Branch and Bound for TSP

Input: A distance matrix D for a graph $G = (V, E)$ with n vertices. If the edge (i, j) is in E then $d(i, j)$ equals the distance from i to j, otherwise $d(i, j)$ equals ∞.

Output: The length $d(C)$ of a shortest circuit C in G visiting each vertex exactly once, and an n-vector of edges $(\{v_1, w_1\}, ..., \{v_n, w_n\})$ constituting C.

Method: Branch and Bound. Bounding function, search trategy, and branching strategy is described below.

Formulation as a mathematical program:

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

s.t.

$$\sum_{k=1}^{i-1} x_{ki} + \sum_{k=i+1}^{n} x_{ik} = 2, \qquad i \in \{1, ..., n\}$$

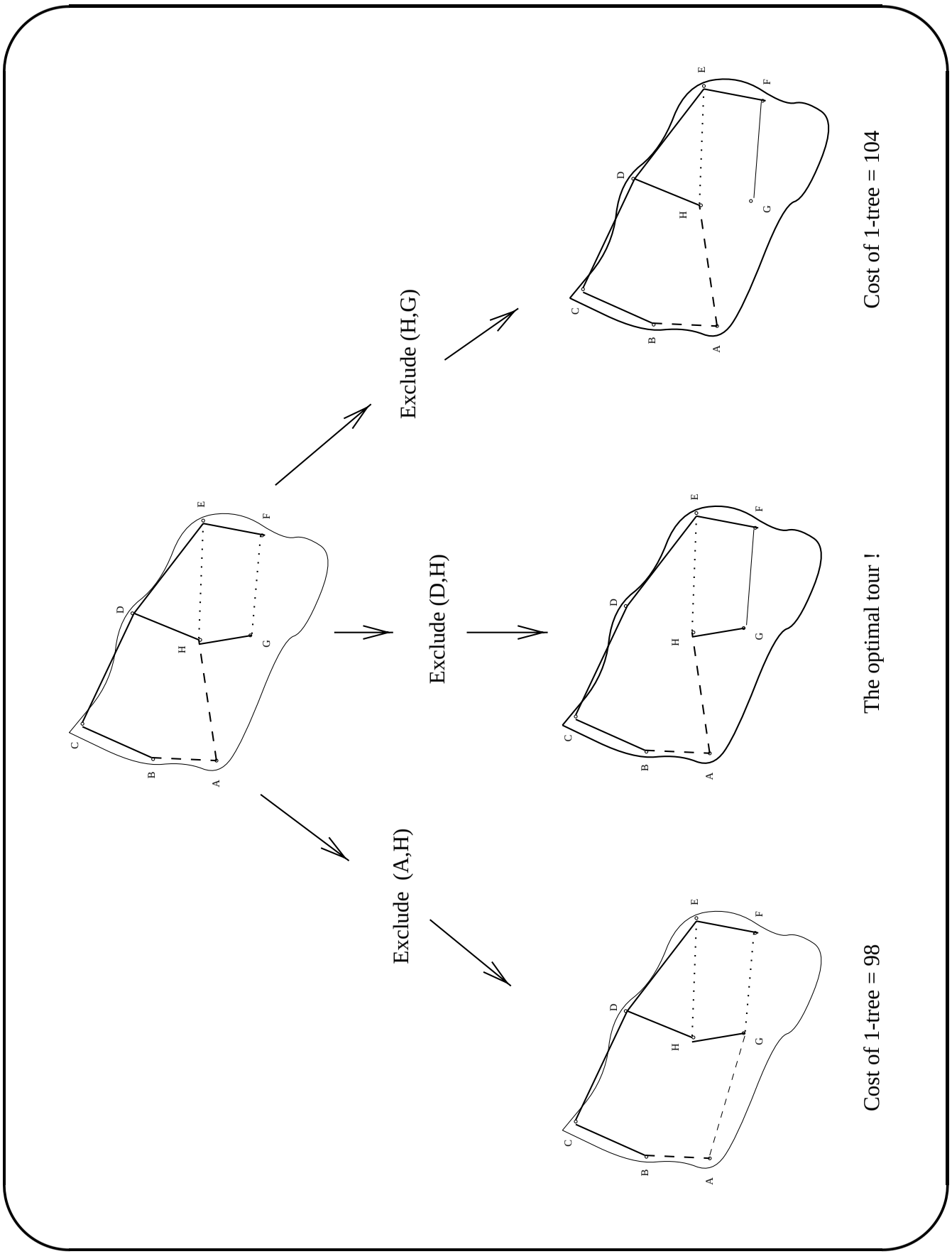
$$\sum_{i,j \in Z} x_{ij} < |Z| \qquad \emptyset \subset Z \subset V$$

