THE 1-MATCHING/COVERING PROBLEM

by

Clovis Perin

Technical Report 81-3

Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

March 1981

Research effort was supported by the Department of Industrial and Operations Engineering of The University of Michigan and by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under grant no. AFOSR 78-4646. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

Abstract

Consider an undirected network consisting of a finite nonempty set N of n nodes partitioned as $N^{\leq} \cup N^{=} \cup N^{\geq} \cup N^{o}$ (where some of these subsets may be empty), a set E of m edges, and a vector of edge costs c. A 1-matching/covering is a subset of edges satisfying the properties: i) every node of $N^{\leq}$ is incident with at most one edge of the subset; ii) every node of $N^{=}$ is incident with exactly one edge of the subset; iii) every node of $N^{\geq}$ is incident with at least one edge of the subset. There are no constraints associated with a node of N^{o}. The minimum cost 1-matching/covering problem is to find a 1-matching/covering which minimizes the total cost of the edges in the subset.

This paper discusses an algorithm for solving this problem, which can be implemented with a worst case time complexity $O(n^3)$.

1 - Introduction

In the late 60's, Edmonds [1,2] introduced the blossom algorithms for solving 1-matching problems which are defined on undirected networks with n nodes and m edges. The minimum cost 1-matching and the minimum cost 1-perfect matching problems have algorithms whose worst case time complexity is $O(n^2m)$. This maximum effort can be further reduced to $O(n^3)$ as shown in [6].

White and Gillenson [11] developed a blossom algorithm for the minimum cost 1-edge covering problem with time complexity $O(nm^2)$. Murty and Perin [9] designed another algorithm for the same problem which can run in $O(n^3)$.

All the above problems constitute special cases of the minimum cost 1-matching/covering problem which is the subject of this paper. We present an $O(n^2m)$ algorithm which can run in $O(n^3)$.

Edmonds, Johnson, and Lockhart [3,4] developed another blossom algorithm for solving a more general problem - the general matching. However, the time complexity of the algorithm is greater than $O(n^2m)$ and the data structure needed for the implementation is much more complex than the one needed for implementing the algorithm presented in this paper.

2 - Formulation of the Problem

Consider an undirected network $G = (N,E,c)$ consisting of a finite nonempty set of n nodes $N = \{ i : i = 1 \text{ to } n \}$, a set of m edges $E = \{ (i;j) : i \neq j \in N \}$, and an edge cost vector $c = [c_{ij}]$. It is given a partition $N^{\leq} \cup N^{=} \cup N^{\geq} \cup N^{o}$ of the set of nodes where some of the subsets may be empty. Let $x = [x_{ij}]$ be a 0-1 vector defined over the set of edges. The minimum cost 1-matching/covering problem is formulated as follows:

Problem I

$$\text{minimize} \quad \Sigma_{ij} [c_{ij} x_{ij} : (i;j) \in E]$$

subject to the node constraints

$$x(i) \begin{cases} \leq 1 & \text{for } i \in N^{\leq} \\ = 1 & \text{for } i \in N^{=} \\ \geq 1 & \text{for } i \in N^{\geq} \\ \text{unconstrained} & \text{for } i \in N^{o} \end{cases}$$

and restricted to

$$x_{ij} = 0 \text{ or } 1 \qquad \text{for } (i;j) \in E$$

where

$$x(i) = \Sigma_{j} [x_{ij} : (i;j) \in E] \qquad \text{for } i \in N.$$

Note that x_{ij}, x_{ji} are two representations of the same variable.

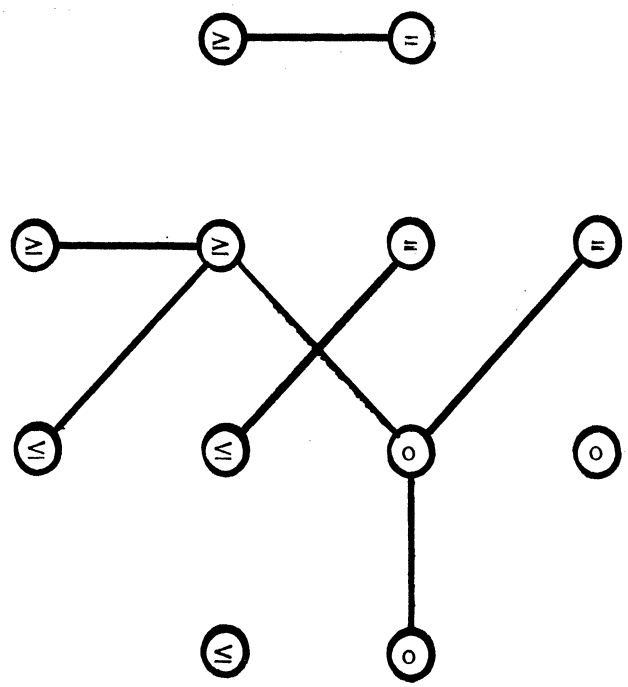

Figure 1 - Example of a 1-matching/covering for a network with 12 nodes. The symbols $\leq$, $=$, $\geq$, o identify the nodes of the subsets $N^{\leq}$, $N^{=}$, $N^{\geq}$, N^{o}, respectively.

Given a 0-1 edge vector $x = [x_{ij}]$, define the set of solution edges

$$E(x) = \{ (i;j) \in E : x_{ij} = 1 \}$$

$E(x)$ is said to be a 1-matching/covering if every node of $N^{=}$ ($N^{\leq}$, $N^{\geq}$) is incident with exactly (at most, at least) one solution edge (i.e., x satisfies the node constraints).

