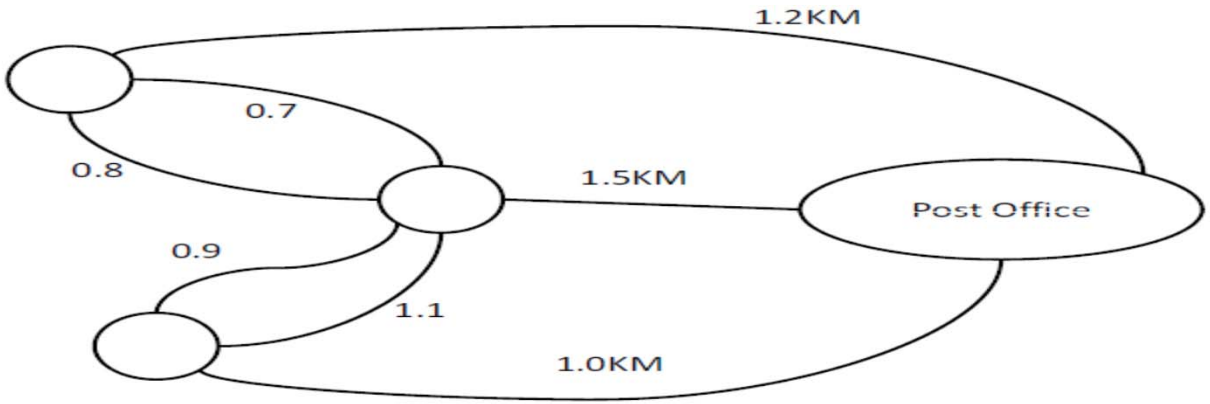
# Lecture 9: Chinese Postman Problem

- Problem:  Find a minimum length closed path (from and back to the post office), with repeated arcs as necessary, which contains every arc of a given undirected network.



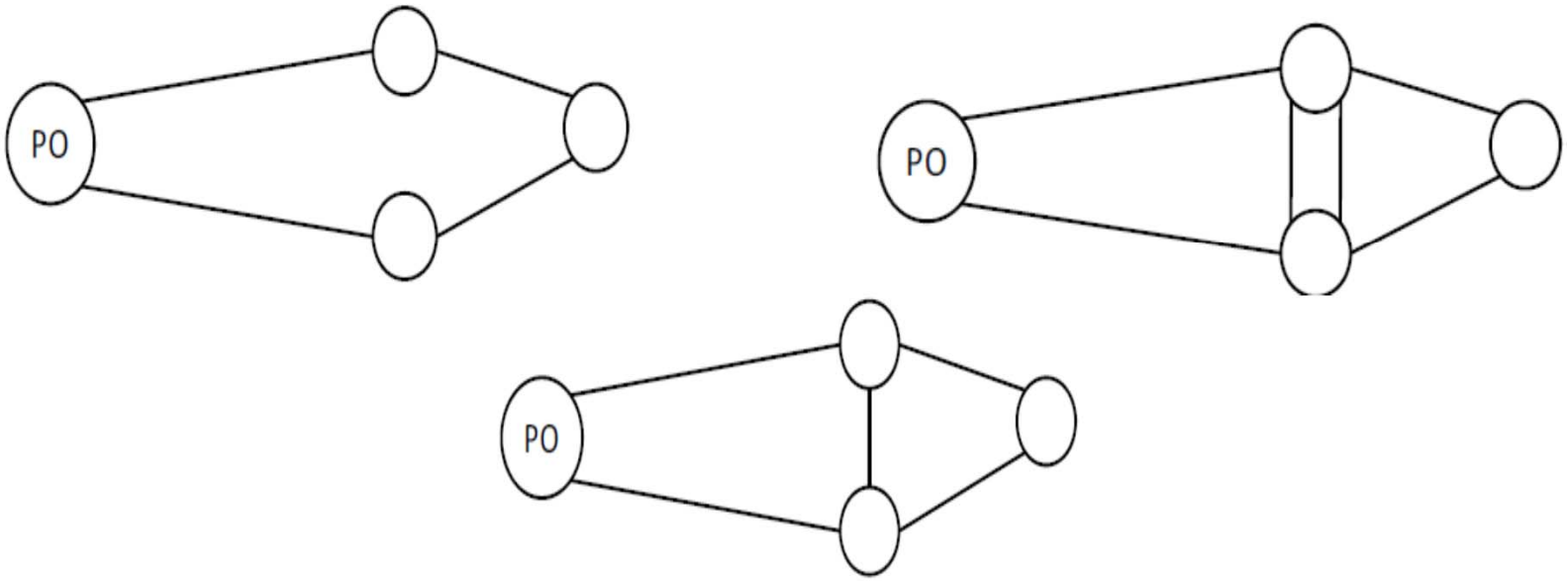

Mei-Ko Kwan, “Graph Programming Using Odd and Even Points", Chinese Math, (1962), 273-277.

# Observation

$$\text{Low bound} = \sum_{(i,j)\in A} l_{ij}$$

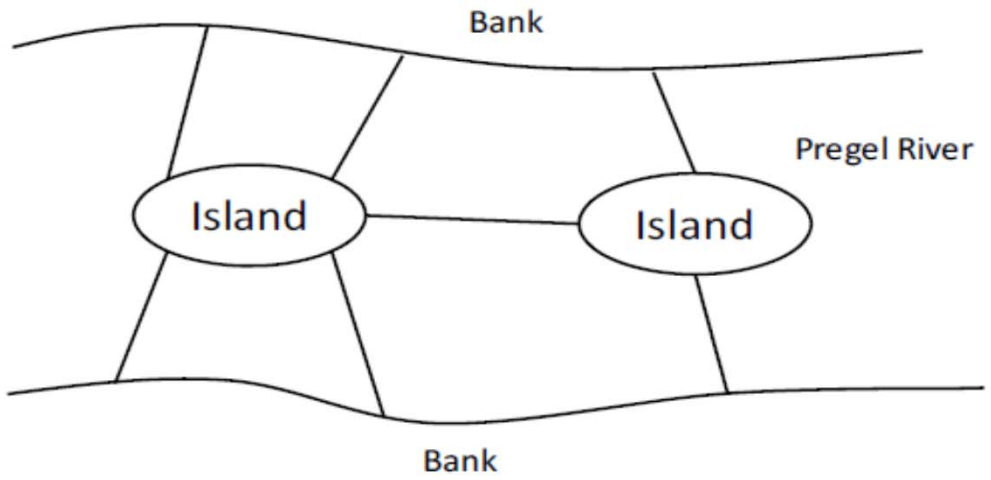
where

$$l_{ij} = \text{length of arc } (i,j) \text{ in } G(N,A)$$



# Basic Concept

- Königsberg Bridge Problem




- Euler Path (Euler graph)

  Given a graph $G$, does there exist a closed path which contains each arc exactly once?

# Main Theorem

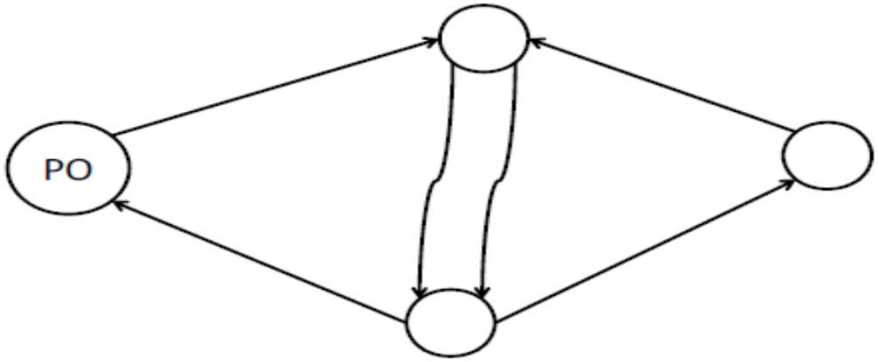
- Theorem 9.1 (Chapter 1)

  A graph G is Eulerian if and only if G is connected and each node of G has an even degree.

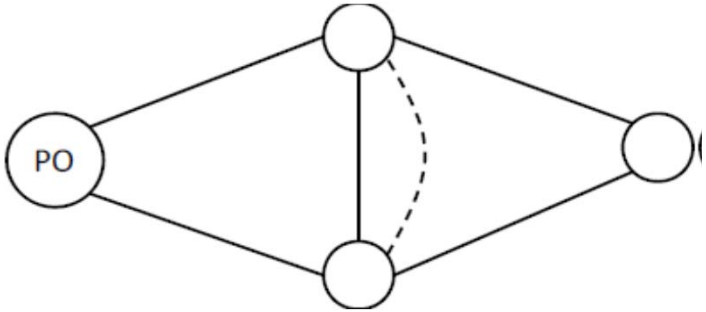
  Proof: Homework

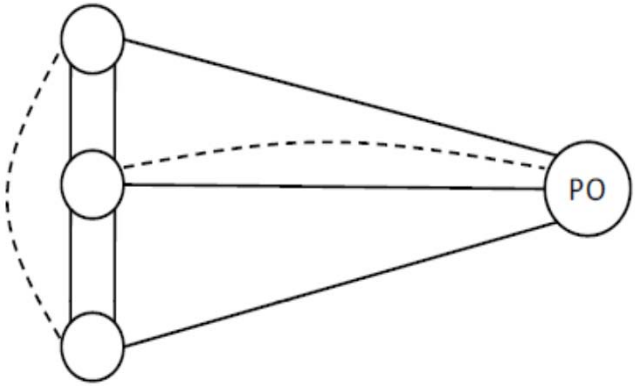
# Questions

- Q1: How to find the Euler path for an Eulerian graph?




- Q2: How to make a graph Eulerian?



# Main Results

- Theorem: There are an even number of nodes with odd degrees in a connected graph.

- Theorem:(Chinese Postman's Problem)
  Pairing the odd degree nodes with the shortest distance in between results in an Eulerian graph that provides solution to the Chinese Postman's Problem.

# Chinese Postman's Algorithm

*Step 1 (Identification of odd Nodes)* Identify the nodes of odd degree in the graph $G$. If there are none, go to Step 4.

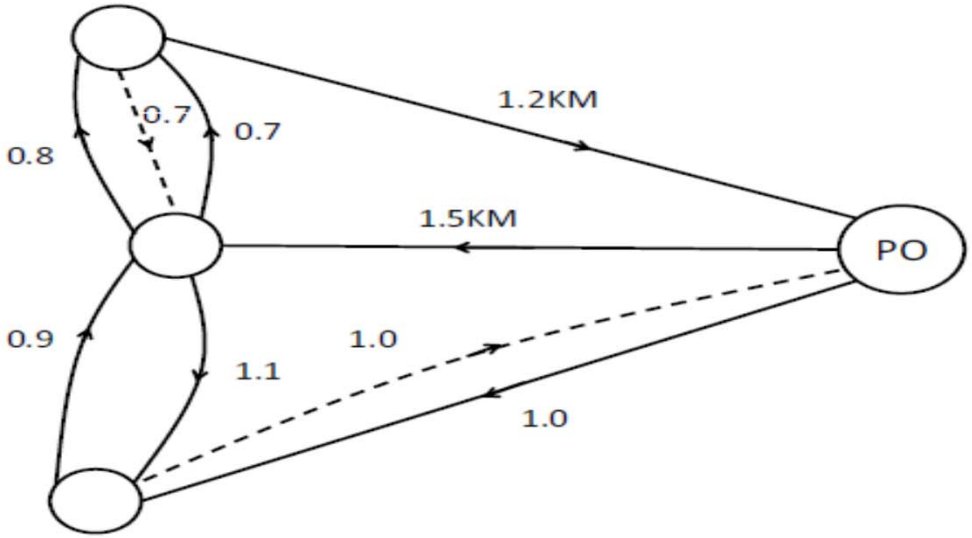
*Step 2 (Shortest Paths)* Compute the shortest paths between all pairs of odd-nodes.

