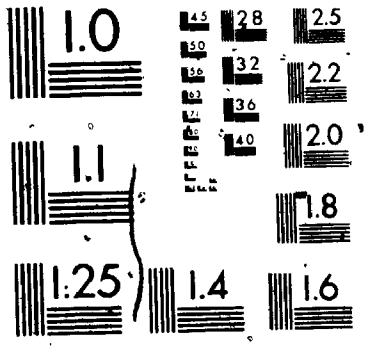
1



National Library Canada | Bibliothèque nationale du Canada | CANADIAN THESES ON MICROFICHE | THÈSES CANADIENNES SUR MICROFICHE

0-315-07709-3

55629

NAME OF AUTHOR/NOM DE L'AUTEUR EBRAHIM ROOHY-LALEH

TITLE OF THESIS/TITRE DE LA THÈSE IMPROVEMENTS TO THE THEORETICAL EFFICIENCY OF THE NETWORK SIMPLEX METHOD

UNIVERSITY/UNIVERSITÉ CARLETON UNIVERSITY

DEGREE FOR WHICH THESIS WAS PRESENTED/ GRADE POUR LEQUEL CETTE THÈSE FUT PRÉSENTÉE Ph.D.

YEAR THIS DEGREE CONFERRED/ANNÉE D'OBTENTION DE CE GRADE 1981

NAME OF SUPERVISOR/NOM DU DIRECTEUR DE THÈSE J. G. DEBANNÉ

DATED/DATÉ Apr. 24, 1981 SIGNED/SIGNÉ

PERMANENT ADDRESS/RÉSIDENCE FIXE 2969 #511 FAIRLEA CR.
OTTAWA ONTARIO K1V 9N2
CANADA

NL-91 (3-74)

IMPROVEMENTS TO THE THEORETICAL EFFICIENCY OF THE NETWORK SIMPLEX METHOD

by

© EBRAHIM ROOHY-LALEH, B.Sc., M.Sc.

A thesis submitted to the Faculty of Graduate Studies in partial fulfilment of the requirements for the degree of Doctor of Philosophy.

Department of Mathematics and Statistics

Carleton University

Ottawa, Ontario

October, 1980

The undersigned recommend to the Faculty of Graduate Studies acceptance of the thesis "IMPROVEMENTS TO THE THEORETICAL EFFICIENCY OF THE NETWORK SIMPLEX METHOD" submitted by Ebrahim Roohy-Laleh, B.Sc., M.Sc., in partial fulfilment of the requirements for the degree of Doctor of Philosophy.

Thesis Supervisor

External Examiner

Chairman, Department of Mathematics and Statistics

Carleton University

ABSTRACT

An example of the optimum assignment problem is given for which, the network simplex method, in the absence of cycling and "stalling" (an exponentially long sequence of consecutive degenerate pivots without cycling), requires an exponentially long simplex sequence due to successively small improvements in the objective function value during the non-degenerate pivots. Then pivoting rules of Cunningham are refined and good worst case computation bounds in the network simplex method for the optimum assignment problem are obtained. These rules use the strongly feasible tree method of Cunningham and the bounds depend logarithmically on the edge costs.

Three new pivot rules are given, each of which combined with the maintenance of strongly feasible trees prevents stalling in the network simplex method for the trans-shipment problem and the upper-bounded network simplex method for the minimum cost flow problem. Moreover one of these rules is refined and the first "good" network simplex method for the optimum assignment problem is obtained. This algorithm solves an n by n optimum assignment problem in no more than $n^3 - 2n^2 + n$ pivots among which at most $n^2 - 2n + 1$ could be non-degenerate.

A new phase I method for the minimum cost flow problem is presented. This method requires a specific starting basis which in the absence of negative cost directed circuits, can be found in a low degree polynomial

amount of work in the size of the problem being solved. This method has the property that the first feasible solution found is also an optimal solution to the minimum cost flow problem. Moreover in this method, both cycling and stalling can be prevented without the imposition of a specific topology such as strong feasibility on the bases, and its worst case computation bound is independent of the edge costs; indeed if the supply-demand and capacity vectors are integral and the method of strongly feasible trees is used, then the total number of simplex pivots in solving a minimum cost flow problem is bounded above by $|V| \cdot b^+$ (where V is the set of vertices and b^+ is the sum of the demands at the demand vertices). Furthermore this phase I method solves an n by n optimum assignment problem in at most n^2 pivots, exactly n of which are non-degenerate.

ACKNOWLEDGMENT

The author would like to thank Professor W.H. Cunningham for his helpful suggestions in the early stages of this work. The author would also like to thank Professor J.G. Debanné for his useful comments and continuous encouragement.

TABLE OF CONTENTS

INTRODUCTION

The network simplex method which is the specialization of the upper-bounded simplex method for the minimum cost flow problem (MCFP), is due mainly to Dantzig [12], but it could be traced back to Hitchcock [28] and Koopmans [36]. The essence of the algorithm is to start with a "primal" feasible tree solution and a vertex numbering (dual solution) associated with a spanning tree and proceed from spanning tree to spanning tree, successively obtained by the addition of a non-tree edge and deletion of a tree edge. At each iteration the feasibility of the new tree solution is maintained and the process is continued until either a tree is obtained, the associated solution of which satisfies certain proven optimality conditions or that the unboundedness of the problem is demonstrated.

The network simplex method which operates combinatorially on graphs rather than arithmetically on matrices (simplex tableaus) is known to be 50 to 100 times faster than the simplex method itself. However in the 50's and 60's the out-of-kilter algorithm of Fulkerson was shown to be superior in practice to the network simplex method. But in the late 60's and 70's due mainly to the rapid development of computers, many researchers turned their attention to the "practical" improvement of the network simplex method. Their objective was to implement the network simplex method, based on a compact representation

of the basis, determination of the leaving edge without trial and error and efficient techniques to update the vertex numbering at each iteration. In 1965, E.L. Johnson [33] and [32], developed one such data structure known as the "triple label method". But most of the recent developments in data structures and computer implementation took place in 70's by many different researchers. An excellent survey of these works can be found in Bradley, Brown and Graves [7]. The state-of-the-art of most of the recent improvements in the implementation of the network simplex method can be found in a paper by Ali, Helgason, Kennington and Lall [1].

The most important aspect of the contemporary implementations of the network simplex method has been to test its efficiency against the other network flow algorithms, in particular against the out-of-kilter algorithm, on large standard test problems. Unlike the experimental results of the 50's and 60's, the recent comprehensive comparisons which are carried out independently by many researchers have verified the unquestionable superiority of the new network simplex implementations over the fastest running out-of-kilter codes.

Unlike the practical aspects of the network simplex method which have received the attention of most researchers, the theoretical aspects of this method have been paid very little attention. In this paper we will study the network simplex method mainly from the theoretical point of view.

An algorithm is said to be "good in input size" in the sense of Edmonds, if its worst case computation bound is a polynomial in the amount of memory space required to input the data and it is said to be "good in problem size" or simply "good" in the sense of Edmonds [18], if its worst case computation bound is a polynomial in the number of variables and the number of constraints.

In any (network) simplex method it can easily be shown that the amount of work per simplex iteration (pivot) is a low degree polynomial in size of the problem being solved. Thus if the network simplex method were to be good in the sense of Edmonds, the total number of pivots must be bounded above by a polynomial in the number of variables and the number of contraints. Experiments on thousands of large and small scale real life and randomly generated problems reveal that, in practice, the total number of simplex pivots is proportional to the number of variables. However the examples of Edmonds [19] and Zadeh [46] show that if the entering and leaving edges of the network simplex method are chosen in a consistently ill-advised way, it would result in undesirable (exponential) number of pivots. Therefore the investigation of the theoretical efficiency of the algorithm must concentrate on restricting the choice of the entering and leaving edges in a systematic way so as to result in a "good" worst-case bound on the total number of pivots.

There are two main difficulties caused by degenerate pivots, i.e. the steps of the algorithm which yield a new tree but not a new

solution. One of these difficulties is "cycling" and the other one is known as "stalling". In the former case the (network) simplex algorithm fails to terminate due to an infinte sequence of consecutive degenerate pivots caused by encountering repeatedly the same sequence of trees. In the latter case the (network) simplex algorithm, in the absence of cycling, encounters an exponential sequence of consecutive degenerate pivots. Recently Bland [6], presented an elegant proof that a certain simple and natural entering and leaving edge rule (pivoting rule) never leads to cycling. Bland's rule is extremely simple, indeed:

(i) Among all the candidate edges to enter the basis, select the edge with the smallest subscript.

(ii) If more than one edge is candidate to leave the tree, choose again the edge with the smallest subscript.

This rule in its simplest form, prevents cycling even in linear programming problems. But Avis and Chvátal [2], using the example of Klee and Minty [34] have shown that the simplex algorithm with Bland's rule could stall in solving linear programming problems. Cunningham [11], using Edmonds' [19] example, has reached the same conclusion as Avis and Chvátal for the network flow problems. Cunningham [10], has imposed a topology on the network simplex bases (trees) and has called a feasible tree with this topology a "strongly feasible" tree. He has shown that not only the network simplex method with the maintenance of strongly feasible trees (strongly feasible network simplex method) leads to an optimal solution (provided that an optimal solution exists), but it also never cycles.

As for Bland's rule, Cunningham [11], using Edmonds' [19], example has shown that, the strongly feasible network simplex method may require an exponential number of consecutive degenerate pivots. However unlike Bland's rule, the method of strongly feasible trees has an interesting feature in that, throughout the algorithm the strongly feasible structure of the trees is preserved only by a specific choice of the leaving edges. This leaves some flexibility in the algorithm, namely the choice of the entering edges to use to prevent stalling. Indeed by restricting the choice of entering edges, Cunningham [11], has given four different pivot rules each of which combined with the maintenance of strong feasibility prevents stalling.

In §2.4 and §2.6 the results of [10] and [11] are reviewed and in chapter 3, three new entering edge rules are presented, each of which prevents stalling in the MCFP when combined with the method of strongly feasible trees. Avis and Chvátal [2], have shown that prevention of stalling in the linear programming simplex algorithm is sufficient to make it a "good" algorithm in the sense of Edmonds. They do this by transforming the given linear programming problem into a "highly" degenerate linear programming problem (i.e. a problem with zero right hand side). But unfortunately Avis and Chvátal's transformation does not preserve the network structure of the network

flow problems; thus prevention of stalling in the network simplex method does not automatically make it a "good" algorithm in the sense of Edmonds. This is to say that the network simplex method in the absence of stalling may still require an exponential number of pivots, caused by successively "small" improvements in the objective function value during the execution of non-degenerate pivots. Indeed this slow improvement in the objective function value is the reason for the poor behaviour of the network simplex method in Zadeh's [46] example. For any integer $n \geq 1$, Zadeh's example is a "circulation problem" with $2n + 2$ vertices and the "return edge" requires $2^n + 2^{n-2} - 2$ units of flow. But the problem is constructed so that every non-degenerate pivot of the (strongly feasible) network simplex method increases the flow in the return edge only by one unit, thus requiring $2^n + 2^{n-2} - 2$ non-degenerate pivots. This example demonstrates the dependence of the worst case computation bound of the (strongly feasible) network simplex method on the supply-demand vector. In §4.2 a "bad" example of the optimum assignment problem is presented, for which the (strongly feasible) network simplex method requires an exponential number of pivots in the absence of the stalling. Once again this "bad" behaviour of the network simplex method is due to the slow improvements in the objective value during the execution of non-degenerate pivots, which unlike the Zadeh's example, is caused by the specific choice of the costs. Therefore the worst case computation bound of the (strongly feasible) network simplex method is dependent

on the supplies and demands as well as the costs. This paper is mainly intended to make a start at reducing or better still, eliminating the dependence of the worst case computation bound of the network simplex method on supplies and demands and/or costs while ensuring that the algorithm will not cycle and will not stall.

In §4.3, the entering edge rules of [11] are refined and "good" worst case computation bounds (in input size of the problem) for the optimum assignment problem are obtained (that is, the bounds depend logarithmically on the costs of edges). In §4.4, the first "good" network simplex method for the optimum assignment problem is presented. This algorithm maintains strongly feasible trees and solves an n by n optimum assignment problem in at most $n^3 - 2n^2 + n$ pivots among which at most $n^2 - 2n + 1$ could be non-degenerate. In §5.1, it is shown that, in solving the maximum flow problem with the network simplex method, cycling and stalling can be prevented with some simple leaving edge rules. In the remainder of chapter 5 a new phase I method for the MCFP is presented. This method uses the well-known "most negative reduced cost" entering edge rule and has the following properties:

(i) The first feasible solution encountered is also optimal. If the capacities and supplies and demands are integers and if the problem does not contain negative cost dicircuits then

(ii) The worst case bound on the number of pivots is $(|E| - |V| + 1)\cdot b^+$.

(iii) If the method of strongly feasible trees is used, then the worst case bound on the number of pivots is $|V|\cdot b^+$.

(Where b^+ is the sum of the demands at the demand vertices)

(iv) This method with the maintenance of strongly feasible trees, solves an n by n optimum assignment problem in at most n^2 pivots, exactly n of which are non-degenerate. Furthermore, this method can be used to solve an optimum assignment problem of given cardinality.

CHAPTER 0

§0.1 SOME GRAPH THEORETIC CONCEPTS

A (finite) DIRECTED GRAPH or DIGRAPH $G = (V, E)$ consists of a finite set $V = V(G)$ of VERTICES (or NODES or POINTS) $u, v, w, \ldots$ and a finite subset E of ordered pairs (u, v) of elements taken from V. Elements of E are called (DIRECTED) EDGES (or ARCS or LINKS) and will be designated by $e, f, g, \ldots$ An edge $e = (u, v)$ in E is said to be directed from its TAIL u to its HEAD v. Head and tail of an edge $e \in E$ are called ENDS of e and are denoted by $h(e)$ and $t(e)$, respectively. Ends of an edge e are said to be ADJACENT vertices; e is said to be INCIDENT to its ends. An edge $e = (u, u)$ is called a LOOP. Two or more edges are MULTIPLE EDGES if they have the same head and the same tail. A graph is called SIMPLE if it contains no loops and no multiple edges. A graph $G = (V, E)$ is said to be an UNDIRECTED GRAPH if E consists of unordered pairs of vertices; this means that, the elements of E have no specific orientations or are undirected. A MIXED GRAPH $G = (V, E)$ is a graph, for which some elements of E are ordered pairs, others are not, in other words, some edges are directed, others are undirected. A SUBGRAPH $G' = (V', E')$ of a graph $G = (V, E)$ is a graph, for which $V' \subseteq V$ and $E' \subseteq E$; if $V' = V$, then the subgraph is called SPANNING.

In a mixed graph, we denote by $\delta(V')$, for $V' \subseteq V$ the set of edges $\{e \in E$: e is undirected and exactly one end e is in V' or e is directed and only $t(e) \in V'\}$; $\gamma(V')$ the set of edges $\{e \in E$: both ends of e are in $V'\}$. A graph $G = (V, E)$ is said to be BIPARTITE if there exists $V' \subseteq V$, for which each element of E has one end in V' and the other in $V \setminus V'$; this means that $\delta(V') = E$; $(V', V \setminus V')$ is called a BIPARTITION of G. A mixed graph is DICONNECTED or STRONGLY CONNECTED if for every $V' \subseteq V$, $\phi \neq V' \neq V$, we have $\delta(V') \neq \phi$. A PATH in a mixed graph G from $v^0 \in V$ to $v^n \in V$ is an alternating sequence of vertices and edges P: $v^0, e^1, v^1, e^2, \ldots, e^n, v^n$, in which each edge is incident to the two vertices immediately preceding and following it. P is said to JOIN v^0 to v^n. A FORWARD [REVERSE] EDGE of P is a directed edge of P such that its tail [head] proceeds its head [tail] in the path. A path is said to be DIRECTED PATH or DIPATH, if every directed edge of it is a forward edge. A path P is EDGE-SIMPLE if all the edges are distincts, and VERTEX SIMPLE or SIMPLE if all the vertices (and thus necessarily all the edges) are distinct. A simple path from u to v with at least one edge is called a CIRCUIT if u = v.

A path P with n (not necessarily distinct) edges is said to have length n. The DISTANCE in G from vertex u to vertex v is the LENGTH of the path in G from u to v if such path exists, and is ∞ otherwise. Forward and reverse edges of a circuit and a DIRECTED CIRCUIT or DICIRCUIT are defined as expected. A FOREST is a graph with no circuits; a connected forest is a TREE.

The (0, 1, -1) vertex - edge INCIDENCE MATRIX of a mixed graph G is the matrix $A = (a_{ij}: i \in V, j \in E)$ where

$$a_{ij} = \begin{cases} -1 & \text{if } j \text{ is directed and } i = t(j), \\ 0 & \text{if } i \text{ is not an end of } j, \\ 1 & \text{if } j \text{ is directed and } i = h(j) \\ & \text{or } j \text{ is undirected and } i \text{ is one of its ends.} \end{cases}$$

The following lemmas are due to Johnson [33].

<u>Lemma 0.1.1</u> If there is a path from u to v in a graph G then there is a simple path from u to v.

<u>Proof</u>. Let $u = v^0, e^1, v^1, e^2, \ldots, e^n, v^n = v$ be a path from u to v. Let v^i for some $0 \leq i \leq n$ be the first vertex which is repeated, and suppose v^j is the last listing of vertex v^i in the path. Omit the segment $e^{i+1}, v^{i+1}, e^{i+2}, \ldots, e^j, v^j$ from the path to form a new connected path $u = v^0, e^1, v^1, e^2, \ldots, e^i, v^i, e^{j+1}, v^{j+1}, \ldots, e^n, v^n = v$ such that the vertices $v^0, v^1, \ldots, v^i$ are not repeated. Clearly this reduction process is finite and the resulting path will be simple. Q.E.D.

The following is an inductive characterization of trees.

<u>Lemma 0.1.2</u> A tree is either a vertex, or is two disjoint trees joined by a single edge incident to one vertex of one tree and one vertex of the other tree.

<u>Proof</u>. Clearly, a graph constructed in this way is a tree. Now we must show that if an edge (if one exists) of a tree is deleted, then the new graph has two components each of which is a tree. If a tree T has no edge then it is a single vertex. Otherwise let $e = (u,v)$ be an edge of T, then there is no simple path joining u to v which

does not contain e because if one exists, then adding e to the path would have formed a circuit, thereby contradicting the definition of a tree. Hence by lemma 0.1.1 there is no path from u to v not containing e. So if e is removed from the tree T, then the remaining graph has at least two connected components T^1 and T^2 with u in T^1 and v in T^2. Let $U[V] \subseteq V(T)$ be such that the unique simple path from an element of U[V] to u[v] in T does not contain e. Now we wish to show that $U = V(T^1)$ and $V = V(T^2)$. Clearly $V(T^1) \subseteq U$. Let $u' \in U$, then $u' \notin V(T^2)$ because otherwise the paths from u' to u and u' to v and e would form a circuit in T. If $u' \notin V(T^1)$, this implies that this path in T from u' to u must contain e; i.e. $u' \notin U$, which is a contradiction. Therefore $U = V(T^1)$, similarly $V = V(T^2)$. Thus removal of e causes the remaining graph to have exactly two connected components T^1 and T^2, both of which are trees because if either had a circuit, so would T. Q.E.D.

An END or LEAF of a tree is a vertex of a tree, which is incident to at most one edge of the tree.

<u>Lemma 0.1.3</u> Every tree has at least one end and if it has an edge, then it has at least two ends.

<u>Proof</u>. This is true for a single vertex and for a tree with one edge. Suppose it is true for two trees. Then adding an edge incident to one vertex of each tree will always leave at least one end in each tree; but by lemma 0.1.2, the new graph is also a tree, hence it will have at least two ends. Q.E.D.

Lemma 0.1.4 A tree with m vertices has m-1 edges.

Proof. The proof is immediate using lemma 0.1.2 and induction as in the proof of lemma 0.1.3. Q.E.D.

Lemma 0.1.5 Every connected graph G contains a spanning tree T.

Proof. Define a subgraph T having the same vertex set as G and an edge set chosen as follows: Initially, choose any edge of G to be in T. Thereafter, choose any edge of G that does not produce a circuit in T. When every edge in G\T produces a circuit if added to T, then T is easily seen to be a spanning tree of G. Q.E.D.

If A is a matrix then the RANK of A denoted by r(A) is the maximal number of its linearly independent columns. It is a well-known result of linear algebra that the rank is also equal to the maximal number of linearly independent rows of A.

Lemma 0.1.6 If T = (V,E) is a tree where the elements of E are directed edges and if A is the incidence matrix of T, then A has full column rank.

Proof. From lemma 0.1.4, $|E| = |V|^{\dagger} - 1$ and from definition of A, it is clear that each column of A contains exactly two non-zero entries one of which is 1 and the other -1; thus, adding up all the rows of A "point wise", results in a zero vector, which means that any row of A can be written as the linear combination of other rows, thus $r(A) \leq |V| - 1$. By induction we will show that $r(A) = |V| - 1$. If tree T has only one edge then the lemma is clearly true.

---

†Where S is a set then |S| denotes the CARDINALITY of S i.e. the number of elements of S.

Suppose two trees T^1 and T^2 have full column rank. Join T^1 to T^2 by an edge say $e = (u,v)$, where $u \in V(T^1)$ and $v \in V(T^2)$ to obtain a bigger tree T. Let A^1, A^2 and A be the incidence matrices of T^1, T^2 and T respectively. By elementary row operations we can make the row corresponding to vertex u of T^1, a zero row, without changing the rank of A^1. The matrix A can be represented as follows:

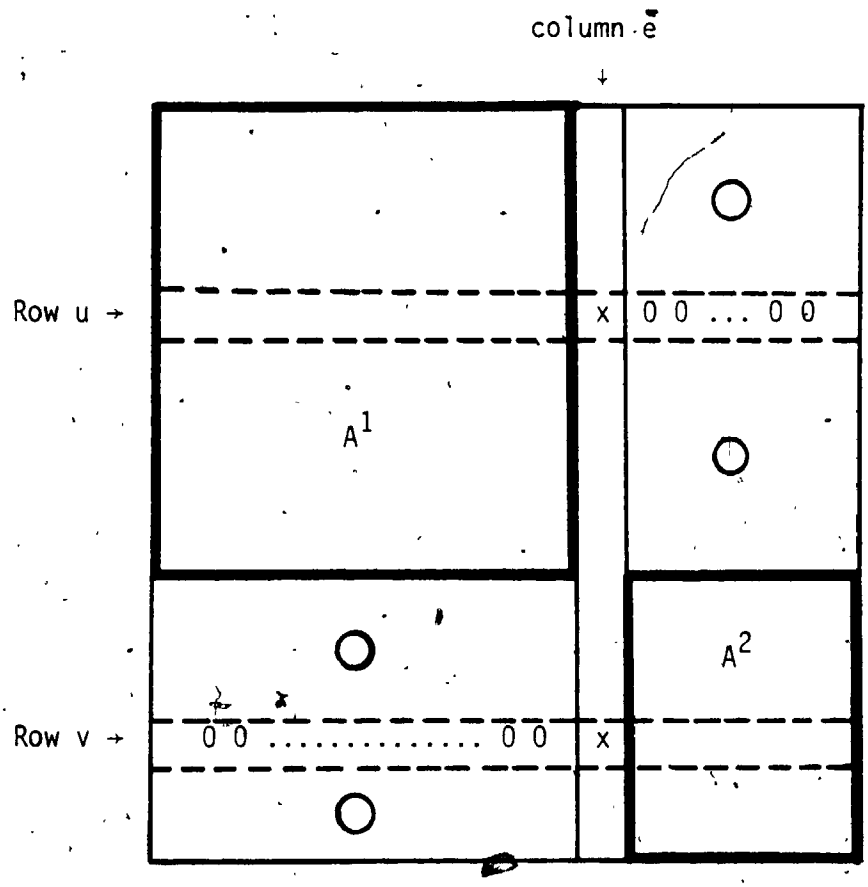


Fig. 0.1.1 Incidence matrix A of tree T obtained by joining vertex u of T^1 to vertex v of T^2 by an edge e.
"x" denotes the non-zero entries of column e.

From Fig. 0.1.1 it is clear that no column of A can be written as a linear combination of other columns and this proves the lemma. Q.E.D.