Define

$$S = \{ s \subset N^{\leq} \cup N^{=} \cup N^{\geq} : s \text{ has odd cardinality } |s| \geq 3 \}$$

$$x^{*}(s) = \Sigma_{ij} [x_{ij} : (i;j) \text{ joins a node inside to a node outside } s] +$$
$$\Sigma_{i} [x(i) : i \in N^{\geq} \text{ inside } s] +$$
$$\Sigma_{i} [-x(i) : i \in N^{\leq} \text{ inside } s]$$

$$a_s = 1 + |N^{\geq} \cap s| - |N^{\leq} \cap s|$$

Theorem 1

Every 0-1 edge vector x which is a feasible solution of problem I satisfies the blossom constraints

$$x^{*}(s) \geq a_s \quad \text{for } s \in S$$

Proof

Each blossom constraint s can be written as follows:

$$\Sigma_{ij} [\, x_{ij} : (i;j) \text{ joins a node inside to a node outside } s \,] + \Sigma_i [\, x(i) - 1 : i \in N^{\geq} \text{ inside } s \,] + \Sigma_i [\, 1 - x(i) : i \in N^{\leq} \text{ inside } s \,] \geq 1$$

It is easily seen that the left hand side must be a nonnegative integer because x is an integer feasible solution. Therefore, the left hand side equal to zero is the only remaining possibility for violating a blossom constraint.

Since every $s \in S$ contains only nodes of $N^{\leq} \cup N^{=} \cup N^{\geq}$, it follows that every node inside s must be incident with exactly one solution edge, and there must be no solution edge joining a node inside with a node outside s. However, this situation cannot occur because every edge joins exactly two nodes and s has odd cardinality. Therefore, every feasible solution of problem I must satisfy the blossom constraints.

The optimality conditions of problem I are obtained by relaxing the integrality restrictions and introducing the blossom constraints. The algorithm presented in this paper constitutes a proof that the extreme points of the set of feasible solutions of this modified problem are the feasible solutions of the original problem.

Problem II: Primal

$$\text{minimize} \quad \Sigma_{ij} [\, c_{ij}\, x_{ij} : (i;j) \in E \,]$$

$$\text{subject to} \quad x(i) \begin{cases} \leq 1 & \text{for } i \in N^{\leq} \\ = 1 & \text{for } i \in N^{=} \\ \geq 1 & \text{for } i \in N^{\geq} \end{cases}$$

$$x^{*}(s) \geq a_s \quad \text{for } s \in S$$

$$0 \leq x_{ij} \leq 1 \quad \text{for } (i;j) \in E$$

The dual problem is obtained by associating to the node and blossom constraints, node prices $y = [y_i]$ and pseudonode prices $z = [z_s]$, respectively.

Problem II: Dual

$$\text{maximize} \quad \Sigma_i [\, y_i \,] + \Sigma_s [\, a_s\, z_s \,] + \Sigma_{ij} [\, u_{ij} \,]$$

$$\text{subject to} \quad y_i \begin{cases} \leq 0 \text{ for} & \text{for } i \in N^{\leq} \\ \text{unrestricted} & \text{for } i \in N^{=} \\ \geq 0 & \text{for } i \in N^{\geq} \\ = 0 & \text{for } i \in N^{o} \end{cases}$$

$$z_s \geq 0 \quad \text{for } s \in S$$

$$u_{ij} \leq 0 \leq v_{ij} \quad \text{for } (i;j) \in E$$

$$y_i + y_j + \Sigma_s [\, (b_i^s + b_j^s)\, z_s : i \text{ or } j \text{ inside } s \,] + u_{ij} + v_{ij} = c_{ij} \quad \text{for } (i;j) \in E$$

$$\text{where} \quad b_i^s \begin{cases} -1 & \text{for } i \in N^{\leq} \text{ inside } s \\ 0 & \text{for } i \in N^{=} \text{ inside } s \\ 1 & \text{for } i \in N^{\geq} \text{ inside } s \\ 1 & \text{for } i \in N \text{ outside } s \end{cases}$$

Defining for each $(i;j) \in E$ a relative cost

$$d_{ij} = y_i + y_j + \Sigma_s [\ (b_i^s + b_j^s)\ z_s : \text{ i or j inside } s \in S\ ]$$

$$= c_{ij} - u_{ij} - v_{ij}$$

the complementary slackness conditions can be written as follows:

Problem II: Complementary Slackness Conditions

$$y_i\ (\ x(i) - 1\ ) = 0 \qquad \text{for } i \in N$$

$$z_s\ (\ x^*(s) - a_s\ ) = 0 \qquad \text{for } s \in S$$

$$\left.\begin{array}{lll} d_{ij} > c_{ij} & \text{implies} & x_{ij} = 1 \\ d_{ij} < c_{ij} & \text{implies} & x_{ij} = 0 \end{array}\right\} \quad \text{for } (i;j) \in E$$

Hence, a solution (x,y,z) is an optimum solution of problem II if: i) x is a 0-1 vector satisfying the node constraints; ii) (y,z) is dual feasible; iii) (x,y,z) satisfies the complementary slackness conditions.

3 - Description of the Algorithm

Let $$E^{-} = \{ (i;j) \in E : i,j \in N^{\geq} \cup N^{o},\ c_{ij} \leq 0 \}$$

Suppose problem I has an optimum solution which does not include all edges of E^{-}; it is possible to construct an alternate optimum solution by introducing the edges of E^{-} which are not present because feasibility is maintained and the objective value does not increase when nonpositive values are added to it. The algorithm maintains all edges of E^{-} in the solution set.