# Chinese Postman's Algorithm

*Step 3 (Weighted Matching)* Partition the odd-nodes into pairs, so that the sum of the lengths of the shortest paths joining the pairs is minimal. Do this by solving a weighted matching problem over the complete graph $G^*$ whose nodes are the odd-nodes of the network, and in which $w_{ij}$, the weight of arc $(i, j)$, is given by the relation $w_{ij} = M - a_{ij}$ where $a_{ij}$ is the length of a shortest path between $i$ and $j$, and $M$ is a large number. (Note that there is a complete matching in a complete graph with an even number of nodes.) The arcs of $G$ in the paths identified with arcs of the matching are arcs which should be traversed twice. Duplicate these arcs in $G$.

*Step 4 (Construction of Tour)* Use any efficient procedure to construct an Euler path in $G$. //

# Example




$$\text{Total Traveling distance}$$
$$= 1.0 + 0.9 + 0.8 + 0.7 + 0.7 + 1.2 + 1.5 + 1.1 + 1.0$$
$$= 8.9\ km$$

- Remaining Work:

  Nonbipartite (Max) Weighted Matching

# Nonbipartite Weighted Matching

- Primal Problem: For a given $G(N, A)$,

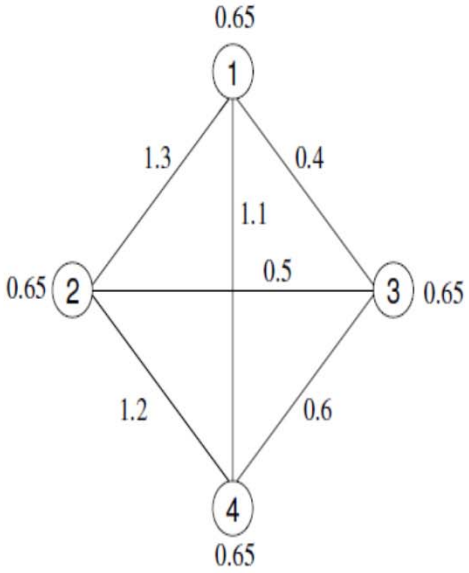
$$\max \sum_{(i,j)\in A} w_{ij} x_{ij}$$

s.t.

$$\begin{aligned} \sum_{j\in N}(x_{ij} + x_{ji}) &\leq 1, && \forall\, i \in N \\ \sum_{(i,j)\in\, A_k} x_{ij} &\leq n_k, && k = 1, ..., m \\ x_{ij} &\geq 0, && \forall\, (i,j) \in A \end{aligned}$$

where $\{R_1, ..., R_m\}$ forms an odd-set cover of $G$ with $2n_k + 1$ nodes in $R_k$ (i.e., $R_k$ is a potential blossom) for $k = 1, ..., m$ and $A_k = \{(i,j) \in A \mid i, j \in R_k\}$.

# Example 1




$$
\begin{array}{lrrrrrrl}
max & 1.3x_{12} & +0.4x_{13} & +1.1x_{14} & +0.5x_{23} & +1.2x_{24} & +0.6x_{34} & \\
s.t. & x_{12} & +x_{13} & +x_{14} & & & & \leq 1 \\
 & x_{12} & & & +x_{23} & +x_{24} & & \leq 1 \\
 & & x_{13} & & +x_{23} & & +x_{34} & \leq 1 \\
 & & & x_{14} & & +x_{24} & +x_{34} & \leq 1 \\
 & x_{12} & +x_{13} & & +x_{23} & & & \leq 1 \\
 & x_{12} & & +x_{14} & & +x_{24} & & \leq 1 \\
 & & x_{13} & +x_{14} & & & +x_{34} & \leq 1 \\
 & & & & x_{23} & +x_{24} & +x_{34} & \leq 1 \\
 & x_{12}, & x_{13}, & x_{14}, & x_{23}, & x_{24}, & x_{34} & \geq 0
\end{array}
$$

# Dual Problem

- Dual Problem:

$$\min \sum_{i \in N} u_i + \sum_{k=1}^{m} n_k z_k$$

s.t.

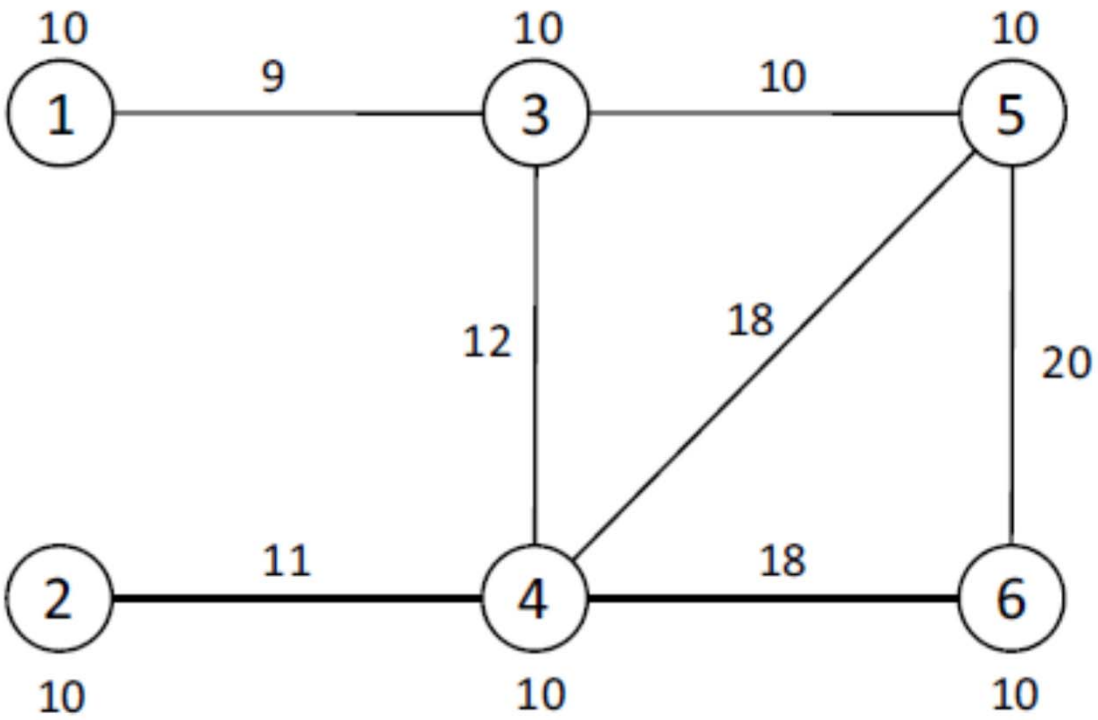
$$\begin{aligned} u_i + u_j + \sum_{(i,j) \in A_k} z_k &\geq w_{ij}, & \forall\ i, j \in N \\ u_i &\geq 0, & \forall\ i \in N \\ z_k &\geq 0, & k = 1, 2, ..., m \end{aligned}$$

# Example 1

$$
\begin{array}{llllllll}
max & 1.3x_{12} & +0.4x_{13} & +1.1x_{14} & +0.5x_{23} & +1.2x_{24} & +0.6x_{34} & \\
s.t. & x_{12} & +x_{13} & +x_{14} & & & & \leq 1 \\
 & x_{12} & & & +x_{23} & +x_{24} & & \leq 1 \\
 & & x_{13} & & +x_{23} & & +x_{34} & \leq 1 \\
 & & & x_{14} & & +x_{24} & +x_{34} & \leq 1 \\
 & x_{12} & +x_{13} & & +x_{23} & & & \leq 1 \\
 & x_{12} & & +x_{14} & & +x_{24} & & \leq 1 \\
 & & x_{13} & +x_{14} & & & +x_{34} & \leq 1 \\
 & & & & x_{23} & +x_{24} & +x_{34} & \leq 1 \\
 & x_{12}, & x_{13}, & x_{14}, & x_{23}, & x_{24}, & x_{34} & \geq 0
\end{array}
$$

$$
\begin{array}{llllllll}
min & u_1 & +u_2 & +u_3 & +u_4 & +z_1 & +z_2 & +z_3 & +z_4 & \\
s.t. & u_1 & +u_2 & & & +z_1 & +z_2 & & & \geq 1.3 \\
 & u_1 & & +u_3 & & +z_1 & & +z_3 & & \geq 0.4 \\
 & u_1 & & & +u_4 & & +z_2 & +z_3 & & \geq 1.1 \\
 & & u_2 & +u_3 & & +z_1 & & & +z_4 & \geq 0.5 \\
 & & u_2 & & +u_4 & & +z_2 & & +z_4 & \geq 1.2 \\
 & & & u_3 & +u_4 & & & +z_3 & +z_4 & \geq 0.6 \\
 & u_1, & u_2, & u_3, & u_4, & z_1, & z_2, & z_3, & z_4 & \geq 0
\end{array}
$$

# Example 2 - Homework



# Optimality Conditions

- Complementary Slackness Conditions:

$$(8.1) \quad x_{ij} > 0 \quad \Rightarrow \quad u_i + u_j + \sum_{(i,j)\in A_k} z_k = w_{ij}$$

$$(8.2) \quad u_i > 0 \quad \Rightarrow \quad \sum_{j\in N}(x_{ij} + x_{ji}) = 1$$

(i.e., node $i$ is covered by matching)

$$(8.3) \quad z_k > 0 \quad \Rightarrow \quad \sum_{(i,j)\in A_k} x_{ij} = n_k$$

(i.e., blossom $k$ is filled by matching)

# Solution Strategy

- (A) Keep primal feasibility (a matching) and dual feasibility (a non-negative node/blossom potential).
- (B) Maintain valid (8.1) and (8.3).
- (C) Reduce the violation of (8.2).

# Initial Solution

(i) $x_{ij} \triangleq 0, \forall\ (i,j) \in A$ (null matching)

(takes care of primal feasibility and (8.1))

(ii)

$$u_i \triangleq \tfrac{1}{2} \max_{(i,j)\in A}\{w_{ij}\}, \quad \forall\ i \in N$$

$$z_k \triangleq 0, \qquad k = 1, 2, ..., m$$

(takes care of dual feasibility and (8.2))

# Observations

- Observation 1: When $\max_{(i,j)\in A}\{w_{ij}\} \leq 0,$

  Null matching is optimal!

- Observation 2:

  In general, (8.2) is violated at some nodes with a positive potential.

# General Step

With a valid matching and valid node/blossom potentials, starting from an exposed node with $u_i > 0$, we try to find an augmenting path within the subgraph obtained by

(i) Shrinking all blossoms $k$ for which $z_k > 0$,

(ii) deleting all arcs $(i, j)$ for which
$u_i + u_j + \sum z_k > w_{ij}$
(i.e., only use those arcs with $u_i + u_j + \sum z_k = w_{ij}$).

# Possible Outcomes

- Case 1: If an augmenting path is found, then we get a new valid matching and reduce the violation of (8.2).

- The new matching takes care of the maximum matching in a blossom automatically for (8.3). Also because we only work on a sub-graph in which

$$u_i + u_j + \sum_k z_k = w_{ij},$$

(8.1) is maintained.

# Possible Outcomes

- Case 2: If no augmentation can be found (Hungarian), then we adjust node/blossom potentials such that

(i) $u_i \rightarrow u_i - \delta$, if $i$ is an S-labeled node or $i$ is contained in an outermost blossom whose pseudonode is S-labeled.

(ii) $u_i \rightarrow u_i + \delta$, if $i$ is an T-labeled node or $i$ is contained in an outermost blossom whose pseudonode is T-labeled.

(iii) $z_k \rightarrow z_k + 2\delta$, if blossom $k$ is an S-labeled outermost blossom.