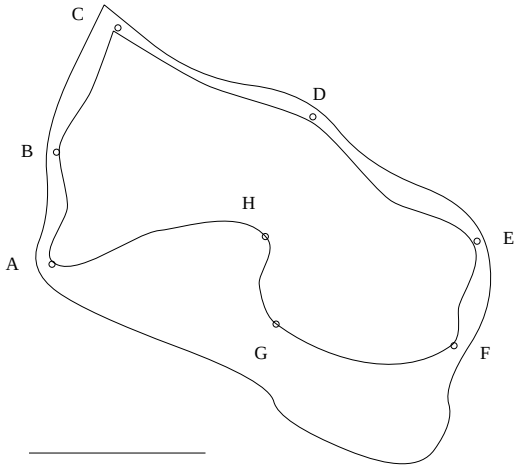
$$x_{ij} \in \{0, 1\}, \qquad i, j \in \{1, ..., n\}$$

Jens Clausen IMM

TSP - B& B components.

- **Bounding function:** Select a vertex - denote this "#1". Remove this and all incident edges from G. Find a minimum spanning tree T_{rest} for the graph $G \setminus$ "#1". Add the two shortest edges e_1, e_2 incident with "#1" to T_{rest} - the result is the 1-tree T_{et} in G wrt.. "#1". $d(T_{et})$ is a lower bound for the length of the optimal TSP tour.
- **Branching:** Select a vertex v with degree $deg(v)$ (i.e. number of incident edges) in T_{et} larger than 2. Construct $deg(v)$ different subproblems by removing exactly one of the $deg(v)$ edges in T_{et} incident with v from G.
- **Search strategy:** Best First - examine the subproblem with smallest lower bound first.
- Improved bounding function: Consider the degrees $deg(.)$ in T_{et} of each vertex. Set $\pi(i) = deg(i) - 2$. Define a new distance matrix D' by $d'(i,j) = d(i,j) + \pi(i) + \pi(j)$. Find T_{et} wrt. D'. This gives a new (and most often better) lower bound and may be repeated. Note that $\sum \pi(i) = 0$ and hence $d(C) = d'(C)$ for any TSP-tour C.

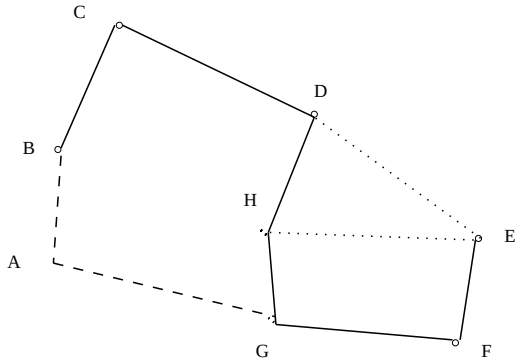

Exclude (H,G)
Exclude (D,H)
Exclude (A,H)
Cost of 1-tree = 104
The optimal tour !
Cost of 1-tree = 98
A
B
C
D
E
F
G
H



Optimal tour, cost = 100

	A	B	C	D	E	F	G	H
A	0	11	24	25	30	29	15	15
B	11	0	13	20	32	37	17	17
C	24	13	0	16	30	39	29	22
D	25	20	16	0	15	23	18	12
E	30	32	30	15	0	9	23	15
F	29	37	39	23	9	0	14	21
G	15	17	29	18	23	14	0	7
H	15	17	22	12	15	21	7	0

(a)



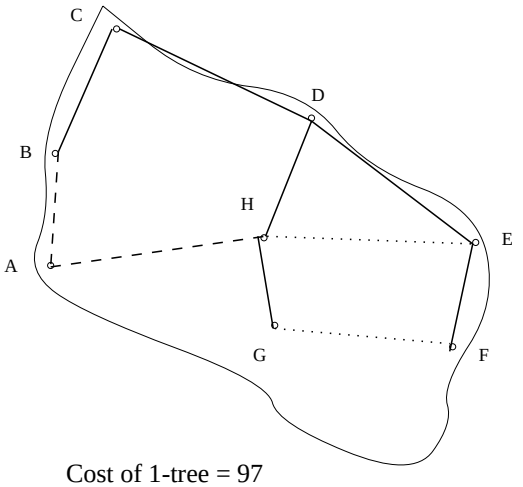
Tree in rest of G

Edge left out by Kruskal's MST algorithm

1-tree edge

Cost of 1-tree = 97

(b)