During the execution of the algorithm, the set of solution edges is partitioned into matching and covering edges. Edges of E^{-} are always covering edges; solution edges which are adjacent to other solution edges are covering edges, too. The remaining solution edges are matching edges. So, components of the solution set consisting of a single edge which is not an edge of E^{-} constitute the set of matching edges. Note that such a classification leads to a partition of the set of nodes: those covered by matching edges; those covered by covering edges; and those which are not covered by the solution edges. This leads to the following classification of nodes (see figure 2):

matched - nodes incident with a matching edge

type 2 - nodes incident with two or more covering edges

type 1 - nodes incident with exactly one covering edge

type 0 - nodes of $N^{\leq} \cup N^{o}$ with null node price incident with no solution edge

exposed - nodes of $N^{\leq}$ with negative price and nodes of $N^{\geq} \cup N^{=}$ incident with no solution edge

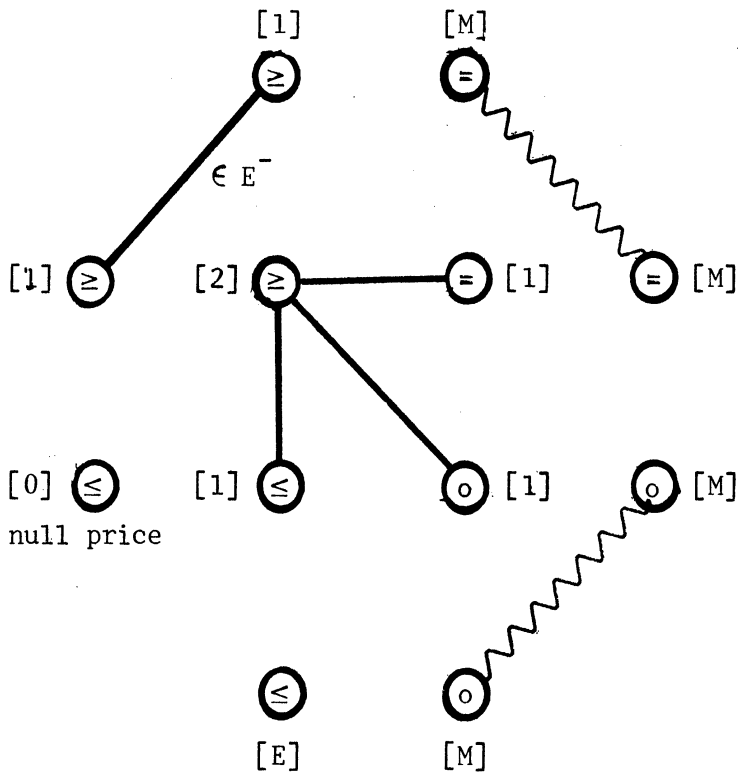

Figure 2 - Example of edge and node classification. Symbols inside the nodes define the node partition. Wavy edges are matching and straight edges are covering. Symbols in brackets identify the node classification: M - matched, 2 - type 2, 1 - type 1, 0 - type 0, E - exposed.

The algorithm works with all solution edges as a subset of $E^{-} \cup E^{=}$ where the set of equality edges $E^{=}$ is defined by

$$E^{=} = \{ (i;j) \in E : d_{ij} = c_{ij} \}$$

Alternating trees are rooted at exposed nodes and grown by incorporating equality edges in such a way that every path in a alternating tree connecting a node with the root is an alternating path of matching/nonsolution equality edges.
Labels with three indices are assigned to the nodes in the alternating trees identifying: i) the tree a node belongs to; ii) the predecessor node in the alternating path to the root; iii) the even (outer "+") or odd (inner "-") length of the alternating path. The roots of the alternating trees receive outer labels with no predecessors.

During the tree growing process, an odd alternating cycle of equality edges may be detected. The set of nodes in such a cycle define a partial network called simple blossom which is shrunken into a pseudonode. To shrink a simple blossom into a pseudonode is to introduce a pseudonode containing inside all nodes of the simple blossom; each node of the simple blossom is said to be inside the pseudonode; and every edge joining a node inside with a node outside the pseudonode is said to be incident with the pseudonode.

The solution/nonsolution role of the edges in a simple blossom may change during the execution of the algorithm, but the following properties are always satisfied: i) there is an odd alternating cycle of solution/nonsolution

equality edges connecting all nodes of the simple blossom; ii) there is an unique node called the base of the simple blossom which is incident with either two solution edges or two nonsolution edges of the odd cycle; iii) every node of the simple blossom but the base is incident with a pair of solution-nonsolution edges of the odd cycle.

The shrinking operation is carried out over simple blossoms which may contain pseudonodes shrunken before. So, it is possible to find a pseudonode inside other pseudonode. At this point, we introduce the terms original node (i.e., a node of N) and current node (i.e., original node or pseudonode which is not inside any pseudonode).

Pseudonodes are classified as matched, type 1, type 0, or exposed. A matched (type 1) pseudonode is incident with a matching (covering) edge. An exposed (type 0) pseudonode is incident with no solution edge and contains inside it an exposed (type 0 or type 2) original node. Note that there are no "type 2" pseudonodes. See figure 2.

The base and the apex of an original node are defined as the original node itself. The base of a pseudonode is the base of its simple blossom. The apex of a pseudonode is the apex of its base. Therefore, exposed, matched, type 1, and type 0 pseudonodes have bases which are exposed, matched, type 1, and (type 0 or type 2), in the same way that they have apices which are exposed, matched, type 1, and (type 0 or type 2), respectively.

