EXTENSIONS OF MATCHING THEORY

by

David B. Hartvigsen

Submitted in partial fulfillment of the
requirements for the Degree
Doctor of Philosophy

Department of Mathematics
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

September, 1984

Table of Contents

Chapter 1

INTRODUCTION

Section 0. Introduction

An optimization problem consists of a set F and a cost function $c : F \to R^1$. The problem is to find

$$\max\{c(f) : f \in F\}.$$

These problems can roughly be divided into two classes. If $|F|$ is uncountably infinite, then the problem is called continuous. If $|F|$ is finite or countably infinite, the problem is called discrete or combinatorial.

The problems we deal with are combinatorial. In Section 4 we define some problems in this class and look at some methods used to solve them. We set the stage for this with some background material in Sections 1 - 3. Many combinatorial optimization (c.o.) problems come from graph theory, so Section 1 contains an introduction to this field (as well as a definition of matroids and some notation we will be using). An important area used in solving c.o. problems is linear programming which is outlined along with some polyhedral theory in Section 2. In Section 3 we look at an area which greatly affects the approach taken to solve c.o. problems -- complexity theory.

In Section 5 we focus upon an area in c.o. known as matching theory and in Section 6 some generalizations of this theory are developed. Finally, a summary of our results is given in Section 7.

Section 1. Graph Theory, Matroids, and Notation

In this section we define the concepts of graph theory which will be useful in the work to follow. For an introduction to graph theory see Bondy and Murty [76].

A graph G is an ordered pair $(V(G), E(G))$ where $V(G)$ is a nonempty set of elements called nodes and $E(G)$ is a collection of unordered pairs, called edges, of elements of $V(G)$ with no pair occurring more than once in $E(G)$ (i.e. our graphs have no multiple edges but may have loops). When it is clear which graph G we are referring to, we may abbreviate $V(G)$ and $E(G)$ with V and E, respectively. We will refer to an edge containing nodes u and v as (u,v) or uv. A graph may be represented on paper by drawing a point for each node and, for each edge in E, joining the corresponding points with a line. For convenience when dealing with a particular graph, we will use the word "graph" to refer to both its abstract definition and any particular graphical representation of it.

An edge is said to be incident with a node if the edge contains that node. We denote the collection of all edges incident with a node v by $\delta(v)$. In general, for a set $S \subseteq V$, we denote the collection of edges with exactly one node in S by $\delta(S)$. We denote the collection of edges with both nodes in S by $\gamma(S)$. The degree of a node v in a graph G, denoted $d_G(v)$, is $|\delta(v)|$. Two edges are said to be adjacent if they are incident with a common node and, similarly, two nodes are said to be adjacent if they are incident with a common edge.

A subgraph $H = (V',E')$ of a graph $G = (V,E)$ is a graph such that $V' \subseteq V$ and $E' \subseteq E$. If V' is nonempty then the subgraph $(V',\gamma(V'))$ is called the <u>subgraph of G induced by V'</u> and is denoted $G[V']$. We will denote the graph $G[V\setminus\{v\}]$ by $G\setminus v$. If $e \in E$, then we will call the graph obtained from G by removing e, $G\setminus e$. If $E' \subseteq E$ is nonempty then the subgraph (V',E'), where V' is the set of nodes incident with an edge in E', is called the <u>subgraph of G induced by E'</u>.

The <u>complete graph</u> K_n, $n \geq 1$, is a graph with n nodes and all possible (i.e. $\binom{n}{2}$) edges. Hence K_1 is a single node. K_2 is an edge, K_3 is a triangle, etc. Complete graphs are also known as <u>cliques</u>. A <u>bipartite graph</u> is a graph with the property that its nodes can be partitioned into two sets X and Y so that each edge has one node in X and the other in Y.

A <u>trail</u> in a graph G is an alternating sequence of nodes and edges $v_0e_1v_1 \ldots e_kv_k$, $k \geq 1$, with the property that $e_i = (v_{i-1},v_i)$ for $1 \leq i \leq k$ and all the edges are distinct. If all the nodes of a trail are distinct, it is called a <u>path</u>. A trail is called <u>closed</u> if $v_0 = v_k$. A closed trail in which all the nodes except the first and last are distinct is called a <u>cycle</u>. A <u>Hamilton cycle</u> or <u>Hamilton tour</u> is a cycle which contains every node of the graph. The <u>length</u> of a trail is equal to its number of edges and determines whether it is odd or even. We may often refer to a path, a closed trail, or a cycle just by the edges which it contains.

A graph is said to be <u>connected</u> if there exists a path between every pair of its nodes. The <u>components</u> of a graph are the subgraphs which are maximal with respect to connectivity.

A <u>tree</u> is a connected graph which contains no cycles. A <u>forest</u> is a graph whose components are trees.

A <u>cutnode</u> v of G is a node such that $G\backslash v$ has more components than G. A connected graph with no cutnodes is called a <u>block</u>. A <u>block of a graph</u> is a subgraph that is a block and is maximal with respect to this property.

A <u>matching</u> of a graph is a set of edges no two of which are adjacent. Any node contained in an edge of a matching M is said to be <u>saturated by</u> M. A <u>perfect matching</u> is one which saturates every node of the graph. A <u>near-perfect matching</u> is a matching which saturates all but one of the nodes. A graph H is <u>hypomatchable</u> if its node set V has odd cardinality and for every $v \in V$, there is a near-perfect matching of H which is deficient at v.

Given a graph $G = (V,E)$, <u>shrinking the set</u> $S \subset V$ yields a graph $\tilde{G}$ obtained from G by removing the nodes in S and replacing them with a single node which is adjacent to every node which was adjacent to a node in S.

We next define matroids. For an introduction to matroids see Welsh [76] or Lawler [76].

A <u>matroid</u> is a pair $(S,\mathcal{I})$ where S is a finite set, called the <u>ground set</u>, and $\mathcal{I}$ is a collection of subsets of S which satisfy the following:

i) $\emptyset \in \mathcal{I}$

ii) $I \subset J \in \mathcal{I} \Rightarrow I \in \mathcal{I}$

iii) $|I_p| = p$ and $|I_{p+1}| = p + 1 \Rightarrow \exists\ e \in I_{p+1} \backslash I_p$ s.t. $I_p \cup \{e\} \in \mathcal{I}$.

There are many known classes of matroids. For example, if for some graph $G = (V,E)$ we take

$$\mathcal{I} = \{I \subseteq E : I \text{ is the set of edges of a subgraph of } G \text{ which is a forest}\}$$

then $(E,\mathcal{I})$ is called a graphic matroid. If S = set of columns of a matrix A and

$$\mathcal{I} = \{I \subseteq S : \text{the columns in } I \text{ are linearly independent}\},$$

then $(S,\mathcal{I})$ is called a representable matroid.

We will use the following notational conventions.

If $x = (x_i : i \in I)$ and $S \subseteq I$, where I is a finite set, then $x(S)$ denotes $\Sigma_{i\in S}\ x_i$.

If α is a number, then $\lfloor\alpha\rfloor$ denotes the greatest integer less than or equal to α.

1 is used to denote the vector of 1's of size appropriate to the context.

Section 2. Linear Programming and Polyhedral Theory

In this section we discuss the notions of linear programming which will be useful later. Linear programming is an example of why it is sometimes difficult to classify optimization problems as either continuous or combinatorial. We define linear programming as a continuous optimization problem but we will see that it also has a combinatorial interpretation, which is what makes it useful to us. For a history of linear programming, see Dantzig [63].

Given an $m \times n$ matrix A, an m-vector b, and an n-vector c (called the cost vector), we may define two related linear programs (LP's), the primal LP and its dual LP, as follows:

$$\text{Primal LP} : \max\{cx : Ax \leq b,\ x \geq 0\}$$
$$\text{Dual LP} : \min\{yb : yA \geq c,\ y \geq 0\}.$$

cx and yb are called objective functions and the inequalities associated with each LP are called its constraints. Any x (y) which satisfies the primal (dual) constraints is called a primal (dual) feasible solution. A primal (dual) feasible solution x^* (y^*) such that

$$cx^* = \max\{cx : Ax \leq b,\ x \geq 0\} \quad (y^*b = \min\{yb : yA \geq c,\ y \geq 0\})$$

is called a primal (dual) optimal solution.

If we let x' be any primal feasible solution and let y' be any dual feasible solution, then

$$cx' \leq y'Ax' \leq y'b$$

or

$$\max\{cx : Ax \leq b,\ x \geq 0\} \leq \min\{yb : yA \geq c,\ y \geq 0\}.$$

Thus we get the Weak Duality Theorem of Linear Programming which states:

Theorem 1.1 (Weak Duality Theorem): If $\bar{x}$ and $\bar{y}$ are primal and dual feasible solutions, respectively, such that $c\bar{x} = \bar{y}b$, then $\bar{x}$ and $\bar{y}$ are primal and dual optimal solutions, respectively.

There is also a Strong Duality Theorem of Linear Programming, which was proved by J. von Neuman (see Dantzig [63]) and Gale, Kuhn and Tucker [51].

Theorem 1.2 (Strong Duality Theorem): If $\max\{cx : Ax \leq b, x \geq 0\}$ exists, then $\min\{yb : yA \geq c, y \geq 0\}$ exists, and $\max\{cx : Ax \leq b, x \geq 0\} = \min\{yb : yA \geq c, y \geq 0\}$.

The following corollary of this theorem, referred to as Complementary Slackness (see Dantzig [63] for a proof), will be useful to us.

Corollary 1.2 (Complementary Slackness): If x and y are primal and dual feasible solutions, respectively, then they are both optimal iff

i) $x_i > 0 \Rightarrow yA_i = c_i$ for $i = 1,\ldots,n$

ii) $y_j > 0 \Rightarrow a_j x = b_j$ for $j = 1,\ldots,m$ where A_i is the ith column of A and a_j is the jth row of A.