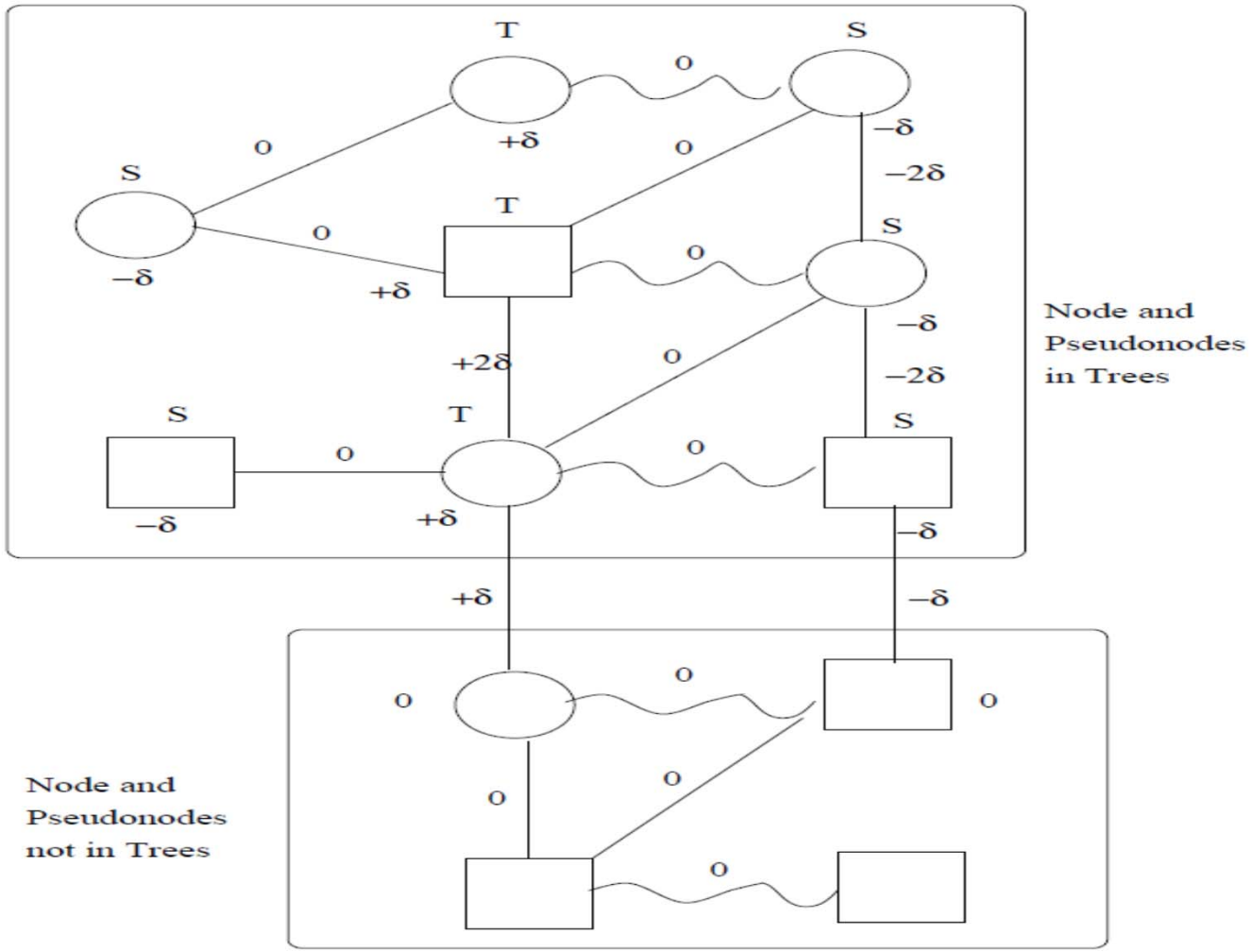
(iv) $z_k \rightarrow z_k - 2\delta$, if blossom $k$ is a T-labeled outermost blossom.

# Effects

(i) If $(i, j) \in A_k$, then $u_i + u_j + \sum_k z_k$ remains the same.

(ii) In general, we have the changes shown in the next page. Some of new arcs with $u_i + u_j + \sum_{(i.j) \in A_k} z_k = w_{ij}$ will be included for finding new augmenting path.

# Example


T
S
0
+δ
0
−δ
−2δ
S
T
0
S
−δ
+δ
0
−δ
+2δ
0
−2δ
S
T
S
0
0
−δ
+δ
−δ
+δ
−δ
Node and
Pseudonodes
in Trees
0
0
0
0
0
0
Node and
Pseudonodes
not in Trees

# A Critical Issue

- How to choose $\delta$:

  (a) Primal feasibility is not an issue.

  (b) Dual feasibility needs attention.

  (c) (8.1) remains valid.

  (d) (8.3) is not an issue.

# Restrictions

(8.4) If $i$ is an S-labeled node or within an S-labeled outermost blossom, then $u_i - \delta \geq 0$ is required.
$\Rightarrow$ Among such $i$'s, we let

$$\delta_1 \triangleq min_i\{u_i\}$$

(8.5) For $(i, j)$, if both $i$ and $j$ are S-labeled or contained within different S-labeled outermost blossoms, then

$$(u_i - \delta) + (u_j - \delta) \geq w_{ij}$$

is required.
$\Rightarrow$ Among such $(i, j)$'s, we let

$$\delta_2 \triangleq \frac{1}{2} \min_{(i,j)}\{u_i + u_j - w_{ij}\}$$

# Restrictions

(8.6) If the pseudonode for an outermost blossom $k$ has a T-label, then

$$z_k - 2\delta \geq 0$$

is required.

$\Rightarrow$ Among such $k$'s, we let

$$\delta_3 \triangleq \frac{1}{2} \min_k \{z_k\}$$

# Restrictions

(8.7) For $(i, j)$, if $i$ is an S-labeled node or contained within an S-labeled outermost blossom while $j$ is either unlabeled or contained within an outermost blossom whose pseudonode is unlabeled, then

$$(u_i - \delta) + u_j \geq w_{ij}$$

is required.

$\Rightarrow$ Among those $(i, j)$'s, we let

$$\delta_4 \triangleq \min_{(i,j)}\{u_i + u_j - w_{ij}\}$$

# Selection Rule

Choose $\delta \triangleq \min\{\delta_1, \delta_2, \delta_3, \delta_4\}$. Then the dual feasibility is maintained!

# Results

(1) If $\delta = \delta_1$, then violation of (8.2) is reduced. (For simultaneous reduction on all exposed nodes with uniform initial value, an optimal solution is reached.)

(2) If $\delta = \delta_2$, then either an augmenting path can be found or a new blossom formed.

(3) if $\delta = \delta_3$, then an outermost blossom can be expanded (unshrunk).

(4) if $\delta = \delta_4$, then at least one new arc can be added to an alternating tree.

# SUMMARY OF MAX WEIGHTED MATCHING ALGORITHM

*Step 0 (Start)* Start with $X = \emptyset$ and $u_i = \frac{1}{2}\max\{w_{ij}\}$, $z_k = 0$ as primal and dual solutions.

*Step 1 (Labeling)* Root an alternating tree at each exposed node, and proceed to construct alternating trees by labeling, using only arcs $(i, j)$ for which

$$u_i + u_j + \sum z_k = w_{ij}.$$

If an augmenting path is found, go to Step 2. If a blossom is formed, go to Step 3. If the trees become Hungarian, go to Step 4.

# SUMMARY OF MAX WEIGHTED MATCHING ALGORITHM

*Step 2 (Augmentation)* Find the augmenting path, tracing the path through shrunken blossoms. Augment the matching, remove all labels from nodes and pseudonodes, and return to Step 1.

*Step 3 (Blossoming)* Identify the blossom and shrink it in the graph. The pseudonode representing the blossom receives an $S$-label, and its $z$-variable is set to zero. Return to Step 1.

*Step 4 (Change in Dual Variables)* Determine the maximum value of $\delta$, according to conditions (8.4) through (8.7), and make the appropriate changes in the dual variables. If condition (8.4) is controlling, halt; the matching and the dual solution are optimal. Otherwise, expand outermost blossoms for which $z_k = 0$ and return to Step 1.//

# Implementation

$O(n^4)$ WEIGHTED MATCHING ALGORITHM

*Step 0 (Start)* The graph $G = (N, A)$ is given, with a weight $w_{ij}$ for each arc $(i, j)$. Set $u_i = \frac{1}{2}\max\{w_{ij}\}$, for each node $i \in N$. Set $\Delta = +\infty$. Set $X = \emptyset$. There are no blossoms and no nodes are labeled.

*Step 1 (Labeling)*

(1.0) Apply the label "$S : \emptyset$" to each exposed node.

(1.1) If there are no unscanned labels, go to Step 4. Otherwise, find a node $i$ with an unscanned label. If the label is an $S$-label, go to Step 1.2; if it is a $T$-label, go to Step 1.3.

# Implementation

(1.2) Scan the $S$-label on node $i$ by carrying out the following procedure for each arc $(i, j) \notin X$ incident to node $i$:

If $b(i) = b(j)$, do nothing; otherwise continue.
If node $b(j)$ has an $S$-label and $\bar{w}_{ij} = 0$, backtrace from the $S$-labels on nodes $i$ and $j$. If different root nodes are reached, go to Step 2; if the same root node is reached, go to Step 3.
If node $b(j)$ has an $S$-label and $\bar{w}_{ij} > 0$, set $\Delta = \min\{\Delta, \frac{1}{2}\bar{w}_{ij}\}$.
If node $b(j)$ is unlabeled and $\bar{w}_{ij} = 0$, apply the label "$T : i, j$" to $b(j)$.
If node $b(j)$ is unlabeled and $\bar{w}_{ij} > 0$, set $\Delta = \min\{\Delta, \bar{w}_{ij}\}$.
When the scanning of node $i$ is complete, return to Step 1.1.

# Implementation

(1.3) Scan the $T$-label on node $i$ by carrying out the following procedure for the unique arc $(i, j) \in X$ incident to node $i$.

If $b(i) = b(j)$, do nothing; otherwise continue.
If node $j$ has a $T$-label, backtrace from the $T$-labels on nodes $i$ and $j$. If different root nodes are reached, go to Step 2; if the same root node is reached, go to Step 3.
Otherwise, give node $j$ the label "$S : i$." The $S$-labels on all nodes within the outermost blossom with base node $j$ are now considered to be unscanned.
Return to Step 1.1.

# Implementation

*Step 2 (Augmentation)* An augmenting path has been found in Step 1.2 or 1.3. Augment the matching $X$. Correct labels on nodes in the augmenting path, as described in the text. Expand blossoms with zero dual variables, resetting the blossom numbers $b(i)$. Remove labels from all base nodes. Remaining labels are set to "scanned" state. Set $\Delta = +\infty$. Go to Step 1.0.

*Step 3 (B1ossoming)* A blossom has been formed in Step 1.2 or 1.3. Determine the membership and base node of the new blossom, as described in the text. Supply missing labels for all nodes, except the base node, in the new blossom. Reset blossom numbers. Set the $z$-variable to zero for the new blossom.

Return to Step 1.2 or 1.3, as appropriate.

# Implementation

*Step 4 (Revision of Dual Solution)* Let $K_s$ denote the set of $S$-blossoms and $K_T$ denote the set of $T$-blossoms.

Find

$$\delta_1 = \min\{u_i\},$$
$$\delta_2 = \tfrac{1}{2}\min\{z_k | k \in K_T\},$$
$$\delta = \min\{\delta_1, \delta_2, \Delta\}.$$

Set $u_i = u_i - \delta$, for each node $i$ such that $b(i)$ has an $S$-label.

Set $u_i = u_i + \delta$, for each node $i$ such that $b(i)$ has a $T$-label.