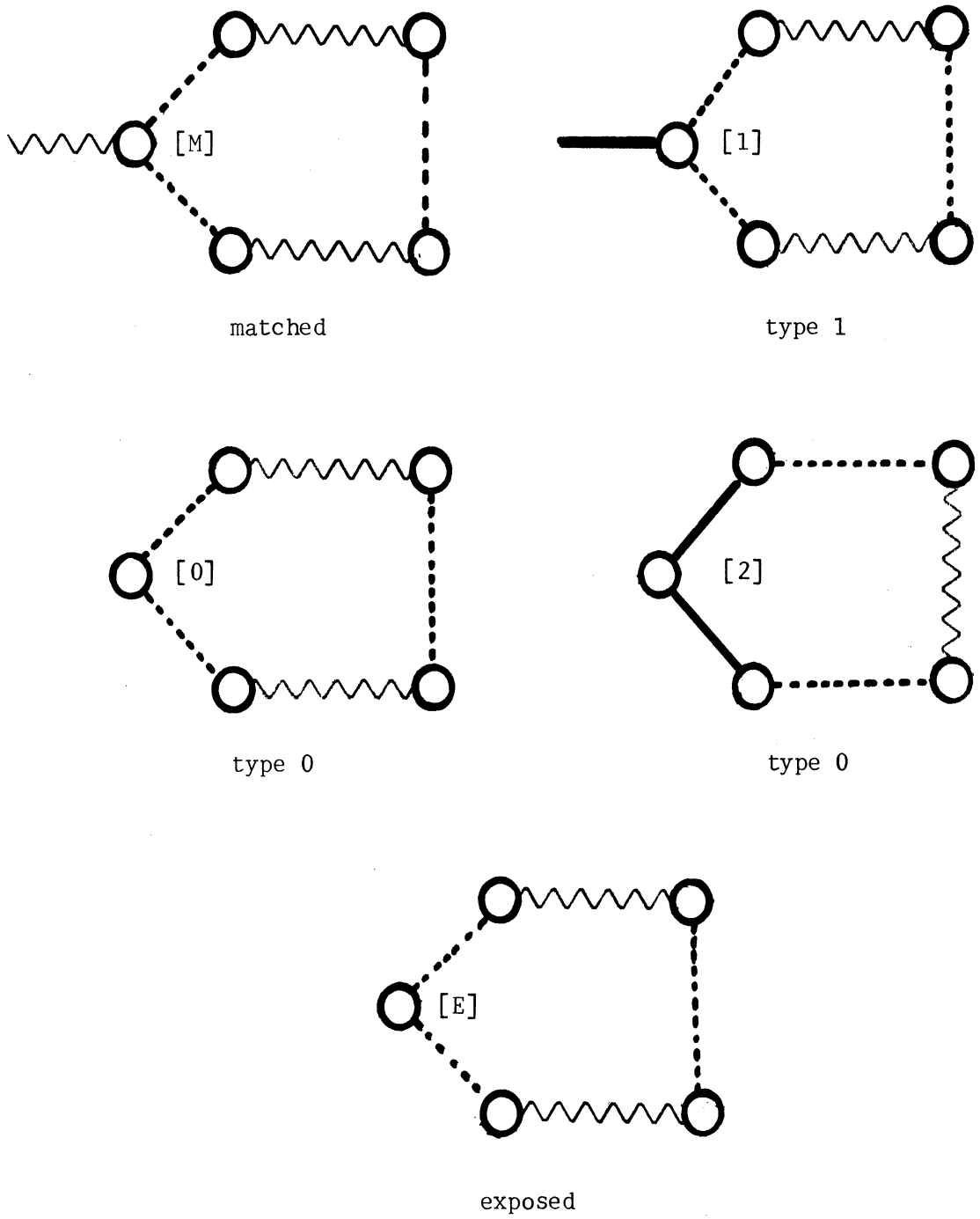

Figure 3 - Simple blossom classification. Wavy edges are matching. Straight edges are covering. Dashed edges are equality. Symbols in brackets identify the node classification of the base of the simple blossom. Simple blossom classification is given in terms of the corresponding pseudonode.

During the execution of the algorithm, nonexposed nodes (i.e., matched, type 2, type 1, type 0) remain nonexposed and at least one exposed node becomes nonexposed when an augmentation is performed, and this occurs when an augmenting path is detected. An augmenting path is an alternating path of solution/nonsolution equality edges connecting an exposed node to either another exposed node, a type 0 node, a type 1 node, a type 2 node, a matched node that can become type 2, or a matched node that can become type 0; every augmenting path has odd length except the one connecting an exposed node to a matched node that can become type 0 which has even length. An augmentation consists of reversing the solution/nonsolution role of the edges in an augmenting path. Note that all edges in such a path are equality edges and that every nonextreme node of an augmenting path remains incident with one [but not the same] solution edge. Extreme nodes of an augmenting path either become incident with a solution edge or become type 0.

After an augmentation, the apex of a pseudonode incident with a solution edge is the original node inside the pseudonode incident with this solution edge, and the apex of a type 0 pseudonode is a node of $N^{\leq} \cup N^{\geq}$ with null node price. The base of a pseudonode is always the node of the simple blossom associated with the pseudonode which either is the apex or contains inside it the apex. The two edges of the odd cycle of equality edges of the simple blossom must be solution edges, if the pseudonode is type 0 and the base is a node of $N^{\geq}$; otherwise, these two edges must be nonsolution edges. Once the role of these two edges have been determined, the solution/nonsolution role of the remaining edges of the simple blossom is uniquely specified by the property that every node of the simple blossom but the base is incident with a pair of solution-nonsolution edges of the odd cycle.