A set of points of the form $\{x : Ax \leq b\}$ is called a polyhedron. For background on polyhedral theory see Rockafellar [70] and Pulleyblank [82]. A polyhedron P in $\mathbb{R}^n$ is bounded if there exist $\ell, u \in \mathbb{R}^n$ such that $\ell \leq x \leq u$ for all $x \in P$. We refer to a bounded polyhedron as a polytope.

Let X be a finite subset of $\mathbb{R}^n$. Then X is said to be affinely independent if whenever $\Sigma(\lambda_x x : x \in X) = 0$ and $\Sigma(\lambda_x : x \in X) = 0$ for $\lambda \in \mathbb{R}^{|x|}$, we have $\lambda = 0$. The dimension of a set $S \subseteq \mathbb{R}^n$, denoted $\dim(S)$, is the maximum number of affinely independent vectors in S.

An inequality $ax \leq \alpha$ is said to be valid for a polyhedron P if $P \subseteq \{x : ax \leq \alpha\}$. If for a valid inequality $ax \leq \alpha$, $F = \{x : ax = \alpha\} \cap P \neq \emptyset$, then the set F is called a face of P. A face F such that $\dim(F) = \dim(P) - 1$ is called a facet of P. We will use the word facet to refer to both the set F and the inequality $ax \leq \alpha$. A facet is said to be $0 - 1$ if the inequality can be scaled (multiplied by a positive scalar) so that the coefficients of x are 0 or 1. A $0 - 1 - 2$ facet is defined similarly. A face F such that $\dim(F) = 1$ is called vertex of P. Hence a vertex is a single point.

Note that, if a polyhedron P in $\mathbb{R}^n$ has full dimension, i.e. $\dim(P) = n$, then there exists a unique minimal defining system for P, up to positive multiples of the inequalities, where the inequalities are the facets of P.

The following theorem about vertices is well known:

Theorem 1.3: Given any polytope $P = \{x : Ax \leq b,\ x \geq 0\}$, and any objective function cx, there exists a vertex x^* of P which is an optimal solution to the LP : $\max\{cx : x \in P\}$.

This implies that all the information necessary to solve an LP over a polytope is contained in the vertices of the polytope.

So the above continuous optimization problem may be rewritten as a c.o. problem as follows:

$$\max\{cx : x \in P\} = \max\{cx : x \in \{\text{vertices of } P\}\}.$$

Thus linear programming has a decided combinatorial interpretation which we will discuss further in Section 4.

Section 3. Computational Complexity

The goal of computational complexity is to measure the difficulty of problems. By a "problem" we mean a fairly general "question" which can be applied to a class of problem "instances". For example, the problem of linear programming consists of the question "What is $\max\{cx : Ax \leq b, x \geq 0\}$?" and a class of instances each of which is a specific choice of c, A, and b. The difficulty of a problem is measured by examining algorithms to solve it. The following ideas were first proposed by Edmonds [65a]. For the history and development of this subject see Cook [71], Karp [72], and Garey and Johnson [79].

Suppose we are considering an algorithm which solves some problem. Let the <u>size</u> of an instance be the number of bits required to encode it and let $f(n)$ be the maximum number of steps that the algorithm takes to run on all instances of size n. Then, if there exists a polynomial $p(n)$ such that $f(n) \leq p(n)$ for all n, we say that the algorithm is <u>good</u>. A problem is said to belong to the class P if there exists a good algorithm to solve it.

For many problems it is not known if they are in P or not. A number of these problems, called NP-complete problems, have the property that if any one of them is in P, then they are all in P.

NP-complete problems seem to be inherently more difficult than problems in P. Algorithms which solve problems in P tend to be very problem specific in that they exploit the problem's structure. Algorithms which solve NP-complete problems tend to use more general approaches, e.g. enumeration, cutting planes, or branch and bound. A problem is called NP-hard if it is at least as difficult to solve as the NP-complete problems.

Linear programming is a problem whose classification was not known until recently. It was shown by Khachian [79] using the ellipsoid method of Shor [70] and Yudin and Nemirowski [76], that linear programming is in P (see Bland, Goldfarb, and Todd [81] for a survey; see also Grotschel, Lovasz, Schrijver [81] for applications of the ellipsoid method in combinatorial optimization).

Section 4. Combinatorial Optimization

A large number of c.o. problems arise from graph theory. In fact it is often possible to prove graph theoretically that many c.o. problems arise in dual pairs where a maximization problem shares its optimum value with a related minimization problem.

A well-known example of this involves matchings on bipartite graphs. Given a bipartite graph, a maximization problem is to find a maximum cardinality matching and a minimization problem is to find a minimum cardinality set of nodes such that every edge contains at least one of these nodes (this is called a minimum cardinality node cover). The following theorem is due to König [31].

Theorem 1.4: For any bipartite graph, the maximum cardinality of a matching is equal to the minimum cardinality of a node cover.

Other examples of c.o. problems like this are Menger's Theorem [27], the Max-flow Min-cut Theorem (see Dantzig [51], Ford and Fulkerson [56], and Elias, Feinstein, and Shannon [56]), and the Lucchesi-Younger Theorem (see Lucchesi and Younger [78]). (For surveys of results in c.o. see Lovasz [79] and Schrijver [83].)

c.o. problems may also be treated algebraically in vector spaces. Let us illustrate this with the bipartite matching example. Given a bipartite graph $G = (V,E)$, we associate with every matching M an incidence vector $x^M \in \mathbb{R}^{|E|}$ where

$$x_e^M = \begin{cases} 1 & \text{if } e \in M \\ 0 & \text{otherwise} \end{cases} .$$

It is well known that the convex hull of a finite collection N of points in $\mathbb{R}^n$ yields a polytope whose vertices are in N. If we let P be the convex hull of the incidence vectors of matchings of G, then we can state the maximum cardinality matching problem as follows:

$$\max\{1 \cdot x : x \in P\}.$$

A more useful description of a polytope is in terms of its defining inequalities which then gives us an LP. We will refer to a system of inequalities which define a given polytope as a (complete) polyhedral characterization of the polytope. (Note that in the above problem the objective function is fixed. When this occurs we may not need to know the entire collection of inequalities defining the polytope in order to solve it as an LP. Thus we will also talk of a polyhedral characterization of a polytope sufficient to solve the given problem as an LP. Such a characterization may consist of only a proper subset of the inequalities in a complete polyhedral characterization.)

A first step toward finding such an LP is to write the problem as an LP with the property that the integral points which satisfy the constraints are exactly the incidence vectors. If we then add the constraint that all solutions must be integral we get an integer program (IP). In our example this is easy to do:

$$\max\{1 \cdot x : x(\delta(v)) \leq 1 \quad \forall\ v \in V,\ x \geq 0,\ x \quad \text{integral}\}.$$

The question to now ask is "If we drop the integrality constraint of our IP, how is the resulting polytope related to the one we seek?" (This process of dropping constraints is known as a relaxation.) The answer is that sometimes the relaxation gives us the polytope we seek and other times the relaxation has fractional vertices which require the addition of more inequalities to the system.

In our example we are lucky in that the polytope $\{x : x(\delta(v)) \leq 1 \ \ \forall\, v \in V,\ x \geq 0\}$ has all integral vertices and is therefore the polytope we seek. (This is a consequence of the fact that the system of inequalities forms a totally unimodular matrix.) (See Hoffman and Kruskal [56] and Hoffman [60].) Thus we are actually addressing the <u>maximum weight bipartite matching problem</u> where a weight is assigned to each edge and we want a matching such that the sum of the weights on its edges is a maximum. As an LP, with c the vector of weights, this problem is

$$\max\{cx : x(\delta(v)) \leq 1 \ \ \forall\, v \in V,\ x \geq 0\}.$$

By examining the dual to this LP,

$$\min\{y \cdot I : y_u + y_v \geq c_e \ \forall\, e = (u,v) \in E,\ y \geq 0\},$$

we may derive some max-min theorems. The polytope for this LP has all integral vertices whenever c is integral (again since the system of inequalities forms a totally unimodular matrix). For the cardinality problem (i.e. when $c \equiv 1$) the dual constraints define the polytope which is the convex hull of incidence vectors of node covers. So by LP duality we get our max-min Theorem 1.4. From the weighted problem we get a more general max-min theorem.

For a graph with weights c_e assigned to each edge $e \in E$, a <u>weighted node cover</u> is an assignment of non-negative integers to the nodes so that for each edge the sum of the integers assigned to its nodes is at least equal to the weight of that edge. So in the minimum weight node cover problem we seek a weighted node cover such that the sum of the integers assigned to the nodes is a minimum. With this definition LP duality gives us:

Theorem 1.5: For any bipartite graph with integer weights c_e for $e \in E$, the maximum weight matching problem and the minimum weight node cover problem have the same optimum value.

Note that the number of constraints in the LP's for the bipartite matching problems is polynomial in the size at the instance (i.e. the size at the graph). Therefore these problems are in P due to Khachian's result. There also exist much more efficient algorithms for these problems which exploit the structure of bipartite graphs. (We will examine in detail this type of algorithm in the next section.) So for both the cardinality and weighted bipartite matching problems we have max-min theorems, polyhedral characterizations, and polynomial algorithms. Results of this type have been found for all the c.o. problems listed at the beginning of this section.

The problem of matchings on general graphs presents difficulties not encountered on bipartite graphs. Although the IP for the bipartite case is also an IP for the general case, the polytope $\tilde{P}$ of the relaxation does not have all integral vertices for general graphs. To overcome this difficulty, c.o. has some more powerful tools which we describe in the next section on matching theory.