Set $z_k = z_k - 2\delta$, for each blossom $k \in K_T$.

Set $z_k = z_k + 2\delta$, for each blossom $k \in K_S$.

If $\delta = \delta_1$, halt; $X$ is a maximum weight matching, and the values of $u_i, z_k$ yield an optimal dual solution.

# Implementation

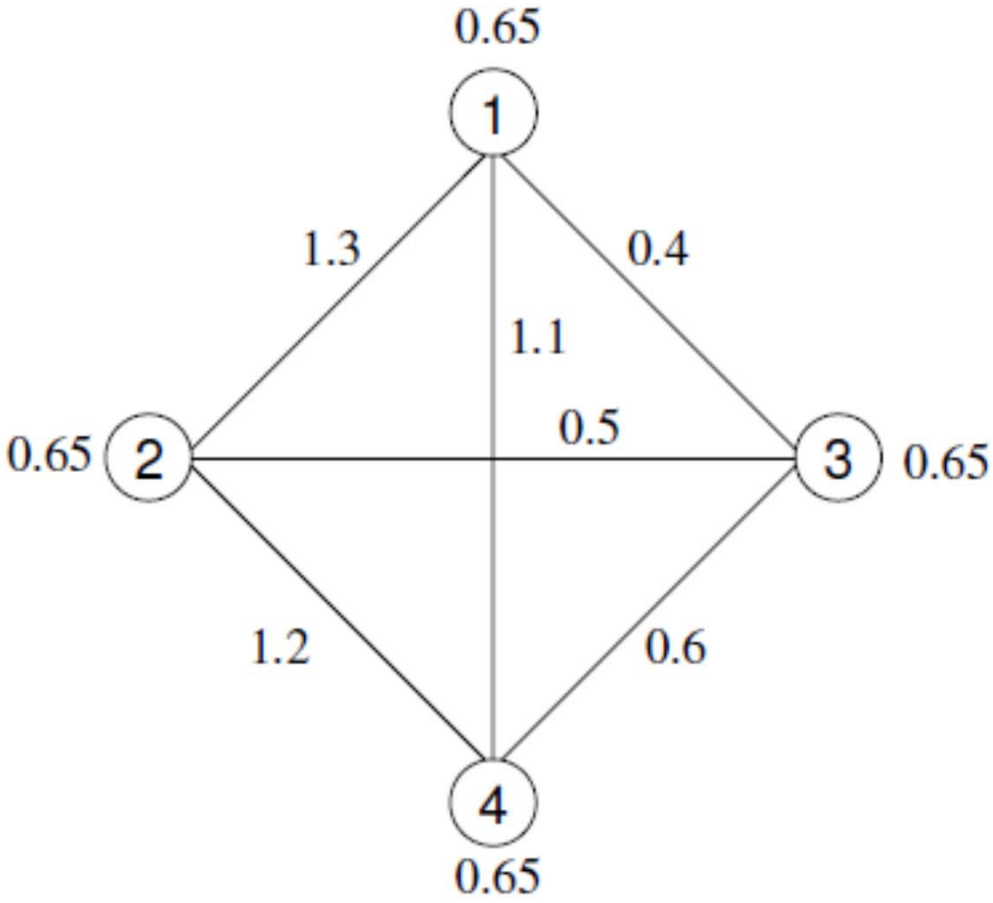
If $\delta = \delta_2$, expand each $T$-blossom $k$ for which $z_k = 0$ by determining the blossoms nested immediately within the $T$-blossom and resetting $b(i)$ for all nodes within the blossom. Remove labels from all new base nodes within the expanded blossom.

All labels on base nodes, and $S$-labels on nodes within $S$-blossoms, are now "unscanned." Remaining labels are in a "scanned" state.

Set $\Delta = +\infty$.

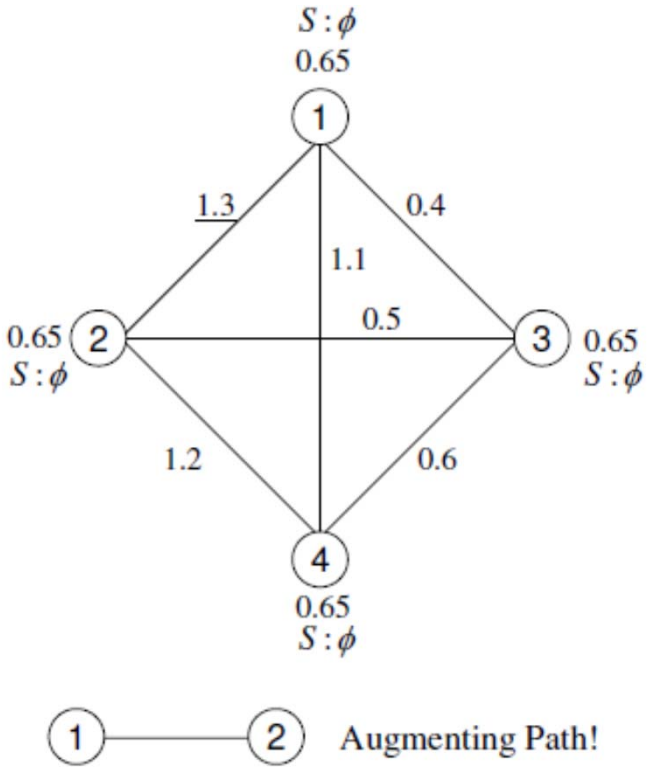
Return to Step l.1.//

# Example 1

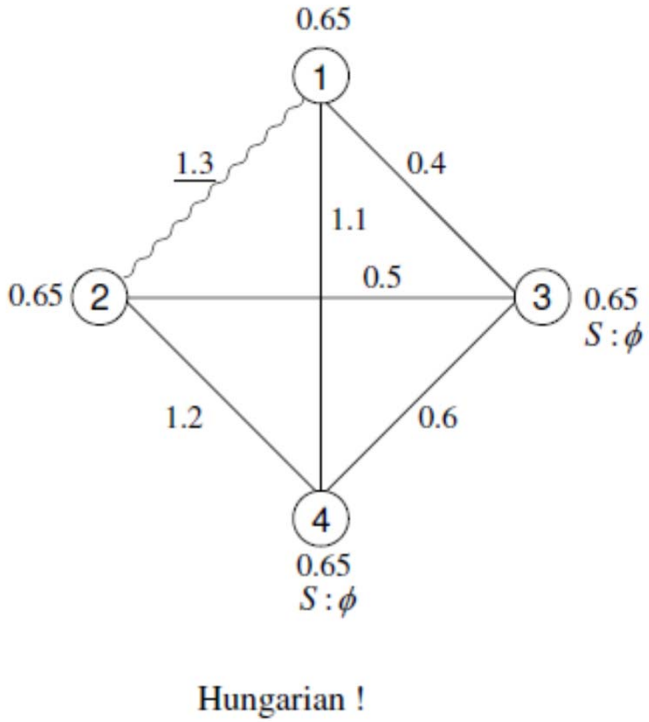


# Iterations (1)

- Iteration 1



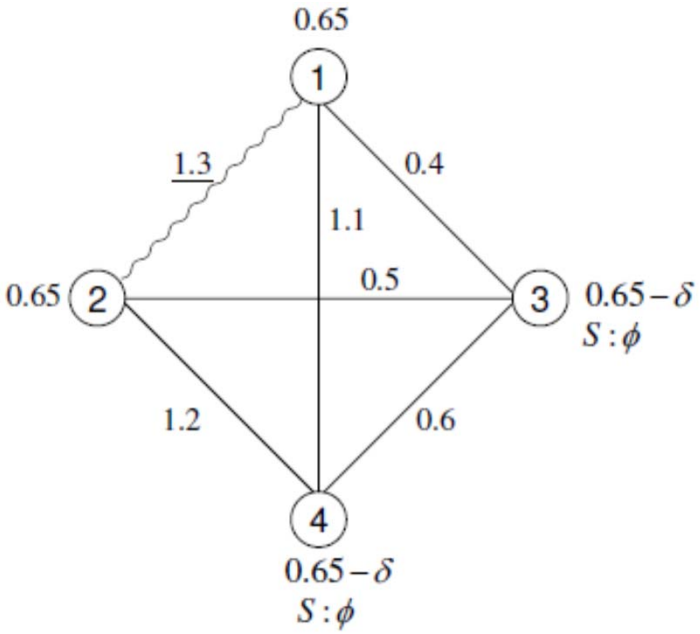

Iteration 2




Hungarian !

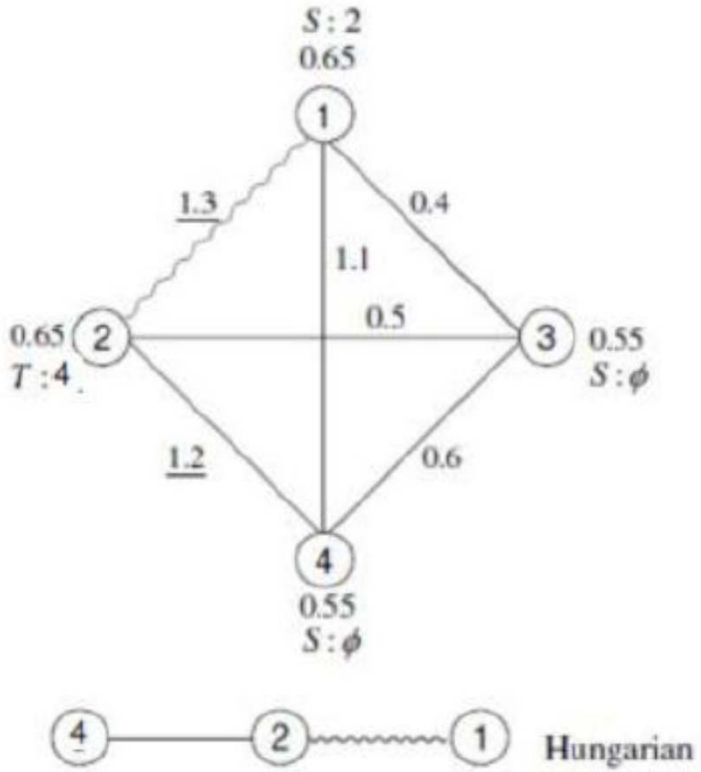
# Iterations (2)

- Iteration 3




$\delta = 0.1$ !