<u>Lemma 0.1.7</u> Let C: $v^1, e^1, v^2, e^2, \ldots, e^n, v^{n+1} = v^1$ be a circuit, where elements of E(C) are directed edges. If A is the incidence matrix of C, then columns of A are linearly dependent.

<u>Proof</u>. Clearly $|V(C)| = |E(C)|$, but since C is a digraph, therefore the row rank of A $\leq |V(C)| - 1$, which implies that the column rank of A $\leq |V(C)| - 1 = |E(C)| - 1$. Q.E.D.

§0.2. LINEAR PROGRAMMING PROBLEMS

A linear programming (LP) problem is a problem of optimizing an n-variable linear function called OBJECTIVE FUNCTION, subject to m equality and/or non-strict linear inequality constraints. Here we will briefly discuss the mathematical theory of the LP problem rather than solution techniques such as Dantzig's Simplex Method. For more details see any linear programming text such as [14], [24], [27]. The linear programs:

$$(\text{LP1}) \quad \begin{cases} \min & \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} & \\ & \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1,2,\ldots,m, \\ & x_j \geq 0, \qquad j = 1,2,\ldots,n. \end{cases}$$

and

$$
(LP2) \quad \begin{cases} \max & \sum_{i=1}^{m} b_i y_i \\ \text{s.t.} & \\ & \sum_{i=1}^{m} a_{ij} y_i \le c_j, \quad j = 1,2,\ldots,n, \\ & y_i \text{ unrestricted in sign}, \quad i = 1,2,\ldots,m, \end{cases}
$$

are said to be DUAL linear programs. (Here a_{ij}, b_i and c_j are constants; the x_j and y_i are variables). (LP1) and (LP2) can also be written in the matrix notation as:

$$
(LP1) \quad \begin{cases} \min & C^T X \\ \text{s.t.} & AX = b; \\ & X \ge 0. \end{cases} \qquad (LP2) \quad \begin{cases} \max & Y^T b \\ \text{s.t.} & Y^T A \le C, \\ & Y \text{ unrestricted in sign}. \end{cases}
$$

where A is an m x n real matrix called CONSTRAINT MATRIX, and the COST vector C and CONSTANT vector b are given real vectors of dimensions n x 1 and m x 1 respectively; real vectors X and Y are primal and dual variables of dimensions n x 1 and m x 1 respectively.

A vector X satisfying AX = b is a SOLUTION OF LP1. If, in addition $X \ge 0$, then X is a FEASIBLE SOLUTION of LP1.

A basis of matrix A is a matrix B consisting of a maximal set of linearly independent columns of A, i.e. number of columns of B is equal to the rank of A. A BASIC SOLUTION to the linear system

$AX = b$, $0 \leq X \leq u$, is a solution X^o for which there is a basis B of A such that $x_j^o = 0$ or $x_j^o = u_j$, unless x_j^o corresponds to a column of B. If in addition $0 \leq x_j^o \leq u_j$, $j = 1,\ldots,n$ then X^o is called a BASIC FEASIBLE SOLUTION. A variable corresponding to a column of B is called a BASIC variable, otherwise it is said to be a NON-BASIC variable.

The following are some well known results of linear programming.

Theorem 0.2.1. If an LP Problem has an optimal solution, it has a basic optimal solution.

Proof. See [14], [24], [27].

Theorem 0.2.2. (WEAK LP DUALITY THEOREM). For any feasible solution X^o of LP1 and any feasible solution Y^o of LP2, $\sum_{j=1}^{n} c_j x_j^o \geq \sum_{i=1}^{m} b_i y_i^o$.

Proof. See [14], [24], [27].

Corollary 0.2.1. If LP1 is UNBOUNDED (has feasible solutions for which $\sum_{j=1}^{n} c_j x_j$ is arbitrarily small) then LP2 is INFEASIBLE (has no feasible solution).

Corollary 0.2.2. If LP2 is UNBOUNDED (has feasible solutions for which $\sum_{i=1}^{m} b_i y_i$ is arbitrarily large) then LP1 is infeasible.

Corollary 0.2.3. If there exist feasible solutions, X^o, Y^o of LP1, LP2 respectively such that $\sum_{j=1}^{n} c_j x_j^o = \sum_{i=1}^{m} b_i y_i^o$, then X^o, Y^o are optimal to their respective problems.

Theorem 0.2.3. (STRONG LP DUALITY THEOREM). If one of LP1, LP2 is bounded and feasible, then the other is, and there exist optimal solutions, X^o, Y^o of LP1, LP2 respectively such that $\sum_{j=1}^{n} c_j x_j = \sum_{i=1}^{m} b_i y_i$.

Proof. See [14], [24], [27].

Theorem 0.2.4. (COMPLEMENTARY SLACKNESS THEOREM "C.S.T."). If X^o, Y^o are feasible to LP1 and LP2 respectively, then they are optimal to their respective problems if and only if they satisfy the following "complementary slackness" conditions: For all $j = 1,2,\ldots,n, x_j^o > 0$ implies $\sum_{i=1}^{m} a_{ij} y_i^o = c_j$ or equivalently $x_j^o(c_j - \sum_{i=1}^{m} a_{ij} y_i^o) = 0$ (i.e. if the jth primal variable is positive, then jth dual constraint holds with equali

Proof. See [14], [24], [27].

If (primal) LP problem is stated in a form other than that of LP1, then using the following simple devices, it can be converted to the form of LP1.

An inequality constraint of the form $\sum_{j=1}^{n} a_{ij} x_j \leq b_i$ $[\sum_{j=1}^{n} a_{ij} x_j \geq b_i]$ can be converted to equality constraint by adding [subtracting] a non-negative slack [surplus] variable to [from] the left hand side of the inequality.

If a variable is unrestricted in sign, then it can be replaced by a difference of two non-negatively restricted variables. Maximization of a function is equivalent to minimization of its negative.

If $0 \le \ell_j \le x_j$ then x_j can be replaced by $x_j' = x_j - \ell_j \ge 0$. In view of the duality theorem, LP1 and LP2 must satisfy one of the following:

(1) LP1 unbounded and LP2 infeasible.

(2) LP1 infeasible and LP2 unbounded.

(3) Both LP1 and LP2 are feasible and have optimal solutions in which case the optimal value of the objective functions are equal.

(4) Both LP1 and LP2 are infeasible.

<u>Lemma 0.2.1</u>. The feasible solution X^o of

$$(LP)\quad \begin{cases} \min \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} \\ \sum_{j=1}^{n} a_{ij} x_j \ge b_i, & i = 1,2,\ldots,m, \\ 0 \le x_j \le u_j, & j = 1,2,\ldots,n. \end{cases}$$

is optimal if and only if there exists a vector $Y^o = (y_i^o : i = 1, \ldots, m)$, such that

$$\begin{cases} \sum_{i=1}^{m} a_{ij} y_i^o < c_j \text{ implies } x_j^o = 0 \\ \sum_{i=1}^{m} a_{ij} y_i^o > c_j \text{ implies } x_j^o = u_j. \end{cases}$$

<u>Proof</u>. See [15], [23], [27].

CHAPTER 1

§1.1. NETWORK FLOW PROBLEM

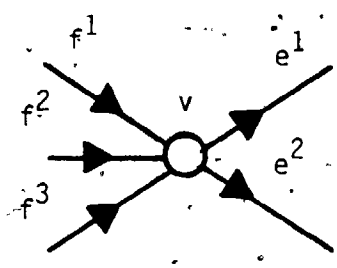
Let $G = (V,E)$ be a digraph and A its vertex-edge incidence matrix. A linear programming problem with A as its constraint matrix is called a NETWORK FLOW (NF) PROBLEM. An NF problem of the form:

$$\begin{cases} \min & C^T X \\ \text{s.t.} & AX = b, \qquad 1.1.1 \\ & 0 \leq X \leq u, \qquad 1.1.2 \end{cases}$$

is called a MINIMUM COST FLOW PROBLEM (MCFP), i.e. A is an $m \times n$ $(0,1,-1)$ matrix in which each column contains only two non-zero entries one of which is $+1$ and the other -1. C, b and u are given constant vectors of costs, supply-demand and upper bounds and are of dimensions $n \times 1$, $m \times 1$ and $n \times 1$ respectively and X is an $n \times 1$ vector of variables. A solution of an NF problem will be called a FLOW and a feasible solution of this problem will similarly be called FEASIBLE FLOW.

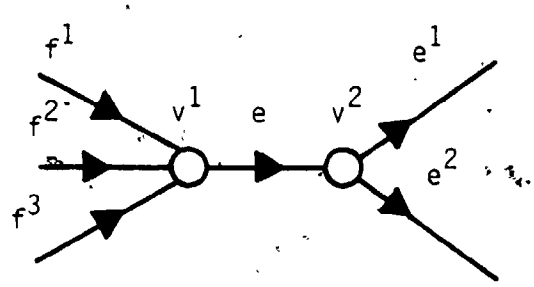
Let P be a path from u to v in G and let $X = (x_e : e \in E)$ be a feasible flow of NF problem. P is said to be a FLOW AUGMENTING PATH if $x_e < u_e$ for each forward edge e and $x_e > 0$ for each reverse edge e in P. A FLOW AUGMENTING CIRCUIT in a specific direction is defined similarly.

It is clear from the definition of A that each row of A corresponds to a vertex of G and each column of A corresponds to an edge of G. If b_v for some vertex v is negative then v is called a SUPPLY VERTEX and if $b_v > 0$ then it is called a DEMAND VERTEX; otherwise v is referred to as a TRANS-SHIPMENT VERTEX. It is assumed that the vertices do not have upper bounds and/or costs. This is not a restriction, in the sense that, if v had an upper bound u_v and a cost c_v, then we could define a new digraph $G' = (V',E')$ from $G = (V,E)$, with $V' = [V \setminus \{v\}] \cup \{v^1, v^2\}$ and $E' = E \cup \{e = (v^1, v^2)\}$, where v^1 and v^2 have no upper bounds, no costs, $c_e = c_v$ and $u_e = u_v$. If $f \in E$ with $h(f) = v$ $[t(f) = v]$ then $f \in E'$ with $h(f) = v^1$ $[t(f) = v^2]$. See Fig. 1.1.1 (a) and 1.1.1 (b).



In G, v has cost c_v and upper bound u_v.

Fig. 1.1.1 (a)



In G', $c_e = c_v$ and $u_e = u_v$. v^1 and v^2 have no costs and no upper bounds.

Fig. 1.1.1 (b)

<u>Remark 1.1.1.</u> A necessary condition for an NF problem to have a feasible solution is that $\sum(b_v : v \in V) = 0$.

<u>Proof.</u> Since each column of A contains only one 1 and one -1, therefore by adding all of the constraints in 1.1.1, we obtain

$$0 = \sum[(1 - 1)x_e;\ e \in E] = \sum(b_v : v \in V). \qquad \text{Q.E.D.}$$

Remark 1.1.1 implies that the $r(A) \leq |V| - 1$.

If B is a basis of A, then let T_B be the digraph corresponding to B (This means that vertices and edges of T_B correspond to rows and columns of B respectively).

<u>Theorem 1.1.1</u>. If B is a basis of A, then T_B is a spanning tree of graph G.

<u>Proof</u>. Since G is connected, therefore by lemma 0.1.5, G has a spanning tree T, but from lemma 0.1.6, $r(T) = |V| - 1$ and since $r(A) \leq |V| - 1$, therefore $r(A) = |V| - 1$ i.e. any basis B of A has $|V| - 1$ columns and $|V|$ rows.

If T_B is not a spanning subgraph of G, then some vertex of G say v^1 of G, is not in T_B. Then, every entry of row 1 of B is zero. But some column of A has a non-zero entry in row 1, and such a column cannot be written as a linear combination of columns of B, thereby contradicting B being a basis. If we show that T_B cannot contain a circuit, then lemma 0.1.4 would imply that T_B is a tree.

If T_B contains a circuit, then by lemma 0.1.7, columns of B corresponding to the edges of the circuit are linearly dependent, contradicting B being a basis. Q.E.D.

Hereafter, a basic solution of the NF problem will be called a TREE SOLUTION, and the MCFP will be written as follows:

$$
(\text{MCFP}) \begin{cases} \min & \sum (c_e x_e;\ e \in E) \\ \text{s.t.} & \\ & \sum (x_e:\ v = h(e)) - \sum (x_e:\ v = t(e)) = b_v, \text{ for all } v \in V, \\ & 0 \le x_e \le u_e, \text{ for all } e \in E. \end{cases}
$$

§1.2. IMPORTANT SPECIAL CASES OF MCFP

(1) <u>The Trans-shipment Problem (TP)</u>: This is a special case in which each $u_j = \infty$.

(2) <u>The Hitchcock Transportation Problem</u>: This is a special case in which all $u_j = \infty$ and the graph is bipartite with bipartition S and T such that all the edges are directed from S to T and $b_s < 0$ for all $s \in S$ and $b_t > 0$ for all $t \in T$.

(3) <u>Optimum Assignment (OA) Problem</u>: This is a special case of the Hitchcock transportation Problem in which $|S| = |T|$, $b_s = -1$ for all $s \in S$, $b_t = 1$ for all $t \in T$, and the problem requires an integer (0 - 1) solution. We can think of elements of S as being men and elements of T as being jobs and each man is to be assigned to one job and each job to be assigned to one man. Where c_{st} is the cost of assigning the man s to job t, the problem is to find an optimum assignment.

(4) Circulation Problem (CP): This is a special case in which each $b_v = 0$. Any MCFP can be cast into a circulation problem by slightly enlarging the graph, as follows:

Let $S = \{v \in V: v \text{ is a supply vertex i.e. } b_v < 0\}$ and

$T = \{v \in V: v \text{ is a demand vertex i.e. } b_v > 0\}$,

here, elements of S are called sources and elements of T are called sinks.

Add a "dummy" source vertex s and a "dummy" sink vertex t to the graph, and for each $v \in S$, add an edge (s,v) to the graph with $c_{sv} = 0$ and $u_{sv} = -b_v$; likewise for each $u \in T$ add an edge (u,t) to the graph with $c_{ut} = 0$ and $u_{ut} = b_u$ and finally add "return edge" (t,s) to the graph with $c_{ts} = -M$ and $u_{ts} = \sum (b_u: u \in T)$, where M is a "large enough" positive integer. See Fig. 1.2.1.

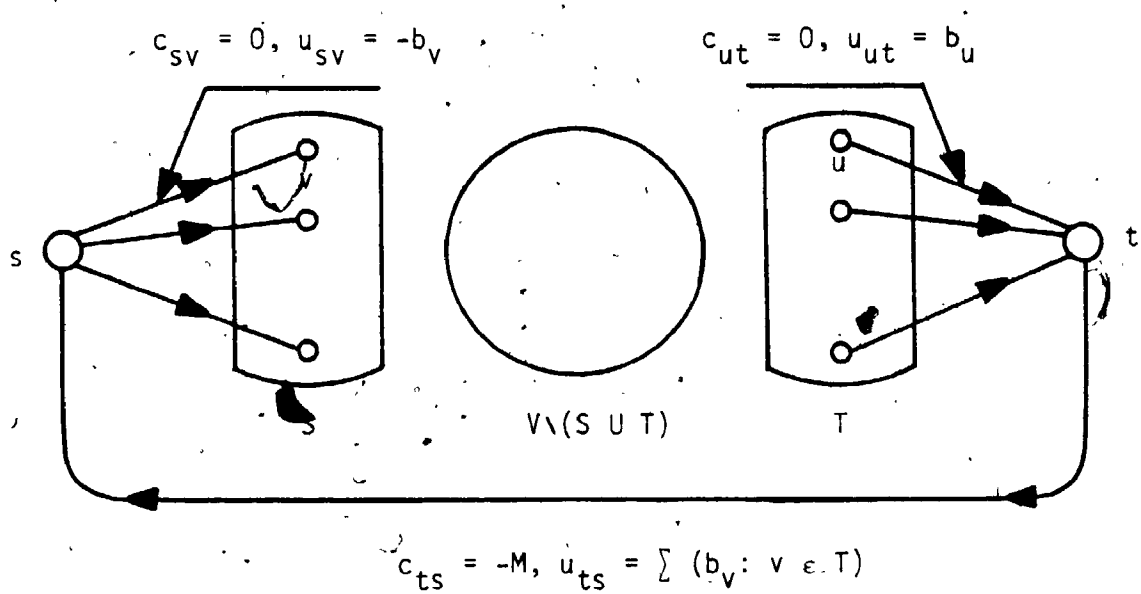


Fig. 1.2.1

(5) Shortest Path Problem: This is a special case of MCFP in which $b_r = 1 - |V|$, $b_v = 1$ for $v \neq r$, all $u_j = \infty$, and we ask for an integer-valued optimum solution; then we have the (one-to-all) shortest paths problem.

Maximum Flow Problem: This is a linear programming problem defined on digraph G as follows:

$$\begin{cases} \max & z \\ \text{s.t.} & \sum (x_e : v = h(e)) - \sum (x_e : v = t(e)) = \begin{cases} -z & \text{if } v = s \\ 0 & \text{if } v \neq s \text{ or } t \\ z & \text{if } v = t \end{cases} \\ & 0 \leq x_e \leq u_e, \text{ for all } e \in E. \end{cases}$$

where s is the only source and t is the only sink in the graph. This can easily be cast into an equivalent circulation problem, by simply adding a return edge (t,s) with $c_{ts} = -1$ and $u_{ts} = \infty$, upper bounds of the other edges are as defined. Then, the new problem is:

$$\begin{cases} \min & -x_{ts} \\ \text{s.t.} & \sum (x_e : v = h(e)) - \sum (x_e : v = t(e)) = 0, \text{ for all } v \in V, \\ & 0 \leq x_e \leq u_e, \text{ for all } e \in E. \end{cases}$$

§1.3. EQUIVALENCE OF HITCHCOCK TRANSPORTATION AND MCFP

Orden [41], has devised a method, which transforms a MCFP into a capacitated transportation problem in which each edge has an upper-bound and Dantzig [13], has shown that, the latter can be transformed into the Hitchcock transportation problem. However Wagner [44], has suggested a method which makes the transition in one step. These transformations increase the problem size by a modest factor, but from the theoretical point of view are not totally unreasonable. However in practice, such transformations are wasteful. These transformations will not be given here and the MCFP will be treated directly.

§1.4. LINEAR PROGRAMMING ANALYSIS OF MCFP

The dual of (MCFP) given at the end of §1.1 is:

$$(\text{DMCFP})\begin{cases} \max & \sum (b_v y_v : v \in V) - \sum (u_e \cdot w_e : e \in E), \\ \text{s.t.} & y_{h(e)} - y_{t(e)} - w_e \leq c_e, \text{ for all } e \in E, \\ & y_v \text{ unrestricted in sign, for all } v \in V, \\ & w_e \geq 0, \text{ for all } e \in E. \end{cases}$$

By theorem 0.2.4 "C.S.T.", X^o and (Y^o, W^o) are optimal to MCFP and DMCFP if and only if

$$\begin{cases} x_e^o > 0 \Rightarrow y^o_{h(e)} - y^o_{t(e)} - w^o_e = c_e \\ w_e^o > 0 \Rightarrow x_e^o = u_e. \end{cases} \qquad (0.C.1)$$

But $x_e^o = u_e \Rightarrow w_e^o \geq 0$ and $x_e^o > 0 \Rightarrow y^o_{h(e)} - y^o_{t(e)} - w_e^o = c_e$

$\Rightarrow y^o_{h(e)} - y^o_{t(e)} \geq c_e$.

On the other hand $x_e^o = 0 \Rightarrow x_e^o < u_e \Rightarrow w_e^o = 0$, then from (DMCFP) we get $y_{h(e)} - y_{t(e)} \leq c_e$, therefore (O.C.1) can be written as:

$$\begin{cases} x_e^o = 0 \Rightarrow & y^o_{t(e)} - y^o_{h(e)} + c_e \geq 0, \\ x_e^o = u_e \Rightarrow & y^o_{t(e)} - y^o_{h(e)} + c_e \leq 0. \end{cases} \qquad \text{(O.C.)}$$

Hereafter the elements of dual vector Y, will be called VERTEX NUMBERING and $y_{t(e)} - y_{h(e)} + c_e$ will be called the REDUCED COST of edge e, and will be denoted by $\bar{c}_e$. With this notation, (O.C.) can be written as

$$\begin{cases} x_e^o = 0 \Rightarrow & \bar{c}_e \geq 0, \\ x_e^o = u_e \Rightarrow & \bar{c}_e \leq 0. \end{cases} \quad \text{or} \quad \begin{cases} \bar{c}_e > 0 \Rightarrow & x_e^o = 0, \\ \bar{c}_e < 0 \Rightarrow & x_e^o = u_e. \end{cases} \qquad \text{(O.C.)}$$

§1.5. GRAPHIC INTERPRETATIONS OF VERTEX NUMBERING AND REDUCED COST

Cost of a dipath P from u to v, with respect to cost vector C, is defined as follows:

$$C(P) = \sum(c_e\text{: e is a forward edge of P}) - \sum(c_e\text{: e is a reverse edge of P}).$$

A path P from u to v with respect to flow X is a FLOW AUGMENTING path if $x_e \leq u_e$ for every forward edge e of P, and $x_e > 0$ for every reverse edge

e of P. The cost of a dicircuit and a flow augmenting circuit in a specific direction, are defined as expected. Let X be a basic feasible solution to MCFP, corresponding to basis B. From theorem 1.1.1, the graph associated with B is a spanning tree T of graph G. To evaluate vertex numbering (dual variables) corresponding to B, we must set to equality, the inequality dual constraints associated with basic variables. This obviously is a system of linear equations, with $|V|$ variables and $|V| - 1$ equations. It is a well-known result of linear algebra, that such a linear system has infinitely many solutions and one of the variables must be considered as a parameter. Let r be an arbitrary but fixed vertex of G. The vertex r will be called the root of the tree T. Set $y_r = 0$, then the system will have a unique solution, and can be evaluated as follows: choose an edge e of T such that one end of e is r. There exists such an edge, since T is a spanning tree of G. If $h(e) = r$, then

$$0 = y_{t(e)} - y_{h(e)} + c_e = y_{t(e)} + c_e \text{ or } y_{t(e)} = -c_e .$$

If $t(e) = r$, then

$$0 = y_{t(e)} - y_{h(e)} + c_e = -y_{h(e)} + c_e \text{ or } y_{h(e)} = c_e .$$

Now besides r, the vertex numbering of another element of V is known. Choose an edge e of T, which is not chosen before, such that one

end of e is incident to one of the vertices with known vertex numbering, then evaluate the vertex numbering of the other end, and continue the process. From the procedure it is clear that y_v, for $v \in V$, is the cost of the unique path in the tree T from r to v. Now consider a non-basic edge e relative to T, and let w be the first common vertex of two paths in the tree T from h(e) and t(e) to the root. Let P_1 and P_2 be the unique paths in T from w to t(e) and w to h(e) respectively; then it is clear that $y_{t(e)} = y_w + C(P_1)$ and $y_{h(e)} = y_w + C(P_2)$. But

$$\bar{c}_e = y_{t(e)} - y_{h(e)} + c_e = y_w + C(P_1) - [y_w + C(P_2)] + c_e$$

$$= C(P_1) - C(P_2) + c_e$$

$$= C(P_1) + C(\bar{P}_2) + c_e \,, \qquad 1.5.1$$

where $\bar{P}_2$ is the unique path in the tree T from h(e) to w. But $P_1 \cup \{e\} \cup \bar{P}_2$ is the unique circuit formed by adding e to the tree T; thus 1.5.1 implies that $\bar{c}_e$ is the cost of the unique circuit formed by adding e to T, traversed in the direction of e. Relative to feasible tree solution X of MCFP, if e is a non-basic edge, then $x_e = 0$ or u_e. Let C(T,e) denote the unique circuit formed by adding a non-tree edge e to T, where C(T,e) is directed in the same direction as e. The unique vertex w as defined above is the INITIAL vertex of C(T,e).

If $x_e = 0$ and $C(T,e)$ is a flow augmenting circuit, then the direction of augmentation, has to be the same as the direction of $C(T,e)$. Likewise if $x_e = u_e$ and $C(T,e)$ is a flow augmenting circuit, then the direction of augmentation has to be in the opposite direction of $C(T,e)$. In the latter case the cost of the circuit in the direction of augmentation is $-\bar{c}_e$. From the calculation of node numbering , it is clear that if $e \in E(T)$, then $\bar{c}_e = 0$; therefore the following is equivalent to O.C.