Section 5. Matchings

In this section we take a detailed look at results on matchings in general graphs. As with the bipartite matching problem we get max-min theorems, polyhedral characterizations, and a polynomial algorithm. We present an approach developed by Edmonds [65] in which a polynomial algorithm is devised and then a polyhedral characterization and max-min theorems are obtained as byproducts.

A first step in Edmonds' approach is to conjecture an LP formulation of the weighted problem; i.e., to give a system of valid inequalities which are conjectured to be a polyhedral characterization of the convex hull of incidence vectors of matchings of a graph. Using this conjecture a polynomial algorithm is devised which solves the maximum weight matching problem. For any objective function, the algorithm constructs a primal feasible solution which is a matching and proves its optimality by constructing a dual feasible solution with the same optimum value. Thus, by the following well-known theorem of convexity theory, every vertex of the polytope is an incidence vector of a matching which means the conjectured polyhedral characterization is correct.

Theorem 1.6: For each vertex of a polytope there exists an objective function which attains its optimum only at that vertex.

As in the bipartite case, a good place to start looking for an LP formulation is as a relaxation of an IP formulation. We noted in the last section that the IP formulation we used for the bipartite case also works for the general case:

$$\max\{cx : x(\delta(v)) \leq 1 \ \ \forall v \in V, \ x \geq 0, \ \ x \text{ integral}\}.$$

The polytope $\tilde{P}$ of the relaxation does not have all integral vertices for general graphs, however, thus making the matching problem a "non-trivial" c.o. problem. Consider the simplest graph which is not bipartite -- the triangle -- with values of 1/2 assigned to each edge. This fractional point is a vertex of $\tilde{P}$. Edmonds took as his conjecture the following system:

$$(1.1) \qquad \begin{cases} x(\delta(v)) \leq 1 & \forall v \in V \\ x(\gamma(S)) \leq \left\lfloor \frac{|S|}{2} \right\rfloor & \forall S \subseteq V, \ |S| \text{ odd.} \\ x \geq 0 \end{cases}$$

Before giving the algorithm we need a theorem due to Berge [57] and some definitions.

Given a graph G and a matching M, an <u>alternating path relative to</u> $\underline{M}$ is a path whose edges are alternately in and out of the matching. An <u>augmenting path</u> is an alternating path whose first and last edges are not in the matching.

<u>Theorem 1.7</u>: A matching M is maximum iff there exist no augmenting paths relative to M.

The primal part of the algorithm is essentially a search for augmenting paths. This is accomplished by growing an alternating forest which consists of node disjoint alternating trees whose structure we now describe.

A surface graph $\tilde{G}$ of G is obtained from G by shrinking a collection of pairwise disjoint subsets $S_1,\ldots,S_k$ of nodes of G where the graphs $(S_i,\gamma(S_i))$ for $i = 1,\ldots,k$ are hypomatchable. In fact, each S_i may be shrunk sequentially in the course of the algorithm so that $G \times S_i = (G \times S_i^1) \times S_i^2 \times\ldots\times S_i^j$ where $(S_i^1 \cup\ldots\cup S_i^m, \gamma(S_i^1 \cup\ldots\cup S_i^m))$ is hypomatchable for $1 \leq m \leq j$. (Note that the order of shrinking is important since the sets S_i^m for $1 \leq m \leq j$ need not be disjoint.) Expanding the shrunk node i which corresponds to S_i yields the graph $\tilde{G}' = G \times S_1 \times\ldots\times S_{i-1} \times (S_i^1 \times\ldots\times S_i^{j-1}) \times S_{i+1} \times\ldots\times S_k$. If a node u of G is contained in a shrunk node v of $\tilde{G}$, then we say surface$(u) = v$.

Let M be a matching of G. An alternating tree is a subgraph of a surface graph which is a tree. Its nodes are therefore either real (nodes of G) or shrunk. An alternating tree T is rooted at some node r which may be real or shrunk. The nodes of T are called odd (even) if the number of edges of T in the path from this node to r is odd (even). Similarly an edge of T is called odd (even) if it is the first edge in a path from an odd (even) node to r. Every odd edge of T is not in M and every even edge of T is in M. Every odd node of T must be incident with exactly two nodes of T and every edge in M which is in $\tilde{G}$ and is incident with a node of T must be an edge of T.

Let $\mathcal{H}$ be the collection of node induced (connected) subgraphs which are hypomatchable. For any $j \in E$, let $\mathcal{H}(j) = \{H \in \mathcal{H} : j \in E(H)\}$.

The primal and dual LP's which we use are:

$$\text{primal} \begin{cases} \text{maximize} & c \cdot x \\ \text{subject to} & x(\delta(v)) \leq 1 \quad \forall\, v \in V \\ & x(E(H)) \leq \left\lfloor \frac{V(H)}{2} \right\rfloor \quad \forall\, H \in \mathcal{H} \\ & x \geq 0 \end{cases}$$

$$\text{dual} \begin{cases} \text{minimize} & \Sigma(y_i : i \in V) + \Sigma\left(y_H \left\lfloor \frac{V(H)}{2} \right\rfloor : H \in \mathcal{H}\right) \\ \text{subject to} & y_u + y_v + y(\mathcal{H}(e)) \geq c_e \quad \forall e = (u,v) \in E \\ & y \geq 0. \end{cases}$$

Note that we have restricted the primal constraints a bit more than in (1.1). Due to the way the algorithm works, however, this will be seen to be sufficient.

Let $E(y) = \{e = (u,v) \in E : y_u + y_v + y(\mathcal{H}(e)) = c_e\}$ and let us call the graph induced by $E(y)$ the <u>equality subgraph</u>.

At each stage of the algorithm we have a primal feasible solution x, which is the incidence vector of a matching, and a dual feasible solution y. By complementary slackness our feasible x and y are optimal iff they satisfy the following:

(1.2) $\qquad x_e > 0 \Rightarrow y_u + y_v + y(\mathcal{H}(e)) = c_e \quad \forall\, e = (u,v) \in E$

(1.3) $\qquad y_H > 0 \Rightarrow x(E(H)) = \left\lfloor \frac{|V(H)|}{2} \right\rfloor \quad \forall\, H \in \mathcal{H}$

(1.4) $\qquad y_i > 0 \Rightarrow x(\delta(i)) = 1 \quad \forall\, i \in V.$

At each stage of the algorithm conditions (1.2) and (1.3) are satisfied by x and y, however, (1.4) is, in general, not satisfied. So the algorithm modifies x and y until (1.4) is satisfied.

The Algorithm:

Step 0 [Initialization]: Set $\tilde{G} = G$, $x_j = 0$ for all $j \in E$, $y_i = \frac{1}{2}\max\{0, \max\{c_j : j \in \delta(i)\}\}$ for all $i \in V$, and $y_H = 0$ for all $H \in \mathcal{H}$.

Step 1 [Optimality Check or Node Selection]: If $y_i = 0$ for every node i such that $x(\delta(i)) = 0$, then we terminate with an optimal feasible solution. Otherwise for each node i such that $x(\delta(i)) = 0$ and $y_i > 0$ make surface(i) a root of F. Thus F consists initially of isolated even nodes.

Step 2 [Edge Selection]: Let $E^*(y)$ be the set of edges of $\tilde{G}$ which are in $E(y)$ but not in F. Search $E^*(y)$ for an edge (u,v) where u is an even node of F and v is not an odd node of F. If no such edge exists, go to Step 6 where we make a dual variable change. If such an edge $j = (u,v)$ exists, consider the following four cases:

Case 1: v is not a node of F and $x(\delta(v)) = 1$. Go to Step 3 where we grow the forest F.

Case 2: v is not a node of F and $x(\delta(v)) = 0$. Go to Step 4 where we augment the matching.

Case 3: v is an even node of F and u and v are in different trees of F. Go to Step 4 where we augment the matching.

Case 4: v is an even node of F and u and v are in the same tree of F. Go to Step 5 where we shrink.

Step 3 [Forest Growth]: Let $k = (v,w)$ be the matching edge incident with v. Grow F by adding j and k to F and making v an odd node and w an even node. Go to Step 2.

Step 4 [Augmentation]: Set $x_j = 1$. If $x(\delta(v)) = 0$, then traverse the path from u to its root by alternately lowering and raising by 1 the x values on the edges encountered. Otherwise, traverse the paths from u and v to their respective roots by alternately lowering and raising by 1 the x values on the edges encountered.

After this change, consider each shrunk node s on the traversed path(s). It is incident with exactly one edge $h = (s,t)$ such that $x_h = 1$. Suppose $h = (s',t')$ in G. Find a near-perfect matching in s so that the deficiency now occurs at s' and change x accordingly. "Throw away" F but keep $\tilde{G}$ and go to Step 1.

Step 5 [Shrinking]: j creates a unique odd cycle in the tree which contains u and v. Shrink the nodes of this cycle into an even node. Thus we change $\tilde{G}$. Go to Step 2.

Step 6 [Dual Change]: We now proceed to decrease the value of the dual solution by an amount $\sigma > 0$ as follows:

$y_i \leftarrow y_i - \sigma$ if surface(i) is an even node of F.

$y_i \leftarrow y_i + \sigma$ if surface(i) is an odd node of F.

$y_H \leftarrow y_H - 2\sigma$ if H corresponds to an odd shrunk node of F.

$y_H \leftarrow y_H + 2\sigma$ if H corresponds to an even shrunk node of F.

The magnitude of σ is bounded by the feasibility conditions of the dual LP:

(1.5) $\sigma \leq y_i$ for all nodes i such that surface(i) is an even node of F

(1.6) $2\sigma \leq y_H$ for all $H \in \mathcal{H}$ which when shrunk yield an odd node of F

(1.7) $\sigma \leq y_u + y_v + y(\mathcal{H}(e)) - c_e$ for all $e = (u,v) \in E$ which join an even node of F to a node not in F.