Iteration 4



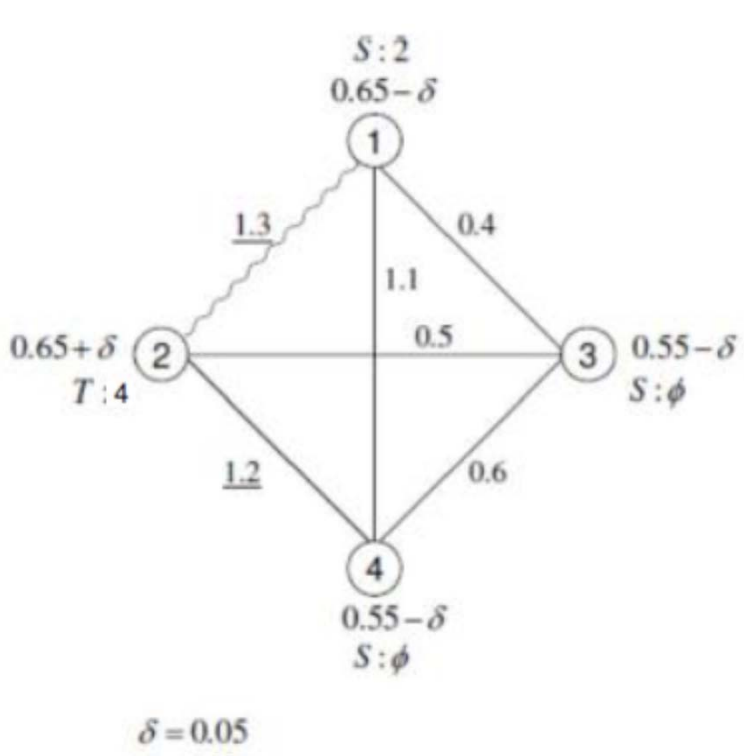




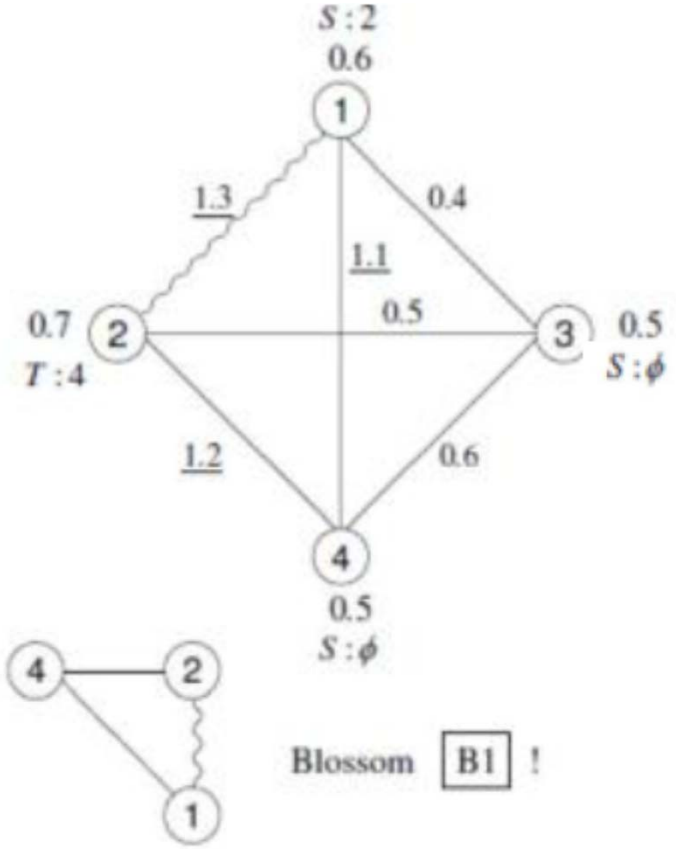
Hungarian

# Iterations (3)

- Iteration 5

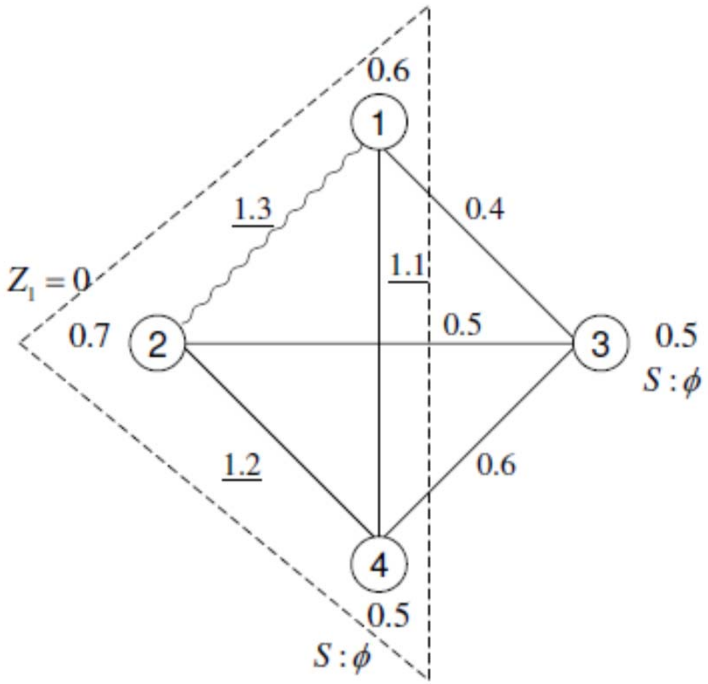



Iteration 6



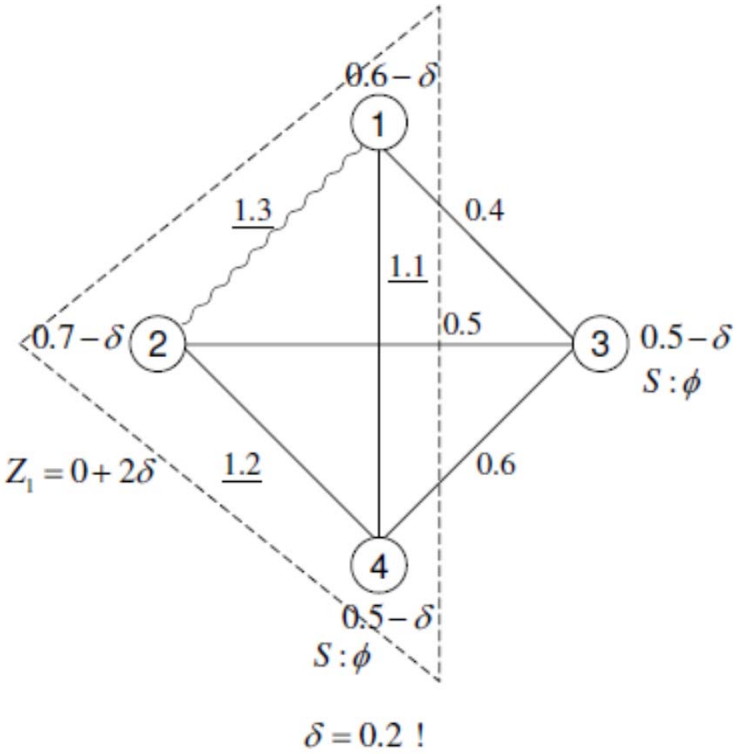
# Iterations (4)

- Iteration 7




Hungarian !

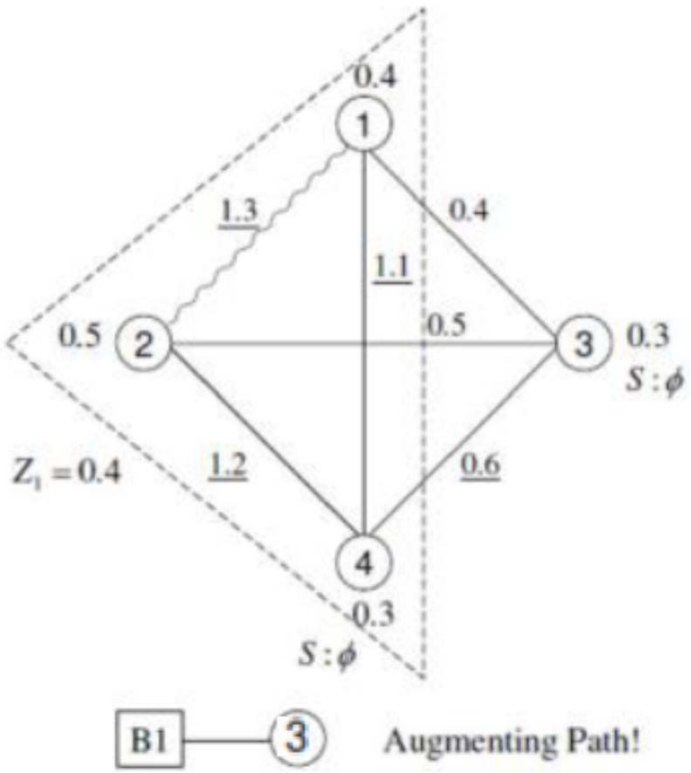
Iteration 8



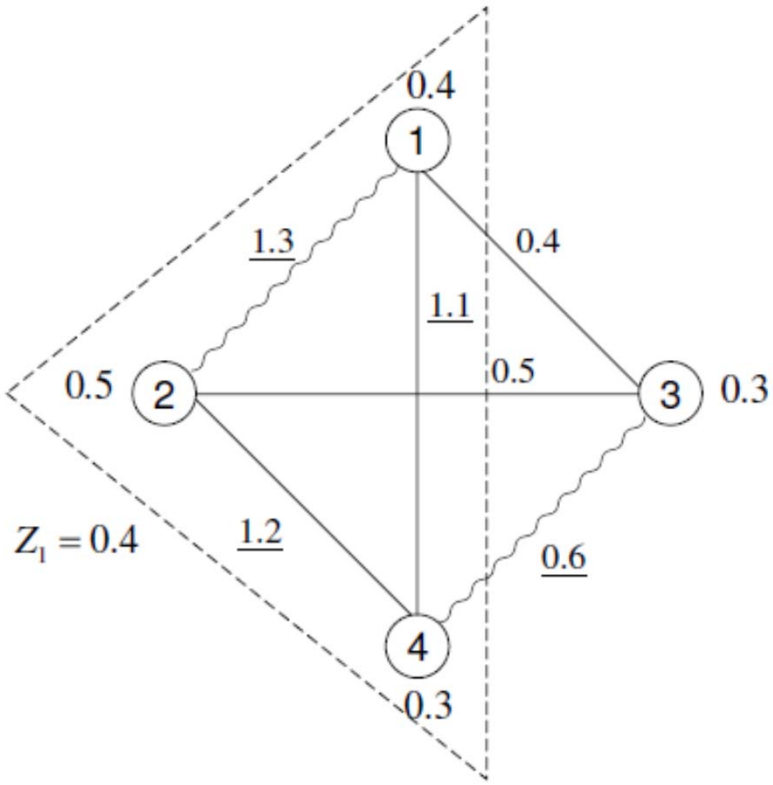

$\delta = 0.2$ !

