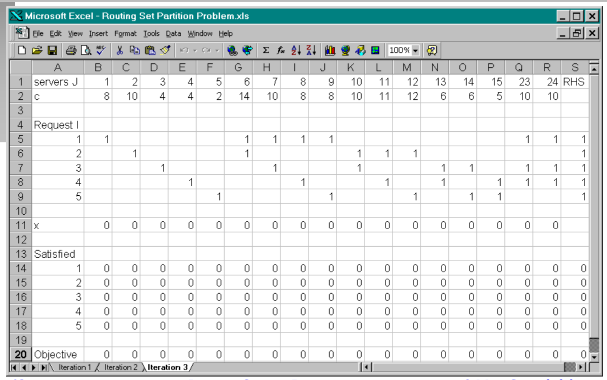
$$p_1 = \frac{12}{26} \cdot 8 = 0.46 \cdot 8 = 3.69 \qquad p_2 = \frac{30}{38} \cdot 10 = 0.79 \cdot 10 = 7.89$$

$$p_4 = \frac{14}{26} \cdot 8 = 0.54 \cdot 8 = 4.31 \qquad p_3 = \frac{8}{38} \cdot 10 = 0.21 \cdot 10 = 2.11$$

Set Partitioning Example: Three Customer Routes

Route	Customer One	Customer Two	Customer Three	Route Demand	Route Length	Savings
16	1	2	3	13	14	-0.31
17	1	3	2	13	16	-2.31
18	2	1	3	13	16	-2.31
19	1	2	5	14	16	-2.42
20	1	5	2	14	18	-4.42
21	2	1	5	14	14	-0.42
22	1	3	4	14	12	-1.89
23	1	4	3	14	10	0.11
24	3	1	4	14	10	0.11
25	1	3	5	13	12	-4.20
26	1	5	3	13	12	-4.20
27	5	1	3	13	10	-2.20

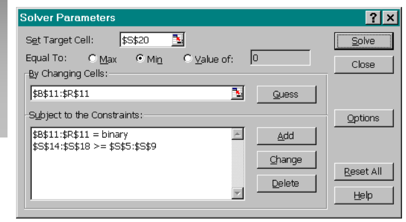
Set Partition Example: Three-Stop Routes Excel Data



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	servers J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	23	24	RHS
2	c	8	10	4	4	2	14	10	8	8	10	11	12	6	6	5	10	10	
3																			
4	Request I																		
5	1	1					1	1	1	1							1	1	1
6	2		1				1				1	1	1						1
7	3			1				1			1			1	1		1	1	1
8	4				1				1			1		1		1	1	1	1
9	5					1				1			1		1	1			1
10																			
11	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
12																			
13	Satisfied																		
14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19																			
20	Objective	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

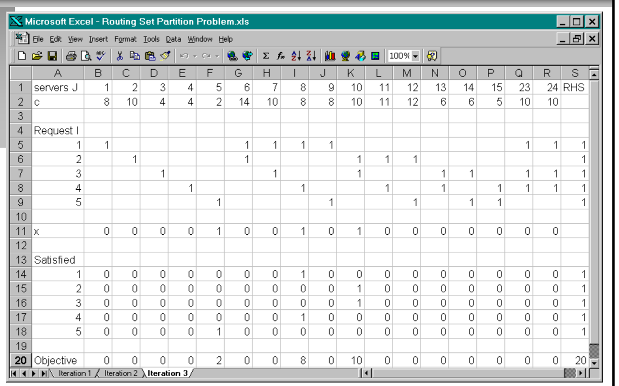
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Set Partition Example: Three-Stop Routes Excel Solver



25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Set Partition Example: Three-Stop Routes Excel Solution



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	servers J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	23	24	RHS
2	c	8	10	4	4	2	14	10	8	8	10	11	12	6	6	5	10	10	
3																			
4	Request I																		
5	1	1					1	1	1	1							1	1	1
6	2		1				1				1	1	1						1
7	3			1				1			1			1	1		1	1	1
8	4				1				1			1		1		1	1	1	1
9	5					1				1			1		1	1			1
10																			
11	x	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	
12																			
13	Satisfied																		
14	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
15	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
16	3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
17	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
18	5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
19																			
20	Objective	0	0	0	0	2	0	0	8	0	10	0	0	0	0	0	0	0	20