Choose the largest σ which satisfies (1.5)-(1.7). If $\sigma > 0$, then make the dual change. If σ was bounded by (1.7), then a new edge has become available in $E^*(y)$. Go to Step 2. If σ was bounded by (1.5), then a node i which is either a real even node or contained in a shrunk even node of $\tilde{G}$ has $y_i = 0$. Go to Step 7 where we do a "pseudo augmentation". Otherwise, σ was bounded by (1.6) and there is an odd shrunk node i with dual variable $y_H = 0$. Go to Step 8 where we expand this shrunk node.

<u>Step 7</u> [Pseudo Augmentation]: There is a node i such that $y_i = 0$ and v = surface(i) is an even node of F. Let r be the root of the tree of F which contains v. If $v \neq r$, then alternately lower and raise by 1 the values of x_j for the edges j in the (even length) path in F from v to r. If

v is shrunk (this includes the case that $v = r$), then find a new near-perfect matching in v which is deficient at i. Also change the matchings inside any shrunk nodes encountered between v and r so that x is still a matching (as in Step 4). "Throw away" F and go to Step 1.

<u>Step 8</u> [Expanding a Shrunk Node]: i is an odd shrunk node. Let (i',j) be the nonmatching edge of F incident with i such that $\text{surface}(i') = i$. Let (i'',k) be the matching edge of F incident with i such that $\text{surface}(i'') = i$. Let P be the path through i from i' to i'' which an augmentation through i follows. Expand i to get $\tilde{G}'$, set $\tilde{G} = \tilde{G}'$, and make every other node along P odd and even such that both i' and i'' are odd. Go to Step 1.

<u>End of Algorithm</u>

<u>Remarks on the Algorithm</u>: Every time a new forest is set up in Step 1, we have a node i contained in the root of each tree such that $x(\delta(i)) = 0$ and $y_i > 0$. The forest is grown until either we augment and make $x(\delta(i)) = 1$ for some such i, or we execute a dual change making $y_i = 0$ for some such i. The algorithm never increases y_v for a node unless $x(\delta(v)) = 1$ and never decreases $x(\delta(v))$ for a node unless $y_v = 0$. Therefore, after at most $|V|$ forest growings we must have (1.4) satisfied. It has been shown that with careful implementation this algorithm can be made to work in $O(|V|^3)$ (for a survey of complexity results on matching problems see Galil [83]). (Note that to prove the validity of the algorithm it must also be shown that augmentations through shrunk nodes can be made efficiently. This is an easy matter which we will not go into here.)

It is important to focus upon the primal part of the algorithm which always seeks to find a maximum cardinality matching on the equality subgraph. To see this, suppose we have a graph G for which we want a maximum cardinality matching. Hence $c \equiv 1$ and a dual feasible solution is $y_i = \frac{1}{2}$ for all $i \in V$ and $y_H = 0$ for all $H \in \mathcal{H}$. In fact, this dual solution sets $E(y) = E$. Suppose we now apply the algorithm. It grows a forest until at some point a dual change is required. After one dual change the algorithm terminates. Using this algorithm for the cardinality problem yields the following max-min theorem known as the Tutte-Berge Theorem (see Tutte [47] and Berge [58]).

<u>Theorem 1.8</u>: For any graph $G = (V,E)$, the cardinality of a maximum matching in G is equal to

$$\min_{V' \subseteq V} \frac{|V| + |V'| - 0(G[V \setminus V'])}{2}$$

where $0(G[V\setminus V'])$ denotes the number of components of $G[V\setminus V']$ which are hypomatchable.

<u>Proof</u>: Let $\nu(G)$ be the cardinality at a maximum matching in G. First we show that

$$\nu(G) \leq \min_{V' \subseteq V} \frac{|V| + |V'| - 0(G[V \setminus V'])}{2} . \tag{1.8}$$

For any $V' \subseteq V$, the number of deficient nodes in a maximum cardinality matching is bounded below by $0(G[V\setminus V']) - |V'|$

since all of the nodes of a component of $0(G[V\backslash V'])$ cannot be matched among themselves. Hence the number of nodes which are saturated in a maximum cardinality matching must be bounded above by

$$|V| - (0(G[V\backslash V']) - |V'|) = |V| + |V'| - 0(G[V\backslash V'])$$

which implies (1.8).

To show that equality holds in (1.8), consider the alternating forest after applying the cardinality algorithm to G as described above. If we let V' be the set of odd nodes, then we see that the hypomatchable components of $G[V\backslash V']$ are precisely the even nodes. Since each even node which is not a root is matched to an odd node, we get that the number of deficiencies equals $0(G[V\backslash V']) - |V'|$. Hence we get equality in (1.8).

There is also a max-min theorem for the weighted case as there was for the weighted bipartite case. For this see Schrijver and Seymour [77], Cunningham and Marsh [79] and Schrijver [83].

A special case of this theorem is Tutte's [47] well-known characterization of graphs which have a perfect matching.

<u>Theorem 1.9</u>: A graph G has a perfect matching iff for all $V' \subseteq V$, $0(G[V\backslash V']) \leq |V'|$ where $0(G[V\backslash V'])$ denotes the number of components of $G[V\backslash V']$ which have an odd number of nodes.

Another special case is the max-min theorem for the cardinality bipartite matching problem which we looked at in the last section.

Section 6. Generalizations of Matchings

In this section we look at several ways in which matchings can be generalized. In particular, we see how some of these generalizations are related to the travelling salesman problem and yield problems which are "near the border" between polynomial and NP-complete problems.

Given a graph $G = (V,E)$ and a vector $b \in \mathbb{Z}_+^{|V|}$, a <u>b-matching</u> is a vector $x \in \mathbb{Z}_+^{|E|}$ such that $x(\delta(v)) \leq b(v)$ for all $v \in V$. If $w \in \mathbb{R}^{|E|}$, then the weight of a b-matching x is wx. A b-matching which satisfies $x(\delta(v)) = b(v)$ for every $v \in V$ is called <u>perfect</u>. A 1-matching, that is $b \equiv 1$, is just an ordinary matching.

Edmonds and Pulleyblank (see Pulleyblank [73],[80]) gave a polynomial algorithm, polyhedral characterization, and max-min theorems for the maximum weight b-matching problem as well as the maximum weight perfect b-matching problem (see also Cook [83]). If we add to the b-matching problem a capacity vector $c \in \mathbb{Z}_+^{|E|}$, then a <u>capacitated b-matching</u> x is a b-matching such that $x_e \leq c_e$ for all $e \in E$. A polynomial algorithm, polyhedral characterization, and max-min theorems for the weighted problem can all be obtained by reducing this problem to the uncapacitated case. (See Belck [50], Tutte [52],[54],[81], Edmonds and Johnson [73], Marsh [79], Schijver [83] and Araoz, Cunningham, Edmonds and Green-Krotki [82].)

Let us now consider how these matching problems can be related to the travelling salesman problem. The 0 - 1 incidence vectors of Hamilton tours of a graph $G = (V,E)$ may be described as the solutions to the system

(1.9) $0 \leq x_j \leq 1$ for all $j \in E$

(1.10) $x(\delta(v)) = 2$ for all $i \in V$

(1.11) $x(\delta(S)) \leq |S| - 1$ for all $S \subset V$, $|S| \geq 3$

(1.12) x integral.

The constraints (1.10) are called <u>degree constraints</u> and the constraints (1.11) are called <u>subtour elimination constraints</u>. This formulation, one of the earliest for the travelling salesman problem, was first given by Dantzig, Fulkerson and Johnson [54].

Let us consider the following relaxation of this system.

(1.9)' $0 \leq x_j$ for all $j \in E$

(1.10) $x(\delta(i)) = 2$ for all $i \in V$

(1.12) x integral.

The solutions to this system are incidence vectors of perfect 2-matchings. As we have seen, a polynomial algorithm for finding a maximum weight perfect 2-matching is known. We now show how this algorithm can be used to find an approximation to the problem of finding a maximum weight Hamilton tour of a complete graph with edge weights $w \in \mathbb{R}_+^{|E|}$. This idea is due to Fisher, Nemhauser, and Wolsey [79].

Algorithm

Input: A complete graph $G = (V,E)$ with edge weight vector $w \in \mathbb{R}_+^{|E|}$

Step 1: Find a maximum weight perfect 2-matching of G. (Treat edges at value 2 as length 2 cycles.)

Step 2: Remove the smallest edge from each cycle.

Step 3: Connect the resulting paths into a Hamilton tour.

Output: The weight of the tour.

Let us now examine how good this approximation is. A Hamilton tour is a perfect 2-matching, so the maximum weight of a perfect 2-matching is greater than or equal to the weight of a maximum weight Hamilton tour. The worst case is that in each cycle of the perfect 2-matching all edges have the same weight and all cycles are of length 2. Thus we get that

$$\text{Approximate value} \geq \frac{1}{2} \text{ Optimum value.}$$

This bound can be improved if we can insure that the smallest cycles in our perfect 2-matching have length greater than 2. With this in mind, we define a hierarchy of problems (P_k) for $k \geq 1$ where (P_k) is the problem of finding a maximum weight perfect 2-matching with all cycles of size greater than k. Performing an analysis as above yields the following:

For (P_1): Approximate value $\geq \frac{1}{2}$ Optimum value

For (P_2): Approximate value $\geq \frac{2}{3}$ Optimum value

For (P_3): Approximate value $\geq \frac{3}{4}$ Optimum value

$\vdots$

For (P_k): Approximate value $\geq \frac{k}{k+1}$ Optimum value.