# Iterations (5)

- Iteration 9

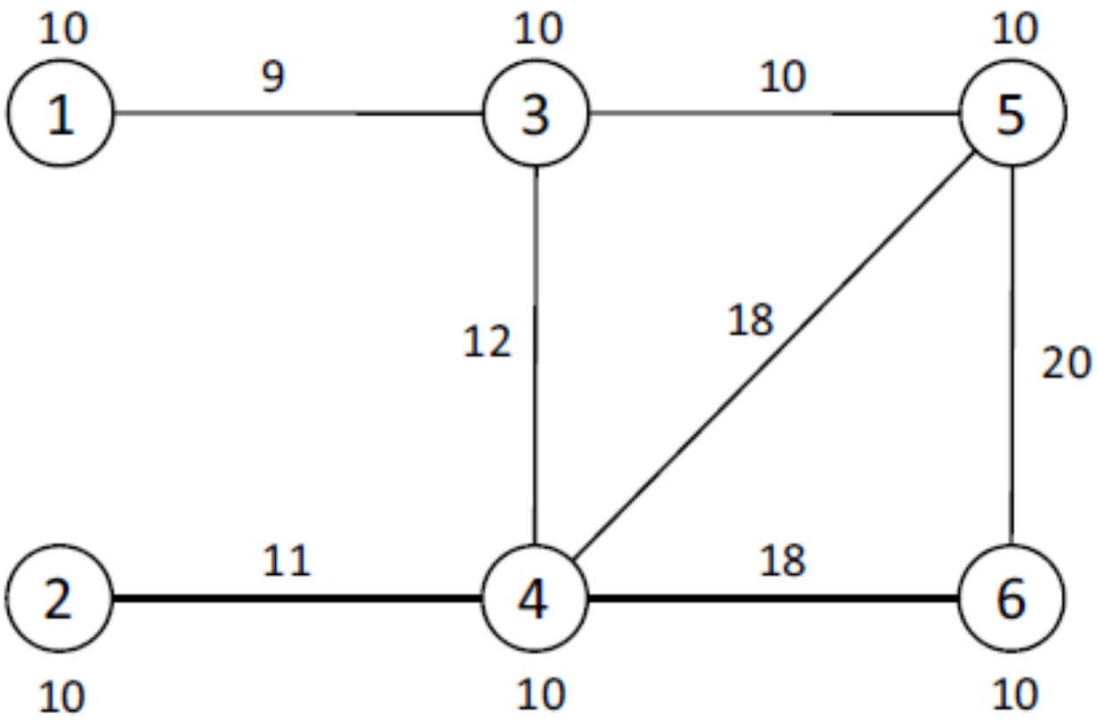



Iteration 10



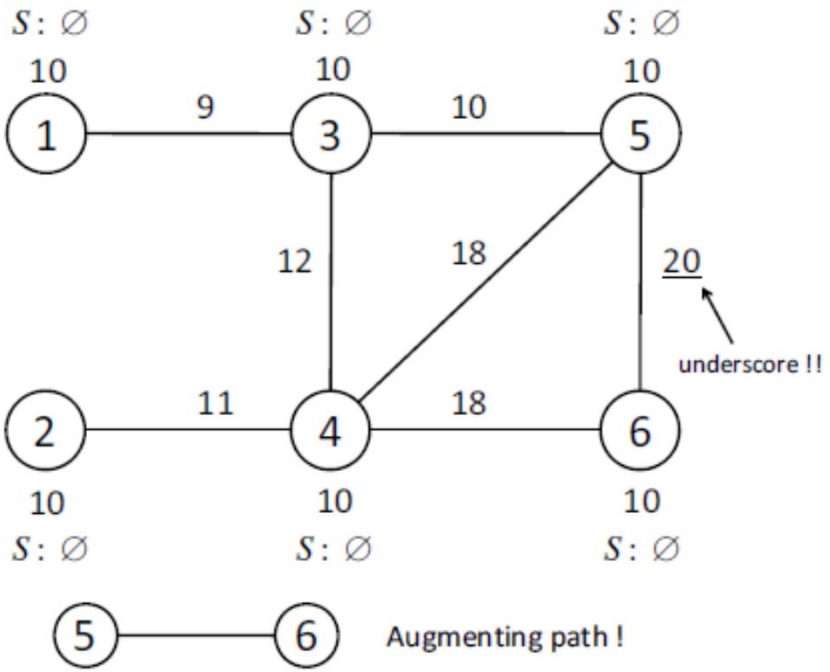

All nodes are covered. Optimal!

# Example 2

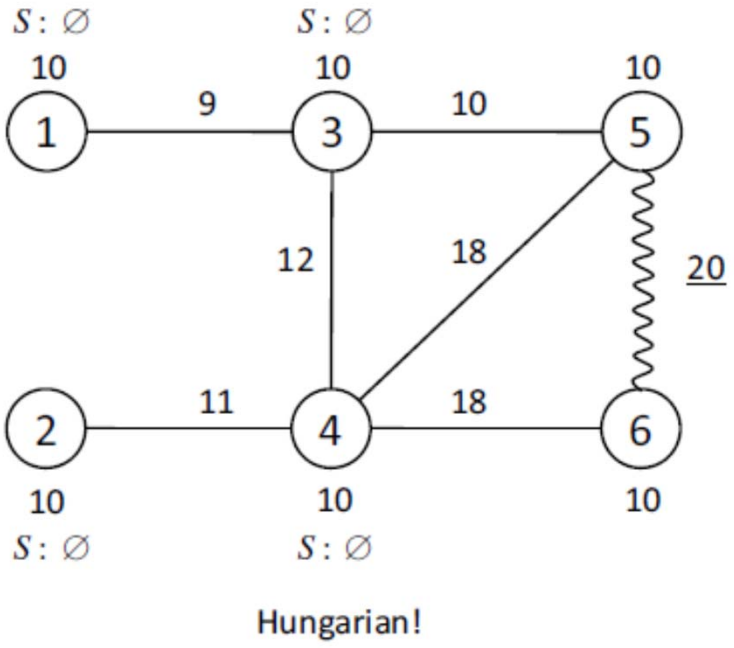


# Iterations (1)

- Iteration 1

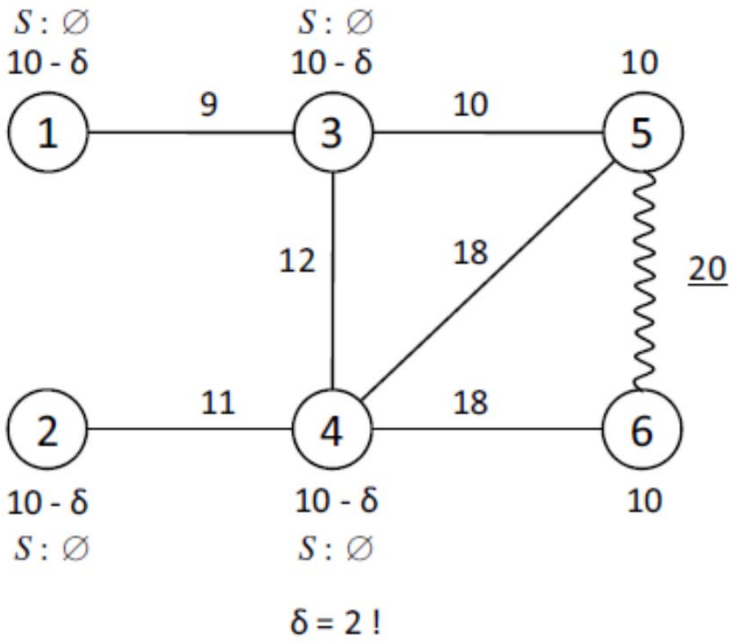



Iteration 2

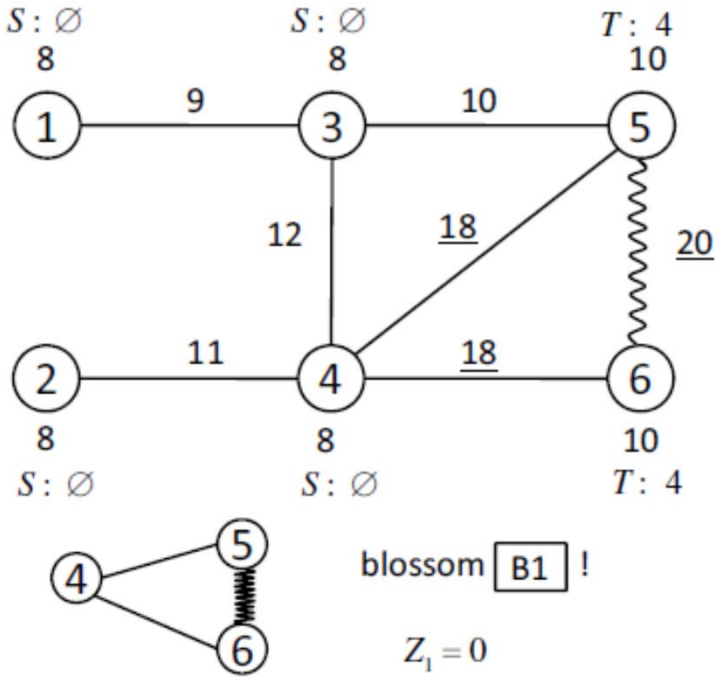


# Iterations (2)

- Iteration 3

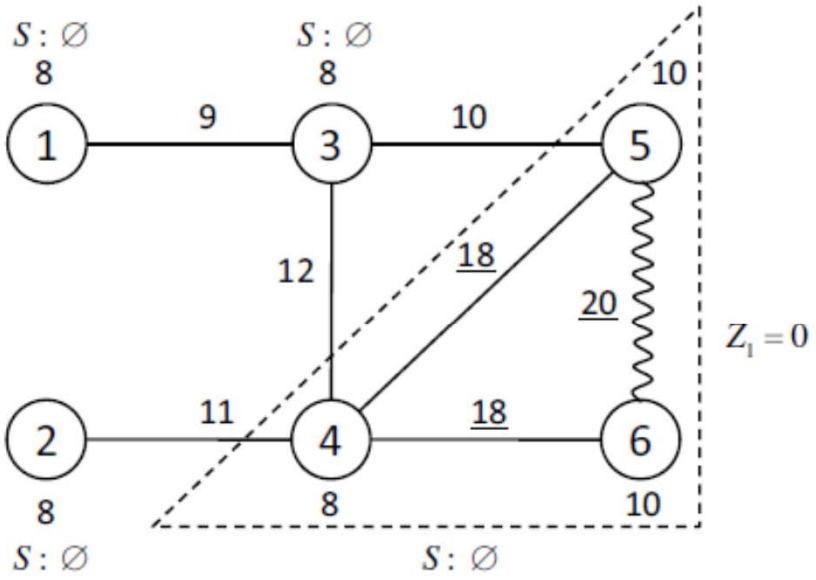



Iteration 4



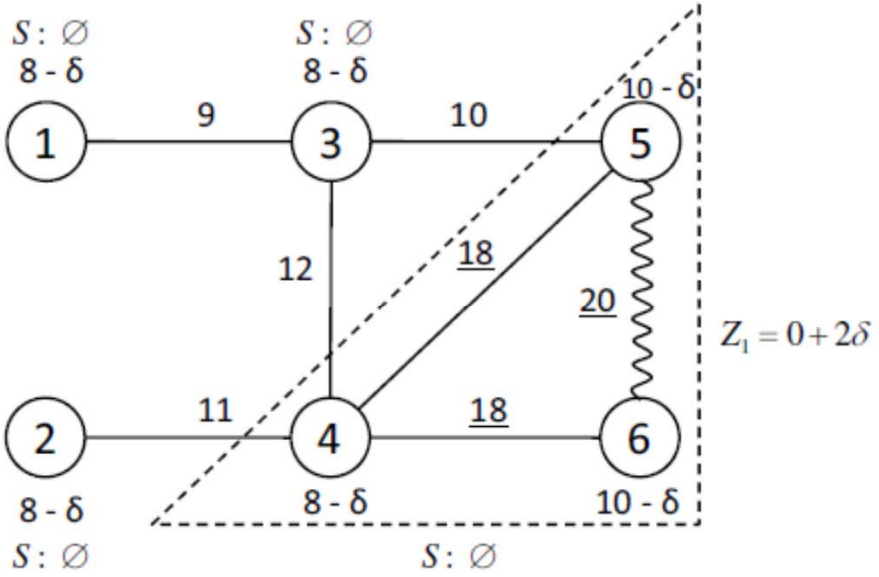
# Iterations (3)

- Iteration 5




Hungarian!