Cost of 1-tree = 97

Modified distance matrix:

	A	B	C	D	E	F	G	H
A	0	11	24	25	29	29	16	15
B	11	0	13	20	31	37	18	17
C	24	13	0	16	29	39	30	22
D	25	20	16	0	14	23	19	12
E	29	31	29	14	0	8	23	14
F	29	37	39	23	8	0	15	21
G	16	18	30	19	23	15	0	8
H	15	17	22	12	14	21	8	0

(c)

The Improved Bounding - Details I

Idea: Reformulate the problem and relax some constraints

The real problem is:

Find the shortest 1-træ, which is also a TSP-tour
i.e. **in which all vertices have degree 2.**

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

$$\sum_{k=1}^{i-1} x_{ki} + \sum_{k=i+1}^{n} x_{ik} = 2, \qquad i \in \{1, ..., n\}$$

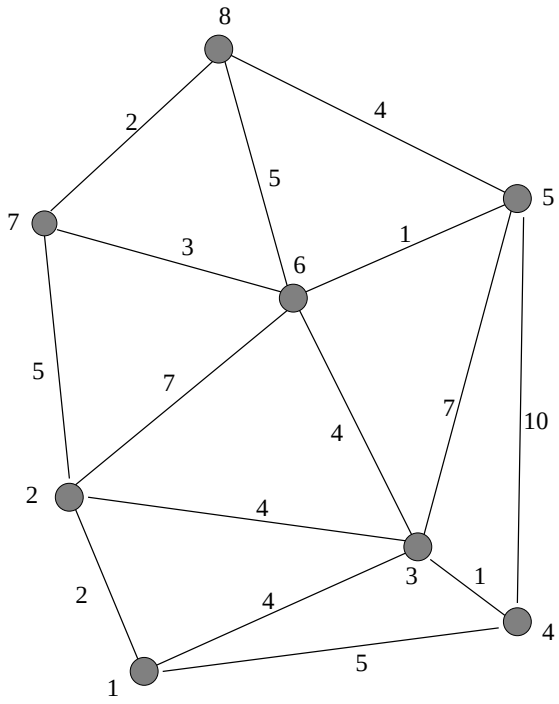
"The edges with $x_{ij} = 1$ constitute a 1-tree"

$$x_{ij} \in \{0, 1\}, \qquad i, j \in \{1, ..., n\}$$

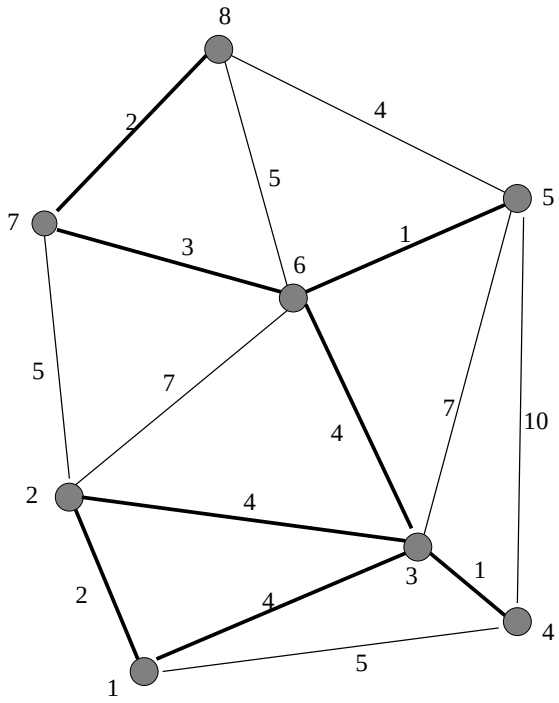
If the constratins are relaxed with multipliers$\pi_1, ..., \pi_n$ the objective function becomes:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (d_{ij} + \pi_i + \pi_j) x_{ij} - 2 \sum_{i=1}^{n} \pi_i$$

Hence: The improved 1-tree bound is nothing but the value of the Lagrangean relaxation of our problem for the given set $\pi_1, ..., \pi_n$ (which incidently sums to).