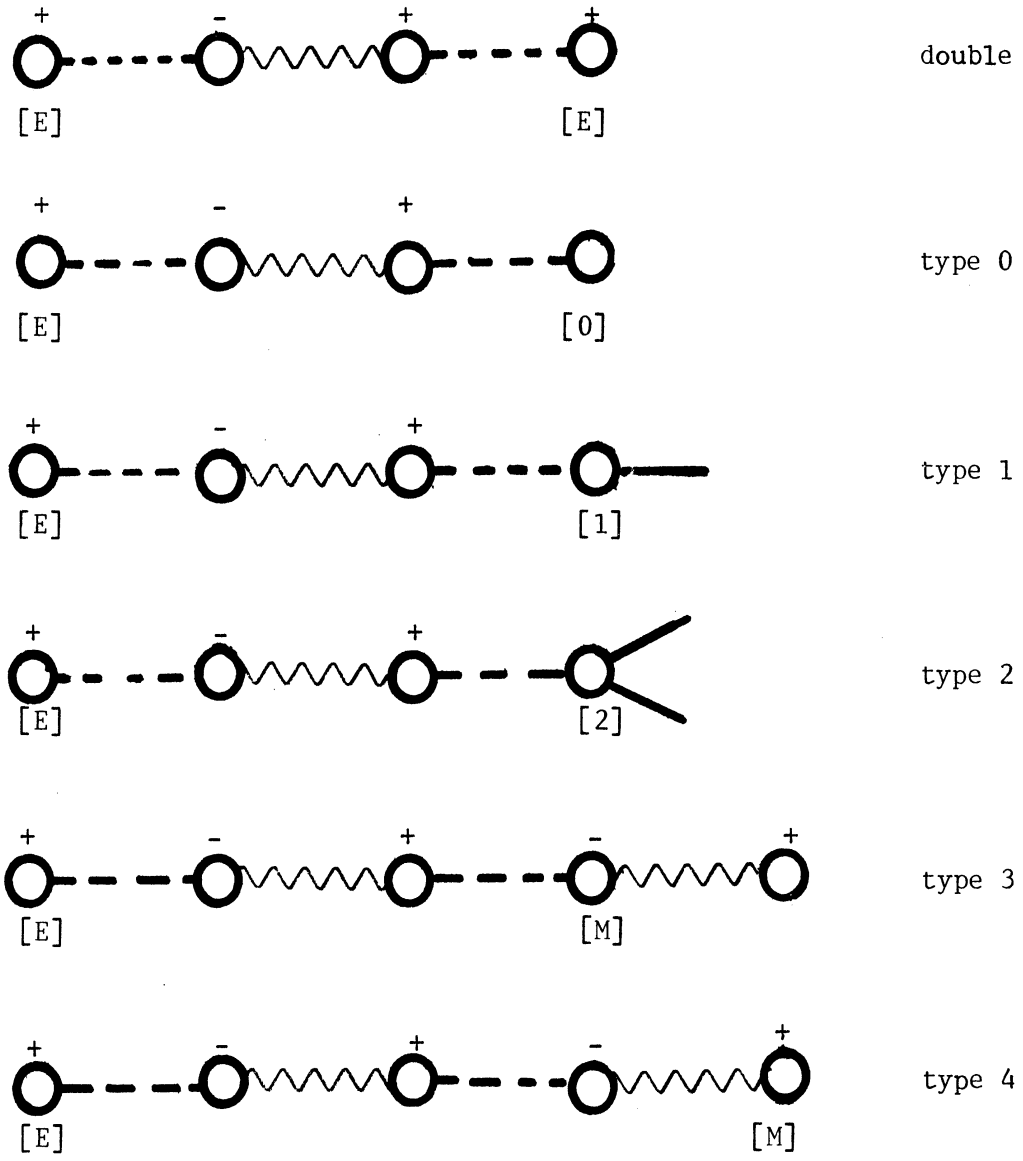

Figure 4 - Examples of augmenting paths. Wavy edges are matching. Straight edges are covering. Dashed edges are equality. Extreme nodes of augmenting paths have their node classification identified by the symbols in brackets. Classification of augmenting path is made in terms of the corresponding augmentation.

Properties I

The following properties are satisfied throughout the execution of the algorithm.

$x_{ij} = 1$ implies $d_{ij} \geq c_{ij}$

$x_{ij} = 0$ implies $d_{ij} \leq c_{ij}$

$x(i) > 1$ implies $i \in N^{\geq} \cup N^{o}$ with $y_i = 0$

$x(i) < 1$ implies either

$$\begin{cases} i \in N^{\leq} \cup N^{o} & \text{with} \quad y_i = 0 \\ i \text{ is exposed} \end{cases}$$

$$y_i \begin{cases} \leq 0 & \text{for } i \in N^{\leq} \\ = 0 & \text{for } i \in N^{=} \\ \geq 0 & \text{for } i \in N^{\geq} \end{cases}$$

$z_s \geq 0$ for $s \in S$

$z_s > 0$ implies either

$$\begin{cases} x^{*}(s) = a_s \\ s \text{ is exposed} \end{cases}$$

Conditions I

The following conditions are required before a dual solution change step be executed:

1) each current node which is exposed must be outer labelled.
2) every inner labelled node must be joined by a matching edge to an outer node.
3) no current equality edge joins an outer node to a noninner (outer or unlabelled) node.
4) no inner node of $N^o \cup N^{\leq}$ has null price.
5) no inner pseudonode has null price.
6) no outer node of $N^o \cup N^{\geq}$ has null price.
7) no outer pseudonode contains inside it a node of $N^{\leq} \cup N^{\geq}$ with null price.

In order to compute the relative cost d_{ij} of an edge joining two current nodes, define for every original node

$$y'_i = \begin{cases} y_i & i \in N^{\leq} \cup N^o \\ y_i + \Sigma_s [\, z_s : i \text{ inside } s \in S \,] & i \in N^{=} \\ y_i + 2\,\Sigma_s [\, z_s : i \text{ inside } s \in S \,] & i \in N^{\geq} \end{cases}$$

so that $d_{ij} = y'_i + y'_j$ for every edge $(i;j) \in E$ joining two current nodes which may be the original nodes i, j or pseudonodes containing i, j inside them. Therefore, an edge with $d_{ij} = y'_i + y'_j = 0$ joining two current nodes is an equality edge.