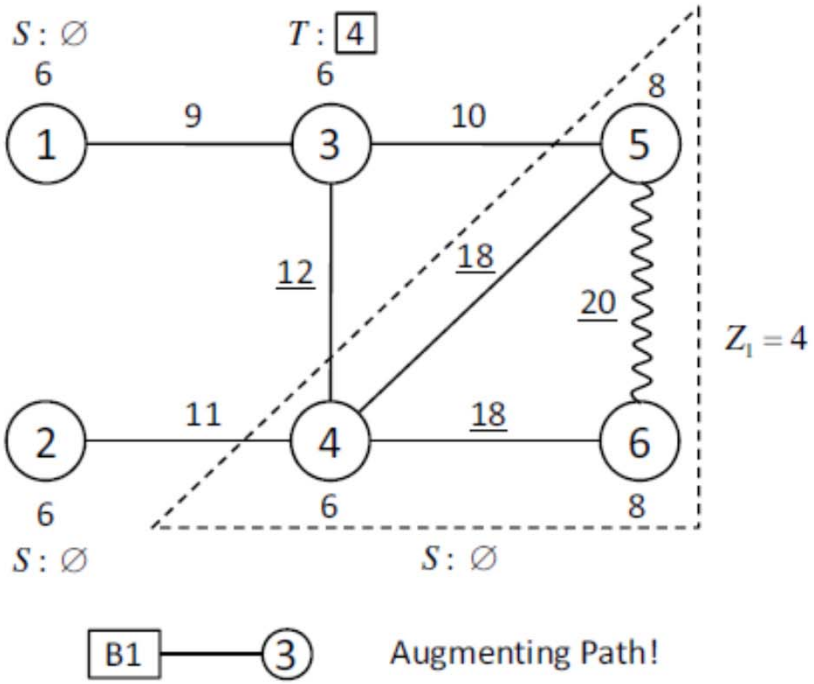
Iteration 6



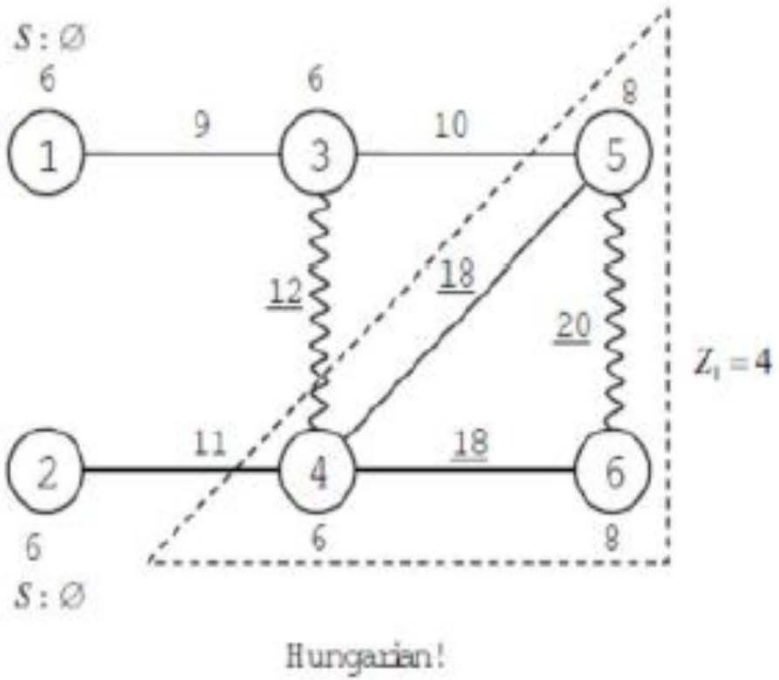

δ = 2 !

# Iterations (4)

- Iteration 7

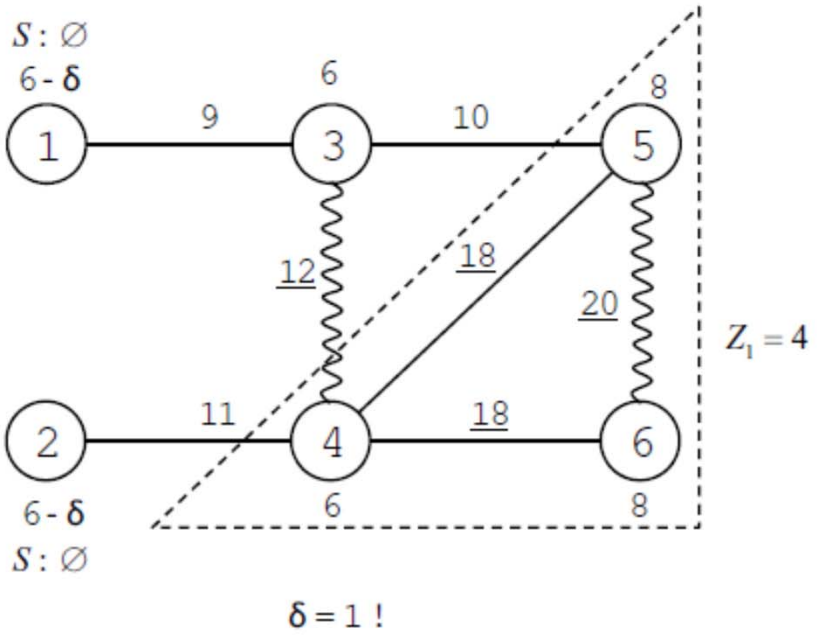



Iteration 8

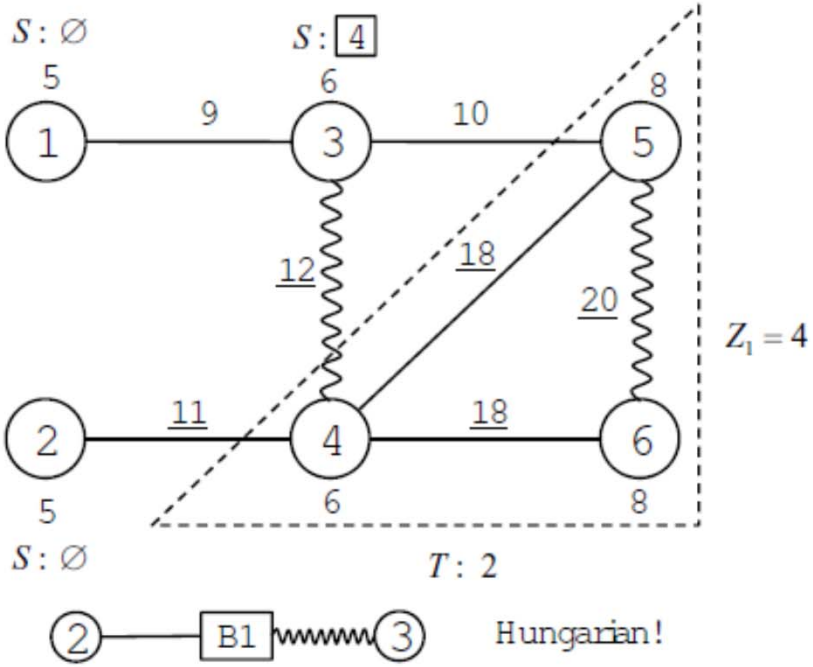


# Iterations (5)

- Iteration 9

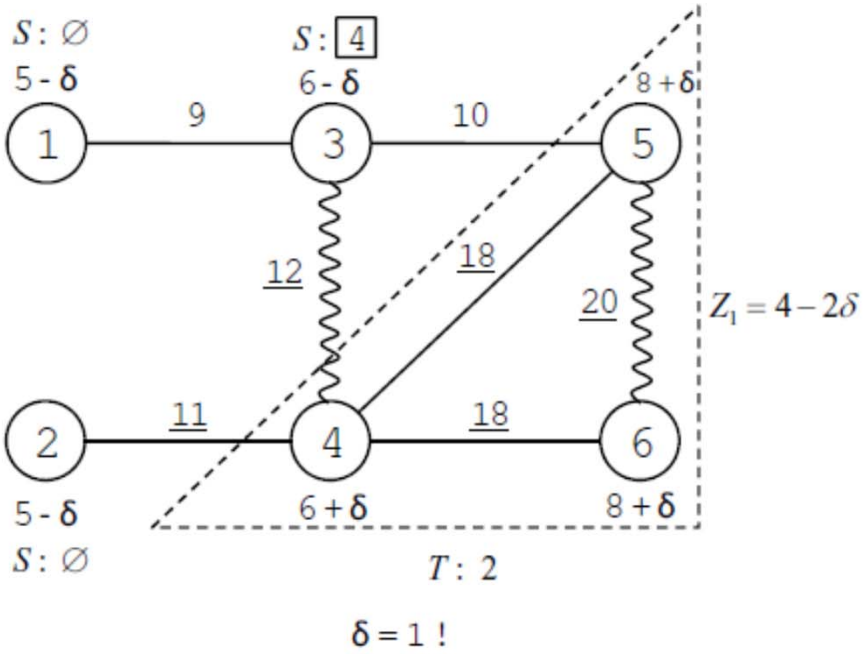



Iteration 10

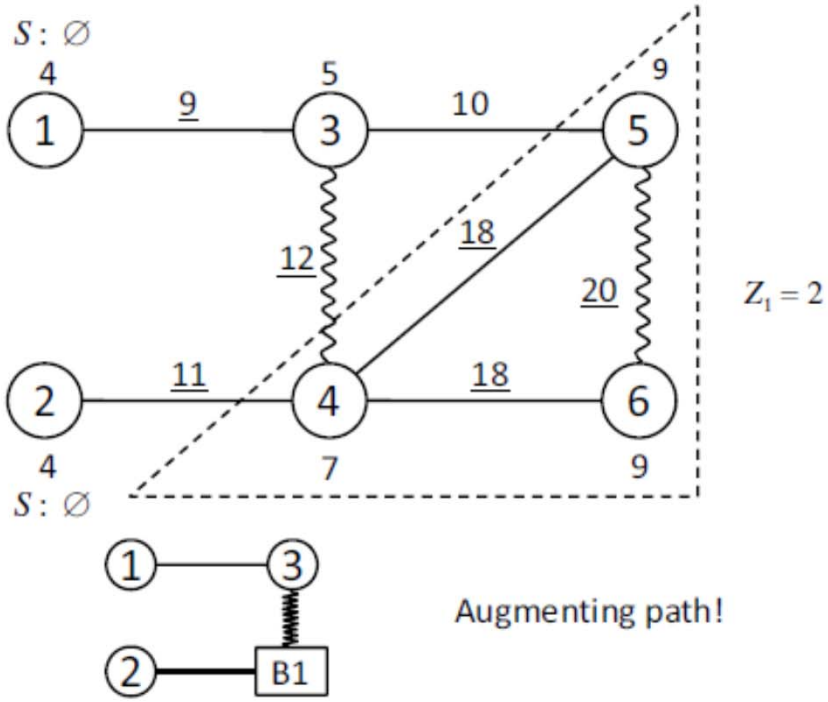


# Iterations (6)

- Iteration 11

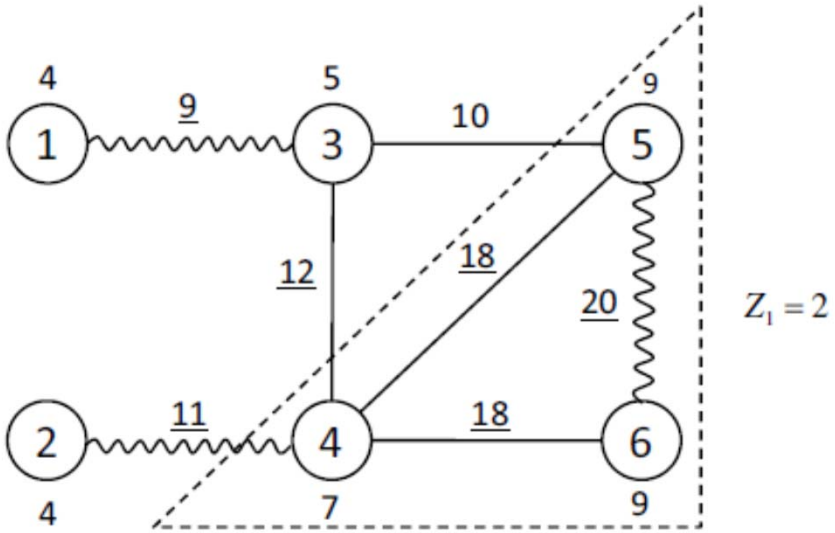



Iteration 12



# Iterations (7)

- Iteration 13




All nodes covered ! Optimal !

# An $O(n^3)$ Weighted Matching Algorithm

Several features must be added to the algorithm presented in the previous section, in order to reduce its complexity to $O(n^3)$. Each of these features has as its objective the avoidance of rescanning labels after revision of the dual solution.

# Rescanning Labels

There are three reasons why labels must be rescanned in the $O(n^4)$ algorithm:

(1) *$T$-labels* Suppose $\bar{w}_{ij}$ is reduced to zero for an arc $(i, j) \notin X$, where has an $S$-label and $b(j)$ is unlabeled. Rescanning the $S$-label on node $i$ results in the application of a $T$-label to $b(j)$.