<u>Lemma 1.5.1</u>. If X^o is a feasible tree solution of MCFP, with respect to tree T, then X^o is optimal if for any $e \notin E(T)$,

$$\text{Cost}[C(T,e)] = \bar{c}_e \begin{cases} \geq 0 & \text{if } x_e^o = 0 \\ \leq 0 & \text{if } x_e^o = u_e \end{cases}$$

This simple and elegant optimality condition suggests a specialization of Dantzig's simplex algorithm for solving MCFP. It is this specialized network simplex algorithm, which we will be discussing hereafter.

CHAPTER 2

NETWORK SIMPLEX METHOD

§ 2.1. PRELIMINARIES

The network simplex method, which is the specialization of the upper-bounded simplex method for MCFP, is mainly due to Dantzig [14], but it could be traced back to Hitchcock [28] and Koopmans [36]. The algorithm starts with a feasible tree solution X^0 and a vertex numbering Y^0 corresponding to a spanning tree T^0. It proceeds from spanning tree to spanning tree, each obtained from the previous one by the addition of a non-tree edge and deletion of a tree edge, ensuring the feasibility of the resulting tree solution, until a tree is obtained, such that its vertex numbering satisfies the optimality conditions of the previous chapter. Before proceeding with the formal description of the network simplex method, we discuss the following, which may be helpful in the understanding of the upcoming discussions.

Where S is a set and $T \subseteq S$, we denote $S \setminus T$ by $\bar{T}$. Where $P = (p_j : j \in J)$ is a real vector and $I \subseteq J$, we abbreviate $\sum(p_j : j \in I)$ to $P(I)$. Hereafter, we will assume that the digraph G associated with the given MCFP is connected, because otherwise, it is easily seen that the problem can be decomposed into two smaller problems of similar type.

Since from Remark 1.1.1, $\sum(b_v: v \in V) = 0$ is a necessary condition for MCFP to have a feasible solution, we shall assume that $\sum (b_v: v \in V) = 0$.

<u>Lemma 2.1.1</u>. For any $S \subseteq V$ and any solution X^o of $AX = b$, $b(S) = X^o(\delta(\bar{S})) - X^o(\delta(S))$.

<u>Proof</u>. If we sum up the rows of $AX^o = b$, corresponding to elements of S, for each $e \in \gamma(S)$, the variable x_e^o appears exactly twice on the left hand side of the summation, once with a negative sign and once with a positive sign, thus cancelling each other out; if $e \in \delta(\bar{S})$ $[\delta(S)]$, then x_e^o appears only once and its sign is positive [negative], because $h(e) \in S$ $[t(e) \in S]$ and the result follows immediately. Q.E.D.

Where T is a spanning tree of G, for $e \in E(T)$, let $R(T,e) = \{v \in V$: the unique path from root r of T to v in T does not contain e$\}$. Where X^o is a feasible tree solution of $AX = b$, $0 \leq x \leq u$, associated with tree T^o and $E' \subseteq E$ then let

$$u(E') = \{e: e \in E' \text{ and } x_e^o = u_e\} \quad \text{and}$$

$$0(E') = \{e: e \in E' \text{ and } x_e^o = 0\}.$$

The following is the immediate consequence of lemma 2.1.1.

Corollary 2.1.1. If X^0 is a feasible tree solution associated with tree T^0, then for any $f \in E(T^0)$,

$$x_f^0 = \sum[b_v : v \in R(T,f)] + \sum [u_e : e \in u(\delta(R(T,f)))]$$

$$- \sum[u_e : e \in u(\delta(\overline{R(T,f)}))]$$

when f is directed towards r in T. Similarly

$$x_f^0 = - \sum[b_v : v \in R(T,f)] - \sum[u_e : e \in u(\delta(R(T,f)))]$$

$$+ \sum[u_e : e \in u(\delta(R(T,f)))]$$

when f is directed away from r in T.

§2.2. NETWORK SIMPLEX METHOD

ALGORITHM 2.2.1

STEP 0 Begin with a feasible tree T^0, having an associated feasible tree solution X^0 and vertex numbering Y^0.

STEP 1 Let $E^1 = \{e : e \in 0(\overline{E(T^0)}) \text{ and } \bar{c}_e < 0\}$ and

$E^2 = \{e : e \in u(\overline{E(T^0)}) \text{ and } \bar{c}_e > 0\}$.

If $E^1 \cup E^2 = \phi$, stop; X^o is optimal. Otherwise, choose $e \in E^1 \cup E^2$; if $e \in E^1$, let direction of $C(T^o,e)$ be the same as the direction of e. Otherwise, let direction of $C(T^o,e)$ be opposite to the direction of e. Go to step 2.

<u>STEP 2</u> Let

$\theta_1 = \min\{x_j^o$: j a reverse edge of $C(T^o,e)\}$,

$\theta_2 = \min\{u_j - x_j^o$: j a forward edge of $C(T^o,e)\}$.

Let $\theta = \min\{\theta_1,\theta_2\}$.

Let $F = \{j$: j a reverse edge of $C(T^o,e)$ and $x_j^o = \theta\} \cup$

$\{j$:j a forward edge of $C(T^o,e)$ and $u_j - x_j^o = \theta\}$.

Define $X^1 = (x_j^1$: $j \in E)$ by

$$x_j^1 = \begin{cases} x_j^o + \theta & \text{j forward edge in } C(T^o,e), \\ x_j^o - \theta & \text{j reverse edge in } C(T^o,e), \\ x_j^o & \text{otherwise.} \end{cases}$$

Choose $f \in F$ and let $T^1 = (T^o \cup \{e\}) \setminus \{f\}$.

Calculate vertex numbering Y^1 with respect to T^1. Replace T^o, X^o and Y^o by T^1, X^1 and Y^1 respectively and go to step 1.

Correctness of the algorithm: From the choice of θ and definition of X^1 from X^0 in step 2, it is clear that X^1 is a feasible tree solution of MCFP. Also from the calculation of Y^1 with respect to T^1, it is obvious that, Y^1 and X^1, satisfy complementary slackness conditions. The stopping criteria of step 1, simply means that the present Y^0 is feasible to DMCFP, therefore X^0 and Y^0 are feasible to MCFP and DMCFP respectively and satisfy complementary slackness conditions; thus by theorem 0.2.4, X^0 and Y^0 are optimal to their respective problems.

From step 1 and step 2 of the algorithm it is clear that incoming edge e and outgoing edge f may not be unique; thus the algorithm may be refined so as to prescribe a specific choice of e in step 1 ("entering-edge rule") and of f in step 3 ("leaving-edge rule"). Following Bland [6], we refer to the algorithm with such refinements as "a network simplex method"; "the network simplex method" is the class of all such algorithms. Hereafter, one application of step 1 followed by one application of step 2, will be called a SIMPLEX PIVOT or simply a PIVOT . If the θ of step 2 is zero (non-zero), then the Pivot is said to be a DEGENERATE (NON-DEGENERATE) pivot.

Given an instance of MCFP having a feasible tree T^0, a tree T^1 is said to be a SUCCESSOR of T^0 if T^1 can be obtained from T^0 as in the algorithm. A SIMPLEX SEQUENCE is a sequence $T^1, T^2, \ldots$ such that T^1 is a feasible tree, and, for all i, T^{i+1} is a successor of T^i.

It follows from the definition of X^1 in step 2 that $CX^1 = CX^0 - \theta|\bar{c}_e|$. Thus the value of the objective function strictly decreases for each pivot for which $\theta \neq 0$. Since the number of feasible bases is finite, the algorithm is finite unless it encounters a degenerate subsequence of infinite length, because of repeated terms (cycling). In the absence of cycling, the algorithm could admit a degenerate sequence of exponential length in $|V|$ (stalling). In studying cycling and stalling, the following are helpful.

If T^0, X^0, Y^0, T^1, X^1, Y^1, e and f conform to the conditions in the algorithm, then the following lemmas are easily established.

<u>Lemma 2.2.1</u>.

$$y_v^1 = \begin{cases} y_v^0 & \text{if } v \in R(T,f), \\ y_v^0 + \bar{c}_e & \text{if } t(e) \in R(T,f) \text{ and } v \notin R(T,f), \\ y_v^0 - \bar{c}_e & \text{if } h(e) \in R(T,f) \text{ and } v \notin R(T,f). \end{cases}$$

i.e. the elements of $V^1 = V \backslash R(T,f)$ all increase or all decrease by $|\bar{c}_e|$.

<u>Corollary 2.2.1</u>. If $\bar{c}_e < 0$ and $t(e) \in R(T,f)$ or $\bar{c}_e > 0$ and $h(e) \in R(T,f)$ then $\sum (y_v^1 : v \in V) < \sum (y_v^0 : v \in V)$.

Where there is a possibility of confusion, we will denote $\bar{c}_e$ relative to tree T by $\bar{c}_e(T)$.

<u>Lemma 2.2.2</u>. Let $V^1 = V \setminus R(T,f)$; then:

(a) If for $v \varepsilon V^1$, $y_v^1 < y_v^0$ then

$$\bar{c}_e(T^1) \begin{cases} < \bar{c}_e(T^0) & \text{if } e \varepsilon \delta(V^1); \\ > \bar{c}_e(T^0) & \text{if } e \varepsilon \delta(V \setminus V^1); \\ = \bar{c}_e(T^0) & \text{otherwise.} \end{cases}$$

(b) If for $v \varepsilon V^1$, $y_v^1 > y_v^0$ then

$$\bar{c}_e(T^1) \begin{cases} > \bar{c}_e(T^0) & \text{if } e \varepsilon \delta(V^1); \\ < \bar{c}_e(T^0) & \text{if } e \varepsilon \delta(V \setminus V^1); \\ = \bar{c}_e(T^0) & \text{otherwise.} \end{cases}$$

§2.3 CYCLING

Although cycling is virtually unknown in practice, and although very few examples of general linear programming and/or network flow problems that cycle when solved by the LP simplex method or the network simplex method have been published in the literature, its potential occurrence is disturbing from a theoretical point of view. Here we mention some of the known examples of cycling, before presenting methods of preventing its occurrence. The first such example for the general linear programming problem is due to Hoffman [29], and Beale [5]

has given an example of cycling for the dual simplex method. Another such example , again due to Beale which appeared in Gass [24], is essentially a generalized network flow problem, i.e. an LP problem in which every column of the constraint matrix contains only two non-zero entries of opposite signs , and has cycle length of 6. Gassner [25], has published the first example of cycling for network flow problems. This example occurs in a "4 by 4 optimum assignment problem" and its cycle length is 12. A simpler example of cycling for the NF problem is due to Cunningham [11]. This example occurs on a graph with 3 vertices and 12 edges and its cycle length is also 12 but its basis size is only 2.

§2.4 PREVENTING CYCLING

In 1952, Charnes [9], developed a method of preventing cycling in LP problems, commonly known as Charnes' perturbation method. Later in 1954, Dantzig, Orden and Wolfe [15], published a " Lexicographically ordered" method, for preventing cycling in LP problems. This method is essentially the same as Charnes' perturbation method. Recently Bland [6], presented a proof that a certain simple and natural entering and leaving variable rule (pivoting rule) never leads to cycling.

The following method of preventing cycling in the network simplex method is due to Cunningham [10], [11] and it is recently shown [21], that its "natural" extension, prevents cycling in the generalized network flow problem. Because of repeated use of this method in the remainder of this paper, we will discuss it in some detail, but the missing proofs and details can be found in [10] and [11].

Let T^0 be a feasible tree and X^0 be its associated feasible tree solution.

<u>Definition 2.4.1</u>. T^0 is said to be STRONGLY FEASIBLE if each $e \in E(T^0)$ with $x_e^0 = 0$ is directed away from root r in T^0, and each $e \in E(T^0)$ with $x_e^0 = u_e$ is directed towards r in T^0. This simply means that the unique path in T^0 from r to every $v \in V$ is a flow augmenting path relative to X^0.

The algorithm 2.2.1 with the following two refinements will be called "strongly feasible network simplex method" or for short SFNSM.

<u>Refinement 1</u>. In step 0, let T^0 be a strongly feasible tree.

<u>Refinement 2</u>. In step 2, select $f \in F$ as follows:

Let w be the first common vertex in the paths from h(e) and t(e) to r. Choose f to be the first member of F encountered in traversing $C(T^0,e)$ beginning at w, where $C(T^0,e)$ is traversed in the direction of e if $x_e^0 = 0$ and in the direction opposite to e if $x_e^0 = u_e$.

Lemma 2.4.1. SFNSM encounters only strongly feasible trees.

Proof. See [10].

Theorem 2.4.1. SFNSM is finite.

Proof. See [10].

Corollary 2.4.1. If a strongly feasible tree T^1 is obtained from a strongly feasible tree T^0 by a degenerate pivot, then $y_v^1 \leq y_v^0$ for all $v \in V$, where Y^0 and Y^1 are vertex numbering of T^0 and T^1 respectively.

§2.5 STALLING

Although Bland's pivot rule and Cunningham's strongly feasible tree method prevent cycling and guarantee finiteness of the network simplex method, degeneracy nevertheless is a symptom of another unpleasant problem; that is to say, the simplex method may remains at the same solution for many pivots. This phenomenon is often called STALLING. Now the natural question to ask is, do any of the above refinements

prevent stalling? This question is answered negatively by Avis and Chvátal [2]. They use the example of Klee and Minty [34], to show that, the LP simplex method with Bland's rule can stall. Cunningham [11], has used Edmonds' [19] "network" example to show that the network simplex method stalls with both Bland's rule and the strongly feasible tree method. Also J.M. Tan (private communication) has used Zadeh's [46] example to show the exponential behaviour of the Bland rule. In the following example, Edmonds [19] shows the existence of a simplex sequence, which provides an easy proof of the exponential behaviour of the strongly feasible tree method.

Problem n (P_n) [Edmonds]; for $n \geq 0$, is a trans-shipment problem on the digraph $G = (V,E)$ of Fig. 2.5.1, where $|V| = 2n + 3$ and $|E| = 4n + 3$. All demands are zero and the costs are as shown in Fig. 2.5.1.

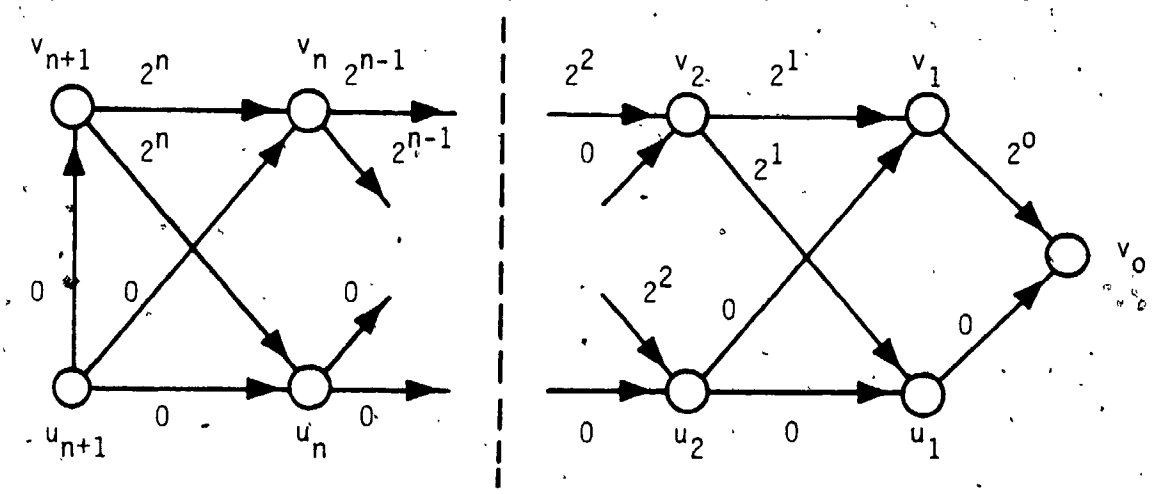
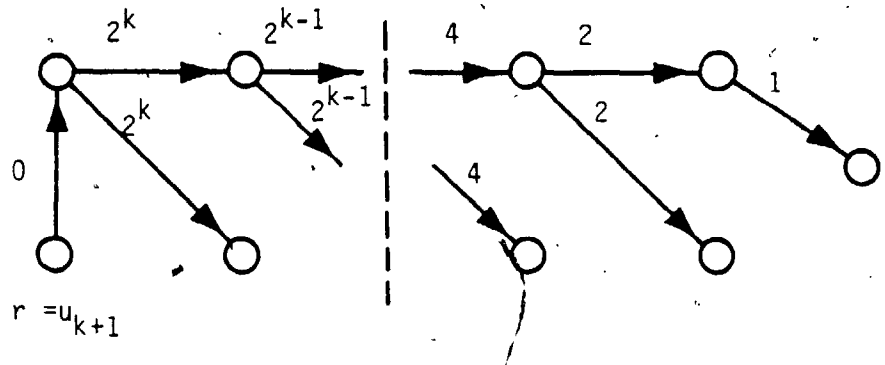


Fig. 2.5.1. Edmonds' Example of Stalling.

Lemma 2.5.1. For each integer $n \geq 0$, there exists a simplex sequence of strongly feasible trees having length at least 2^n for P_n.

Proof. For P_k, start with the following strongly feasible tree T_k^o rooted at u_{k+1}.

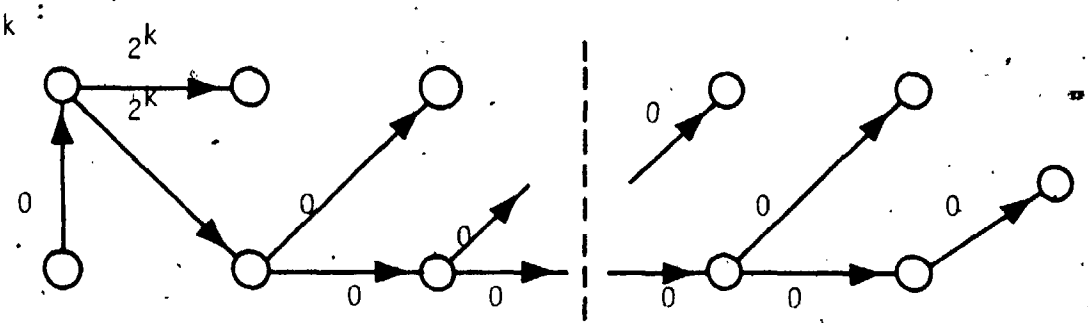
T_k^o:



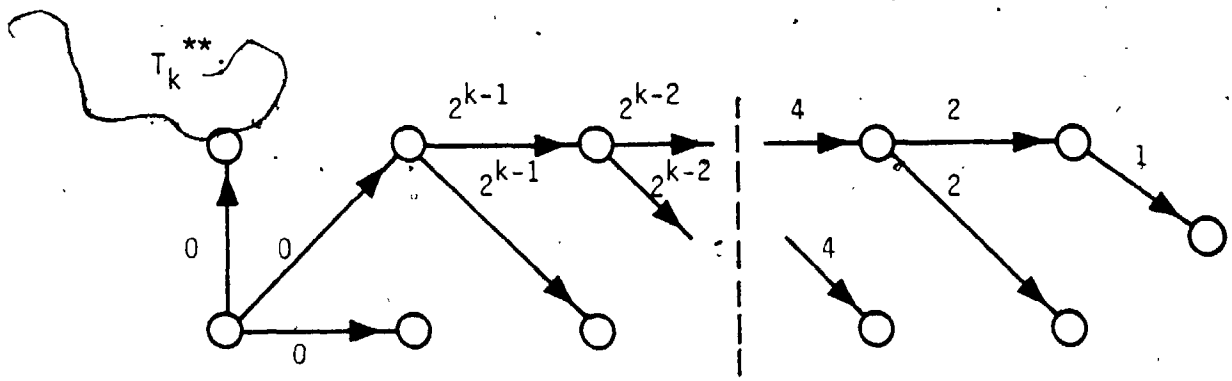
Let f(n) denote the total number of pivots to obtain the optimal tree T_n^{op} from T_n^o for P_n.

Observations. (1) Since the cost of edge (u_{k+1}, v_{k+1}) is zero and both edges (v_{k+1}, v_k) and (v_{k+1}, u_k) have costs 2^k, clearly, the same sequence of entering edges to obtain T_{k-1}^{op} from T_{k-1}^o can be used to obtain the following tree T_k^* from T_k^o for P_k [without deleting from the basis any of the three edges (u_{k+1}, v_{k+1}), (v_{k+1}, v_k) and (v_{k+1}, u_k)].

T_k^*:



(2) Now with 2k+1 pivots on the following edges (in sequence) (u_{k+1},v_k), (v_k,v_{k-1}), $(v_{k-1},v_{k-2}),\ldots,(v_2,v_1)$, (v_1,v_0), (v_2,u_1), $(v_3,u_2),\ldots,(v_k,u_{k-1})$, (u_{k+1},u_k) we build the tree T_k^{**} from T_k^{*}.

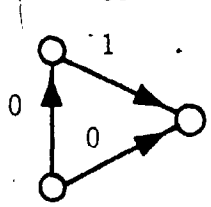


(3) Relative to T_k^{**} we have, $y_{u_k} = y_{v_k} = 0$; therefore once again the same sequence of entering edges to obtain T_{k-1}^{op} from T_{k-1}^{o} can be used to obtain T_k^{op} from T_k^{**}.

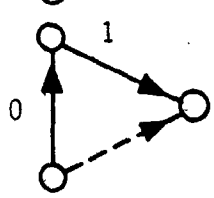
From the three above observations, we have $f(k) = 2f(k-1) + 2k + 1$. 2.5.1

This recursive relation suggests a proof by induction on n.

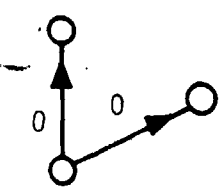
For n = 0, G_o:



T_o^{o}:

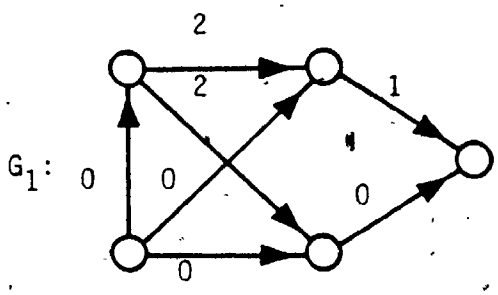


T_o^{op}:

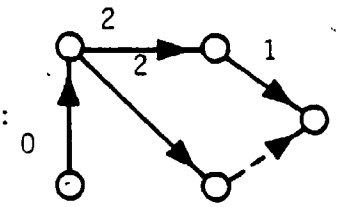


therefore $f(0) = 1 \geq 2^0$

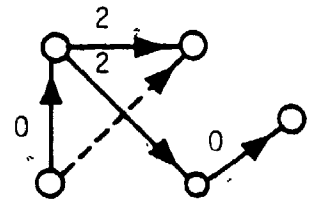
For n = 1, G_1:

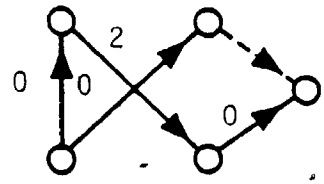


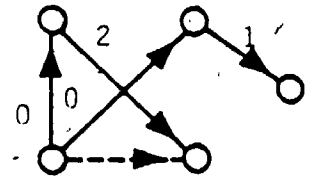
T_o^o:



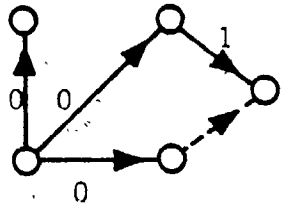
T_o^*:



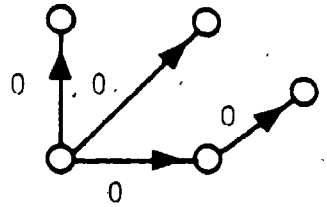




T_o^{**}:



T_o^{op}:



<u>Note</u>: Dotted lines indicate the entering edges.

$f(1) = 2f(0) + 2x1 + 1 = 5 \geq 2^1$.