4 - Algorithm

Step 1: Initialization

Set

$$y_i = y_i' = \begin{cases} 0 & \text{for } i \in N^{\geq} \cup N^{o} \\ \min_{jk} \{ 0, c_{jk} : (j;k) \in E \} & \text{for } i \in N^{\leq} \cup N^{=} \end{cases}$$

$$z_s = 0 \quad \text{for } s \in S$$

The set of solution edges is initialized with the edges of

$$E^{-} = \{ (i;j) \in E : c_{ij} \leq 0, \; i,j \in N^{\geq} \cup N^{o} \}.$$

Identify all exposed nodes. Assign outer labels to these nodes. They are the roots of the alternating trees. Push all such nodes into the list of unscanned nodes. Go to step 2: checking the list.

Step 2: Checking the List

If there are no exposed nodes, then terminate.

If the list of unscanned nodes contains no current outer nodes, then empty the list and go to step 3: dual solution change.

Select a current outer node from the list of unscanned nodes. For every equality edge incident with this node, perform procedure 1: edge scanning, while this node remains outer labelled.

Go back to the beginning of step 2.

Step 3: Dual Solution Change

Compute:

$$\delta_1 = \min_i \{ -y_i : i \in N^{\leq} \text{ with outer label} \}$$

$$\delta_2 = \min_i \{ y_i : i \in N^{\geq} \text{ with inner label} \}$$

$$\delta_3 = \min_i \{ -y_i : i \in N^{\leq} \text{ inside } s \in S \text{ with outer label} \}$$

$$\delta_4 = \min_i \{ y_i : i \in N^{\geq} \text{ inside } s \in S \text{ with outer label} \}$$

$$\delta_5 = \min_s \{ z_s : s \in S \text{ with inner label} \}$$

$$\delta_6 = \min_{ij} \{ (c_{ij} - y'_i - y'_j)/2 : \text{edge } (i;j) \text{ joins two distinct outer nodes} \}$$

$$\delta_7 = \min_{ij} \{ c_{ij} - y'_i - y'_j : \text{edge } (i;j) \text{ joins an outer node to a current unlabelled node} \}$$

$$\delta = \min \{ \delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6, \delta_7 \}$$

If $\delta = \infty$, then stop because the problem is infeasible. Otherwise, update the dual solution.

$$y_i = \begin{cases} y_i + \delta & \text{for } i \in N \text{ with outer label} \\ & \text{or } i \in N^{\leq} \text{ inside } s \in S \text{ with outer label} \\ & \text{or } i \in N^{\geq} \text{ inside } s \in S \text{ with inner label} \\ y_i - \delta & \text{for } i \in N \text{ with inner label} \\ & \text{or } i \in N^{\leq} \text{ inside } s \in S \text{ with inner label} \\ & \text{or } i \in N^{\geq} \text{ inside } s \in S \text{ with outer label} \\ y_i & \text{for } i \in N \text{ current and unlabelled} \\ & \text{or } i \in N^{=} \text{ noncurrent} \\ & \text{or } i \in N^{\leq} \cup N^{\geq} \text{ inside } s \in S \text{ current and unlabelled} \end{cases}$$

$$z_s = \begin{cases} z_s + \delta & \text{for } s \in S \text{ with outer label} \\ z_s - \delta & \text{for } s \in S \text{ with inner label} \\ z_s & \text{for } s \in S \text{ noncurrent or unlabelled} \end{cases}$$

$$y_i' = \begin{cases} y_i' + \delta & \text{for } i \in N \text{ with outer label} \\ & \text{or } i \in N \text{ inside } s \in S \text{ with outer label} \\ y_i' - \delta & \text{for } i \in N \text{ with inner label} \\ & \text{or } i \in N \text{ inside } s \in S \text{ with inner label} \\ y_i' & \text{for } i \in N \text{ current and unlabelled} \\ & \text{or } i \in N \text{ inside } s \in S \text{ current and unlabelled} \end{cases}$$

Go to step 4: Inspection.

Step 4: Inspection

If $\delta = \delta_1$, then check whether there is an outer node of $N^{\leq}$ with null node price. While there is such a node, perform procedure 7: type 4 augmentation.

If $\delta = \delta_2$, then check whether there is an inner node of $N^{\geq}$ with null node price. While there is such a node, perform procedure 6: type 3 augmentation.

If $\delta = \delta_3$ or $\delta = \delta_4$, then check whether there is an outer pseudonode containing inside it a node of $N^{\leq} \cup N^{\geq}$ with null node price. While there is such a node, perform procedure 7: type 4 augmentation.

If $\delta = \delta_5$, then check whether there is an inner pseudonode with null pseudonode price. While there is such a pseudonode, perform procedure 9: pseudonode unshrinking.

If $\delta = \delta_6$, then check whether there is an equality edge joining two outer nodes. While there is such an edge, perform procedure 2: double augmentation if the outer nodes are in different alternating trees, otherwise perform procedure 8: simple blossom shrinking.

If $\delta = \delta_6$ or $\delta = \delta_7$, then check whether there is an equality edge joining an outer node with an unlabelled node. While there is such an edge, perform procedure 1: edge scanning.

Go back to step 2: checking the list.

Procedure 1: Edge Scanning

This procedure is performed when there is an outer node i joined by an equality edge to a current node j.

If node j is outer labelled and nodes i, j are in the same alternating tree, then perform procedure 8: simple blossom shrinking, and return.

If node j is outer labelled and nodes i, j are in different alternating trees, then perform procedure 2: double augmentation, and return.

If node j is a type 0 node, then perform procedure 3: type 0 augmentation, and return.

If node j is a type 1 node, then perform procedure 4: type 1 augmentation, and return.

If node j is a type 2 node, then perform procedure 5: type 2 augmentation, and return.