(2) *Augmenting paths and blossoms* Suppose $\bar{w}_{ij}$ is reduced to zero for an arc $(i, j) \notin X$, where both $i$ and $j$ have $S$-labels. Rescanning the $S$-label on either $i$ or $j$ results in the discovery of either an augmenting path or a new blossom.

(3) *Expansion of $T$-blossoms* Suppose a $T$-blossom is expanded because its $z$-variable is reduced to zero. Rescanning the $S$-labels on nodes adjacent to thc expanded blossom may result in the labeling of nodes and blossoms contained within the expanded blossom.

# An $O(n^3)$ Weighted Matching Algorithm

*Step 0 (Start)* The graph $G = (N, A)$ is given, with a weight $w_{ij}$ for each arc $(i, j)$. Let $W = \frac{1}{2}\max\{w_{ij}\}$. Set $u_i = W, \gamma_i = \pi_i = \tau_i = +\infty$ and $b(i) = i$ for each node $i \in N$. For each node pair $i$, $j$ set $C(i, j) = \emptyset$. Set $X = \emptyset$. There are no blossoms and no nodes are labeled.

# An $O(n^3)$ Weighted Matching Algorithm

*Step 1 (Labeling)*

(1.0) Apply the label "$S : \emptyset$" to each exposed node.

(1.1) If there is no node $i$ with an unscanned $S$-label or an unscanned $T$-label with $\pi_i = 0$, go to Step 4. Otherwise, find such a node $i$. If the label is an $S$-label, go to Step 1.2; if it is a $T$-label, go to Step 1.3.

(1.2) Scan the $S$-label on node $i$ by carrying out the following procedure for each arc $(i, j) \notin X$ incident to node $i$:

If $b(i) = b(j)$, do nothing; otherwise continue.
If node $b(j)$ has an $S$-label and $\bar{w}_{ij} = 0$, backtrace from the $S$-labels on nodes $i$ and $j$. If different root nodes are reached, go to Step 2; if the same root node is reached, go to Step 3.
If node $b(j)$ has an $S$-label and $\bar{w}_{ij} > 0$, then carry out the following procedure. Set $\gamma_{b(i)} = \min\{\gamma_{b(i)}, \frac{1}{2}\bar{w}_{ij}\}$, $\gamma_{b(j)} = \min\{\gamma_{b(j)}, \frac{1}{2}\bar{w}_{ij}\}$. Find $C(b(i), b(j)) = (p, q)$. If $\bar{w}_{ij} < \bar{w}_{pq}$, then set $C(b(i), b(j)) = (i, j)$.
If node $b(j)$ has no $S$-label and $\bar{w}_{ij} < \pi_{b(j)}$, then apply the label "$T : i, j$" to $b(j)$, replacing any existing $T$-label, and set $\pi_{b(j)} = \bar{w}_{ij}$.

# An $O(n^3)$ Weighted Matching Algorithm

If node $b(j)$ has no $S$-label and $\bar{w}_{ij} < \tau_j$, then set $\tau_j = \bar{w}_{ij}$ and set $t(j) = i$.
When the scanning of node $i$ is complete, return to Step 1.1.

(1.3) Scan the $T$-label on node $i$ (where $\pi_i = 0$) by carrying out the following procedure for the unique arc $(i, j) \in X$ incident to node $i$.

If $b(i) = b(j)$, do nothing; otherwise continue.
If node $j$ has a $T$-label and $\pi_j = 0$, backtrace from the $T$-labels on nodes $i$ and $j$. If different root nodes are reached, go to Step 2; if the same root node is reached, go to Step 3.
Otherwise, give node $j$ the label "$S : i$." The $S$-labels on all nodes within the outermost blossom with base node $j$ are now considered to be unscanned.
Return to Step 1.1.

# An $O(n^3)$ Weighted Matching Algorithm

*Step 2 (Augmentation)* An augmenting path has been found in Step 1.2, 1.3, or 4.2. Augment the matching $X$. Correct labels on nodes in the augmenting path, as described in the text. Expand blossoms with zero dual variables, resetting the blossom numbers. Remove labels from all base nodes. The remaining labels are set to the "scanned" state. Set $\gamma_i = \pi_i = \tau_i = +\infty$, for all $i$, and $C(i, j) = \emptyset$, for all $i, j$. Go to Step 1.0.

# An $O(n^3)$ Weighted Matching Algorithm

*Step 3 (B1ossoming)* A blossom has been formed in Step 1.2, 1.3, or 4.2. Determine the membership and base node of the new blossom, as described in the text. Supply missing labels for all nodes, except the base node, in the new blossom.

Reset blossom numbers. Set the $z$-variable to zero for the new blossom.

Let $b$ be the base node of the new blossom, and $I$ be the set of (old) base nodes contained in the blossom. Let $J$ be the complementary set of base nodes. For each $j \in J$, find arc $C(i,j) = (p', q')$, for which $\bar{w}_{p'q'} = \min_{i \in I}\{\bar{w}_{pq} | C(i,j) = (p,q)\}$, and set $C(b,j) = (p', q')$. Then set $r_b = \min_{j \in J}\{\bar{w}_{pq} | C(b,j) = (p,q)\}$.

Return to Step 1.2, 1.3, or 4.2, as appropriate.

# An $O(n^3)$ Weighted Matching Algorithm

(4.1) Let $K_s$ denote the set of $S$-blossoms and $K_T$ denote the set of $T$-blossoms. i.e., outermost blossoms whose base nodes $b$ have $T$-labels with $\pi_b = 0$.

Find

$$\delta_1 = \min\{u_i\},$$
$$\delta_2 = \tfrac{1}{2}\min\{z_k | k \in K_T\},$$
$$\delta_3 = \min\{\gamma_i | b(i) = i\},$$
$$\delta_4 = \min\{\pi_i | \pi_i > 0\},$$
$$\delta = \min\{\delta_1, \delta_2, \delta_3, \delta_4\}.$$

# An $O(n^3)$ Weighted Matching Algorithm

Set $u_i = u_i - \delta$, for each node $i$ such that $b(i)$ has an $S$-label.

Set $u_i = u_i + \delta$, for each node $i$ such that $b(i)$ has an $T$-label and $\pi_{b(i)} = 0$.

Set $\gamma_i = \gamma_i - 2\delta$, for each node $i$ such that $b(i) = i$.

Set $\pi_i = \pi_i - \delta$, if $\pi_i > 0$.

Set $\tau_i = \tau_i - \delta$, for each node $i$ such that $\pi_{b(i)} > 0$.

Set $z_k = z_k - 2\delta$, for each blossom $k \in K_T$.

# An $O(n^3)$ Weighted Matching Algorithm

Set $z_k = z + 2\delta$, for each blossom $k \in K_S$.
If $\delta = \delta_1$ halt; $X$ is a maximum weight matching, and the values of $u_i$, $z_k$ yield an optimal solution.
If $\delta = \delta_2$, carry out the following procedure to expand each $T$-blossom $k$ for which $z_k = 0$. Determine the blossoms nested immediately within the $T$-blossom and reset $b(i)$ for all nodes within the blossom. Remove labels from all new base nodes within the blossom. For each new base node $b$, find $\tau_i = \min\{\tau_j | b(j) = b\}$, and if $\tau_i < +\infty$, apply the (unscanned) label "$T : t(i), i$" to $b$ and set $\pi_b = \tau_i$. Remaining labels on nodes within the blossom are in a "scanned" state.

# An $O(n^3)$ Weighted Matching Algorithm

(4.2) If $\gamma_b > 0$, for all base nodes $b$, go to Step 1.1. Otherwise, find a base node $b$ for which $\gamma_b = 0$ and a base node $b'$ such that $\bar{w}_{ij} = 0$ for $(i, j) = C(b, b')$.Backtrace from the $S$-labels on $i$ and $j$. If different root nodes are reached, go to Step 2. If the same root node is reached, go to Step 3, later returning to Step 4.2.//

# References


1. M. Fujii, T. Kasami, and K. Ninomiya, "Optimal Sequencing of Two Equivalent Processors," *SIAM J. Appl. Math.*, 17 (1969) 784-789.

   "Erratum." *SIAM J. Appl. Math.*, 20 (1971), 141.

2. E. G. Coffman, Jr. and R. L. Graham, "Optimal Scheduling for Two-Processor Systems," *Acta Informatica*, 1 (1972) 200-213.

3. J. Edmonds, "An Introduction to Matching," mimeographed notes, Engineering Summer Conference, The University of Michigan, Ann Arbor, 1967.

4. C. Berge, "Two Theorems in Graph Theory," *Proc. Natl.. Acad. Sci. U.S.*, 43 (1957) 842-844.

# References


5. R. Z. Norman and M. O. Rabin, "An Algorithm for a Minimum Cover of a Graph," *Proc. Amer. Math. Soc.*, 10 (1959) 315-319.

6. J. Edmonds, "Path, Trees, and Flowers," *Can. J. Math.*, 17 (1965) 449-467.

7. C. Witzgall and C. T. Zahn, Jr., "Modification of Edmonds' Algorithm for Maximum Matching of Graphs," *J. Res. NBS*, 69B (April-June 1965) 91-981.

8. M. L. Balinski, "Labelling to Obtain a Maximum Matching," mimeographed paper, 1967.

# References


9. S. Even and O. Kariv, "An $O(n^{2.5})$ Algorithm for Maximum Matching in General Graphs," *Proc. 16th Annual Symp. on Foundations of Computer Science*, IEEE, New York, 1975, pp. 100-112.

10. W. T. Tutte, "The Factorization of Linear Graphs," *J. London Math. Soc.*, 22 (1947) 107-111.

11. W. T. Tutte, "The Factors of Graphs," *Can. J. Math.*, 4 (1952) 314-328.

12. W. T. Tutte, "A Short Proof of the Factor Theorem for Finite Graphs," *Can. J. Math.*, 6 (1954) 347-352.

# References


13. J. Edmonds, "Maximum Matching and a Polyhedron with 0, 1 Vertices," *J. Res. NBS*, 69B (April-June 1965) 125-130.

14. E. L. Johnson, "Networks, Graphs, and Integer Programming," Ph.D. thesis, issued as Report ORC 65-1, Operations Research Center, The University of California, Berkeley, 1965.

15. R. J. Urquhart, "Degree Constrained Subgraphs of Linear Graphs," Ph.D. dissertation, The University of Michigan, Ann Arbor, 1967.

16. L. J. White, "A Parametric Study of Matchings and Coverings in Weighted Graphs," Ph.D. dissertation, The University of Michigan, Ann Arbor, 1967.

17. J. Edmonds and E. L. Johnson, "Matching: A Well-Solved Class of Integer Linear Programs," in *Combinatorial Structure and Their Applications*, R. Guy, editor, Gordon and Breach, New York, 1970, pp. 89-92.

# References


18. J. Edmonds, E. L. Johnson, S. Lockhart, "Blossom I, A Code for Matching," unpublished report, IBM T. J. Watson Research Center, Yorktown Heights, New York, 1969.

19. S. Lockhart, "An Annotated Fortran Program for Matching," unpublished report, IBM T. J. Watson Research Center, Yorktown Heights, New York, 1969.

20. H. Gabow, "An Efficient Implementation of Edmond's Maximum Matching Algorithm," Tech. Report 31, Stanford Univ. Comp. Science Dept., June 1972.

21. Mei-ko Kwan, "Graphic Programming Using Odd and Even Points," *Chinese Math.*, 1 (1962) 273-277.

# References


22. J. Edmonds, "The Chinese Postman Problem," *Operations Research*, 13, Suppl. 1 (1965) 373.

23. J. Edmonds and E. L. Johnson, "Matching, Euler Tours and the Chinese Postman," *Math. Programming*, 5 (1973) 88-124.