G(0)

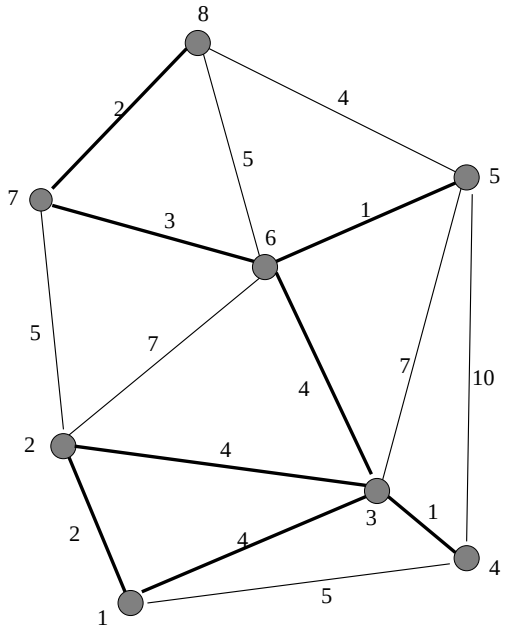


G(pi1)

(pi1 , ... , pi8) = (deg1 - 2 , ... , deg8 - 2) =

= (0 , 0 , 2 , -1 , -1 , 1 , 0 , -1)

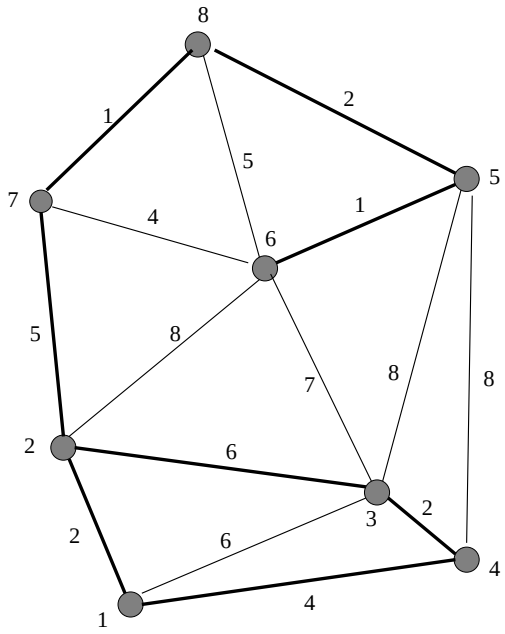
The Improved Bounding - Example I



G(pi1)

(pi1 , ... , pi8) = (deg1 - 2 , ... , deg8 - 2) =

= (0 , 0 , 2 , -1 , -1 , 1 , 0 , -1)

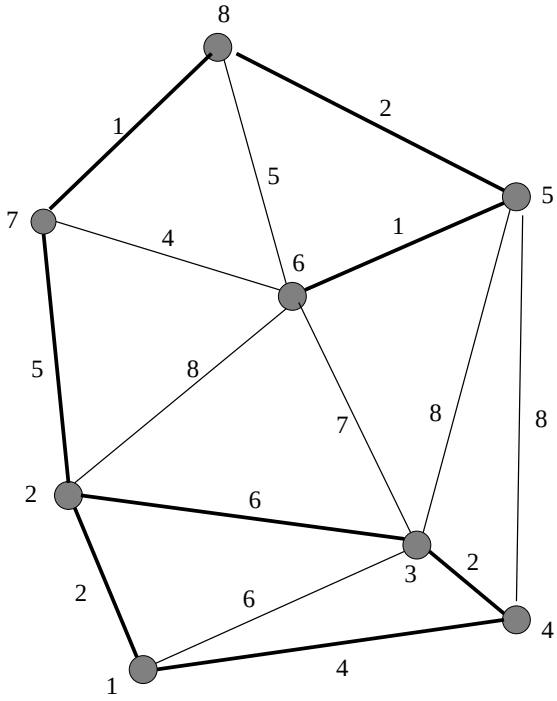


G(pi2)

(pi1 , ... , pi8) = (deg1 - 2 , ... , deg8 - 2) =

= (0 , 1 , 0 , 0 , 0 , -1 , 0 , 0)