If node j is a matched node, then let node k be the current node joined to j by a matching edge. Assign to nodes j, k inner and outer labels of the alternating tree of node i whose predecessors are nodes i, j, respectively. Push node k into the list of unscanned nodes. If $j \in N^{\geq} \cup N^{0}$ with $y_j = 0$, then perform procedure 6: type 3 augmentation and return. If $j \in S$ with $z_j = 0$, then perform procedure 9: pseudonode unshrinking. If $k \in N^{\leq} \cup N^{0}$ with $y_k = 0$, then perform procedure 7: type 4 augmentation and return. If $k \in S$ and contains inside it a node of $N^{\leq} \cup N^{\geq}$ with null node price, then perform procedure 7: type 4 augmentation and return.

Return.

Procedure 2: Double Augmentation

This procedure is performed when two outer nodes i, j of different alternating trees are joined by an equality edge. Let $P(i)$, $P(j)$ denote the alternating paths connecting nodes i, j to

with the roots of their alternating trees; such paths are obtained by backtracking the predecessor indices. The augmenting path determined by $P(i) \cup \{(i;j)\} \cup P(j)$ connects two exposed nodes. Reverse the solution/nonsolution role of the equality edges in this path. Every matched node remains matched, and the two exposed nodes become matched. Erase the label of all nodes in the alternating trees of nodes i, j. Push the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 3: Type 0 Augmentation

This procedure is performed when an outer node i is joined by an equality edge to a current type 0 node j. Let $P(i)$ be the alternating path connecting node i with the root of its alternating tree. The augmenting path determined by $P(i) \cup \{(i;j)\}$ connects an exposed node with a type 0 node. Reverse the solution/nonsolution role of the edges in this path. Every matched node remains matched, and the exposed as well as the type 0 node become matched. Erase the label of all nodes in the alternating tree of node i. Push the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 4: Type 1 Augmentation

This procedure is performed when an outer node i is joined by an equality edge to a current type 1 node j. Let $P(i)$ be the alternating path connecting node i with the root of its alternating tree.

If $j \in N^{\geq} \cup N^{o}$ with $y_j = 0$, then the augmenting path determined by $P(i) \cup \{(i;j)\}$ connects an exposed node with a type 1 node. Reverse the solution/nonsolution role of the edges in this path. Every matched node but node i remains matched. Node j becomes type 2. Node i becomes type 1. The exposed node, if other than node i, becomes matched.

Otherwise, let node k be the type 2 node joined to node j by a covering edge. The augmenting path determined by $P(i) \cup \{(i;j), (j;k)\}$ connects an exposed node with a type 2 node and goes through a type 1 node. Reverse the solution/nonsolution role of the edges in this path. Every matched node remains matched. The exposed node becomes matched. Node j becomes matched. Node k becomes matched, type 1, or remains type 2 depending on the solution edges incident with it.

Erase the label of all nodes in the alternating tree of node i. Pushing the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 5: Type 2 Augmentation

This procedure is performed when an outer node i is joined by an equality edge to a current type 2 node j. Let P(i) be the alternating path connecting node i with the root of the alternating tree. The augmenting path $P(i) \cup \{(i;j)\}$ connects an exposed node with a type 2 node and does not goes through a type 1 node. Reverse the solution/nonsolution role of the edges in this path. Every matched node but node i remains matched. Node j remains type 2. Node i becomes type 1. The exposed node, if other than node i becomes matched.

Erase the label of all nodes in the alternating tree of node i. Push the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 6: Type 3 Augmentation

This procedure is performed when there is an inner matched node $j \in N^{\geq} \cup N^{o}$ with $y_j = 0$. Let $P(j)$ denote the alternating path connecting node j with the root of the alternating tree. The augmenting path $P(j)$ connects an exposed node with a matched node. Reverse the solution/nonsolution role of the edges in this path. Node j becomes type 2. The node which was joined by a matching edge to node j and the node which was the predecessor of node j in the alternating path, both become type 1. The remaining matched nodes remain matched. The exposed node, if other than the predecessor of node j, becomes matched. Erase the label of all nodes in the alternating tree of node j. Push the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 7: Type 4 Augmentation

This procedure is performed when there is an outer node $i \in N^{\leq} \cup N^{o}$ with $y_i = 0$ or there is an outer pseudonode $i \in S$ containing inside it a node $k \in N^{\leq} \cup N^{\geq}$ with $y_k = 0$. Let $P(i)$ be the alternating path connecting node i with the root of the alternating tree. The augmenting path $P(i)$ connects an exposed node with a matched node. Reverse the solution/nonsolution role of the edges in this path. Node i becomes type 0; if node i is a pseudonode, then node k becomes the new apex of node i. Every other matched node remains matched. The exposed

node, it other than node i, becomes matched. Erase the label of all nodes in the alternating tree of node i. Push the remaining outer nodes into the list of unscanned nodes. Return.

Procedure 8: Simple Blossom Shrinking

This procedure is performed when two outer nodes i, j of the same alternating tree are joined by an equality edge. Let P(i), P(j) be the alternating paths connecting nodes i, j with the root of the alternating tree, respectively. The odd alternating cycle of equality edges determined by $[\ P(i) \cup \{(i;j)\} \cup P(j)\ ] \setminus [\ P(i) \cap P(j)\ ]$ defines a simple blossom. Introduce a pseudonode containing inside all nodes and edges of this simple blossom. The base of this pseudonode is the only node incident with two distinct edges of P(i), P(j). The apex of this pseudonode is the apex of its base. The pseudonode receives an outer label of the same alternating tree and with the same predecessor of its base. The label of every node in the simple blossom is erased.

Check whether there is a node of $\overset{\geq}{N} \cup \overset{\leq}{N}$ with null node price inside the pseudonode. Perform procedure 7: type 4 augmentation.