<u>Induction hypothesis</u>. Suppose for some $k \geq 1$ it is true that $f(k) \geq 2^k$. We want to show that: for $n = k + 1$ it is also true that $f(k + 1) \geq 2^{k+1}$. But from 2.5.1 we have,

$$f(k+1) = 2f(k) + 2(k+1) + 1$$
$$\geq 2.2^k + 2(k+1) + 1 \qquad \text{[by induction hypothesis]}$$
$$\geq 2^{k+1} \qquad \text{and this completes the proof.}$$

At the end of this section it is worth noting that, unlike the example of cycling which consists of a trans-shipment problem and a simplex sequence for it, an "example" of stalling requires an infinite sequence of problems.

§2.6 PREVENTING STALLING

Avis and Chvátal [2], have shown that, if stalling could be prevented for the general linear programming simplex algorithm, then it becomes a "good" algorithm in the sense of Edmonds, i.e. the amount of work will be bounded above by a "fixed" polynomial in the problem size (i.e. number of constraints and number of variables). They do this by enlarging the given LP problem in such a way that the constant vector of the enlarged problem becomes zero which means that the new problem is "totally" degenerate. Unfortunately this transformation does not preserve the network structure of the NF problem. Thus preventing the stalling in the network simplex method will not make it "good" in the sense of Edmonds, but it is still important to be able to prevent such a disturbing phenomenon.

The results of this section are due to Cunningham [11]; for simplicity of discussion we will use the trans-shipment problem (TP) instead of the MCFP. Missing details, proofs and extension of results

to MCFP can be found in [11]. Let $T^0, T^1, T^2, \ldots$ be a simplex sequence for TP. Put $S(0) = 0$ and, where $k \geq 1$ and $S(k-1)$ is defined, let $S(k)$ be the least integer, if one exists, satisfying $\max(\bar{c}_e(T^i): S(k-1) \leq i \leq S(k)) \geq 0$ for all $e \in E$, and $S(k) > S(k-1)$. Otherwise $S(k)$ is undefined. If $S(k)$ exists, and $k \geq 1$ then STAGE k of the simplex sequence $T^0, T^1, T^2, \ldots$ is the simplex sequence $T^{S(k-1)+1}, T^{S(k-1)+2}, \ldots, T^{S(k)}$.

<u>Theorem 2.6.1</u>. A degenerate sequence $T^0, T^1, T^2, \ldots, T^m$ each of whose terms is a strongly feasible tree for X^0, can have at most k stages, where $k = |V| - 1 - |\{e \in E, x_e^0 > 0\}|$. (That is, where T is any tree for X^0, $k = |\{e \in E(T): x_e^0 = 0\}|$). Moreover, if it has k stages, then $S(k) = m$.

<u>Proof</u>. See [11].

This theorem suggests several easy entering edge selection rules to avoid stalling, namely;

<u>Rule 2.6.1. Least Recently Considered (LRC) Rule</u>. Let $e_1, e_2, \ldots, e_n$ be an arbitrary, fixed ordering of E. Where e_i was the last choice for entering edge, and the current tree is T, the next entering edge is the first element e of the list $e_{i+1}, e_{i+2}, \ldots, e_n, e_1, \ldots, e_i$ for which $\bar{c}_e(T) < 0$. This rule clearly ensures that each stage has length not exceeding $|E|$.

Rule 2.6.2. Least Recently Basic (LRB) Rule. This rule chooses among edges $e \notin E(T)$ such that $\bar{c}_e(T) < 0$, the edge that was least recently an edge of a tree.

Rule 2.6.3. Inward Most Negative (IMN) Rule. Let $v_1, v_2, \ldots, v_m$ be an arbitrary but fixed ordering of V. Let e be the edge which entered when T became the current feasible tree, and suppose that $h(e) = v_i$. Then IMN rule chooses e' as the next entering edge, where h(e') is the first vertex in the list $v_{i+1}, v_{i+2}, \ldots, v_m, v_1, \ldots, v_i$ which is the head of an edge f having $\bar{c}_f(T) < 0$, and $\bar{c}_{e'}(T) = \min(\bar{c}_f(T): h(f) = h(e'))$.

Rule 2.6.4. Altered Inward Most Negative (AIMN) Rule. This rule is the same as IMN, except that the list used is $v_i, v_{i+1}, \ldots, v_m, v_1, \ldots, v_{i-1}$.

Theorem 2.6.2. Let $T^0, T^1, \ldots, T^m$ be a degenerate sequence of strongly feasible trees, and let X^0 be the associated tree solution. Let $k = |V| - 1 - |\{e \in E: x_e^0 > 0\}|$; then:

(a) If the LRC rule has been used, then $m \leq k \cdot |E|$,

(b) If the LRB rule has been used, then $m \leq k \cdot |E|$,

(c) If the IMN rule has been used, then $m \leq k \cdot |V|$,

(d) If the AIMN rule has been used, then $m \leq k \cdot |V|$.

Proof. See [11].

CHAPTER 3

§3.1 THREE NEW PREVENTING STALLING RULES

In this chapter three new pivot rules are presented, each of which prevent stalling in the minimum cost flow problem (MCFP) when combined with the method of strongly feasible trees of [10]. The first of these rules follow from the preventing stalling method of Cunningham [11]. The other two rules use the strongly feasible tree method of [10]. As it will be seen in chapter 4, one of these rules can be refined so as to produce a "good" network simplex method for the optimum assignment problem.

Rule 3.1.1. Altered Outward Most Negative (AOMN) Rule. As in the AIMN rule let $v_1, v_2, \ldots, v_m$ be an arbitrary but fixed ordering of V. Let e be the edge which entered when T became the current feasible tree, and suppose that $t(e) = v_i$. Then the AOMN rule chooses e' as the next entering edge, where t(e') is the first vertex in the list, $v_i, v_{i+1}, \ldots, v_m, v_1, \ldots, v_{i-1}$ which is the tail of an edge f having $\bar{c}_f(T) < 0$ and $\bar{c}_{e'}(T) = \min(\bar{c}_f(T): t(f) = t(e'))$.

Theorem 3.1.1. Let $T^0, T^1, \ldots, T^m$ be a degenerate sequence of strongly feasible trees for the trans-shipment problem (TP), and let X^0 be the associated tree solution. Let $k = |V| - 1 - |\{e \in E: x_e^0 > 0\}|$. Then if the AOMN rule has been used, then $m \leq k \cdot |E|$.

Proof. Let "stage" be defined as in section 2.6. It follows from the strong feasibility of the trees and degeneracy of the pivots that, during each pivot, the vertex numbering of no vertex will increase. Therefore it is easily seen that, while the entering edges are successively outward edges relative to a fixed vertex, say v_i, the reduced cost of any edge e with $t(e) = v_i$ will either increase or remain unchanged. This is to say that while vertex v_i is being considered, there can be no more pivots than the number of outward edges reletive to v_i. But each edge has only one tail, or each edge is outward reletive to only one vertex; thus the length of each stage is at most $|E|$, and m-max $S(\ell) \leq |E|$, so the result follows from theorem 2.6.1. Q.E.D.

NOTE: The AOMN rule can be relaxed so as to choose e' such that $\bar{c}_{e'}(T) < 0$ and $t(e')$ is the first vertex in the list $v_i, v_{i+1}, \ldots, v_m, v_1, \ldots, v_{i-1}$. Obviously this relaxed version has the same theoretical bound as the AOMN rule itself; but in practice, the AOMN rule will generally have fewer pivots than its relaxed form.

In what follows the upper-bounded network simplex method for the MCFP is considered directly.

Where T^o is a feasible tree for MCFP and X^o is its associated tree solution, then, relative to a vertex v, an edge e is said to be INWARD if $x_e^o = 0$ and $h(e) = v$ or $x_e^o = u_e$ and $t(e) = v$. An OUTWARD edge relative to v is defined similarly. Where T is a tree $LEVEL_i$ (L_i) of T contains all the vertices v in V such that the unique path from v in T to root r has length i. For example L_o contains only the root r and L_1 contains all the vertices

which are adjacent to r and so on. The number of elements of L_i of a tree T will be denoted by $|L_i(T)|$. If $v \in L_i(T)$, then i is said to be the DEPTH of v and is denoted by $d_v(T) = i$. Following Barr, Glover and Klingman [4], we will define a non-tree edge e, relative to T to be an UP [a DOWN] edge if t(e) [h(e)] is in the unique path in T from h(e) [t(e)] to root r of T, and e will be called a CROSS edge if it is neither an up nor a down edge.

An edge $e \in E$ is said to be PIVOT ELIGIBLE if (a) $x_e^0 = 0$ and $\bar{c}_e(T^0) < 0$; or (b) $x_e^0 = u_e$ and $\bar{c}_e(T^0) > 0$. If $E' \subseteq E$ then $e \in E'$ is the MOST NEGATIVE REDUCED COST element of E' if $|\bar{c}_e(T^0)| = \max\{|\bar{c}_f(T^0)|: f \in E'$ and f is a pivot eligible edge$\}$.

For $V' \subseteq V$ let $OW(V') = \{e \in E$: e is an outward edge relative to some $v \in V'\}$ and let $ON(V') = \{e \in OW(V')$: e is a pivot eligible edge$\}$; and further, let OMN(V') denote the most negative reduced cost element of ON(V').

<u>Rule 3.1.2</u>. Where T is the initial feasible tree or a tree obtained from a non-degenerate pivot, then rule 3.1.2 applies starting at $L_0(T)$ and traverses the tree upwards by moving from L_i to L_{i+1} of the current tree T´ only if $ON(L_i(T´)) = \phi$, otherwise, it chooses the $OMN(L_i(T´))$ to enter the basis.

<u>Theorem 3.1.2</u>. Let $T^0, T^1, T^2, \ldots, T^m$ be a degenerate sequence of strongly feasible trees and let X^0 be the associated tree solution. If rule 3.1.2 has been used, then $m \leq |V|^2$.

<u>Proof.</u> To prove this theorem we need the following observations which are the generalization of the observations made by Barr, et.al. [4] for the optimum assignment problem.

Remark 3.1.1. Let T^0 be a strongly feasible tree and X^0 be its associated tree solution. Then if the entering edge e is an up edge with $x_e^0 = u_e$ or a down edge with $x_e^0 = 0$ then the pivot is necessarily non-degenerate.

Proof. The result is immediate from the definitions of strong feasibility and of the up [down] edge. Q.E.D.

Remark 3.1.2. If T' is a strongly feasible tree obtained from T by a degenerate pivot on an edge $e \in ON(\{u\})$ for some $u \in V$ and if $V' \subseteq V$ is the set of vertices the vertex numbering of which changed during the pivot; then:

(a) For all $v \in V \setminus V'$, $d_v(T') = d_v(T)$.

(b) If $d_u(T) = k$, then for all $v \in V'$, $d_v(T') > k$.

Proof. If e is a cross edge or a down edge such that $h(e) = u$ or an up edge such that $t(e) = u$ then the results follow from strong feasibility of T and the degeneracy of the pivot. On the other hand e cannot be a down edge with $t(e) = u$ or an up edge with $h(e) = u$ because otherwise by definition of an outward edge, in the former case $x_e = 0$ and in latter case $x_e = u_e$; thus in either case, by remark 3.1.1, the pivot will be non-degenerate, thereby contradicting the assumption on pivot degeneracy. Q.E.D.

Corollary 3.1.1. If T, T', u and e are defined as in remark 3.1.2 then for all $i \leq k$, $L_i(T') \subseteq L_i(T)$.

Proof. Immediate consequence of remark 3.1.2. Q.E.D.

<u>Lemma 3.1.1</u>. Let $T^0, T^1, \ldots, T^\ell$ be a degenerate sequence of strongly feasible trees and let X^0 be the associated tree solution. Then if for $i \geq 1$, T^i is obtained from T^{i-1} by pivoting on the $OMN(L_k(T^{i-1}))$ for some fixed $k \geq 0$, then;

(a) $\ell \leq |V|$ and

(b) If $k > 0$ and for all $j \leq k - 1$, $ON(L_j(T^0)) = \phi$, then for $i = 1, \ldots, \ell$, $ON(L_j(T^i)) = \phi$.

<u>Proof</u>. For $i \geq 1$, let $OMN(L_k(T^{i-1}))$ be denoted by e^{i-1} and let $V^i \subseteq V$ be the set of vertices the vertex numbering of which changed during the pivot on the edge e^{i-1} and for $i \geq 0$, let Y^i denote the vertex numbering of T^i. To prove (a), first we will show by induction on i that, (a1), for $i \geq 1$ if $e \in ON(L_k(T^i))$, then neither end of e can be an element of $\bigcup_{p=1}^{i} V^p$. For i =1, since the pivot on e^0 is degenerate; therefore from lemma 2.2.1, we have:

$$y_v^1 = \begin{cases} y_v^0 & \text{if } v \notin V^1, \\ y_v^0 - |\bar{c}_{e^0}(T^0)| & \text{if } v \in V^1. \end{cases} \qquad 3.1.1$$

Now let $f \in OW(L_k(T^1))$, then it follows from corollary 3.1.1 that $f \in OW(L_k(T^0))$. If $x_f^0 = 0$, $t(f) \in L_k(T^1)$ and $h(f) \in V^1$ then

$$\bar{c}_f(T^1) = \bar{c}_f(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[using 3.1.1]},$$

but if $\bar{c}_f(T^0) \geq 0$ then $\bar{c}_f(T^1) > 0$ and if $\bar{c}_f(T^0) < 0$ then

$$\bar{c}_f(T^1) = -|\bar{c}_f(T^0)| + |\bar{c}_{e^0}(T^0)|$$

$$\geq 0 \qquad \text{[because } e^0 = OMN(L_k(T^0))\text{]}.$$

Similarly, it can be shown that, if $x_f^0 = u_f$, $h(f) \in L_k(T^1)$ and $t(f) \in V^1$ then $\bar{c}_f(T^1) \leq 0$. The truth of (a1) follows from definition of an outward edge.

Now suppose that for some $i \geq 1$, it is true that if $e \in ON(L_k(T^i))$, then neither end of e belongs to $\bigcup_{p=1}^{i} V^p$. We need to demonstrate that if $e \in ON(L_k(T^{i+1}))$ then neither end of e belongs to $\bigcup_{p=1}^{i+1} V^p$. Since the pivot on the edge e^i is degenerate thus from lemma 2.2.1 we have :

$$y_v^{i+1} = \begin{cases} y_v^i & \text{if } v \notin V^{i+1}, \\ y_v^i - |\bar{c}_{e^i}(T^i)| & \text{if } v \in V^{i+1}. \end{cases} \qquad 3.1.2$$

Let $f \in OW(L_k(T^{i+1}))$; then from corollary 3.1.1 it is clear that $f \in OW(L_k(T^i))$. If $x_f^0 \neq 0$ and $t(f) \in L_k(T^{i+1})$ then, if $h(f) \in [\bigcup_{p=1}^{i} V^p] \setminus V^{i+1}$ then from 3.1.2 we have $\bar{c}_f(T^{i+1}) = \bar{c}_f(T^i)$ and from induction hypothesis $\bar{c}_f(T^i) \geq 0$. If $h(f) \in V^{i+1}$ then from 3.1.2 we have $\bar{c}_f(T^{i+1}) = \bar{c}_f(T^i) + |\bar{c}_{e^i}(T^i)|$ in which case if $\bar{c}_f(T^i) \geq 0$ then $\bar{c}_f(T^{i+1}) \geq 0$ otherwise $\bar{c}_f(T^i) = -|\bar{c}_f(T^i)|$ or $\bar{c}_f(T^{i+1}) = -|\bar{c}_f(T^i)| + |\bar{c}_{e^i}(T^i)| \geq 0$ [from the choice of e^i]. Similarly it is easily shown that if $x_f^0 = u_f$ and $h(f) \in L_k(T^{i+1})$ and $t(f) \in \bigcup_{p=1}^{i+1} V^p$ then $\bar{c}_f(T^{i+1}) \leq 0$. Now the result follows from these and the definition of outward edge .

To complete the proof of (a) note that for $i \geq 1$, (a1) together with the definition of V^i imply that, when e^i enters the tree T^i then $C(T^i, e^i)$ (i.e. the unique circuit formed by adding e^i to T^i)

does not contain any vertex from $\bigcup_{p=1}^{i} V^p$ and thus clearly $V^{i+1} \setminus [\bigcup_{p=1}^{i} V^p] \neq \phi$ or that $|\bigcup_{p=1}^{i+1} V^p| > |\bigcup_{p=1}^{i} V^p|$ and this together with (a1) imply (a).

(b) This follows inductively from corollary 3.1.1, by the outward edge definition and 3.1.2. Q.E.D.

Proof of Theorem 3.1.2. The result of this theorem follows directly from lemma 3.1.1 and the fact that, no tree can have more than $|V|$ levels.

THE MOST NEGATIVE (MN) RULE. The [network] simplex method with the MN rule always chooses the most negative reduced cost [edge] variable to enter the basis. If there is more than one candidate, then one such variable is arbitrarily chosen as the entering variable. This rule was first suggested by Dantzig [14] and is commonly known as Dantzig's pivot rule. If the most negative reduced cost element of a pre-specified subset of variables is to be chosen as the entering edge, then, we will refer to such a rule as the "Altered Most Negative (AMN)" rule. It is natural to assume that the theoretical and practical behaviour of the (network) simplex method with the AMN rule will be dependent on the choice of the subset of variables. Thus, we can correctly say that the AMN rule is a class of such rules. However, in what follows we will use the term "AMN rule" to describe a particular instance of this class.

Rule 3.1.3. Altered Most Negative (AMN) rule. The network simplex method with the AMN rule will choose the entering edges as follows:

S1: Let T be the initial or current feasible tree. Go to S2.

S2: Set $T^0 \leftarrow T$ and let $E_2^0 = \{e \in E$: e is a pivot eligible edge relative to $T^0\}$ and let $E_1^0 = E \setminus E_2^0$. Go to S3.

S3: Choose the most negative reduced cost element of E_2^0 to enter the basis, pivot and let T be the next tree. If the pivot is non-degenerate then go to S2. Otherwise go to S4.

S4: If no edge in E_1^0 is pivot eligible relative to the current tree T, go to S2. Otherwise go to S5.

S5: Choose the most negative reduced cost element of E_1^0 to enter the basis and let the resulting current tree be T. If the pivot is degenerate, then go to S4, otherwise go to S2.

Definition 3.1.1. Let a PERIOD of the strongly feasible network simplex method with the AMN rule be a simplex sequence $T^0, T^1, T^2, \ldots, T^k$ of strongly feasible trees, where T^0 is defined as in the AMN rule, T^1 is obtained from T^0 by a degenerate pivot on the most negative reduced cost element e^0 of E_2^0. For $1 \le i \le K-2$, T^{i+1} is obtained from T^i by a degenerate pivot on the most negative reduced cost element of E_1^0 (relative to T^i) and finally T^k is obtained from T^{k-1} by a pivot on the most negative reduced cost element e^{K-1} of E_1^0 (relative to T^{k-1}) and either the pivot was non-degenerate or the pivot was degenerate and no element of E_1^0 is pivot eligible relative to T^k.

Theorem 3.1.3. The length of a period of a strongly feasible network simplex method with the AMN rule is bounded above by $|V|$, i.e. $k - 1 \leq |V|$.

Proof. To prove this theorem we need the following lemmas.

Lemma 3.1.2. Let $E^1 \subseteq E$ and T be a feasible tree and let X and Y be its associated tree solution and vertex numbering respectively. If T^1 is obtained from T by a simplex pivot on the most negative reduced cost element e of E^1 and if X^1 and Y^1 are its corresponding tree solution and vertex numbering and $V^1 \subseteq V$ is the set of vertices such that, the vertex numbering of its elements changed during the pivot, then:

$$\text{if} \quad y_v^1 = \begin{cases} y_v & \text{if } v \notin V^1, \\ y_v - |\bar{c}_e(T)| & \text{otherwise,} \end{cases}$$

then

(i) $\bar{c}_f(T^1) \geq 0$ for all $f \in E^1 \cap \{O(\delta(V \setminus V^1))\}$;

(ii) $\bar{c}_f(T^1) \leq 0$ for all $f \in E^1 \cap \{u(\delta(V^1))\}$.

Proof. (i) Let $f \in E^1 \cap \{O(\delta(V \setminus V^1))\}$ then $h(f) \in V^1$ and

$$\bar{c}_f(T^1) = y^1_{t(f)} - y^1_{h(f)} + c_f$$

$$= y_{t(f)} - [y_{h(f)} - |\bar{c}_e(T)|] + c_f \qquad \text{[by hypothesis]}$$

$$= \bar{c}_f(T^0) + |\bar{c}_e(T)| \qquad 3.1.3$$

If $\bar{c}_f(T) \geq 0$ then from 3.1.3 we get $\bar{c}_f(T^1) \geq 0$. If $\bar{c}_f(T) < 0$ then $f \notin E(T)$ and thus $x_f = x_f^1 = 0$ which means that, relative to T, f was a pivot eligible edge. But from the choice of e we have $|\bar{c}_f(T)| \leq |\bar{c}_e(T)|$. However $\bar{c}_f(T) = -|\bar{c}_f(T)|$; therefore $-\bar{c}_f(T) \leq |\bar{c}_e(T)|$ or $|\bar{c}_e(T)| + \bar{c}_f(T) \geq 0$ and this latest inequality together with 3.1.3 implies that $\bar{c}_f(T^1) \geq 0$.

(ii) Let $f \in E^1 \cap \{u(\delta(V^1))\}$ then $t(f) \in V^1$ and

$$\bar{c}_f(T^1) = y^1_{t(f)} - y^1_{h(f)} + c_f$$

$$= [y_{t(f)} - |\bar{c}_e(T)|] - y_{h(f)} + c_f \qquad \text{[by hypothesis]}$$

$$= \bar{c}_f(T) - |\bar{c}_e(T)| \qquad 3.1.4$$

If $\bar{c}_f(T) \leq 0$ then from 3.1.4 we have $\bar{c}_f(T^1) \leq 0$. If $\bar{c}_f(T) > 0$ then f must have been a non-tree edge and thus $x_f = x_f^1 = u_f$ and therefore, relative to T, f has been a pivot eligible edge; but from the choice of e we have $\bar{c}_f(T) = |\bar{c}_f(T)| \leq |\bar{c}_e(T)|$ or $\bar{c}_f(T) - |\bar{c}_e(T)| \leq 0$; therefore this latest inequality together with 3.1.4 implies that $\bar{c}_f(T^1) \leq 0$.

<u>Lemma 3.1.3</u>. For some integer $k > 1$, let $T^0, T^1, \ldots, T^k$ be the sequence of strongly feasible trees in a period, where for $i = 1, \ldots, k$, T^i is obtained from T^{i-1} by pivoting on the edge e^{i-1}. Let V^i be the set of vertices, such that the vertex numbering of its elements changed during the pivot on edge e^{i-1}. Then, for all $i = 1, 2, \ldots, k - 1$

(a) $V^1, V^2, \ldots, V^i$ are mutually disjoint.

(b) if $e \in O\{[\gamma(\bigcup_{j=1}^{i} V^j)] \cap E_1^{\sigma}\}$ then $\bar{c}_e(T^i) \geq 0$;

and if $e \in u\{[\gamma(\bigcup_{j=1}^{i} V^j)] \cap E_1^{o}\}$ then $\bar{c}_e(T^i) \leq 0$.

(c) The entering edge e^i is such that either

$$x^o_{e^i} = 0,\ t(e^i) \in \bigcup_{j=1}^{i} V^j \text{ and } h(e^i) \notin \bigcup_{j=1}^{i} V^j \quad \text{or}$$

$$x^o_{e^i} = u_{e^i},\ h(e^i) \in \bigcup_{j=1}^{i} V^j \text{ and } t(e^i) \notin \bigcup_{j=1}^{i} V^j .$$

(d) $|\bar{c}_{e^{i-1}}(T^{i-1})| \geq |\bar{c}_{e^i}(T^i)|$.