The incidence vectors of the feasible solutions for (P_2) may be described as the solutions to the following relaxation of the system for the travelling salesman problem:

(1.9) $\quad 0 \leq x_j \leq 1 \quad$ for all $j \in E$

(1.10) $\quad x(\delta(i)) = 2 \quad$ for all $i \in V$

(1.11) $\quad x$ integral.

A system describing the incidence vectors of the feasible solutions for (P_i) $i \geq 3$ may be obtained from the system for (P_2) by adding the appropriate subtour elimination constraints (i.e. $x(\gamma(S)) \leq |S| - 1$ for $S \subset V$, $3 \leq |S| \leq i$). Thus these problems form a hierarchy of relaxations of the travelling salesman problem which are progressively closer to the travelling salesman problem itself.

Let us now consider the difficulty of these problems. The problem (P_5) with weights of 1 on the edges was shown to be NP-complete by Papadimitriou (see Cornuejols and Pulleyblank [80]). Hence all the problems (P_k) for $k \geq 5$, both cardinality and weighted versions, are NP-hard. The problem (P_4) was shown to be NP-hard by Vornberger [79], however, the status of (P_4) with weights of 1 is not known.

Let us now consider problems (P_1), (P_2), and (P_3). In fact, let us discuss them in a slightly more general framework where the degree constraints in the systems defining their incidence vectors are replaced by $x(\delta(i)) \leq 2$ for all $i \in V$. (These problems are theoretically nicer to work with because the associated polytopes are full dimensional and hence have uniquely defined facets.) Let us call these problems (P_i') for $i = 1,2,3$.

The incidence vectors of (P_i') have values 0, 1, and 2. The 2's correspond to edges and the 1's correspond to paths and cycles all of which are node disjoint. Actually, since any path and even length cycle can be written $.5x_1 + .5x_2$ where x_1 and x_2 are matchings with 2's on their edges, the convex hull of incidence vectors satisfying the system for (P_1') is just the convex hull of incidence vectors of 2-matchings with edges at value 2 and odd cycles with edges at value 1 which are node disjoint. The problem (P_1') has a polynomial algorithm which is just a simplified version of Edmonds' algorithm for matchings given in the last section. (In fact, the system for (P_1') is totally unimodular so it can be solved polynomially as an LP.)

The incidence vectors satisfying the system for (P_2') correspond to node disjoint paths and cycles of length ≥ 3 all of whose edges are at value 1; that is, simple 2-matchings.

This problem also has a polynomial algorithm due to Edmonds, Johnson, and Lockhart [68]. We give an algorithm for this problem in Chapter 2 (although we do not use bidirected edges as they do). It, too, is modeled after the 1-matching algorithm and yields a polyhedral characterization and max-min theorems. As in the 1-matching case, this polyhedral characterization is non-trivial in that it requires a new collection of inequalities.

The incidence vectors satisfying the system for (P_3') correspond to node disjoint paths and cycles of length > 3 all of whose edges are at value 1; that is, *triangle-free simple 2-matchings*. In Chapter 3, we present an Edmonds' style polynomial algorithm for the cardinality case $w \equiv 1$. We show that an Edmonds' style algorithm for the weighted problem, which uses the cardinality algorithm to make primal improvements, will this time be much more complicated. ~~We give a polyhedral characterization sufficient to solve the cardinality problem as an LP and~~ We conjecture what ~~the entire~~ a complete polyhedral characterization looks like. We also conjecture that the weighted problem is in P.

We now discuss a second hierarchy of relaxations of the travelling salesman problem (see Cornuejols and Pulleyblank [80],[83]). Let us begin by defining problem $(Q_1) = (P_1)$; that is, (Q_1) is the problem of finding a maximum weight perfect 2-matching. Suppose we no longer consider an edge at value 2 as a length 2 cycle as we did for the (P_k) hierarchy. Then we may define (Q_k) as the problem of finding a maximum weight perfect 2-matching with all cycles of size $> k$. (So we may now use edges at value 2 for all k.) Hence $(Q_2) = (Q_1)$. (Q_3) is the problem of finding a triangle-free maximum weight perfect 2-matching. Note that $(Q_4) = (Q_3)$ and in general $(Q_{2k}) = (Q_{2k-1})$ for $n \geq 1$ since the value of every even cycle can be equaled or exceeded by a perfect matching of the cycle with edges at value 2. Thus, in what follows, we will refer only to problems (Q_k) for k odd.

Let us consider the set of incidence vectors of solutions to (Q_k). Since $(Q_1) = (P_1)$, we know that the vertices of the convex hull of incidence vectors satisfying the following system are the feasible solutions to (Q_1).

(1.9)' $\quad 0 \leq x_j \quad$ for all $j \in E$

(1.10) $\quad x(\delta(i)) = 2 \quad$ for all $i \in V$

(1.12) $\quad x$ integral.

Similarly, the incidence vectors of the feasible solutions to (Q_k), for $k \geq 3$ and odd, are the vertices of the convex hull of incidence vectors satisfying appropriate systems. These systems may be obtained from the system for (Q_1) by adding the appropriate subtour elimination constraints (that is, $x(\gamma(S)) \leq |S| - 1$ for $S \subset V$, $3 \leq |S| \leq k$, and $|S|$ odd). Thus these problems also form a hierarchy of relaxations of the travelling salesman problem.

We next consider another way in which these problems are related to the travelling salesman problem. Suppose a graph

$G = (V,E)$ has an odd number of nodes. Then all solutions to the problem (Q_k) for $k \geq \frac{|V| + 1}{2}$ with weight vector $w \equiv 1$ are Hamilton tours of G. So, needless to say, this variant of the problem, where k is a function of $|V|$, is NP-complete. This idea is introduced and developed by Cornuejols and Pulleyblank [82].

Let us discuss what is known about the complexity of this hierarchy of problems. As with the hierarchy (P_k), we consider slightly more general problem (Q_k') where the degree constraints in the system defining a problem's incidence vectors are replaced by $x(\delta(i)) \leq 2$ for all $i \in V$. (This again allows us to work with full dimensional polytopes.) The incidence vectors of (Q_k') have values 0,1,2 where the 2's correspond to edges and the 1's correspond to paths and cycles of length $> k$ all of which are node disjoint. As we have seen, however, the paths and even length cycles may be expressed as convex combinations of matchings with edges at value 2 so that the convex hull of incidence vectors satisfying our altered system has vertices which are 0,1,2 incidence vectors where the 2's correspond to edges and the 1's correspond to odd cycles of size $> k$. Hence the only difference between the solutions to (Q_k) and (Q_k') is that the solutions to (Q_k) cover all the nodes and the solutions to (Q_k') need not cover all the nodes.

Cornuejols and Pulleyblank [83] showed by constructing an Edmonds' style algorithm that when $w \equiv 1$ and k is fixed, all the (Q_k') are in P. They also showed [80] that the weighted problem (Q_3') is in P. (For an application of (Q_3') in biophysics see Havel [82].) We conjecture that the problem (Q_5') is also in P although it, like (P_3'), will require an algorithm which is significantly more complicated than the associated cardinality algorithm. We discuss this in Chapter 5. The status of problems (Q_k') for $k \geq 7$ is unknown.

One final generalization of matchings may be obtained from the cardinality problems for (Q_i') $i \geq 1$. In the problems (Q_i') we optimize over the 0,1,2 incidence vectors which correspond to node disjoint edges and odd cycles where isolated edges are at value 2 and edges in odd cycles are at value 1. Any such incidence vector x has value $1 \cdot x$ which is just the number of nodes which are incident with a non-zero edge in x. Problems of this sort may be generalized as follows.

Given a graph $G = (V,E)$ and a family F of subsets of V, an F-packing is a subfamily $J \subseteq F$ such that every node of G belongs to at most one member of J. Let us say a subset of nodes is <u>hypomatchable</u> if it induces a hypomatchable subgraph of G. Then, when H denotes a family of hypomatchable node sets and $F = E \cup H$, F-packings are called <u>hypomatchings</u>. Given a hypomatching J, any node which belongs to one member of J is said to be <u>covered</u> by J. A <u>maximum hypomatching</u> is one which covers the maximum number of nodes of G. (See Cornuejols, Hartvigsen, and Pulleyblank [82] and Cornuejols and Hartvigsen [83].)

So the maximum cardinality matching problem is a maximum hypomatching problem where $H = \emptyset$ as is (Q_1') where

H = node sets of odd cycles of size > i. Another variant of this is dynamic matchings which was introduced by Orlin [82]. In this case, weights are associated with the edges of an auxiliary graph G and $S \in H$ if and only if S is the node set of an odd cycle of G for which the sum of the edge weights is zero.

Another example of hypomatchings is that of clique-packing where H is a family of cliques of size ≥ 3 (actually this is equivalent to having H consist of odd cliques since even cliques can be perfectly matched). This problem has an interpretation known as "the research director problem" due to Laurence Wolsey. Suppose a researcher wishes to divide his researchers into groups of mutually compatible members so that the number of researchers working alone is minimized. This is the same as looking for a clique packing of a graph which contains a node for each researcher and an edge joining two nodes if the corresponding researchers are compatible.

It is not difficult to see that this last problem is equivalent to the problem of packing edges and triangles. Clearly any packing of edges and triangles is a clique-packing and any packing of edges and cliques may be transformed into a packing of edges and triangles by perfectly packing any even clique with edges and packing any odd clique with edges and one triangle.

The hypomatching problem, in general, and the edge and triangle packing problems, in particular, have been looked at independently by Hell and Kirkpatrick [84]. They have found a polynomial algorithm and max-min theorems for these problems as well as explored the complexity of some related packing problems (see Hell and Kirkpatrick [78],[81] and Kirkpatrick and Hell [83]). Work of a similar nature has been done on "F-factors" by Mühlbacher [79].