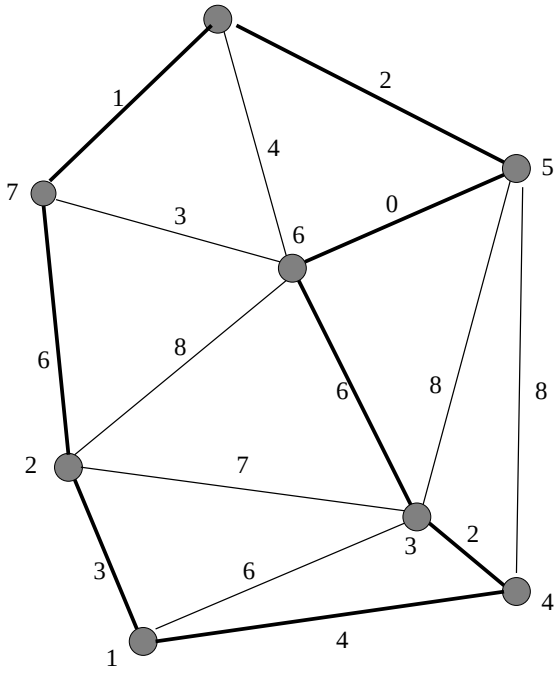
The Improved Bounding - Example II



G(pi2)

(pi1 , ... , pi8) = (deg1 - 2 , ... , deg8 - 2) =

= (0 , 1 , 0 , 0 , 0 , -1 , 0 , 0)



G(pi3)

Branch and Bound for ATSP

Input: A distance matrix D for a *digraf* $G = (V, E)$ with n vertices (i.e. $d(a, b)$ may be different $d(b, a)$). If the edge (i, j) is in E the $d(i, j)$ equals the distance from i to j, otherwise $d(i, j)$ equals ∞.
Output: The length $d(C)$ of a shortest directed circuit C in G visiting each vertex exactly once, and an n-vector of edges $(\{v_1, w_1\}, ..., \{v_n, w_n\})$ constituting C.
Method: Branch and Bound. Bounding function, search trategy, and branching strategy is described below.

Formulation as a mathematical program:

$$min \sum_{i=1}^{n} d_{ij} x_{ij}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i \in \{1, ..., n\}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j \in \{1, ..., n\}$$

$$\sum_{i,j \in Z} x_{ij} < |Z| \qquad \emptyset \subset Z \subset V$$

$$x_{ij} \in \{0, 1\}, \qquad i, j \in \{1, ..., n\}$$

ATSP - example

$$\begin{pmatrix} - & 3 & 93 & 13 & 33 & 9 \\ 4 & - & 77 & 42 & 21 & 16 \\ 45 & 17 & - & 36 & 16 & 28 \\ 39 & 90 & 80 & - & 56 & 7 \\ 28 & 46 & 88 & 33 & - & 25 \\ 3 & 88 & 18 & 46 & 92 & - \end{pmatrix}$$

Lower bound calculation by reduction::

$$\begin{pmatrix} - & 0 & 75 & 2 & 30 & 6 \\ 0 & - & 58 & 30 & 17 & 12 \\ 29 & 1 & - & 12 & 0 & 12 \\ 32 & 83 & 58 & - & 49 & 0 \\ 3 & 21 & 48 & 0 & - & 0 \\ 0 & 85 & 0 & 35 & 89 & - \end{pmatrix}$$

$$\text{LB} =$$
$$(3 + 4 + 16 + 7 + 25 + 3) + (0 + 0 + 15 + 8 + 0 + 0).$$

ATSP - B & B components.

- **Bounding function:** Perform row and column reductions as for the assignment problem - subtract the smallest element in each row from all the row elements, and then the smallest element in each column of the resulting matrix from all column elements. The sum of the subtracted elements constitutes a lower bound for the length of the optimal ATSP-tour.
- **Branching:** Find that 0-position (i, j) in the reduced matrix, for which the sum of the next-smallest elements in row i and column j is largest("the most expensive edge to leave out in a new solution"). Construct two subproblems: one in which (i, j) is forced into the solution (remove row i and column j from the matrix and set $d(j, i)$ equal to ∞), and one in which (i, j) is excluded from the solution (set $d(i, j)$ equal to ∞).
- **Search strategy:** Depth First: - examine the subproblem with the *included* edge first. Because (i, j) is *most expensive* to leave out, there is a fair chance that the other subproblems is fathomed in a later iteration.