<u>Proof</u>: For $i = 0,1,\ldots,k$, let Y^o be the associated vertex numbering of T^i. We will prove the lemma by mathematical induction on i. Suppose $i = 1$, in which case (a) is clearly true; since $k > 1$ therefore e^o has formed a degenerate pivot and:

$$y_v^1 = \begin{cases} y_v^o - |\bar{c}_{e^o}(T^o)| & \text{if } v \in V^1, \\ y_v^o & \text{otherwise.} \end{cases} \qquad 3.1.5$$

From 3.1.5 it is clear that for any $f \in [\gamma(V^1)] \cup [\gamma(V \setminus V^1)]$,

$$\bar{c}_f(T^0) = \bar{c}_f(T^1) \qquad 3.1.6$$

and thus (b) follows from 3.1.6. From the definition of E_2^0 and choice of e^0, it is clear that, e^0 is the most negative reduced cost element of E relative to T^0 and clearly (b) and lemma 3.1.2, imply (c).

To prove (d), there are two cases to consider.

<u>Case 1</u>. $x^0_{e^1} = 0$, $t(e^1) \in V^1$ and $h(e^1) \notin V^1$ and $\bar{c}_{e^1}(T^1) < 0$, in which case:

$$|\bar{c}_{e^1}(T^1)| = -\bar{c}_{e^1}(T^1) = -(y^1_{t(e^1)} - y^1_{h(e^1)} + c_{e^1})$$

$$= -[(y^0_{t(e^1)} - |\bar{c}_{e^0}(T^0)|) - y^1_{h(e^1)} + c_{e^1}],$$

[by 3.1.5]

$$= -\bar{c}_{e^1}(T^0) + |\bar{c}_{e^0}(T^0)|$$

$$\leq 0 + |\bar{c}_{e^0}(T^0)| \qquad [\text{since } e^1 \in E_1^0 \text{ and } x^0_{e^1} = 0].$$

<u>Case 2.</u> $x^o_{e^1} = u_e$, $h(e^1) \in V^1$ and $t(e^1) \notin V^1$ and $\bar{c}_{e^1}(T^1) > 0$, in which case:

$$|\bar{c}_{e^1}(T^1)| = \bar{c}_{e^1}(T^1) = y^1_{t(e^1)} - y_{h(e^1)} + c_{e^1}$$

$$= y^o_{t(e^1)} - (y_{h(e^1)} - |\bar{c}_{e^o}(T^o)|) + c_{e^1}, \text{ [by 3.1.5]}$$

$$= \bar{c}_{e^1}(T^o) + |\bar{c}_{e^o}(T^o)|$$

$$\leq 0 + |\bar{c}_{e^o}(T^o)| \qquad \text{[since } e^1 \in E_1^o \text{ and } x^o_{e^1} = u_{e^1}\text{]}$$

and this completes the proof of (d) for the case $i = 1$.

Induction hypothesis: suppose that for some $i = n$, (a), (b), (c) and (d) are true. We want to show that for $i = n + 1 \leq k - 1$, (a), (b), (c) and (d) are also true. From the induction hypothesis, it is clear that to prove (a), it is sufficient to show that $V^{n+1} \subseteq V \setminus \bigcup_{j=1}^{n} V^j$. But from the AMN rule, e^n must have formed a degenerate pivot and from part (c) of induction hypothesis, either $x^o_{e^n} = 0$, $t(e^n) \in \bigcup_{j=1}^{n} V^j$ and $h(e^n) \notin \bigcup_{j=1}^{n} V^j$ or $x^o_{e^n} = u_{e^n}$, $h(e^n) \in \bigcup_{j=1}^{n} V^j$ and $t(e^n) \notin \bigcup_{j=1}^{n} V^j$. In either case, if follows from the strong feasibility of the trees that the leaving edge belongs to $[\gamma(V \setminus \bigcup_{j=1}^{n} V^j)] \cap E(T^n)$; thus $V^{n+1} \subseteq V \setminus \bigcup_{j=1}^{n} V^j$, thereby completing the proof of (a).

From the degeneracey of the pivot on e^n, we have

$$y^{n+1}_v = \begin{cases} y^n_v - |\bar{c}_{e^n}(T^n)| & \text{if } v \in V^{n+1}, \\ y^n_v & \text{otherwise;} \end{cases} \qquad 3.1.7$$

From (a), since $V^1, V^2, \ldots, V^{n+1}$ are mutually disjoint

$$y_v^{n+1} = \begin{cases} y_v^o - |\bar{c}_{e^o}(T^o)| & \text{if } v \in V^1, \\ y_v^o - |\bar{c}_{e^1}(T^1)| & \text{if } v \in V^2, \\ \vdots & \vdots \\ y_v^o - |\bar{c}_{e^n}(T^n)| & \text{if } v \in V^{n+1}, \\ y_v^o & \text{if } v \notin \bigcup_{i=1}^{n+1} V^i. \end{cases} \qquad 3.1.8$$

It follows from 3.1.7, the induction hypothesis and lemma 3.1.2 that, to prove (b), we need only to consider the following two cases:

(i) If $f \in E_1^o$, $x_f^o = 0$ and $t(f) \in V^{n+1}$ and $h(f) \in V^i$, for some $i = 1,\ldots,n$, then:

$$\bar{c}_f(T^{n+1}) = y_{t(f)}^{n+1} - y_{h(f)}^{n+1} + c_f$$

$$= [y_{t(f)}^o - |\bar{c}_{e^n}(T^n)|] - [y_{h(f)}^o - |\bar{c}_{e^{i-1}}(T^{i-1})|] + c_f,$$

[from 3.1.8]

$$= \bar{c}_f(T^o) - |\bar{c}_{e^n}(T^n)| + |\bar{c}_{e^{i-1}}(T^{i-1})|$$

$$\geq 0 + |c_{e^{i-1}}(T^{i-1})| - |\bar{c}_{e^n}(T^n)| \quad [x_f^o = 0 \text{ and } f \in E_1^o]$$

$$\geq 0 \quad [\text{part (d) of induction hypothesis}].$$

(ii) If $f \in E_1^o$, $x_f^o = u_f$ and $h(f) \in V^{n+1}$ and $t(f) \in V^i$, for some $i = 1,2,\ldots,n$, then:

$$\bar{c}_f(T^{n+1}) = y^{n+1}_{t(f)} - y^{n+1}_{h(f)} + c_f$$

$$= [y^o_{t(f)} - |\bar{c}_{e^{i-1}}(T^{i-1})|] - [y^o_{h(f)} - |\bar{c}_{e^n}(T^n)|] + c_f ,$$

[from 3.1.8]

$$= \bar{c}_f(T^o) - |\bar{c}_{e^{i-1}}(T^{i-1})| + |\bar{c}_{e^n}(T^n)|$$

$$\leq 0 - |\bar{c}_{e^{i-1}}(T^{i-1})| + |\bar{c}_{e^n}(T^n)|, \quad [x_f^o = u_f \text{ and } f \in E_1^o]$$

$$\leq 0 , \quad \text{[from part (d) of induction hypothesis].}$$

and this completes the proof of (b).

Part (c) follows immediately from (b) and lemma 3.1.2. To prove (d), if neither end of e^{n+1} belongs to V^{n+1}, then from 3.1.7, we have $\bar{c}_{e^{n+1}}(T^{n+1}) = \bar{c}_{e^{n+1}}(T^n)$. Thus, from the choice of e^n we have $|\bar{c}_{e^n}(T^n)| \geq |\bar{c}_{e^{n+1}}(T^n)| = |\bar{c}_{e^{n+1}}(T^{n+1})|$. If $t(e^{n+1}) \in V^{n+1}$, then from (c), it must be that, $h(e^{n+1}) \in V\,[\bigcup_{i=1}^{n+1} V^i]$ and $x^o_{e^{n+1}} = 0$. Thus $\bar{c}_{e^{n+1}}(T^{n+1}) < 0$, therefore

$$|\bar{c}_{e^{n+1}}(T^{n+1})| = -\bar{c}_{e^{n+1}}(T^{n+1}) = -y^{n+1}_{t(e^{n+1})} + y^{n+1}_{h(e^{n+1})} - c_{e^{n+1}}$$

$$= -[y^{o}_{t(e^{n+1})} - |\bar{c}_{e^{n}}(T^{n})|] + y^{o}_{h(e^{n+1})} - c_{e^{n+1}},$$

[from 3.1.8]

$$= -\bar{c}_{e^{n+1}}(T^{o}) + |\bar{c}_{e^{n}}(T^{n})|$$

$$\leq 0 + |\bar{c}_{e^{n}}(T^{n})| , \qquad [e^{n+1} \in E_1^{o} \text{ and } x^{o}_{e^{n+1}} = 0]$$

If $h(e^{n+1}) \in V^{n+1}$, then from (c), it must be that, $t(e^{n+1}) \in V\backslash[\bigcup_{i=1}^{n+1} V^{i}]$ and $x^{o}_{e^{n+1}} = u_{e^{n+1}}$. Thus $\bar{c}_{e^{n+1}}(T^{n+1}) > 0$. Therefore,

$$|\bar{c}_{e^{n+1}}(T^{n+1})| = \bar{c}_{e^{n+1}}(T^{n+1}) = y^{n+1}_{t(e^{n+1})} - y^{n+1}_{h(e^{n+1})} + c_{e^{n+1}}$$

$$= y^{o}_{t(e^{n+1})} - [y^{o}_{h(e^{n+1})} - |\bar{c}_{e^{n}}(T^{n})|] + c_{e^{n+1}},$$

[from 3.1.8]

$$= \bar{c}_{e^{n+1}}(T^{n+1}) + |\bar{c}_{e^{n}}(T^{o})|$$

$$\leq 0 + |\bar{c}_{e^{n}}(T^{o})| , \qquad [e^{n+1} \in E_1^{o} \text{ and } x^{o}_{e^{n+1}} = u_{e^{n+1}}],$$

and this completes the proof of (d). Q.E.D.

Corollary 3.1.2. Let $T^0, T^1, \ldots, T^k$ be the sequence of strongly feasible trees in a period and let $e^0, e^1, \ldots, e^{k-1}$ be the entering edges chosen by the AMN rule as in lemma 3.1.3. Then for $i = 0,1,\ldots,k-2$, after e^i enters the tree then it remains a tree edge as long as the subsequent pivots in the period are degenerate and if e^{k-1} also forms a degenerate pivot, then none of the entering edges will be deleted during the period.

Proof. From the proof of part (a) of lemma 3.1.3. for i $1,\ldots,1-2$ during the pivot on e^i an element of $[\gamma(V \setminus \bigcup_{j=1}^{i} V^j)] \cap E(T^i)$ is deleted from T^i. Since from part (c) of lemma 3.1.3 one end of e^i is an element of $\bigcup_{j=1}^{i} V^j$; thus $e^i \in [\gamma(\bigcup_{j=1}^{i+1} V^j)] \cap E(T^{i+1})$, which means that during the subsequent degenerate pivots, e^i can not be deleted from the basis. Consequently if e^{k-1} does also form a degenerate pivot then e^i can not be deleted during the period. On the other hand, since e^0 forms a degenerate pivot, therefore it can not be the leaving edge when it enters the basis. But for $i=1,\ldots,k-1$, relative to T^i, e^0 has one end in $V^1 \subseteq \bigcup_{j=1}^{i} V^i$ and the other end in $V \setminus \bigcup_{j=1}^{i} V^j$ or in other words $e^0 \notin [\gamma(V \setminus \bigcup_{j=1}^{i} V^j)] \cap E(T^i)$; thus e^0 also can not be deleted during the subsequent degenerate pivots. If e^{k-1} does also form a degenerate pivot, e^0 can not be deleted during the period. Q.E.D.

Proof of theorem 3.1.3. If $k = 1$, then there is nothing to prove, otherwise the entering edges chosen by the AMN rule are the same as $e^0, e^1, \ldots, e^{k-1}$ of lemma 3.1.3 and by corollary 3.1.2 none of them will be deleted from the tree at least until e^{k-1} enters the tree T^{k-1}. But by lemma 0.1.4, $|E(T)| = |V| - 1$; therefore $k - 2 \leq |V| - 1$ or $k - 1 \leq |V|$. Q.E.D.

Theorem 3.1.4. Let $T^0, T^1, T^2, \ldots, T^m$ be a degenerate sequence of strongly feasible trees for the MCFP. If the AMN rule has been used, then $m \leq |V| \cdot (|E| - |V| + 1)$.

Proof. By theorem 3.1.3, there are at most $|V|$ pivots during one period of the strongly feasible network simplex method with the AMN rule where all but possibly the last pivot are degenerate. By corollary 3.1.2, if the last pivot of the period is also degenerate then at the end of the period, the total number of pivot eligible edges will decrease by at least one. Moreover, since with respect to any tree there can be at most $|E| - |V| + 1$ (i.e. the number of non-basic edges) pivot eligible edges, therefore $m \leq |V| \cdot (|E| - |V| + 1)$. Q.E.D.

The most efficient pivot rule in practice (at this time) is developed by Bradley et.al. [7], mulvey [39] and others, run roughly as follows:

relative to a tree T^0, scan the non-basic edges and select k most negative reduced cost edges and perform at most $k_1 \leq k$ pivots by choosing the entering edges from this set. Then rescan the edges relative to the last tree, to choose another such set and continue. The numbers k and k_1 are obtained by experiment and seem to be a small fraction of the $|E|$. The obvious and perhaps the main reason for the efficiency of this rule is that during the k_1 pivots very little time and effort is necessary to choose the entering edge and that the set E of edges is scanned only once for every k_1 pivots. Unfortunately the AMN rule lacks this property, that is, even though the number of pivot eligible edges in E_1^0 may not be big, however, to find the entering edge the algorithm must scan the set E_1^0 every time and since cardinality of

E_1^o is monotonically increasing the choice of the entering edge will be more and more costly towards the end of the algorithm. Thus unless otherwise a method is devised to find the most negative reduced cost element of E_1^o quickly, the AMN rule may not be very efficient in practice. But the primary value of AMN rule will be clear in the next chapter, where we will show that a slight modification of it will produce the first polynomial-bounded network simplex algorithm for the optimum assignment problem.

CHAPTER 4

OPTIMUM ASSIGNMENT (OA) PROBLEM

§4.1 PRELIMINARIES

The OA problem is a special case of the trans-shipment (TP) on a bipartite diagraph $G = (V,E)$ with bipartitions $V = (I,J)$; and $\delta(I) = E$ (i.e. each edge $e \in E$ is directed from an element of I to an element of J). For all $v \in I$, $b_v = -1$ and for all $v \in J$, $b_v = 1$ and we want an integer-valued (therefore 0-1 valued) optimal solution. From remark 1.1.1, $|I| = |J|$ is a necessary condition for the feasibility of the OA problem. If $|I| = |J| = n$ then the OA problem is called an n by n OA problem. It is well known that the coefficient matrix of a network flow problem is totally unimodular and it is also well known that if the coefficient matrix of the linear programming problem is totally unimodular and if the constant vector b and upper-bound vector u are integral-valued and if the problem has a feasible solution, then all its basic feasible solutions are integral-valued. Therefore the integrality requirement of the OA problem does not cause any difficulty at all. A (optimal) feasible solution of the OA problem is called a (optimal) PERFECT MATCHING [18]. The OA problem can be solved by any of the algorithms for the minimum cost flow problem. However

there are several specialized "good" algorithms to solve this problem. Most such algorithms are of "dual" nature, in that, each such algorithm starts with a dual feasible solution and at every intermediate step it preserves dual feasibility while improving the primal infeasibility. Kuhn's Hungarian method [37] is, perhaps, the best known in this category. There are several computational improvements to the Hungarian method, chiefly by Munkres [40] and Silver [43]. The other dual algorithms include that of Desler and Hakimi [16] and of Burkardt and Zimmerman [8]. Another "good" algorithm for the OA problem is due to Edmonds and Karp [20]. This algorithm uses Dikjstra's shortest path algorithm [17] and has a computational bound of $O(n^3)$. Recently Hung and Rom [31] gave another $O(n^3)$ algorithm for the OA problem which is essentially the same as Edmonds and Karp's algorithm. Engquist [22] has also developed a successive shortest path algorithm for the OA problem in which the shortest path subproblems are solved by a label correcting algorithm. Among the best known "primal" algorithms for the OA problem are that of Blanski and Gomory [3] and the network simplex method. The worst case computational bound of the former algorithm is $O(n^4)$. This algorithm is implemented in $O(n^3)$ by Cunningham and Marsh [47]. In the following sections we will show that with certain pivot rules, the network simplex method is also a polynomial bounded algorithm for the OA problem.

§4.2 NETWORK SIMPLEX METHOD FOR OA PROBLEM

Let T^0 be a feasible tree for an n by n OA problem and let X^0 be its associated tree solution. It is easily seen that X^0 is (0,1)-valued. Thus there is a perfect matching $M^0 \subseteq E(T^0)$ such that $x_e^0 = 1$ if and only if $e \in M^0$ and $x_e^0 = 0$, otherwise.

Since, $|I| = |J| = n$ therefore $|M^0| = n$. On the other hand $|E(T^0)| = 2n - 1$; thus roughly half of the edges of T^0 have zero flow. Hence the OA problem is a highly degenerate instance of the TP. Barr et al. [4] independently developed an alternating basis network simplex method for the OA problem which is a specialization of the strongly feasible network simplex method for the OA problem. They observed that in solving an OA problem with this specialized strongly feasible network simplex method certain simplifications occur which make the degenerate pivots easier to perform. These observations are as follows:

<u>Lemma 4.2.1</u>. If T^0 is strongly feasible and its "root" $r \in J$, then r is incident with just one edge of T^0, and that edge is an edge of M^0. Every other element v of J is incident to exactly 2 edges of T^0, and the edge of T^0 incident to v which is in the simple path from r to v in T^0 is not in M^0.
is incident to an element of M^0, therefore each $v \in J\backslash\{r\}$ is incident with exactly two elements of $E(T^0)$. Q.E.D.

<u>Lemma 4.2.2</u>. Let T^0 be a strongly feasible tree and let M^0 be the associated perfect matching. If the entering edge e is up or cross then the pivot is degenerate, and the leaving edge is that edge $f \in E(T^0)\backslash M^0$ such that $h(f) = h(e)$. If e is down, then the pivot is non-degenerate, and the leaving edge is $f \in M^0$ such that $h(f) = h(e)$.

In the remainder of this section we will show that the strongly feasible network simplex algorithm is not a "good" algorithm by showing (in the following example†) that, it could admit an exponential number of pivots among which an exponential number of them are non-degenerate.

PROBLEM n (P_n). For a given non-negative integer n, consider the OA problem on the network of Fig. 4.2.1, with $4(n + 2)$ vertices and $6n + 4$ edges with costs as shown in Fig. 4.2.1.

Theorem 4.2.1. The strongly feasible network simplex method, applied to problem n, admits an exponential number of combined degenerate and non-degenerate pivots.

Proof. We will prove the theorem by induction on n. Start with the strongly feasible tree of Fig. 4.2.2 and let I_n denote the total number of pivots for P_n; then it is easily seen that $I_0 = 1 \geq 2^0$ and $I_1 = 5 \geq 2^1$ [see Fig. 4.2.6]. Suppose that for some $n = k - 1 \geq 1$ we have $I_{k-1} \geq 2^{k-1}$; then we want to show that for $n = k$ we also have $I_k \geq 2^k$. In the initial strongly feasible tree of Fig. 4.2.1, the edges (v_{2k+1}, u_{2k}) and (v_{2+1}, u_{2k+1}) are oppositely directed and have the same costs; therefore their effect combined with zero cost edge (v_{2k}, u_{2k+1}) on any basis of P_k is exactly the same as the effect of zero cost edge (v_{2k}, u_{2k}) on the problem P_{k-1}; therefore P_{k-1} is implicitly embedded in P_k. Hence by induction hypothesis it will require I_{k-1} pivots to obtain the strongly feasible tree of Fig. 4.2.3 from the initial tree of Fig. 4.2.2. Now by pivoting

(†) This example is obtained from Edmonds' [19] shortest path example of stalling by converting it to an optimum assignment problem. See Lawler [38], page 186 for more details.

on the following 2k + 1 edges in the order, $(v_{2(k+1)},u_{2k})$, $(v_{2k-1},u_{2(k-1)})$, $(v_{2k-3},u_{2(k-2)}),\ldots,(v_5,u_4),(v_3,u_2),(v_1,u_1),(v_3,u_3),((v_5,u_5),\ldots,$ $(v_{2k-3},u_{2k-3}),(v_{2k-1},u_{2k-1}),(v_{2(k+1)},u_{2k+1})$ (where all but one of the pivots, namely the pivot on the edge (v_1,u_1), are degenerate), we obtain the strongly feasible tree of Fig. 4.2.4 from the strongly feasible tree of Fig. 4.2.3. Since the edges $(v_{2(k+1)},u_{2k})$, $(v_{2(k+1)},u_{2k+1})$ and (v_{2k},u_{2k+1}) have zero costs, therefore once again the P_{k-1} is embedded in P_k and by induction hypothesis I_{k-1} pivots are needed to obtain the optimal tree of Fig. 4.2.5 from the strongly feasible tree of Fig. 4.2.4, thus $I_k = 2\ I_{k-1} + 2k+1 \geq 2\cdot 2^{k-1} + 2k + 1 \geq 2^k$. Q.E.D.

<u>Corollary 4.2.1</u>. Among the I_n simplex pivots for P_n, exactly $2^{n+1} - 1$ pivots are non-degenerate.

<u>Proof</u>. (Proof by induction on n).

Let f(n) denote the total number of non-degenerate pivots for P_n, then clearly;

$$f(o) = 1 = 2^{0+1} - 1$$

$$f(1) = 3 = 2^{1+1} - 1 \qquad \text{[see Fig. 4.2.6]}$$

Now suppose that for some $n \geq 2$ we have $f(n-1) = 2^{(n-1)+1} - 1 = 2^n - 1$; then we want to show that $f(n) = 2^{n+1} - 1$. From the proof of exponentiality of the total number of pivots for P_n we have:

$I_n = 2I_{n-1} + 2n + 1$, where only one of the 2n + 1 intermediate pivots is a non-degenerate pivot, therefore:

$$
\begin{aligned}
f(n) &= 2f(n-1) + 1 \\
&= 2(2^n - 1) + 1 && \text{[by induction hypothesis]} \\
&= 2^{n+1} - 1 && \text{Q.E.D.}
\end{aligned}
$$

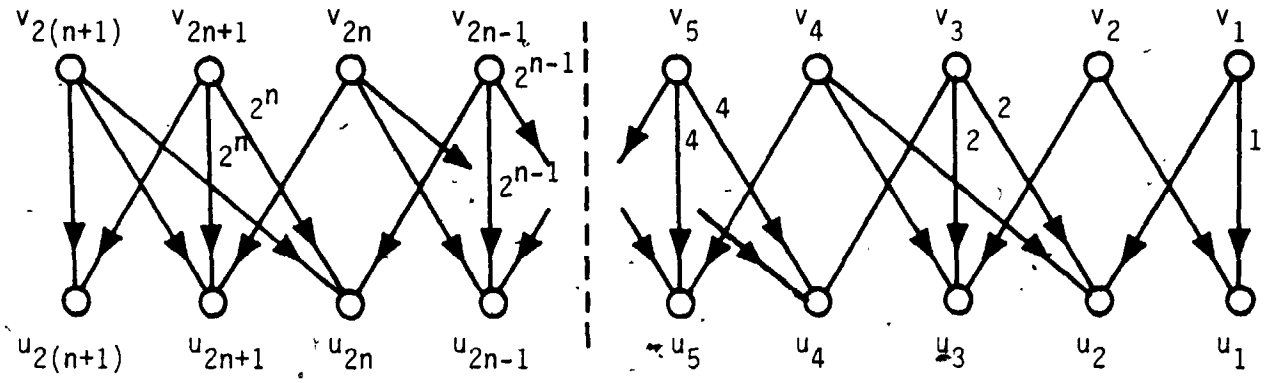


Fig. 4.2.1. Problem n. (numbers beside the edges represent the cost of the edges and default cost values are zero).