In Chapter 4 we see how several theorems on cardinality 1-matchings and 2-matchings generalize to hypomatchings. Also, after showing how hypomatchings generate a matroid analogous to the matching matroid, we give a greedy type algorithm which solves a more general maximum hypomatching problem in which arbitrary weights are assigned to the nodes. Finally, we give a polynomial algorithm for solving some versions of the maximum cardinality hypomatching problem and follow this with a max-min theorem and a polyhedral result.

Section 7. Summary of Results

Looking at the collection of matching problems discussed in the last section, we see that each matching problem has two subproblems: a cardinality problem and a weighted problem. Each subproblem may have several characteristics:

(1) a polynomial algorithm

(2) a polyhedral characterization sufficient to solve the problem as an LP

(3) a max-min theorem.

Any subproblem of a matching problem we have discussed either has all three characteristics or none of them. If a weighted subproblem has all three, then, of course, so does the associated cardinality subproblem. However, the converse is not true as the (P_k) hierarchy, for example, shows.

Let us now consider problems for which both subproblems have characteristics (1)-(3) and let us compare these characteristics between subproblems. We have seen that for all capacitated b-matching problems the polynomial algorithm and polyhedral characterization for both subproblems are essentially the same. (Although a dual change is used in the weighted algorithms, it does not increase the difficulty of the problem significantly.) This reflects the fact that information equivalent to a complete polyhedral characterization is necessary to solve the cardinality problems. In Chapter 2 we show, by giving an algorithm, due to Edmonds [65] and Edmonds, Johnson and Lockhart [68], that this relationship between subproblems holds for the simple 2-matching problem. In particular, we give a polynomial algorithm and from this derive a polyhedral characterization and max-min theorem for both subproblems.

This sets the stage for Chapter 3 in which we look at the triangle-free simple 2-matching problem. For the cardinality subproblem, we give a polynomial algorithm (styled after the cardinality simple 2-matching algorithm), which implies a polyhedral characterization, and max-min theorem. In particular, we characterize those graphs which have a perfect triangle-free simple 2-matching. We then show that a polynomial algorithm and polyhedral characterization for the weighted problem (if they exist as we conjecture they do) must significantly differ from those for the cardinality subproblem. The reason for this difference hinges on the fact that for the cardinality subproblem (and, in fact, for all capacitated b-matching problems) only 0 - 1 inequalities are needed whereas 0 - 1 - 2 inequalities must be used in the weighted subproblem. This means that a complete polyhedral characterization is not needed to solve the cardinality subproblem. We describe some of the 0 - 1 - 2 inequalities and look at some problems involved in finding a polynomial algorithm. In particular we show how the dual change does increase the difficulty of the problem significantly. This problem is also of interest because of its relation to the travelling salesman problem, as described in the preceding section.

In Chapter 5 we are again faced with the same situation. Here we look at the problem (Q_5') and see that although the cardinality subproblem has characteristics (1)-(3), a polynomial algorithm and polyhedral characterization for the weighted problem must differ significantly from these. Again the difference hinges on the appearance of 0 - 1 - 2 inequalities and a more complicated dual change. We describe some of these 0 - 1 - 2 inequalities and look at problems involved in the dual change.

In Chapter 4 we deal with a slightly different generalization of matching theory -- the problem of packing edges and hypomatchable subgraphs in a graph. We generalize a number of fundamental results in matching theorey to this context and give a (sometimes) polynomial algorithm. A special case of this problem is the problem of packing edges and triangles. Packing just edges is in P while packing just triangles is NP-complete since it includes the 3-dimensional assignment problem as a special case (see Garey and Johnson [79]). We show that packing edges and triangles is in P. We end this section with a max-min theorem and a polyhedral result for the edge and triangle problem and the cardinality versions of (Q_k').

Results about matching problems can be proved in two basically different ways. One way is to derive the polyhedral characterization and max-min theorems from the polynomial algorithm. This is what we do in Chapters 2 and 3. However, one may also take a more pure graph theoretic approach to proving these sorts of matching results. This is what we do in much of Chapter 4.

Chapter 2

THE MAXIMUM WEIGHT SIMPLE 2-MATCHING PROBLEM

Section 1. Introduction, Blossoms, and the Alternating Structure

In this chapter we give a polynomial algorithm for the maximum weight simple 2-matching problem. This problem was solved by Edmonds [65] and Edmonds, Johnson and Lockhart [68]. (In contrast to their treatment we do not use bidirected edges.) From the algorithm we derive a polyhedral characterization and a Tutte-type max-min theorem just as was done in the 1-matching case.

Before we describe the algorithm we must define critical graphs, blossoms, and alternating forests.

Let G be a graph, M a simple 2-matching and x its incidence vector. Let H be any subgraph of G. For $v \in V(H)$ we define $b_v^H = \min\{2, |\delta_H(v)|\}$, where $\delta_H(v)$ is the set of edges of H incident with v. We say that M <u>saturates</u> H if

$$x(E(H)) = \lfloor b^H(V(H))/2 \rfloor,$$

where $\lfloor a \rfloor$ denotes the integer part of a. If $b^H(V(H))$ is even then $x(\delta_H(v)) = b_v^H$ for every $v \in V(H)$, i.e. every node of H is saturated. If $b^H(V(H))$ is odd, then there exists a unique $i \in V(H)$ such that $x(\delta_H(i)) = b_i^H - 1$. For all the other nodes $v \in V(H)$, $v \neq i$, $x(\delta_H(v)) = b_v^H$. We say that the simple 2-matching M is <u>deficient</u> at node i in H. If, for every node $u \in V(H)$, there exists a saturating simple 2-matching deficient at u, then the graph H is said to be <u>simple</u> 2-<u>matching</u> <u>critical</u>, or just <u>critical</u>.

A blossom B is a pair (S,T) where $S \subseteq V$ and T is an odd cardinality subset of $\delta(S)$ such that no two edges of T are adjacent. (See Figure 2.1.) The set S will be called the center of the blossom, the edges of T its petals, and the nodes of $V \backslash S$ incident with a petal the tips of the blossom. We define the edge set of the blossom as $E(B) = \gamma(S) \cup T$.

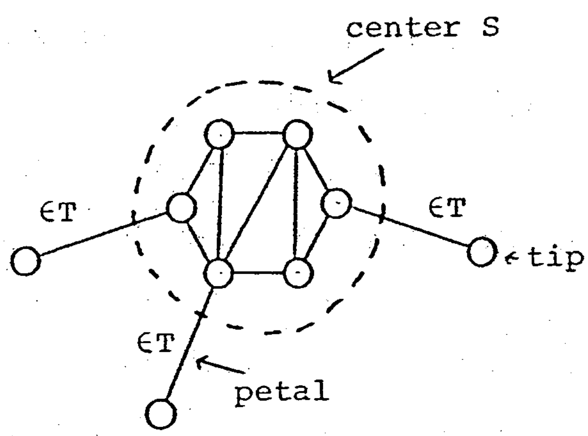

Figure 2.1. A Blossom

Let x be the incidence vector of a simple 2-matching M, and let $B = (S,T)$ be a blossom. Then, for every $v \in S$, $x(\delta(v)) \leq 2$. Moreover, for every $j \in T$, $x_j \leq 1$. Adding these inequalities gives $2x(E(B)) \leq 2|S| + |T|$. Therefore $x(E(B)) \leq |S| + |T|/2$. Since x is integer and $|T|$ is odd, it follows therefore that

$$(2.1) \qquad x(E(B)) \leq |S| + (|T| - 1)/2.$$

When equality holds in (2.1), the simple 2-matching M <u>saturates</u> the blossom (S,T). A blossom is <u>critical</u> if the graph induced by $E(B)$ is simple 2-matching critical. It will be shown later that every simple 2-matching critical graph is a blossom.

In the algorithm we make use of alternating forests which play a similar role to those constructed in the 1-matching algorithm. An alternating forest is defined with respect to a simple 2-matching M as follows (see Figure 2.2).

A surface graph $\tilde{G}$ is obtained from G by shrinking a collection of pairwise disjoint subsets $S_1,\ldots,S_k$ of nodes of G. Each set S_i is the center of a critical blossom which is saturated by M. In fact, each S_i may be shrunk sequentially in the course of the algorithm so that $G \times S_i = (G \times S_i^1) \times S_i^2 \times\ldots\times S_i^j$ where each set $S_i^1 \cup S_i^2 \cup\ldots\cup S_i^m$ for $1 \leq m \leq j$ is the center of a critical blossom which is saturated by M. (Note that the order of shrinking is important since the sets S_i^m for $1 \leq m \leq j$ need not be disjoint.) <u>Expanding the shrunk node</u> i which corresponds to S_i yields the graph

$$\tilde{G}' = G \times S_1 \times S_2 \times\ldots\times S_{i-1} \times (S_i^1 \times\ldots\times S_i^{j-1}) \times S_{i+1} \times\ldots\times S_k.$$

If a node u of G is contained in a shrunk node v of $\tilde{G}$ then we say surface $(u) = v$.