Return.

Procedure 9: Pseudonode Unshrinking

This procedure is performed when an inner pseudonode s has null pseudonode price. The pseudonode is eliminated from the network, so that the nodes in its simple blossom become current. Let node i

be the base of pseudonode s and let node j be the node in the simple blossom s which is incident with an equality edge joining it with the predecessor of pseudonode s. Let $P(i;j)$ be the even length path on the edges of the simple blossom s connecting the nodes i, j. Assign inner/outer labels to the nodes in $P(i;j)$ starting with an inner label for node j and finishing with another inner label for node i. Push all the newly labelled nodes into the list of unscanned nodes. Check whether any newly outer labelled node is a node of $N^{\leq}$ with null price or is a pseudonode containing a node of $N^{\leq} \cup N^{\geq}$ with null price; perform procedure 7: type 4 augmentation and return. Check whether any newly inner labelled node is a node of $N^{\geq}$ with null price; perform procedure 6: type 3 augmentation and return. Check whether any newly inner labelled node is a pseudonode with null node price; perform this procedure 9. Check whether there was any equality edge joining a pseudonode s with an outer node other than its predecessor; for every such an edge perform procedure 1: edge scanning. Return.

5. PROOF OF THE ALGORITHM

Theorem 2. If there is no exposed node at the end of the algorithm, the final solution is optimal.

Proof. It can be easily verified that properties I are satisfied throughout the algorithm; since there is no exposed nodes at the end of the algorithm, the final solution is primal feasible, dual feasible, and satisfies the complementary slackness conditions. Therefore, it is an optimal solution for the problem.

Theorem 3. At each execution of the dual solution change step, the dual objective value strictly increases.

Proof. It can be easily verified that conditions I are satisfied before each execution of the dual solution step. It follows that the variable h is always strictly positive or some condition would be violated. By the way the dual solution change is performed, the dual objective value increases by the positive amount

$$(\#\text{ exposed nodes}) \cdot \delta$$

Theorem 4. If there is an exposed node at the end of the algorithm, then the problem is infeasible.

Proof. The algorithm stops with exposed nodes only if $\delta = \infty$. This implies that the dual problem is unbounded and therefore the primal problem is infeasible.

Theorem 5. The time complexity of the algorithm is at most $O(n^2m)$.

Proof.

At each augmentation at least one original node becomes nonexposed and all nonexposed nodes remain nonexposed. Since there are at most n original nodes which are exposed at the beginning of the algorithm, it is possible to occur at most n augmentations during the execution of the algorithm.

Based on the facts that between two consecutive augmentations:

1) there are at most $n/2$ pseudonodes at any time
2) a simple blossom shrinking produces an outer labelled pseudonode
3) a pseudonode unshrinking is performed on an inner labelled pseudonode and produces at least one inner labelled node
4) a node labelling produces an outer-inner pair of labelled nodes
5) labelled nodes remain labelled while they are current

it follows that there are at most $n/2$ node labellings, at most $n/2$ simple blossom shrinkings, and at most $n/2$ pseudonode unshrinkings between two consecutive augmentations.

Finally, between two operations of labelling, shrinking, or unshrinking, there is at most one execution of the dual solution change step, which requires an effort $O(n+m)$ to be performed.

In short, there are $O(n)$ augmentations, $O(n)$ dual solution changes per augmentation, and an effort $O(n+m)$ at each execution of the dual solution change step. So, the overall time complexity is at most $O(n^2m)$ [assuming $m > n$].

Comment

It is possible to have an implementation with time complexity $O(n^3)$. Such an implementation for the minimum cost 1-perfect matching is presented by Lawler [6] and the same approach can be applied for the minimum cost 1-matching/covering problem. Essentially, it consists of spending an effort $O(n)$ at each dual solution change; this means that only the nodes are examined during the dual solution change step at the moment the variable δ is computed. This implies that the examination of the edges should be done before, at the same time that the edges are being scanned during the tree growing process. Essential edge information must be transformed and stored as node information. In addition, the simple blossom shrinking and the pseudonode unshrinking operations become more complex because more information associated with the nodes need to be updated. Finally, after each augmentation, all alternating trees must be erased and new alternating trees should be rooted and grown again.

REFERENCES

1 - J. Edmonds, "Paths, Trees, and Flowers", Canad. J. Math. 17(1965), 449-467.

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

3 - J. Edmonds and E.L. Johnson, "A Well-Solved Class of Integer Linear Programs" in: "Combinatorial Structure and Their Applications", R. Guy (ed.), Gordon and Breach, New York, 1970, 89-92.

4 - J. Edmonds, E.L. Johnson, and S.C. Lockhart, "Blossom I: Computer Code for the Matching Problem", IBM T.J. Watson Center, Yorktown Heights, New York, 1969.

5 - J. Edmonds and W. Pulleyblank, "The Matching Problem and the Blossom Algorithm", lecture notes, The Johns Hopkins University (1975).

6 - E.L. Lawler, "Combinatorial Optimization: Networks and Matroids", Holt-Rinehart-Winston, New York, 1976.

7 - E. Minieka, "Optimization Algorithms for Networks and Graphs", Dekker New York, 1978.

8 - K.G. Murty - manuscripts for Combinatorial Programming.

9 - K.G. Murty and C. Perin, "A 1-Matching Blossom Type Algorithm for Edge Covering Problems", tech. rept. 79-6, Dept. of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1979.

10 - C. Perin, "Matching and Edge Covering Algorithms", Ph.D. dissertation, Dept. of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1980.

11 - L.J. White and M.L. Gillenson, "An Efficient Algorithm for Minimum k-Covers in Weighted Graphs", Math. Prog. 8(1975), 20-42.