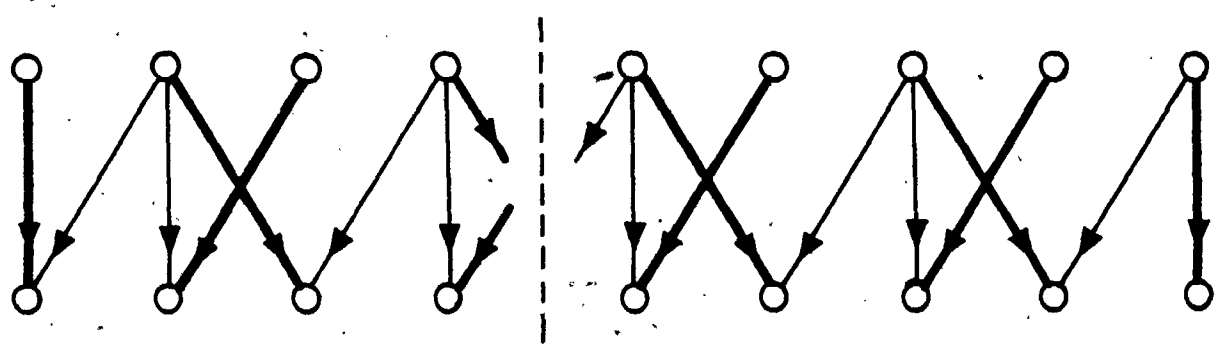


Fig. 4.2.2. Initial strongly feasible tree for problem k with $r = u_1$.

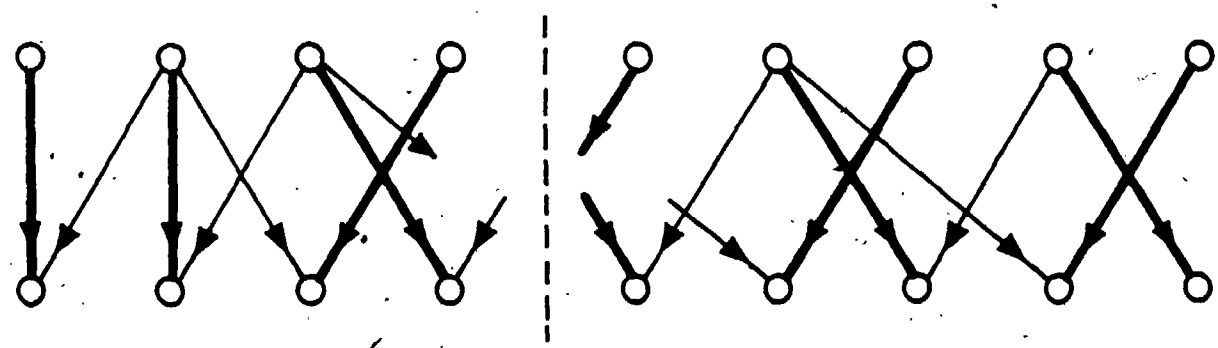


Fig. 4.2.3.

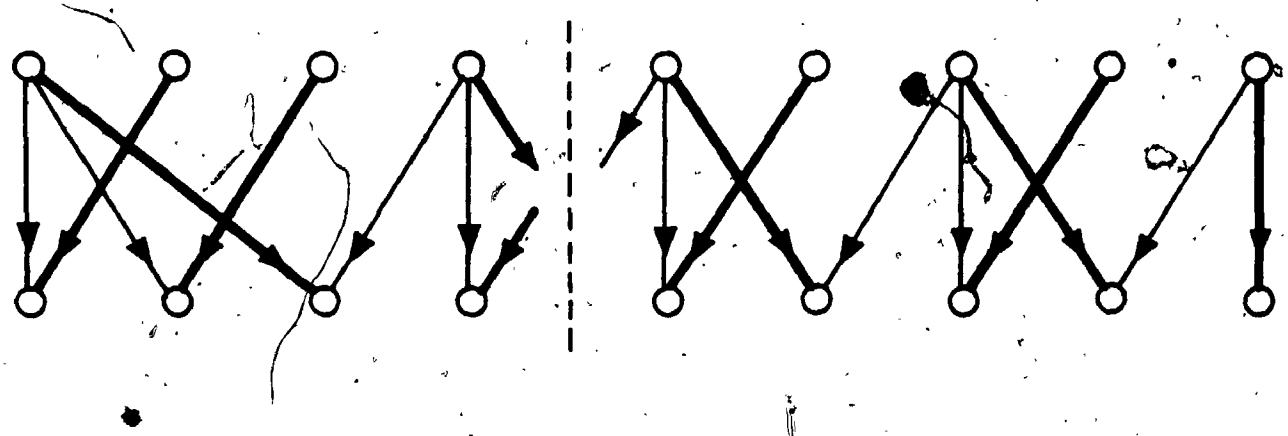


Fig. 4.2.4.

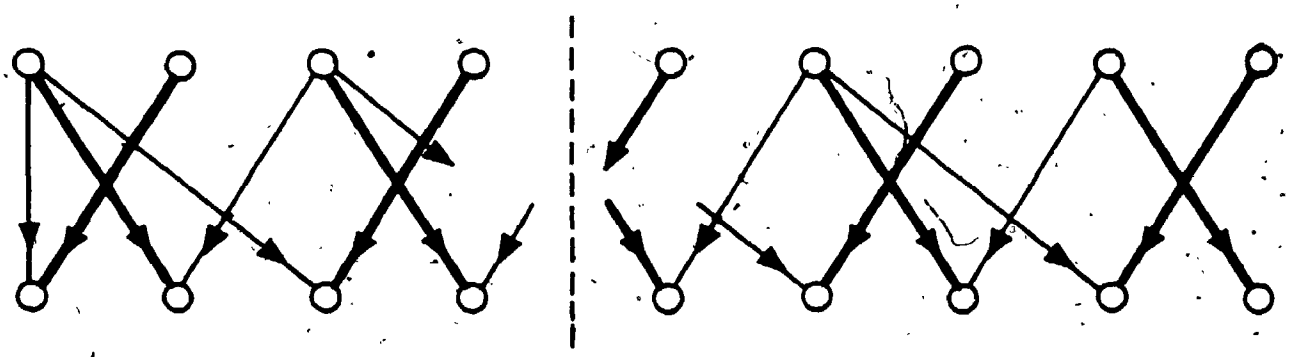


Fig. 4.2.5. Optimal tree

Thick edges represent the matching edges.

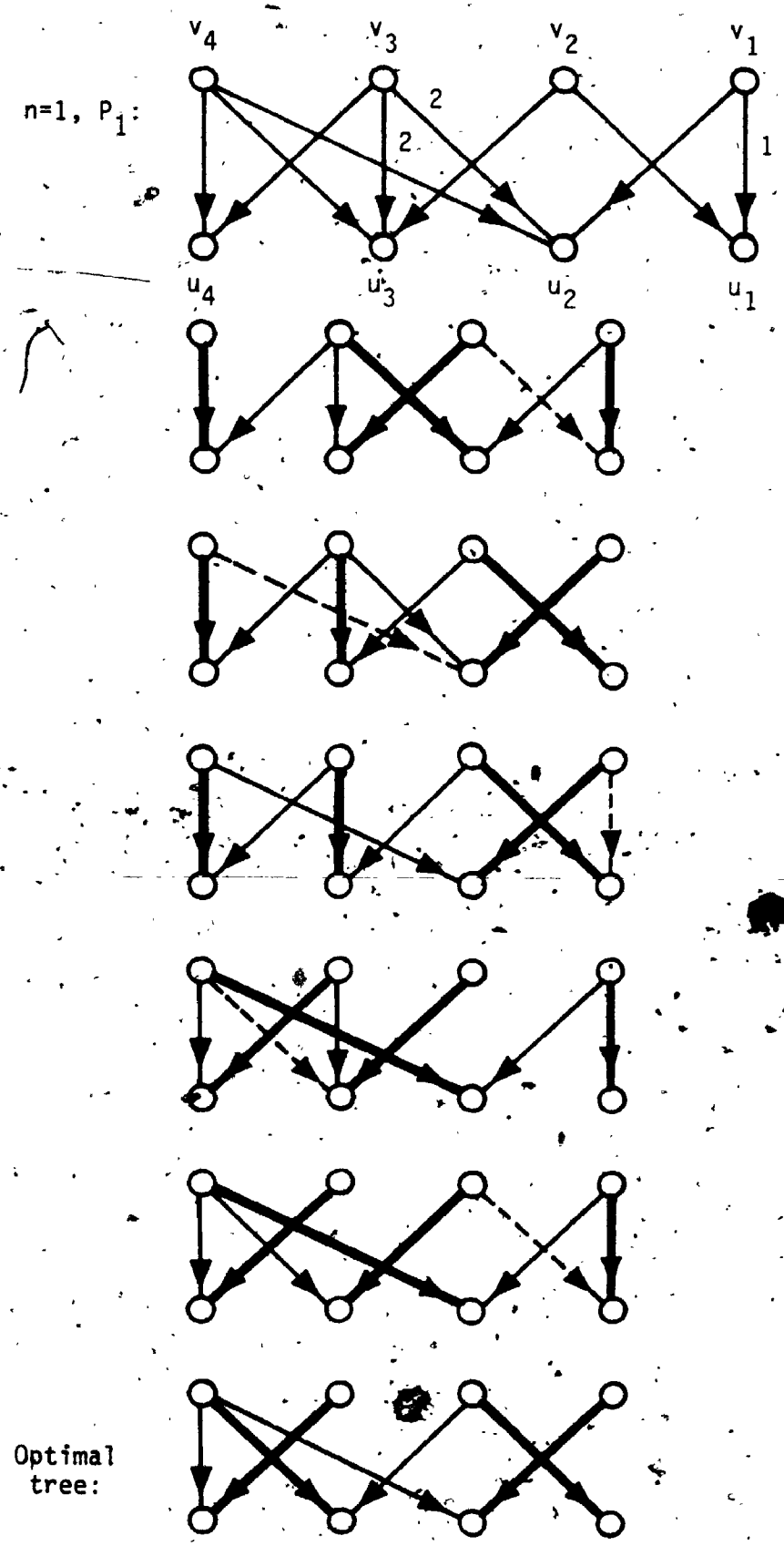


Fig. 4.6.2. Thick edges represent the matching edges and the dotted edge represents the entering edge.

REFINEMENTS OF NETWORK SIMPLEX METHOD FOR OA PROBLEM

In chapters 2 and 3 we have dealt with the degenerate network simplex pivots and resolved the questions of cycling and stalling, but to develop a "good" network simplex method, we must refine the algorithm so as to provide a "good" bound on the total number of non-degenerate pivots and prevent cycling and stalling at the same time. In the following two sections we will give two such refinements to the network simplex method for the OA problem.

§4.3 FIRST REFINEMENT

The following refinment is essentially the scaling method of Edmonds and Karp [20]; however unlike Edmonds and Karp, scaling here is done on the costs rather than capacities and is carried out implicitly.

Consider an n by n OA problem with integral cost vector C and let L be the least positive integer such that $\max\{|c_e| : e \in E\} \leq 2^L$. For $k \leq L$, let period k of the network simplex method be the simplex sequence $T^{k_0}, T^{k_1}, \ldots, T^{k_s}$ such that (i) For $k = L$, T^{k_0} is the initial feasible tree and for $k < L$, T^{k_0} is such that, for all $e \in E$, $\bar{c}_e(T^{k_0}) > -2^{k+1}$; (ii) For $k \leq L$, T^{k_s} is such that, for all $e \in E$, $\bar{c}_e(T^{k_s}) > -2^k$; (iii) For $i = 0,1,\ldots,s$, $T^{k_{i+1}}$ is obtained from T^{k_i} by pivoting on an edge e such that $\bar{c}_e(T^{k_i}) \leq -2^k$. In what follows, if there is no ambiguity, then in period k we will drop "k" from the super-scripts (e.g. T^0 would mean T^{k_0} and etc.).

<u>Theorem 4.3.1</u>. For $k \leq L$, the total number of non-degenerate pivots during period k cannot exceed 2n.

Proof. Let $C(\overline{M})$ denote the cost of a perfect matching $\overline{M}$. Then it is clear that $-n\cdot 2^L \leq C(\overline{M}) \leq n\cdot 2^L$ 4.3.1

If M^* is an optimal perfect matching and $\overline{M}$ is any perfect matching, then from 4.3.1 we get; $C(\overline{M}) - C(M^*) \leq 2n\cdot 2^L$ or

$$C(\overline{M}) - 2n\cdot 2^L \leq C(M^*) \qquad 4.3.2$$

But 4.3.2 simply means that $C(\overline{M}) - 2n\cdot 2^L$ is a lower bound on the cost of an optimal perfect matching and since each non-degenerate pivot improves the objective value by an amount equal to the reduced cost of the incoming edge, therefore, if $\overline{M}$ is the initial perfect matching and $k = L$, then every non-degenerate pivot of period L will decrease the objective value by at least 2^L. Thus, from 4.3.2, after at most $2n$ such non-degenerate pivots, period $k = L$ must terminate. Now for $k \leq L - 1$ let $T^0, T^1, \ldots, T^s$ be the simplex sequence of period k; then by the definition of period k, for

$$e \in E,\ \overline{c}_e(T^0) > -2^{k+1} \qquad 4.3.3$$

and if M^0 is the perfect matching associated with T^0 and M is any improved perfect matching, then:

$$\begin{aligned} C(M) &= C(M^0) + \sum[\overline{c}_e(T^0)\cdot x_e : e \in E\setminus E(T^0)] \\ &> C(M^0) - 2^{k+1}\cdot \sum[x_e : e \in E\setminus E(T^0)] && \text{[from 4.3.3]} \\ &\geq C(M^0) - n\cdot 2^{k+1} && \text{[because } |M| = n\text{]} ; \end{aligned}$$

therefore $C(M^0) - 2n \cdot 2^k$ is a lower bound on any improved perfect matching; thus by an argument similar to that of case $k = L$, during period k, we can have at most $2n$ non-degenerate pivots. Q.E.D.

PREVENTING CYCLING AND STALLING DURING PERIOD k.

Since maintenance of strong feasibility is independent of the choice of incoming edge, cycling in period k can be prevented by maintaining strongly feasible trees.

Now for some integer $k \geq 0$, let period k of a strongly feasible network simplex method for the trans-shipment problem be defined similarly to the period k of the OA problem. Then by imitating Cunningham [11], we will show that the pivot rules of §2.6 to prevent stalling can be refined slightly so as to prevent stalling in period k.

For a TP let L and period k for $k \leq L$, be defined as for the OA problem and let $T^0, T^1, T^2, \ldots$ be a strongly feasible simplex sequence in period k (i.e. for all $e \in E$, $\bar{c}_e(T^0) > -2^{k+1}$ and for $i \geq 0$; T^{i+1} is obtained from T^i by a pivot on an edge e such that $\bar{c}_e(T^i) \leq -2^k$).
Put $S(0) = 0$, and where $P \geq 1$ and $S(P-1)$ is defined, let $S(P)$ be the least integer, if one exists, satisfying $\max\{\bar{c}_e(T^i): S(P-1) \leq i \leq S(P)\} > -2^k$, for all $e \in E$ and $S(P) > S(P-1)$; otherwise $S(P)$ is undefined. If $S(P)$ exists and $P \geq 1$, then stage P of period k is defined as the simplex sequence $T^{S(P-1)+1}, T^{S(P-1)+2}, \ldots, T^{S(P)}$.

Remark 4.3.1. Let T and T' be strongly feasible trees in period k such that T' is obtained from T by a degenerate pivot. Then either $y_v(T') = y_v(T)$ or $y_v(T') \leq y_v(T) - 2^k$.

Proof. This follows from lemma 2.2.1 and the definition of period k.

Theorem 4.3.2. A degenerate sequence $T^0, T^1, T^2, \ldots, T^m$ in period k each term of which is a strongly feasible tree for X^0, can have at most P stages, where $P = |\{e \in E(T^0): x_e^0 = 0\}|$; furthermore if it has P stages then $S(P) = m$.

Proof. Let $G(X^0)$ be a spanning subgraph of G having edge-set $\{e \in E: x_e^0 > 0\}$; then if v, w are vertices of the same component of $G(X^0)$, then from lemma 2.2.1, for all i and j we have:

$$y_v(T^i) - y_w(T^i) = y_v(T^j) - y_w(T^j).$$

Since any path in T^m from root r to $v \in V$ can have at most P zero edges, therefore the truth of the theorem will be established by showing that $y_v(T^{S(i)}) = y_v^m$ for every $v \in V$ such that the path in T^m from root r to v has at most i zero edges, provided S(i) is defined. This is clearly true for i = 0, since in this case the only such vertices v are those belonging to the component of $G(X^0)$ which contains r. Assume that for some $n \geq 1$, it is true for all $i < n$. We want to show that it is also true for i = n. To show this, assume the contrary. This means that, for some v in V, $y_v(T^{S(n)}) \neq y_v(T^m)$. But $y_v(T^m)$ is the cost of the path Q from root r to v in T^m and since the trees are strongly feasible and the pivots are degenerate, thus from corollary 2.4.1 we have $y_v(T^m) < y_v(T^{S(n)})$, and from remark 4.3.1, we get

$y_v(T^m) \leq y_v(T^{S(n)}) - 2^k$. Let e be the last zero edge of Q and Q' be the subpath of Q from r to t(e). Then by induction hypothesis, since Q' has at most n-1 zero edges, $y_{t(e)}(T^{S(n-1)}) = y_{t(e)}(T^m)$. 4.3.4

Also, since v and h(e) are in the same component of $G(X^o)$, $y_{h(e)}(T^m) < y_{h(e)}(T^{S(n)})$ or from remark 4.3.1, $y_{h(e)}(T^m) \leq y_{h(e)}(T^{S(n)}) - 2^k$

or $y_{h(e)}(T^{S(n)}) + 2^k \leq y_{h(e)}(T^{S(n)})$. 4.3.5

It follows that for any P such that $S(n-1) \leq P \leq S(n)$,

$$\begin{aligned}
\bar{c}_e(T^P) &= y_{t(e)}(T^P) - y_{h(e)}(T^P) + c_e \\
&\leq y_{t(e)}(T^m) - y_{h(e)}(T^{S(n)}) + c_e && \text{[from 4.3.4]} \\
&\leq y_{t(e)}(T^m) - y_{h(e)}(T^m) - 2^k + c_e && \text{[from 4.3.5]} \\
&= -2^k && [e \in E(T^m)]
\end{aligned}$$

which contradicts the definition of stage n. Therefore it is true for i = n, and so it is true in general by induction. This completes the proof of the theorem. Q.E.D.

Theorem 4.3.2, suggests several simple entering edge rules to prevent stalling in period k of the algorithm, namely:

<u>Rule 4.3.1. Refined Least Recently Considered (RLRC) Rule</u>. Let $e_1, e_2, \ldots, e_n$ be an arbitrary, fixed ordering of E. In period k of the algorithm, where e_i was the last choice for the entering edge, and the current tree is T, the next entering edge is the first element e of the list $e_{i+1}, e_{i+2}, \ldots, e_n, e_1, \ldots, e_i$ for which $\bar{c}_e(T) \leq -2^k$. This rule

clearly ensures that each stage of period k has length not exceeding $|E|$.

Rule 4.3.2. Refined Least Recently Basic (RLRB) Rule. In period k of the algorithm this rule chooses among non-tree edges e such that $\bar{c}_e(T) \leq -2^k$, the edge that was least recently an edge of a tree.

Rule 4.3.3. Refined Inward Most Negative (RIMN) Rule. Let $v_1, v_2, \ldots, v_m$ be an arbitrary but fixed ordering of V. In period k of the algorithm, let e be an edge which entered when T became the current feasible tree, and suppose that $h(e)=v_i$. Then RIMN rule chooses e' as the next entering edge, where h(e') is the first vertex in the list $v_{i+1}, v_{i+2}, \ldots, v_m, v_1, \ldots, v_i$ which is the head of an edge f having $\bar{c}_f(T) \leq -2^k$, and $\bar{c}_{e'}(T) = \min\{\bar{c}_f(T): h(f)=h(e')\}$.

Rule 4.3.4. Refined Altered Inward Most Negative (RAIMN) Rule. This rule is the same as RIMN rule, except that the list used is $v_i, v_{i+1}, v_{i+2}, \ldots, v_m, v_1, \ldots, v_{i-1}$.

Lemma 4.3.1. Suppose that a degenerate pivot takes strongly feasible tree T to strongly tree T' and suppose that the associated entering edge e satisfies $\bar{c}_e(T) = \min. \{\bar{c}_f(T): h(f) = h(e)\} \leq -2^k$, for some integer $k > 0$. If there exists an edge f having $h(f) = h(e)$ and $\bar{c}_e(T') \leq -2^k$, then entering f to T' will not yield a degenerate pivot.

Proof. Let $V' \subseteq V$ be the set of vertices, for which, the vertex numbering of its elements changed during the pivot on e. Then from the strong feasibility of T and lemma 2.2.1 we have

$$y'_v = \begin{cases} y_v + \bar{c}_e(T) & \text{if } v \in V', \\ y_v & \text{otherwise.} \end{cases} \qquad (*)$$

If $t(f) \in V \setminus V'$, then

$$c_f(T') = y'_{t(f)} - y'_{h(f)} + c_f$$

$$= y_{t(f)} - y_{h(f)} - \bar{c}_e(T) + c_f \qquad \text{[from (*)]}$$

$$= \bar{c}_f(T) - \bar{c}_e(T) \geq 0. \qquad \text{[Since } \bar{c}_f(T) \geq \bar{c}_e(T)\text{]}$$

It follows that $t(f) \in V'$, in which case from strong feasibility of T' the unique path in T' from $h(e)$ to $t(e)$ is a flow augmenting path; thus the unique circuit framed by adding f to T' is a flow augmenting circuit or in other words the pivot is non-degenerate. Q.E.D.

<u>Theorem 4.3.3</u>. Let $T^0, T^1, T^2, \ldots, T^m$ be a degenerate sequence of strongly feasible trees in period k, and let X^0 be the associated tree solution. Let $P = \{|e \in E(T^0): x_e^0 = 0\}|$. Then

(i) If the RLRC rule has been used, then $m \leq P.|E|$;

(ii) If the RLRB rule has been used, then $m \leq P.|E|$;

(iii) If the RIMN rule has been used, then $m \leq P.|V|$;

(iv) If the RAIMN rule has been used, then $m \leq P.|V|$.

<u>Proof</u>. It is not difficult to see that both RLRC and RLRB rules have the property that the length of each stage during a period can not exceed $|E|$; thus the validity of (i) and (ii) follow from theorem 4.3.2. To prove (iv) note from lemma 4.3.1 that ,after e enters to foem a tree T^i, either the next pivot is non-degenerate or that for any f with $h(f)=h(e)$, $\bar{c}_e(T^i) \geq 0$. Thus the length of each stage of any period with RAIMN rule is $\leq |V|$, so the validity of (iv) also follows from theorem 4.3.2.

To prove (iii), we prove by induction on i, that if the path Q in T^m from r to v has at most i zero edges and $i \cdot |V| \leq m$, then $y_v(T^{i \cdot |V|}) = y_v(T^m)$. This is clearly true for i = 0. Assume that for some $n \geq 1$, it is true for all $i < n$. If it is not true for i = n, then there exists $v \in V$ such that the path Q in T^m from r to v has n zero edges and $y_v(T^{n \cdot |V|}) > y_v(T^m)$ or from remark 4.3.1,

$$y_v(T^{n \cdot |V|}) - 2^k \geq y_v(T^m).$$

Let f be the last zero edge of Q; then by induction hypothesis

$$y_{t(f)}(T^{(n-1) \cdot |V|}) = y_{t(f)}(T^m).$$

Moreover,

$$y_{h(f)}(T^{n \cdot |V|}) > y_{h(f)}(T^m) \qquad 4.3.6$$

or from remark 4.3.1 $y_{h(f)}(T^{n \cdot |V|}) - 2^k \geq y_{h(f)}(T^m)$.

or $$y_{h(f)}(T^{n \cdot |V|}) \geq y_{h(f)}(T^m) + 2^k. \qquad 4.3.7$$

It follows that, for all q such that $(n-1) \cdot |V| \leq q < n \cdot |V|$

$$\bar{c}_f(T^q) = c_f + y_{t(f)}(T^q) - y_{h(f)}(T^q)$$

$$\leq c_f + y_{t(f)}(T^m) - y_{h(f)}(T^{n \cdot |V|})$$

$$\leq c_f + y_{t(f)}(T^m) - y_{h(f)}(T^m) - 2^k \qquad \text{[from 4.3.7]}$$

$$= -2^k \qquad \text{[since } f \in E(T^m)\text{]}.$$

Therefore during the simplex sequence $T^{(n-1)\cdot|V|}, T^{(n-1)\cdot|V|+1}, \ldots, T^{n\cdot|V|-1}$, vertex $h(f)$ must have been considered once say at T^q, and any incoming edge e with $h(e) = h(f)$, would have $\bar{c}_e(T^q) \leq \bar{c}_f(T^q)$, therefore:

$$y_{h(f)}(T^{n\cdot|V|}) \leq y_{h(f)}(T^q) + \bar{c}_e(T^q) \leq y_{h(f)}(T^q) + \bar{c}_f(T^q)$$

$$= y_{h(f)}(T^q) + y_{t(f)}(T^q) - y_{h(f)}(T^q) + c_f$$

$$= y_{t(f)}(T^q) + c_f = y_{h(f)}(T^m),$$

which contradicts 4.3.6. Therefore $y_v(T^{n\cdot|V|}) = y_v(T^m)$, and since no such path can have more than p zero edges, hence (iii) is also proved. Q.E.D.