The <u>alternating forest</u> $\mathcal{S}$ is a subgraph of $\tilde{G}$ which is a forest. Thus the alternating forest has two types of nodes, <u>real</u> nodes, which are simply nodes of G, and <u>shrunk</u> nodes. Each node of the alternating forest is also called either <u>even</u>

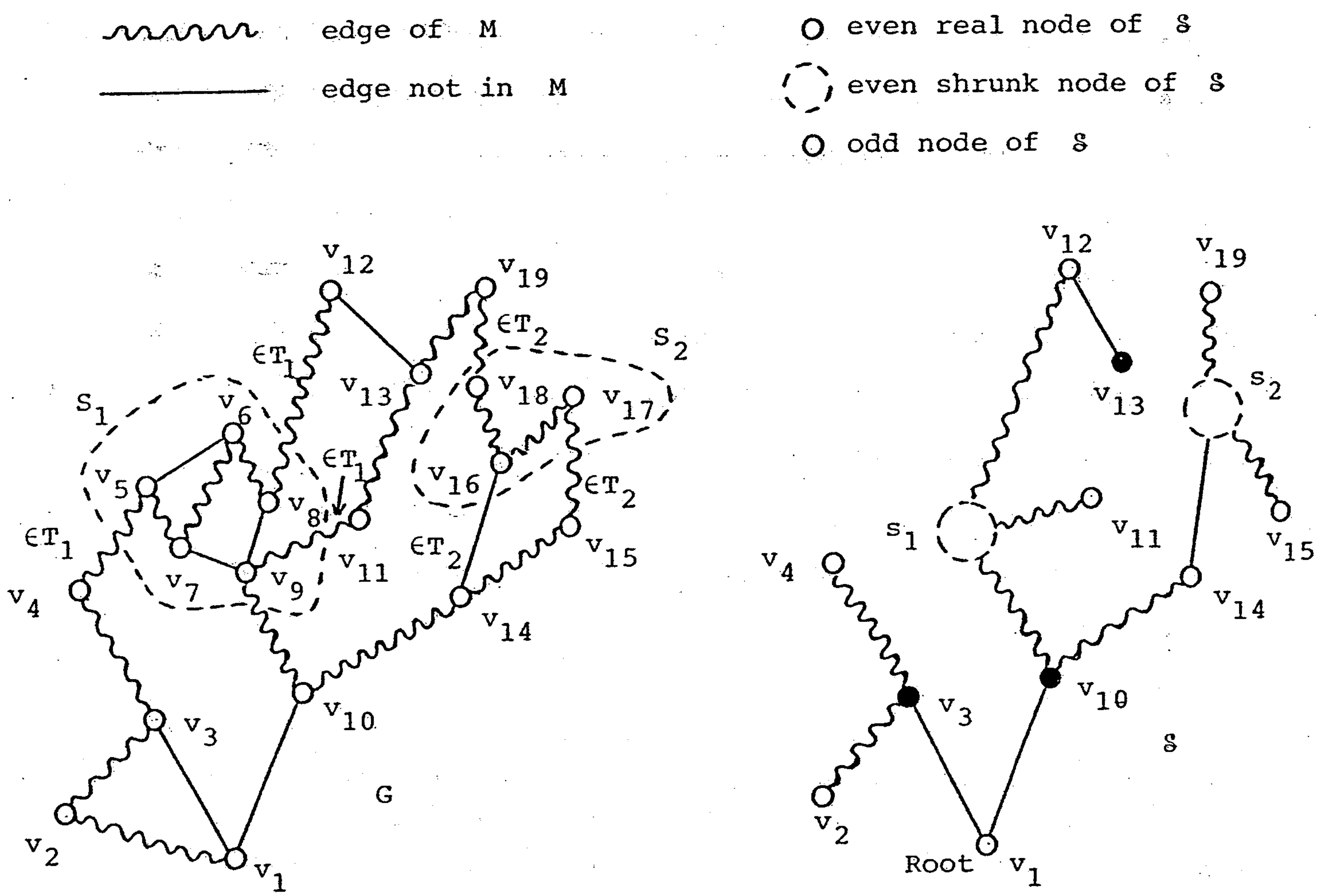

Figure 2.2. A Graph G, a Simple 2-Matching M and an Alternating Forest S

or odd. Consider the subgraph of $\mathcal{S}$ induced by the even shrunk nodes x of $\mathcal{S}$. We call the components of this subgraph even blossom clusters. We will refer to the components of the subgraph of $\mathcal{S}$ induced by the odd nodes as odd blossom clusters.

A node of $\tilde{G}$ is deficient if either it is a deficient real node (i.e. $x(\delta(v)) < 2$) or else it is a shrunk node obtained by shrinking a set S which contains a deficient node of G. A blossom cluster is said to be deficient if it contains a deficient node. Every deficient real node and blossom cluster of $\tilde{G}$ is called a root of the alternating forest.

The edges of the alternating forest are edges of $\tilde{G}$ (and thus, by extension, also edges of G). If an edge j of $\mathcal{S}$ joins two nodes v and w and if, in $\tilde{G}$, an edge of M joins v to w, then $j \in M$.

Every real node and blossom cluster v of the alternating forest, other than the roots, has a unique predecessor node $p(v)$ in the forest such that an edge of the forest joins v to $p(v)$ and, if we follow the path of predecessors from any real node or blossom cluster v, we reach a root. Clearly this path $(v,p(v),p(p(v)),\ldots,r)$ is unique. Every edge of the forest is of the form $vp(v)$.

The roots are even real nodes and/or even blossom clusters. The odd real nodes are exactly the real nodes v of the forest, which are not roots, such that the edge $v(p(v)) \notin M$. For any real node or blossom cluster v, the edges in the path $(v,p(v),\ldots,r)$, which are not blossom petals, are alternately in M and out of M, and the number of these edges is odd if v is odd and is even if v is even. Every edge of $\mathcal{S}$ which is not

Primal

$$\begin{cases} \text{maximize} & w \cdot x & \\ \text{subject to} & x(\delta(v)) \le 2 & \forall\ v \in V \\ & x(\gamma(S)) + x(T) \le s_B & \forall\ B \in \mathcal{B} \\ & x_e \le 1 & \forall\ e \in E \\ & x_e \ge 0 & \forall\ e \in E \end{cases}$$

Dual

$$\begin{cases} \text{minimize} & 2 \sum\limits_{v \in V} y_v + \sum\limits_{B \in \mathcal{B}} s_B \pi_B + \sum\limits_{e \in E} z_e & \\ \text{subject to} & y_u + y_v + \sum\limits_{B \in \mathcal{B}(e)} \pi_B + z_e \ge c_e & \\ & & \forall\ e = (u,v) \in E \\ & y_v \ge 0 & \forall\ v \in V \\ & \pi_B \ge 0 & \forall\ B \in \mathcal{B} \\ & z_e \ge 0 & \forall\ e \in E. \end{cases}$$

Let $E(y,\pi,z) = \{e = (u,v) \in E : y_u + y_v + \sum\limits_{B \in \mathcal{B}(e)} \pi_B + z_e = c_e\}$ and let us call the graph induced by $E(y,\pi,z)$ the <u>equality subgraph</u>.

At each stage of the algorithm we have a primal feasible solution x, which is the incidence vector of a simple 2-matching, and a dual feasible solution y. By complementary slackness, our feasible x and y are optimal iff they satisfy the following:

(2.2) $\quad x_e > 0 \Rightarrow y_u + y_v + \sum\limits_{B \in \mathcal{B}} \pi_B + z_e = c_e \qquad \forall\ e = (u,v) \in E$

(2.3) $\quad \pi_B > 0 \Rightarrow x(\gamma(S)) + x(T) = s_B \qquad \forall\ B = (S,T) \in \mathcal{B}$

(2.4) $\quad z_e > 0 \Rightarrow x_e = 1 \qquad \forall\ e \in E$

a blossom petal has one odd endnode and one even endnode.

Every odd real node of the alternating forest is saturated by M.

If u is a blossom cluster of the alternating forest and u is not a root, then the nodes of G belonging to u are all saturated by M, and they constitute the set S of a blossom $B = (S,T)$. The simple 2-matching M saturates the blossom, i.e. all but one of the nodes of B are saturated in B. Two cases occur depending on whether the edge $j = up(u)$ is an edge of M or not. The edge j is called the base edge of the blossom cluster. If $j \in M$, then the edge set T of the blossom B is defined as $T = (M \cap \delta(S)) \setminus \{j\}$. The end node of j which is in S is the unique deficient node in B. On the other hand, if $j \notin M$, then T is defined as $T = (M \cap \delta(S)) \cup \{j\}$ and the tip of j is the unique deficient node in B. (Note the distinction between deficiency in G and deficiency in B. The simple 2-matching M is deficient in B at node v if its incidence vector x satisfies $x(\delta_B(v)) < b_v^B$ where $b_v^B = 2$ if $v \in S$ and $b_v^B = 1$ if v is a tip of the blossom.) Since B will be a critical blossom, a saturating matching which leaves any given node of B deficient in B will always exist.

Let $\mathcal{B}$ be the collection of blossoms $B = (S,T)$ of G. For any $j \in E$, let $\mathcal{B}(j) = \{B \in \mathcal{B} : j \in E(B)\}$ and for every $B \in \mathcal{B}$ let $s_B = |S| + (|T| - 1)/2$.

The primal and dual LP's which we use are as follows.

(2.5) $y_i > 0 \Rightarrow x(\delta(i)) = 2$ $\forall\ i \in V.$

At each stage of the algorithm conditions (2.2)-(2.4) are satisfied by x and y, however, (2.5) is, in general, not satisfied. So the algorithm modifies x and y until (2.5) is satisfied.

Section 2. The Algorithm

Step 0 [Initialization]: Let M be any simple 2-matching and let x be its incidence vector. Let y be a dual feasible solution such that x and y satisfy complementary slackness conditions (2.2)-(2.4). (For example, set $x_j = 0$ for $j \in E$, $y_i = \frac{1}{2} \max\{0, \max\{c_j : j \in \delta(i)\}\}$ for all $i \in V$, $\pi_B = 0$ for all $B \in \mathcal{B}$, and $z_e = 0$ for all $e \in E$.) Go to Step 1.

Step 1 [Optimality Check and Node Selection]: If $y_i = 0$ for every node i such that $x(\delta(i)) = 0$, then we terminate with optimal feasible solutions. Otherwise, identify the blossom clusters, and for each node i such that $x(\delta(i)) < 2$ and $y_i > 0$, make i a root if it is real in $\tilde{G}$ or make the blossom cluster containing i in $\tilde{G}$ a root if surface (i) is shrunk in $\tilde{G}$. Go to Step 2.