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Set Partitioning References

- *Cullen F., Jarvis J. and Ratliff H., (1981). "Set Partitioning Based Heuristics for Interactive Routing." Networks, Vol. 11, No. 2, pp. 125-143.*
- *Balas E. and Padberg M., (1976). "Set Partitioning: A Survey." SIAM Review, Vol. 18, No. 4, pp. 710-760.*

25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Linehaul-Backhaul Problem - VRPB

- #1 savings technique in routing
- Problem definition
- Mathematical formulation
- Heuristics algorithms
- Software
- Conclusions

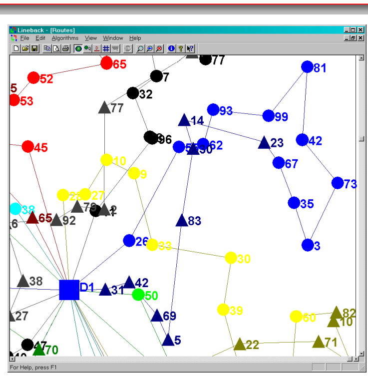
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

VRPB Problem Definition

- Single depot
- N_L customers (x_i, y_i, dem_i) and N_B suppliers (x_i, y_i, sup_i)
- M equal size vehicles (cap_j)
- Rear loaded vehicles
  - all customers before any supplier
- Minimize total travel distance
- Travel distance norm

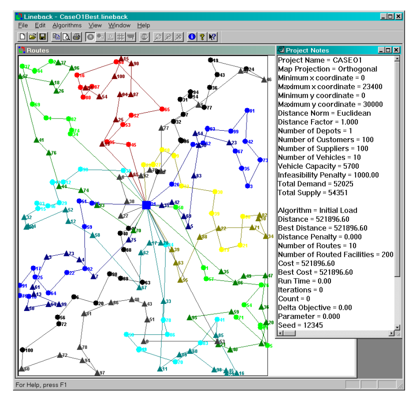
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

VRPB Route Illustration



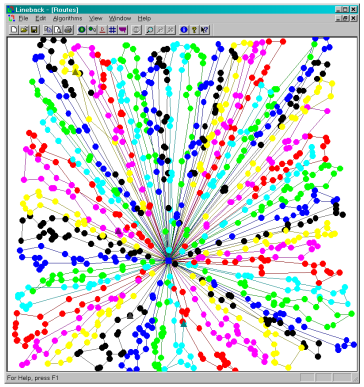
25-Mar-03 Logistics Systems Design © Marc Goetschalckx

Lineback Illustration: 100 x 100 x 10

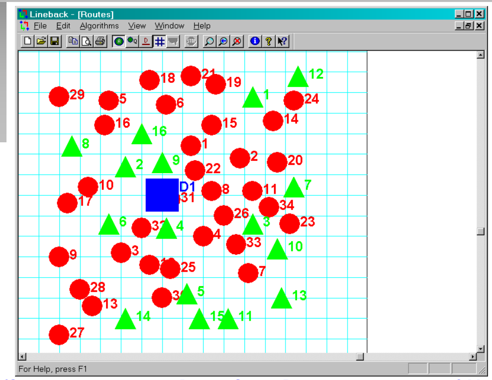


25-Mar-03 © Marc Goetschalckx

Lineback Illustration: 1024 x 4 x 60



VRPB Exercise: Case 51-66 Data



Vehicle Routing with Time Windows VRPTW

- Time window is a restriction on the time of visiting the customer
- Types of time windows
  - Exclusion versus mandatory
  - Single versus multiple
  - Hard versus soft

VRPTW Characteristics

- Much more difficult problem than classical VRP
  - No simple algorithms
  - No easy feasible test
  - No easy human intuition
  - No intuitive route structure
- Solution procedures use mathematical programming or stochastic search

Vehicle Routing References

- Golden, B. and A. Assad, (Editors.), 1988. Vehicle Routing: Methods and Studies. North Holland, Amsterdam.
- Halse, K. 1992. "Modeling and Solving Complex Vehicle Routing Problems." Ph.D. Dissertation, Technical University of Denmark.
- Ball, M., T. Magnanti, C. Monma, and G. Nemhauser (Editors), 1995. Network Routing. North-Holland, Amsterdam.