<u>Theorem 4.3.4</u>. Let $T^0, T^1, T^2, \ldots, T^m$ be the sequence of strongly feasible trees for an n by n CA problem, where T^0 and T^m are the initial and optimal trees respectively. Let C be an integral cost vector and let L be the least integer such that $\max\{|c_e| : e \in E\} \leq 2^L$. Then,

(i) If the RLRC rule has been used, then $m \leq 2n(n-1)\cdot|E|\cdot(L+1)$;

(ii) If the RLRB rule has been used, then $m \leq 2n(n-1)\cdot|E|\cdot(L+1)$;

(iii) If the RIMN rule has been used, then $m \leq 2n^2(n-1)\cdot(L+1)$;

(iv) If the RAIMN rule has been used, then $m \leq 2n^2(n-1)\cdot(L+1)$.

Proof. From theorem 4.3.1, each period of the algorithm can have, at most 2n non-degenerate pivots and from definition of L, it is clear that the algorithm will have L+1 periods. Therefore the total number of the non-degenerate pivots during the algorithm can not exceed 2n. (L+1). Now (i) and (ii) follow from theorem 4.3.3 and the fact that every feasible tree for an nxn OA problem contains exactly (n-1) zero edges. (iii) and (iv) also follow from theorem 4.3.3 by further noting that, in the bipartite graph of an nxn OA problem there are only n vertices with inward edges incident to them. Q.E.D.

Even though the results of theorem 4.3.4 are considered to be "good" in the sense of Edmonds, they are not completely satisfactory because (a) the bounds are dependent on the input size of the costs and (b) there are several specialized "good" algorithms for the OA problem with much better worst case computation bounds. In the following section we will give another refinement of the strongly feasible network simplex method for the OA problem with a worst case computation bound which is independent of the costs.

§4.4. SECOND REFINEMENT

An n by n OA problem has $2^{n-1}\cdot n^{n-2}\cdot n!$ feasible trees, $n^{n-2}\cdot n!$ strongly feasible trees, and n! feasible solutions. Thus, for a (finite) simplex method, any simplex sequence has length not exceeding $2^{n-1}\cdot n^{n-2}\cdot n!$ [11]. The method of strongly feasible trees of Cunningham [10], has improved this bound to $n^{n-2}\cdot n!$, and theorem 2.6.2 has further improved it to $n\cdot(n-1)\cdot n!$. In this section we will show that with the method of strongly feasible trees, it is possible to reduce the bound to $n\cdot(|E|-|V|+1)$. To obtain this bound, we will refine the "altered most negative (AMN)" rule (rule 3.1.3) of chapter 3 as follows.

REFINED "ALTERED MOST NEGATIVE" (RAMN) RULE FOR THE OA PROBLEM

Let T^0 be a strongly feasible tree and let X^0 and Y^0 be its associated tree solution and vertex numbering respectively. Let $E_1^0 = \{e \in E: \bar{c}_e(T^0) \geq 0\}$ and $E_2^0 = \{e \in E: \bar{c}_e(T^0) < 0\}$. If $E_2^0 = \phi$, stop; X^0 and Y^0 are optimal primal and dual solutions respectively; otherwise, enter the most negative reduced cost element e^0 of E_2^0 to the

basis to obtain the strongly feasible tree T^1 and its associated tree solution X^1 and vertex numbering Y^1. If $\bar{c}_e(T^1) \geq 0$ for all $e \in E_1^0$, set $T^0 \leftarrow T^1$, $X^0 \leftarrow X^1$ and $Y^0 \leftarrow Y^1$; and redefine E_1^0 and E_2^0 and continue; else perform simplex pivots by entering the most negative reduced cost element of E_1^0 relative to the current tree to the basis, until a strongly feasible tree T^* with the tree solution X^* and vertex numbering Y^* is obtained such that $\bar{c}_e(T^*) \geq 0$ for all $e \in E_1^0$; then set $T^0 \leftarrow T^*$, $X^0 \leftarrow X^*$ and $Y^0 \leftarrow Y^*$ and redefine E_1^0 and E_2^0 and continue.

Let a PERIOD of a strongly feasible network simplex method for the OA problem with the RAMN rule be a strongly feasible simplex sequence $T^0, T^1, \ldots, T^k$, where T^0 is defined as in the RAMN rule and T^1 is obtained from T^0 by entering the most negative reduced cost element e^0 of E_2^0 to the basis and for $1 \leq i \leq k-1$, T^{i+1} is obtained from T^i by entering the most negative reduced cost element e^i of E_1^0 (relative to T^i) to the basis and T^k is such that $\bar{c}_e(T^k) \geq 0$ for all $e \in E_1^0$.

<u>Theorem 4.4.1</u>. Length of each period of a strongly feasible network simplex method with the RAMN rule for an n by n OA problem is bounded above by n i.e. $k \leq n$.

<u>Proof</u>. Let $T^0, T^1, \ldots, T^k$ and $e^0, e^1, \ldots, e^{k-1}$ be as in the definition of the period and let for $i = 1, \ldots, k$, $V^i \subseteq V$ be the set of vertices such that the vertex numbering of its elements changed when the edge e^{i-1} entered the tree. Then there are three cases to consider.

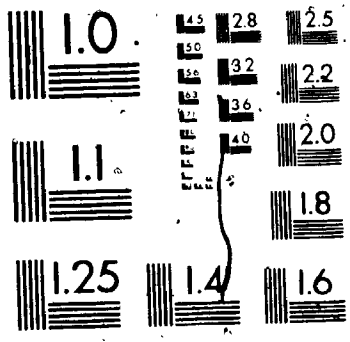
<u>Case 1</u>. All of the pivots during the period are degenerate, in which case the theorem is a special case of theorem 3.1.3 and thus the result follows.

Case 2. The entering edge e^0 forms a non-degenerate pivot, in which case since the trees are strongly feasible therefore leaving edge f^0 is such that $h(e^0) = h(f^0)$; thus $h(e^0) \in R(T^0, f^0)$ and by lemma 2.2.1 we have:

$$y_v^1 = \begin{cases} y_v^0 & \text{if } v \notin V^1, \\ y_v^0 - \bar{c}_{e^0}(T^0) & \text{if } v \in V^1. \end{cases} \qquad 4.4.1$$

But $\bar{c}_{e^0}(T^0) < 0$; therefore $y_v^1 > y_v^0$ for $v \in V^1$ and by lemma 2.2.2 if for some $e \in E_1^0$, $\bar{c}_e(T^1) < 0$, then $e \in \delta(V \setminus V^1)$ and in particular $e^1 \in \delta(V \setminus V^1)$; but it is easily seen that such an edge is an up or a cross edge; therefore by lemma 4.2.2 the pivot is degenerate and if f^1 is the leaving edge, then $h(f^1) = h(e^1)$. Therefore $f^1 \in \gamma(V^1)$; thus $V^2 \subseteq V^1$. By an argument similar to the one in the proof of lemma 3.1.3 it can be shown that for all $i \geq 2$, if $\bar{c}_e(T^i) < 0$ for some $e \in E_1^0$, then $e \in \delta[(V \setminus V^1) \cup (\bigcup_{j=2}^{i} V^j)]$ and in particular $e^i \in \delta[(V \setminus V^1) \cup (\bigcup_{j=2}^{i} V^j)]$ and again such an edge is an up or a cross edge and the leaving edge is an element of $\gamma[V^1 \setminus (\bigcup_{j=2}^{i} V^j)]$. This shows that in case 2 there is only one non-degenerate pivot which is formed by e^0; therefore $e^0 \in M^1$ and remains a tree edge during subsequent (degenerate) pivots of the period and furthermore it is obvious from the proof that for all $i = 1,\ldots,k-1$, whenever e^i enters the basis it remains part of the tree during the subsequent pivots of the period.

2 OF / DE 2



However, since each tree has exactly n-1 zero edges therefore the total number of degenerate pivots cannot exceed n-1; thus $k \leq n-1+1 = n$.

<u>Case 3</u>. Suppose that for some $p \geq 1, e^0, e^1, \ldots, e^{p-1}$ form degenerate pivots and e^p forms a non-degenerate pivot. As in lemma 3.1.3 we can show that:

(a) For $i = 1, \ldots, p$, if $\bar{c}_e(T^i) < 0$ for some $e \in E_1^o$, then $e \in \delta(\bigcup_{j=1}^{i} V^j)$ and

(b) $\bar{c}_{e^0}(T^0) \leq \bar{c}_{e^1}(T^1) \leq \ldots \leq \bar{c}_{e^{p-1}}(T^{p-1}) \leq \bar{c}_{e^p}(T^p)$.

Note that as in corollary 3.1.2, (a) implies that the entering edges $e^0, e^1, \ldots, e^{p-1}$ will not be deleted except possibly during the pivot on edge e^p. But since T^p is a strongly feasible tree, so e^p is a down edge and if f^p is the leaving edge then $h(f^p) = h(e^p)$; but such an edge cannot be any of the entering edges, hence thus far none of the entering edges have been deleted. From (a) and the fact that e^p is a down edge, it is clear that $\bigcup_{i=1}^{p} V^i \subset V^{p+1}$ and that $e^0 \in M^{p+1}$, where M^{p+1} is the perfect matching associated with the current tree T^{p+1}. Let $\underline{V}^{p+1} = V^{p+1} \setminus (\bigcup_{i=1}^{p} V^i)$, then:

<u>CLAIM</u>. If $\bar{c}_e(T^{p+1}) < 0$ for some $e \in E_1^o$, then $e \in \delta(V \setminus \underline{V}^{p+1})$.

<u>Proof of the claim.</u> Because of degeneracy of pivots on e^i, $i = 0,\ldots,p-1$ and (a), we have

$$y_v^p = \begin{cases} y_v^0 + \bar{c}_{e^0}(T^0) & \text{if } v \in V^1, \\ y_v^0 + \bar{c}_{e^1}(T^1) & \text{if } v \in V^2, \\ \vdots & \vdots \\ y_v^0 + \bar{c}_{e^{p-1}}(T^{p-1}) & \text{if } v \in V^p, \\ y_v^0 & \text{if } v \in V\setminus(\bigcup_{i=1}^{p} V^i). \end{cases} \qquad 4.4.2$$

But since the pivot on e^p is non-degenerate and $h(e^p) \in R(T,f^p)$ then by lemma 2.2.1 we have

$$y_v^{p+1} = \begin{cases} y_v^p - \bar{c}_{e^p}(T^p) & \text{if } v \in V^{p+1}, \\ y_v^p & \text{otherwise} \end{cases} \qquad 4.4.3$$

Combining 4.4.2 and 4.4.3 gives

$$y_v^{p+1} = \begin{cases} y_v^0 + \bar{c}_{e^0}(T^0) - \bar{c}_{e^p}(T^p) & \text{if } v \in V^1, \\ y_v^0 + \bar{c}_{e^1}(T^1) - \bar{c}_{e^p}(T^p) & \text{if } v \in V^2, \\ \vdots & \vdots \\ y_v^0 + \bar{c}_{e^{p-1}}(T^{p-1}) - \bar{c}_{e^p}(T^p) & \text{if } v \in V^p, \\ y_v^0 - \bar{c}_{e^p}(T^p) & \text{if } v \in \underline{V}^{p+1}, \\ y_v^0 & \text{if } v \in V\setminus V^{p+1}. \end{cases} \qquad 4.4.4$$

Let $e \in E_1^o$; then:

(3.1). If $e \in [\gamma(\underline{V}^{p+1})] \cup [\gamma(V \setminus V^{p+1})] \cup [\gamma(\bigcup_{i=1}^{p} V^i)]$, then from (a) and 4.4.3 and 4.4.4 it is easy to see that $\bar{c}_e(T^{p+1}) \geq 0$.

(3.2) If $h(e) \in V \setminus V^{p+1}$ and $t(e) \in V^{p+1}$, then

$$\bar{c}_e(T^{p+1}) = y^{p+1}_{t(e)} - y^{p+1}_{h(e)} + c_e$$

$$= [y^p_{t(e)} - \bar{c}_{e^p}(T^p)] - y^p_{h(e)} + c_e \qquad \text{[from 4.4.3]}$$

$$= \bar{c}_e(T^p) - \bar{c}_{e^p}(T^p)$$

$$\geq 0 - \bar{c}_{e^p}(T^p) \qquad \text{[from (a)]}$$

$$\geq 0. \qquad \text{[since } \bar{c}_{e^p}(T^p) < 0\text{]}$$

(3.3) If $h(e) \in \bigcup_{i=1}^{p} V^i$ and $t(e) \in V \setminus V^{p+1}$, then assuming that $h(e) \in V^i$ for some $i = 1, \ldots, p$ we have

$$\bar{c}_e(T^{p+1}) = y^{p+1}_{t(e)} - y^{p+1}_{h(e)} + c_e$$

$$= y^o_{t(e)} - [y^o_{h(e)} + \bar{c}_{e^{i-1}}(T^{i-1}) - \bar{c}_{e^p}(T^p)] + c_e \qquad \text{[from 4.4.4]}$$

$$= \bar{c}_e(T^o) - [\bar{c}_{e^{i-1}}(T^{i-1}) - \bar{c}_{e^p}(T^p)]$$

$$\geq 0 - [\bar{c}_{e^{i-1}}(T^{i-1}) - \bar{c}_{e^p}(T^p)] \qquad \text{[since } e \in E_1^{\,0}\text{]}$$

$$\geq 0. \qquad \text{[from (b)]}$$

(3.4) If $h(e) \in \bigcup_{i=1}^{p} V^i$ and $t(e) \in \underline{V}^{p+1}$ then from 4.4.3 it is easy to see that $\bar{c}_e(T^{p+1}) = \bar{c}_e(T^p)$; but from (a) we have $\bar{c}_e(T^p) \geq 0$; thus $\bar{c}_e(T^{p+1}) \geq 0$.

In (3.1),...,(3.4) we have considered all possibilities but $e \in \delta(V \setminus \underline{V}^{p+1})$; thus the claim is proven.

Now let $e \in \delta(V \setminus \underline{V}^{p+1})$; then:

If $t(e) \in V \setminus V^{p+1}$, then $\bar{c}_e(T^{p+1}) = \bar{c}_e(T^p) + \bar{c}_{e^p}(T^p)$ [Using 4.4.3]

$$\geq 0 + \bar{c}_{e^p}(T^p) \qquad \text{[From (a)]}$$

If $t(e) \in \bigcup_{i=1}^{p} V^i$, then $\bar{c}_e(T^{p+1}) = \bar{c}_e(T^p)$ [Using 4.4.3]

$$\geq \bar{c}_{e^p}(T^p) \qquad \text{[By the choice of } e^p\text{]}$$

Thus in particular $\bar{c}_{e^{p+1}}(T^{p+1}) \geq \bar{c}_{e^p}(T^p)$. 4.4.5

The claim and 4.4.5 mean that we are in a situation similar to the case 1, and as in that case we can show that $e^{p+1},\ldots,e^{k-1}$ form degenerate pivots and corresponding leaving edges are elements of $\gamma(\underline{V}^{p+1})$ and hence during the period, none of the incoming edges, in particular the matching edge e^0, will be deleted. On the other hand, since $\underline{V}^{p+1}$ is disjoint from $\bigcup_{i=1}^{p} V^i$ therefore the total number of degenerate pivots during the period cannot exceed n-1 (where n-1 is the number of zero edges in the tree T^0) and thus $k \leq n-1+1 = n$. Q.E.D.

<u>Theorem 4.4.2.</u> Let $T^0, T^1, \ldots, T^m$ be a sequence of strongly feasible trees for an n by n OA problem, where T^0 and T^m are the initial and optimal trees respectively. Then if the RAMN rule is used then:

(a) $m \leq n^3 - 2n^2 + n$

(b) If L is the total number of non-degenerate pivots during the algorithm then $L \leq |E| - 2n + 1 \leq n^2 - 3n + 1$.

<u>Proof.</u> (a) It follows from theorem 4.4.1 that during each period of the algorithm, the cardinality of the set of negative reduced cost (pivot eligible) edges is reduced by at least 1 and consequently the cardinality of the set of non-negative reduced cost edges is increased by 1, but relative to T^0 the cardinality of the negative reduced cost edges is $\leq |E| - |V| + 1 \leq n^2 - 2n + 1$

$$= n^2 - 2n + 1$$

thus the algorithm can have at most $n^2 - 2n + 1$ periods and from theorem 4.4.1 each period can have at most n pivots; thus

$$m \leq n(n^2 - 2n + 1) = n^3 - 2n^2 + n.$$

(b) Again it follows from the proof of theorem 4.4.1 that each period can have at most one non-degenerate pivot and thus $L \leq n^2 - 2n + 1$. Q.E.D.

It is not difficult to see that the amount of work to execute one pivot of a strongly feasible network simplex method with the RAMN rule for an n by n optimum assignment problem is of $O(|E|)$ or of $O(n^2)$ and thus the worst case computation bound of this algorithm is of $O(n^5)$. Therefore, from a theoretical point of view, the algorithm does not compare favourably with the existing $O(n^3)$ algorithms for the OA problem which are mentioned at the beginning of this chapter. But despite this unpleasant fact, this is very important in that, it is the first polynomial bounded network simplex algorithm for the optimum assignment problem.

Even though the optimum assignment problem is a highly structured instance of network flow problems and the methods of this chapter may not extend to more general network flow problems, these results simply show that as Cunningham [11] has pointed out "the notion that the simplex method is theoretically beyond repair seems to have been accepted on the basis of very little evidence".

CHAPTER 5

CIRCULATION PROBLEM

§5.1 THE MAXIMUM FLOW PROBLEM

Let $G' = (E',V')$ be a finite digraph and let $u' = (u_e' : e \in E)$ be a non-negative capacity vector. Given two vertices s and t called the SOURCE and the SINK, we call an (s,t)-flow (or simply a flow) any real-valued function X defined on E' satisfying the (Kirchhoff) conservation law $v(X) = \sum[x_e : v = h(e)] - \sum[x_e : v = t(e)] = 0$, for all $v \in V' \setminus \{s,t\}$. If in addition, for all $e \in E'$, $0 \le x_e \le u_e'$, then X is a feasible flow. Since G' is a digraph, then as in remark 1.1.1, it is easy to show that $t(X) = -s(X) = z$ for a feasible flow X. The quantity $t(X) = z$ is called the AMOUNT of flow X.

The MAXIMUM FLOW PROBLEM is the problem of finding a feasible flow of maximum amount:

$$\text{(MFP1)} \qquad \begin{array}{l} \max z \\ \text{s.t.} \\ v(X) = \begin{cases} -z & \text{if } v = s, \\ 0 & \text{if } v \in V' \setminus \{s,t\}, \\ z & \text{if } v = t, \end{cases} \\ 0 \le x_e \le u_e' \text{ for all } e \in E'. \end{array}$$

MFP1 can easily be solved as a MCFP by simply adding a "return" edge $g = (t,s)$ with $u_g = \infty$ and $c_g = -1$ and letting $c_e = 0$ for all $e \in E'$ and $u_e = u_e'$. If the new graph is $G = (E,V)$ with $E = E' \cup \{g\}$ and $V = V'$ then the new problem is:

$$\text{MFP} \quad \begin{cases} \min & -x_g \\ \text{s.t.} & v(X) = 0 \quad \text{for all } v \in V, \\ & 0 \le x_e \le u_e \quad \text{for all } e \in E. \end{cases}$$

A starting basic feasible solution is readily available in MFP; i.e. $X = 0$.

In the more general MCFP we must find a starting basic feasible solution. This is commonly known as the PHASE I network simplex method. In phase I, the MCFP is transformed to an MFP by adding a dummy source s and a dummy sink t to the graph $G'' = (E'',V'')$ of the MCFP and a vertex $v \in V''$ is joined to t by an edge (v,t) if and only if $b_v > 0$. The capacity of (v,t) is set at b_v and its cost at zero. Likewise s is joined to a vertex $v \in V''$ by an edge (s,v) if and only if $b_v < 0$. The capacity of (s,v) is set at $-b_v$ and its cost at zero. Furthermore an edge $g = (t,s)$ is also added to the graph with $u_g = \infty$ and $c_g = -1$; the cost of other edges are also set at zero and their capacities are

as given and $b_v = 0$ for all $v \in V'' \cup \{s,t\}$. The resulting MFP on the new graph $G = (E,V)$ is solved by starting with $X = 0$. If X^* is an optimal solution with amount $x_g^* = \sum(b_v > 0: v \in V'')$ [i.e. the flow on the artificial edges (v,t) for all $v \in V''$ such that $b_v > 0$, are at their capacity], then it is easily demonstrated that $\overline{X}^* = (x_e^*: e \in E'')$ is a feasible solution of the MCFP. But if $x_g^* < \sum(b_v > 0; v \in V'')$, then it can easily be seen that, the MCFP is infeasible.

§5.2 NETWORK SIMPLEX METHOD FOR THE MFP

Let T be a feasible tree and let X and Y be its associated basic feasible solution and vertex numbering respectively. Since $u_g = \infty$, therefore for any feasible solution X such that $x_g > 0$, g must be a tree edge. Furthermore for $X = 0$ we can also choose g as a tree edge; thus we will assume that during the execution of the algorithm g is a tree edge. By deleting g from T we obtain two subtrees T_s and T_t with vertex sets V_s and V_t such that $s \in V_s$ and $t \in V_t$ and $V_s \cup V_t = V$ and $V_s \cap V_t = \phi$. Letting $y_s = 0$, it is easy to see that

$$y_v = \begin{cases} 0 & \text{if } v \in V_s, \\ 1 & \text{if } v \in V_t. \end{cases} \qquad 5.2.1$$

From 5.2.1 for non-basic edges we have

$$\bar{c}_e = \begin{cases} 1 & \text{if } e \in \delta(V_t), \\ -1 & \text{if } e \in \delta(V_s), \\ 0 & \text{otherwise.} \end{cases} \qquad 5.2.2$$

Thus an edge e can enter the basis if and only if $e \in \{u[\delta(V_t)]\} \cup \{0[\delta(V_s)]\}$ but such an edge will form an s-t path P and $g \in C(T,e)$ and the vertex s is the first (and indeed the only) common vertex in the unique paths from h(e) and t(e) to the root s of T. To maintain the basis some edge of P must be deleted. The candidate edges to be deleted are those forward edges of P the new flow of which is at the edge capacity or those reverse edges of P with a new flow equal to zero. If the method of strongly feasible trees is used, the nearest blocking edge to s in P will be deleted. Hereafter this rule will be referred to as Rule 5.2.1. But since in a strongly feasible tree the subpath P_1 of P from s to h(e) if $x_e = 0$ (t(e) if $x_e = u_e$) will necessarily be a flow-augmenting path, thus if the pivot is degenerate then the outgoing edge f will have to be an element of $E(T_t)$. Consequently the new V_s will have at least one more vertex than the old V_s; hence there can be no more than $|V| -2$ consecutive degenerate pivots [10]. Johnson's [33] algorithm also has the property that, the length of a sequence of consecutive degenerate pivots can not exceed $|V| -2$. Johnson's method does not keep a full basis, indeed when an edge enters the basis if there are more than one blocking edges, the nearest and the furthest blocking edges to the root are both deleted. In the remainder

of this section we will show that the prevention of cycling and stalling for MFP need not require imposition of special topology such as strong feasibility [10] or not maintaining a full basis [33]. Indeed very simple leaving edge rules such as rule 5.2.1 will prevent both cycling and stalling.