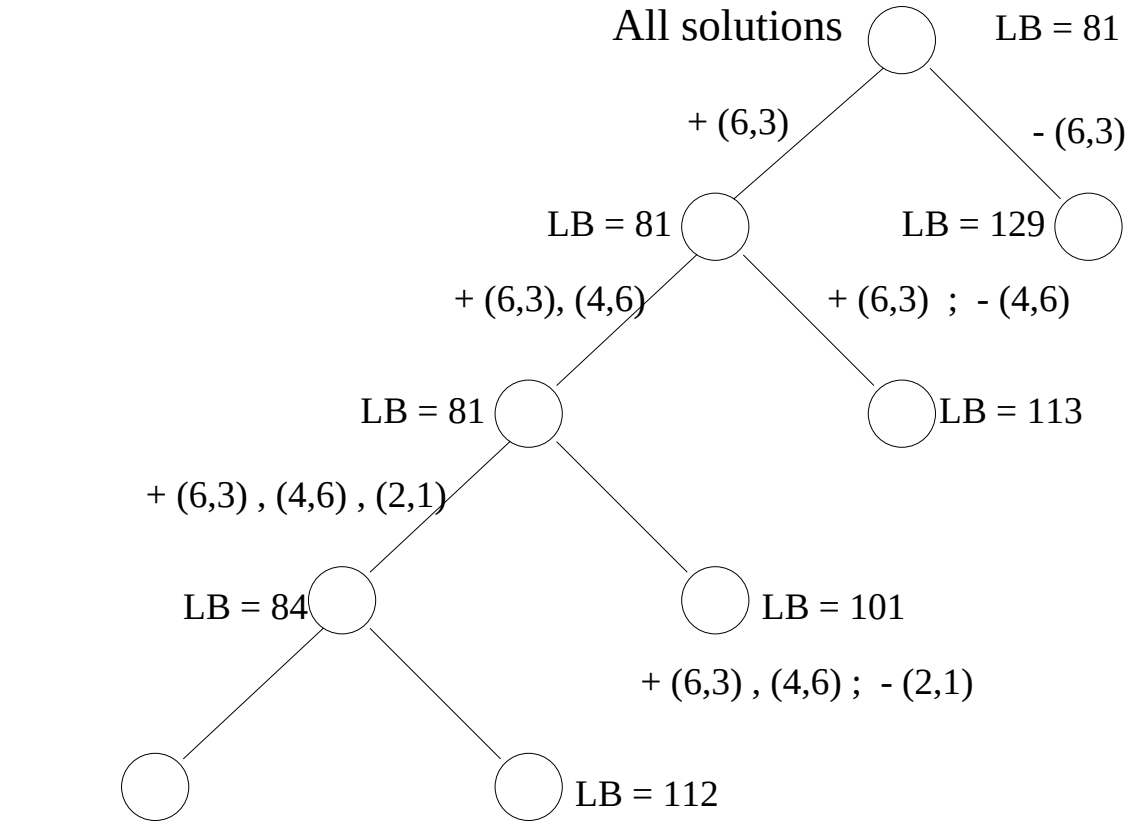
ATSP - an example.

$$\begin{pmatrix} - & 0 & 75 & 2 & 30 & 6 \\ 0 & - & 58 & 30 & 17 & 12 \\ 29 & 1 & - & 12 & 0 & 12 \\ 32 & 83 & 58 & - & 49 & 0 \\ 3 & 21 & 48 & 0 & - & 0 \\ 0 & 85 & 0 & 35 & 89 & - \end{pmatrix}$$

Branching from (6,3) yields:

$$\begin{pmatrix} - & 0 & 2 & 30 & 6 \\ 0 & - & 30 & 17 & 12 \\ 29 & 1 & 12 & 0 & - \\ 32 & 83 & - & 49 & 0 \\ 3 & 21 & 0 & - & 0 \end{pmatrix} \quad \textit{and} \quad \begin{pmatrix} - & 0 & 75 & 2 & 30 & 6 \\ 0 & - & 58 & 30 & 17 & 12 \\ 29 & 1 & - & 12 & 0 & 12 \\ 32 & 83 & 58 & - & 49 & 0 \\ 3 & 21 & 48 & 0 & - & 0 \\ 0 & 85 & - & 35 & 89 & - \end{pmatrix}$$

ATSP - search tree with solution.



Solution with value 104