Step 2 [Edge Selection]: Let $E^*(y,\pi,z)$ be the set of edges of $\tilde{G}$ which are in $E(y,\pi,z)$ but not in M or $\mathcal{S}$. Search $E^*(y,\pi,z)$ for an edge vw where v is an even node of $\mathcal{S}$ and w is not an odd node of $\mathcal{S}$. If no such edge exists, go to Step 6 where we make a dual variable change. If such an edge $j = vw$ exists, consider the following three cases:

Case 1: w is not a node of $\mathcal{S}$ and $x(\delta(w)) = 2$. Go Step 3 where we grow the forest.

Case 2: w is not a node of $\mathcal{S}$ and $x(\delta(w)) < 2$. Let $P = (w, v, p(v), p(p(v)), \ldots, r)$ and go to Step 4 where we augment the matching.

Case 3: w is an even node of $\mathcal{S}$. Go to Step 5 where we augment or shrink.

Step 3 [Forest Growth]:

Case 1: w is shrunk. Let C be the blossom cluster which contains w.

Case 1.a: vw is the base of C. (See Figure 2.3.) Make every shrunk node of C an even node of $\mathcal{S}$, define $p(C) = v$, and add vw to $\mathcal{S}$. Consider each petal $p \neq vw$ of C and its tip t. If $z_p > 0$, do nothing. If $z_p = 0$ and t is not in $\mathcal{S}$, then make t an even node at $\mathcal{S}$, define $p(t) = C$, and add p to $\mathcal{S}$. (By design of the algorithm, every petal tip t of C for which $z_p = 0$ is real and either even or not in $\mathcal{S}$. If t were odd, then C and hence w would have been in $\mathcal{S}$.) Go to Step 2.

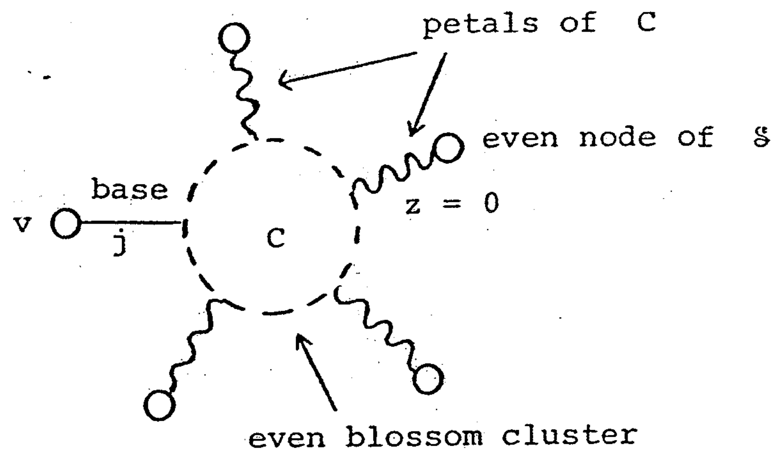

Figure 2.3

<u>Case 1.b</u>: vw is not the base of C. Say $b = cy$ is the base of C where $c \in C$. Make every shrunk node of C an odd node of $\mathcal{S}$, define $p(C) = v$, and add vw to $\mathcal{S}$.

Suppose $cy \in M$, $z_b = 0$, and y is not in $\mathcal{S}$. If y is shrunk, then go to Step 3 Case 1.a taking v and w to be c and y, respectively. If y is real (see Figure 2.4), then make y an even node of $\mathcal{S}$, define $p(y) = C$, and add cy to $\mathcal{S}$. (As above, if $z_b = 0$, then y must be real and either even or not in $\mathcal{S}$.) If $cy \in M$ and $z_b > 0$, do nothing. Go to Step 2.

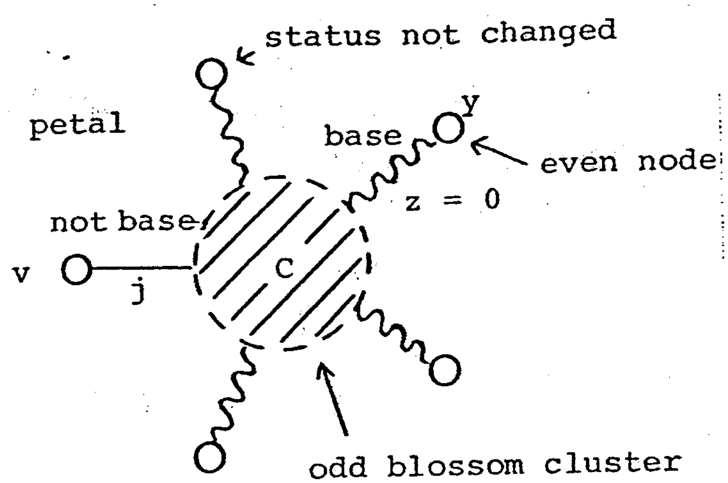

Figure 2.4

Suppose $cy \notin M$. If y is odd, then add cy to S and go to Step 2. (By design of the algorithm, y must be real.) Otherwise (see Figure 2.5), go to Step 2 taking v and w to be c and y, respectively.

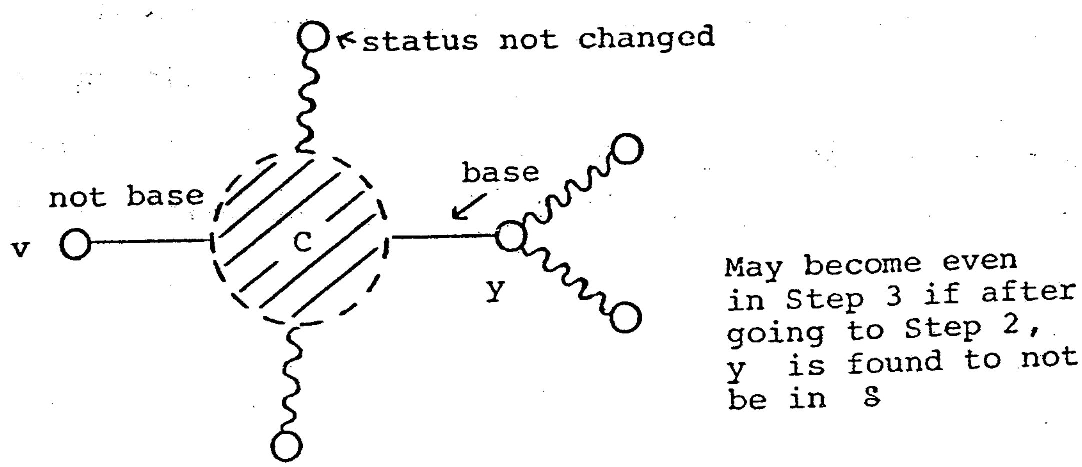

Figure 2.5

<u>Case 2</u>: w is real. Make w an odd node of $\mathcal{S}$ and add vw to $\mathcal{S}$. If v is an odd shrunk node, define $p(w) = C$ where C is the cluster containing v. Otherwise, define $p(w) = v$. Let wu_1 and wu_2 be the two edges of M incident with w. If u_i is in $\mathcal{S}$ or if $z_e > 0$ for $e = wu_i$ we do nothing to it. If u_i is not in $\mathcal{S}$ and $z_e = 0$, consider the following cases for $i = 1,2$ then go to Step 2.

<u>Case 2.a</u>: u_i is real. (See Figure 2.6.) Make u_i an even node of $\mathcal{S}$, define $p(u_i) = w$, and add the edge wu_i to $\mathcal{S}$.

<u>Case 2.b</u>: u_i is shrunk. Let C be the blossom cluster which contains u_i.

<u>Case 2.b.1</u>: wu_i is the base of C. (See Figure 2.6.) Make every shrunk node of C an even node of $\mathcal{S}$, define $p(C) = w$, and add wu_i to $\mathcal{S}$. Consider each petal p of C and its tip t. If $z_p > 0$, do nothing. If $z_p = 0$ and t is not in $\mathcal{S}$, then make t an even node of $\mathcal{S}$, define $p(t) = C$, and add p to $\mathcal{S}$. (As in Case 1.a, every petal tip t of C for which $z_p = 0$ is real and either even or not in $\mathcal{S}$.)

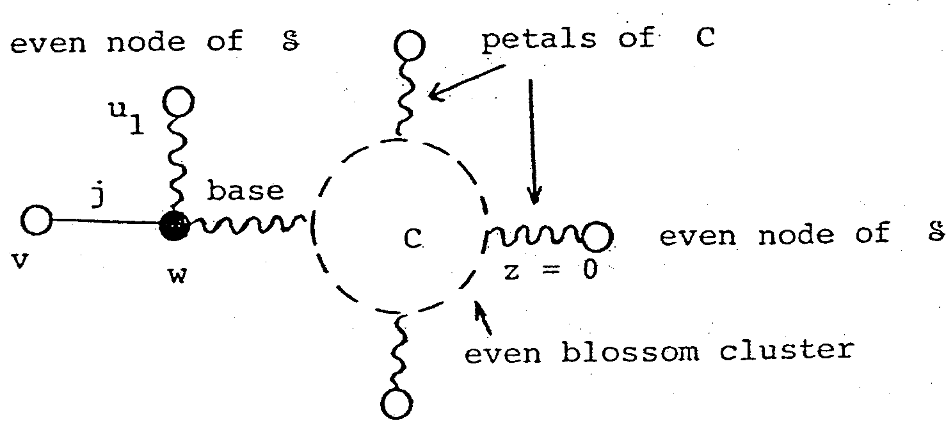

Figure 2.6