Lemma 5.2.1. Let $T^0, T^1, \ldots, T^k$ be a degenerate simplex sequence for the MFP. Then if rule 5.2.1 is used, none of the leaving edges can re-enter the basis during the sequence.

Proof. For some $i = 0,1,\ldots,k-1$, let e^i enter the tree T^i and f^i leave the tree T^i. Let P^i be the s-t path formed by addition of e^i to T^i, then:

Case 1: $x_{f^i} = u_{f^i}$. In this case f^i is a forward edge of P^i and the subpath P_1^i of P^i from s to $t(f^i)$ is a flow-augmenting path and all the vertices in P_1^i (in particular $t(f^i)$) will be elements of V_s^{i+1}. Furthermore none of the edges in P_1^i could be deleted during the subsequent pivots in the sequence, thus $y^{i+1}_{t(f^i)} = y^{i+2}_{t(f^i)} = \ldots = y^k_{t(f^i)} = 0$. Now if for some $j = i+1, i+2, \ldots, k$, $h(f^i) \in V_t^i$, then $y^j_{h(f^i)} = 1$, thus $\bar{c}_{f^i}(T^j) = -1$. But since $x_{f^i} = u_{f^i}$, hence f^i cannot enter the tree T^j. On the other hand if $h(f^i) \in V_s^j$ for some $j = i+2, \ldots, k$, then $y^j_{h(f^i)} = 0$ and hence $\bar{c}_{f^i}(T^j) = 0$ and once again f^i cannot enter the tree T^j.

Case 2: $x_{f^i} = 0$. This is similar to case 1 and the details are omitted. Q.E.D.

Corollary 5.2.1. Network simplex method with rule 5.2.1 for MFP, will not

(a) cycle

(b) stall.

Proof. (a) This is a direct consequence of lemma 5.2.1.

(b) From lemma 5.2.1, it is clear that the number of trees in any degenerate simplex sequence cannot exceed $|E| - |V| + 1$. Q.E.D.

Another property of rule 5.2.1 is given in the following lemma.

Lemma 5.2.2. Let P: $s, e^1, v^1, e^2, v^2, \ldots, e^k, t$ be an s-t simple path in the graph G. Let T^0 be a feasible tree for the MFP and e be the entering edge such that the s-t path formed by addition of e to T^0 is P. Then if rule 5.2.1 is being used, the path P will never repeat again.

Proof. Let S: $T^0, T^1, T^2, \ldots, T^k$ be any simplex sequence. If S is a degenerate simplex sequence, then the result follows from lemma 5.2.1. If some of the pivots in the sequence were non-degenerate, then let $F = \{f \in E(P)$: f has left the basis at least once during the sequence S$\}$. Let $f^* \in F$ be the closest element of F to s in P, i.e. no edge on the path from s to $t(f^*)$ or $h(f^*)$ (whichever is closer to s in P) has been deleted during the sequence S. Now as in lemma 5.2.1, we can argue that after f^* was deleted from the basis it must have remained a non-tree edge and thus P could not be repeated during the sequence S. Q.E.D.

Rule 5.2.1 can be altered so as to choose the furthest blocking edge to root s (or equivalently the closest blocking edge to t) to be deleted. Another rule for choosing the leaving edge is to let f_1 be the closest blocking edge to s and f_2 be the closest blocking edge to t and if d_1 and d_2 represent the distance of f_1 from s and f_2 from t respectively then choose f_1 to be deleted if and only if $d_1 \leq d_2$ otherwise delete f_2. Both of these refinements have the same properties as rule 5.2.1.

If the capacities are all integers and X = 0 is the starting flow then it is easily seen that each non-degenerate pivot will increase the flow on the return edge g by at least 1 unit and if Z^* is the amount of the maximum s-t flow, then the optimum solution will be obtained after no more than Z^* non-degenerate pivots. If the method of strongly feasible trees is used then the total number of pivots will not exceed $(|V| - 2)\cdot Z^*$. Note that if the maximum flow problem is obtained from a MCFP, then $Z^* \leq \sum(b_v : v \text{ a demand vertex})$ and equality holds only if there is a feasible solution to the MCFP.

For the more general MCFP, the bounds such as $\sum(b_v : v \text{ is a demand vertex})$ on the total number of non-degenerate pivots (in the case of integral data) and $|V|$ on the number of consecutive degenerate pivots do not seem to be available. But since the circulation problem (introduced in §1.2) is structurally quite similar to the MFP, then it is natural to ask the following question; can one obtain bounds similar to those of the MFP for the circulation problem? This is the question that we will try to answer positively in the following sections.

§5.3 MINIMUM COST MAXIMUM FLOW PROBLEM (MCMFP)

A MCMFP is a MFP in which there is a non-zero cost vector associated with the edge set of the network; thus the objective is to find a maximum s-t flow of minimum cost.

Any MCFP on a digraph $G' = (V',E')$ can be cast into an equivalent MCMFP as follows: add a dummy source s and a dummy sink t to the vertices of G'. For each $v \in S = \{v \in V': b_v' < 0\}$ add an edge from s to v with cost 0 and capacity $-b_v$ and let E^s denote the set of all such edges. For each $v \in T = \{v \in V': b_v' > 0\}$ add an edge from v to t with cost 0 and capacity b_v and let E^t denote the set of all such edges. In the new digraph $G = (V,E)$ where $V = V' \cup \{s,t\}$ and $E = E' \cup E^s \cup E^t$ let $b_v = 0$ for all $v \in V'$, let costs and capacities of edges in E' be as given and require a maximum s-t flow of minimum cost. The following lemma will show that this MCMFP is equivalent to the MCFP in that, if the MCFP is feasible then the optimal solution to the MCMFP will produce an optimal solution to the MCFP. For a MCFP let $b^+ = \sum(b_v': v \in T)$ and let X be a feasible flow for the MCMFP. As for the MFP1, it is easily seen that for a feasible flow X the amount of flow out of the source is equal to the amount of flow into the sink. We will abbreviate "the amount of flow into the sink" to "amount of flow" and denote it by Z and will let x_0 denote the cost of the optimal flow of amount Z.

Lemma 5.3.1. Let MCMFP be obtained from a MCFP. If Z^* is the amount of maximum flow and X^* is the minimum cost flow of amount Z^* for the MCMFP with cost x_o^* then;

(i) If $Z^* < b^+$, then the MCFP is infeasible and

(ii) If $Z^* = b^+$, then $\overline{X}' = (x_e^*: e \in E')$ is an optimal flow for the MCFP with objective value $\overline{x}_o' = x_o^*$.

Proof. From the transformation of the MCFP into a MCMFP, it is clear that $b^+ = \sum(u_e: e \in E^t)$; and since E^t is an s-t cut set, thus for any feasible flow X, the amount of flow $Z \leq b^+$. On the other hand from remark 1.1.1 we have $b^+ = \sum(-b_v: v \in S)$ and thus clearly $b^+ = \sum(u_e: e \in E^s)$. Now to prove (i) assume the contrary i.e. that $Z^* < b^+$ and the MCFP is feasible and let X' be such a feasible flow. Define $X = (x_e: e \in E)$ as follows:

$$x_e = \begin{cases} x_e' & \text{if } e \in E', \\ b_v & \text{if } e \in E^t \text{ and } t(e) = v, \\ -b_v & \text{if } e \in E^s \text{ and } h(e) = v. \end{cases} \qquad 5.3.00$$

Then X is a feasible flow of amount $Z = b^+ > Z^*$ for the MCMFP, thereby contradicting the maximality of the Z^*; thus the MCFP is infeasible.

To prove (ii) notice that $Z^* = b^+ = \sum(u_e : e \in E^t) = \sum(u_e : e \in E^s)$ or that for all $e \in E^t \cup E^s$, $x_e^* = u_e$. From feasibility of X^* to the MCMFP we have

$$0 = v(X^*) = \begin{cases} v(\overline{X}') - b_v & \text{if } v \in T, \\ v(\overline{X}') + b_v & \text{if } v \in S, \\ v(\overline{X}') & \text{if } v \in V' \setminus \{S \cup T\}, \end{cases}$$

or

$$v(\overline{X}') = \begin{cases} b_v & \text{if } v \in T, \\ -b_v & \text{if } v \in S, \\ 0 & \text{if } v \in V' \setminus \{S \cup T\}. \end{cases}$$

Therefore $\overline{X}'$ is a feasible flow for the MCFP and if $\overline{x}_0'$ is the objective value corresponding to $\overline{X}'$ then $\overline{x}_0' = x_0^*$. To show that $\overline{X}'$ is an optimal flow, we assume the contrary and let X' be an optimal flow of MCFP with objective value $x_1' < x_0^*$. Once again let X be defined from X' as in 5.3.00. Thus X is a feasible flow of amount $Z = b^+$ to the MCMFP which in turn implies that X is a feasible flow of maximum amount with objective value (cost) $x_0 = x_0' < x_0^*$. This is to say that we have found a feasible flow of maximum amount to the MCMFP with cost less than that of X^*, thereby contradicting the optimality of the X^* assumption. Therefore $\overline{X}'$ is an optimal flow for the MCFP.

To solve the MCMFP with the network simplex method, we add a return edge $g = (t,s)$ with $u_g = \infty$ and $c_g = -M = -[(\max|c_e| : e \in E)\cdot|V|+1]$. The resulting problem is a special case of the MCFP in which for all $v \in V$, $b_v = 0$. This problem is known as a CIRCULATION PROBLEM (CP). There are two reasons to choose M so large. (1) It guarantees that the maximum possible amount of flow will pass through the return edge g and using this it is easily seen that the CP and MCMFP are equivalent in the sense that the optimal solution of one will provide an optimal solution to the other. (2) It guarantees negativity of the cost of any circuit containing the return edge g, directed in the direction of g, or in other words it guarantees that for any given tree T containing g, if T_s, T_t, V_s and V_t are defined as for the MFP (§5.2), then

$$\bar{c}_e(T) \begin{cases} < 0 & \text{if } e \in \delta(V_s), \\ > 0 & \text{if } e \in \delta(V_t). \end{cases} \qquad 5.3.0$$

The importance of 5.3.0 will be clear in the following theorems. The flow on the return edge g will be called "the amount of flow". Jewell and Busacker and Gowen independently have shown that in solving a MCMFP, if X is a minimum cost flow of amount z then an augmentation of amount θ on a minimum cost s-t flow augmenting path results in a minimum cost flow X' of amount $z' = z + \theta$ [23]. This is the reason why the labeling method of Ford and Fulkerson [23] requires no more than b^+ flow.

augmentations on a problem with non-negative costs and integral capacities. The essence of the following theorems is to show that, with the method of this chapter for solving CP it is possible to perform all of the non-degenerate pivots (augmentations) on essentially minimum cost s-t augmenting paths and that any such path can be found in a polynomial amount of work in the size of the problem being solved.

<u>Theorem 5.3.1</u>. Let T^0 be a feasible tree (for CP) containing the return edge g and let e^0 be its associated most negative reduced cost edge and suppose that:

(I) An edge $e \in E$ is a pivot eligible edge if and only if $e \in \{O(\delta(V_s^0))\} \cup \{u(\delta(V_t^1))\}$.

(II) If $\{O(\delta(V_t^0))\} \neq \phi$, then for any $f \in \{O(\delta(V_t^0))\}$, $\bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \geq 0$.

(III) If $\{u(\delta(V_s^0))\} \neq \phi$, then for any $e \in \{u(\delta(V_s^0))\}$, $\bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \leq 0$.

Let T^1 be obtained from T^0 by entering e^0 to the basis and deleting f^0 from it. Let e^1 be the most negative reduced cost edge relative to T^1. Then:

(a) An edge $e \in E$ is a pivot eligible edge relative to T^1 if and only if $e \in \{O(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\}$,

(b) $|\bar{c}_{e^1}(T^1)| \leq |\bar{c}_{e^0}(T^0)|$.

(c) If $\{O(\delta(V_t^1))\} \neq \phi$, then for any $f \in \{O(\delta(V_t^1))\}$,
$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1)| \geq 0$.

(d) If $\{u(\delta(V_s^1))\} \neq \phi$, then for any $e \in \{u(\delta(V_s^1))\}$,
$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| \leq 0$.

<u>Proof</u>. Let X^0 and Y^0 [X^1 and Y^1] be the tree solution and vertex numbering associated with T^0[T^1] and let $V^1 \subseteq V$ be the set of vertices such that the vertex numbering of its elements changed during the pivot. Then:

(a) ($\Rightarrow$): If $e \in \{O(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\}$, it follows from 5.3.0 that e is a pivot eligible edge.

(a) ($\Leftarrow$): To prove the only if part of (a), it is sufficient to show that:

(a1) If $f \in A = \{O(\gamma(V_t^1))\} \cup \{O(\gamma(V_s^1))\}$, then
$\bar{c}_f(T^1) \geq 0$,

(a2) If $e \in B = \{u(\gamma(V_t^1))\} \cup \{u(\gamma(V_s^1))\}$, then
$\bar{c}_e(T^1) \leq 0$.

The sufficiency of (a1) and (a2) follow from the fact that the pivot ineligibility of any edge in $\{u(\delta(V_s^1))\} \cup \{O(\delta(V_t^1))\}$ is guaranteed by 5.3.0.

To prove (a), (b), (c) and (d) there are three cases to consider and in each case we will complete the proof of (a) by proving (a1) and (a2) and then we will use (a) to prove (b) and use (b) to prove (c) and (d).

<u>Case 1. $V^1 = \phi$</u>. i.e. $e^o = f^o$ this can happen only if the pivot is non-degenerate; hence for any $e \in E \setminus E(T^1)$ we have:

$$x_e^1 = \begin{cases} u_{e^o} - x^o_{e^o} & \text{if } e = e^o, \\ x_e^o & \text{otherwise,} \end{cases} \qquad 5.3.1$$

and furthermore

$$\begin{aligned} y_v^1 &= y_v^o \quad \text{for all } v \in V \text{ or} \\ \bar{c}_e(T^1) &= \bar{c}_e(T^o) \quad \text{for all } e \in E. \end{aligned} \qquad 5.3.2$$

In this case (a1) and (a2) follow immediately from 5.3.1, 5.3.2 and (I). (b) follows directly from (a) and the choice of e^o. To show (c) and (d) we must consider the following two subcases.

<u>Case 1.1. $e^o \in \{O(\delta(V_s^o))\}$</u>; in which case $\bar{c}_{e^o}(T^o) < 0$ or

$$\bar{c}_{e^o}(T^o) = -|\bar{c}_{e^o}(T^o)| \qquad 5.3.3$$

and also from 5.3.1 we have $x^1_{e^o} = u_{e^o}$; thus

$$\{O(\delta(V_t^1))\} = \{O(\delta(V_t^o))\} \qquad 5.3.4$$

and $\{u(\delta(V_s^1))\} = \{u(\delta(V_s^0))\} \cup \{e^0\}$. 5.3.5

(c) Let $f \in \{O(\delta(V_t^1))\}$, then

$$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1))| \geq \bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \qquad \text{[from 5.3.2 and (b)]}$$

$$\geq 0 \qquad \text{[from 5.3.4 and (II)]}.$$

(d) Let $e \in \{u(\delta(V_s^1))\}$, then

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| \leq \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[from 5.3.2 and (b)]}$$

$$\text{but } \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \begin{cases} = 0 & \text{if } e = e^0 \quad \text{[from 5.3.3]} \\ \leq 0 & \text{otherwise} \quad \text{[from 5.3.5 and (III)]} \end{cases}$$

and thus the result follows.

<u>Case 1.2. $e^0 \in \{u(\delta(V_t^0))\}$</u>; in which case $\bar{c}_{e^0}(T^0) > 0$ or

$$\bar{c}_{e^0}(T^0) = |\bar{c}_{e^0}(T^0)| \qquad 5.3.6$$

and from 5.3.1 we have $x^1_{e^0} = 0$; thus

$$\{O(\delta(V_t^{\,1}))\} = \{O(\delta(V_t^{\,0}))\} \cup \{e^0\} \qquad 5.3.7$$

$$\text{and } \{u(\delta(V_s^{\,1}))\} = \{u(\delta(V_s^{\,0}))\} \qquad 5.3.8$$

(c) Let $f \in \{O(\delta(V_t^{\,1}))\}$, then

$$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1)| \geq \bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \qquad \text{[from 5.3.2 and (b)]}$$

$$\text{but } \bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \begin{cases} = 0 & \text{if } f = e^0 \quad \text{[from 5.3.6]} \\ \geq 0 & \text{otherwise} \quad \text{[from 5.3.7 and (II)]} \end{cases}$$

and hence the result follows.

(d) Let $e \in u(\delta(V_s^{\,1}))$, then

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| \leq \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[from 5.3.2 and (b)]}$$

$$\leq 0 \qquad \text{[from 5.3.8 and (III)]}$$

<u>Case 2. $V^1 \subseteq V_{t-}^{o}$.</u> This means that $f^o \in E(T_t^{o})$ and from lemma 2.2.1 it is easily seen that

$$y_v^{1} = \begin{cases} y_v^{o} & \text{if } v \notin V^1, \\ y_v^{o} - |\bar{c}_{e^o}(T^o)| & \text{if } v \in V^1. \end{cases} \qquad 5.3.9$$

(a1) Let $A^1 = \{f \in E: x_f^{1} = 0,\ t(f) \in V^1 \text{ and } h(f) \in V_s^1 \setminus V^1\} \subseteq \{O(\delta(V_t^{o}))\}$,

$A^2 = \{f \in E: x_f^{1} = 0,\ t(f) \in V_s^1 \setminus V^1 \text{ and } h(f) \notin V^1\} \subseteq \{O(\delta(V_s^{o}))\}$

and

$A^3 = \{O(\gamma(V_t^{1}))\} \cup \{O(\gamma(V_t^{1}))\} \cup \{O(\gamma(V_s^1 \setminus V^1))\}$; then

$A = A^1 \cup A^2 \cup A^3$. Let $f \in A$ then:

<u>if $f \in A^1$</u>, then $\bar{c}_f(T^1) = \bar{c}_f(T^o) - |\bar{c}_{e^o}(T^o)|$ [using 5.3.9]

≥ 0 [from (II)].

<u>if $f \in A^2$</u>, then from lemma 3.1.2 we have $\bar{c}_f(T^1) \geq 0$ and

<u>if $f \in A^3$</u>, then $c_f(T^1) = \bar{c}_f(T^0)$ [from 5.3.9]

≥ 0 [from (I)].

(a2) Let $B^1 = \{e \in E: \bar{x}_e^1 = u_e^1,\ t(e) \in V^1 \text{ and } h(e) \in V_s \setminus V^1\}$,

$B^2 = \{e \in E: x_e^1 = u_e^1,\ t(e) \in V_s^1 \setminus V^1 \text{ and } h(e) \in V^1\}$ and

$B^3 = \{u(\gamma(V_t^1))\} \cup \{u(\gamma(V^1))\} \cup \{u(\gamma(V_s^1 \setminus V^1))\}$, then

$B = B^1 \cup B^2 \cup B^3$. Let $e \in B$ then:

<u>if $e \in B^1$</u>, then by lemma 3.1.2 we have $\bar{c}_e(T^1) \leq 0$,

<u>if $e \in B^2$</u>, then $\bar{c}_e(T^1) = \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)|$ [using 5.3.9]

≤ 0 [from (III)]

and

<u>if $e \in B^3$</u>, then $\bar{c}_e(T^1) = \bar{c}_e(T^0)$ [from 5.3.9]

≤ 0 [from (I)].

(b) From (a) we have $e^1 \in \{0(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\}$. But it is clear that $\{0(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\} \subseteq E \setminus E(T^0)$, therefore

<u>if $e^1 \in \{O(\delta(V_s{}^1))\}$</u>, then $x^1_{e^1} = x^0_{e^1} = 0$.

If $t(e^1) \in V_s^1 \setminus V^1 = V_s{}^0$, then $e^1 \in \{O(\delta(V_s{}^0))\}$ and

$$|\bar{c}_{e^1}(T^1)| = |\bar{c}_{e^1}(T^0)| \qquad \text{[from 5.3.9]}$$

$$\leq |\bar{c}_{e^0}(T^0)|. \qquad \text{[from the choice of e}$$

If $t(e^1) \in V^1$, then $e^1 \in O(\gamma(V_t{}^0))$ and

$$|\bar{c}_{e^1}(T^1)| = -\bar{c}_{e^1}(T^1) \qquad \text{[from 5.3.0]}$$

$$= -[\bar{c}_{e^1}(T^0) - |\bar{c}_{e^0}(T^0)|] \qquad \text{[using 5.3.9]}$$

$$\leq |\bar{c}_{e^0}(T^0)| \qquad [\bar{c}_{e^1}(T^0) \geq 0; \text{ from (I)}].$$

<u>If $e^1 \in \{u(\delta(V_t{}^1))\}$</u>, then $x^1_{e^1} = x^0_{e^1} = u_{e^1}$.

If $h(e^1) \in V_s^1 \setminus V^1 = V_s{}^0$, then $e^1 \in \{u(\delta(V_t{}^0))\}$ and

$$|\bar{c}_{e^1}(T^1)| = |\bar{c}_{e^1}(T^0)| \qquad \text{[from 5.3.9]}$$

$$\leq |\bar{c}_{e^0}(T^0)| \qquad \text{[from the choice of e}$$

If $h(e^1) \in V^1$, then $e^1 \in \{u(\gamma(V_t^0))\}$ and

$$|\bar{c}_{e^1}(T^1)| = \bar{c}_{e^1}(T^1) \qquad \text{[from 5.3.0]}$$

$$= \bar{c}_{e^1}(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[using 5.3.9]}$$

$$\leq |\bar{c}_{e^0}(T^0)| \qquad [\bar{c}_{e^1}(T^0) \leq 0; \text{ from (I)}].$$

(c) If $\{O(\delta(V_t^1))\} = \phi$, then there is nothing to prove. Otherwise let $f \in \{O(\delta(V_t^1))\}$, then:

if $h(f) \in V_s^1 \setminus V^1 = V_s^0$, then $f \in \{O(\delta(V_t^0))\}$ and furthermore from 5.3.9 we have $\bar{c}_f(T^1) = \bar{c}_f(T^0)$ and also $f \neq f^0$. Therefore

$$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1)| = \bar{c}_f(T^0) - |\bar{c}_{e^1}(T^1)|$$

$$\geq \bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \qquad \text{[from (b)]}$$

$$\geq 0 \qquad \text{[from (II)]}$$

and if $h(f) \in V^1$, then $f \in \{O(\gamma(V_t^0))\} \cup \{f^0\}$ and furthermore from 5.3.9 we have $\bar{c}_f(T^1) = \bar{c}_f(T^0) + |\bar{c}_{e^0}(T^0)|$; thus

$$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1)| = \bar{c}_f(T^0) + |\bar{c}_{e^0}(T^0)| - |\bar{c}_{e^1}(T^1)|$$

$$\geq \bar{c}_f(T^0) \qquad \text{[from (b)]}$$

$$\text{but } \bar{c}_f(T^0) \begin{cases} = 0 & \text{if } f^0 \in \{0(\delta(V_t{}^1))\} \text{ and } f = f^0 \\ \geq 0 & \text{otherwise} \end{cases} \qquad \text{[from (I)]}$$

and thus the result follows.

(d). If $\{u(\delta(V_s{}^1))\} = \phi$, then there is nothing to prove. Otherwise let $e \in \{u(\delta(V_s{}^1))\}$ then:

if $t(e) \in V_s^1 \setminus V^1 = V_s{}^0$, then $e \in \{u(\delta(V_s{}^0))\}$, $e \neq f^0$ and from 5.3.9 we have $\bar{c}_e(T^1) = \bar{c}_e(T^0)$. Therefore:

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| = \bar{c}_e(T^0) + |\bar{c}_{e^1}(T^1)|$$

$$\leq \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[from (b)]}$$

$$\leq 0 \qquad \text{[from (III)]}$$

and if $t(e) \in V^1$, then $e \in \{u(\gamma(V_t{}^0))\} \cup \{f^0\}$ and furthermore from 5.3.9 we have $\bar{c}_e(T^1) = \bar{c}_e(T^0) - |\bar{c}_{e^0}(T^0)|$, thus:

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| = \bar{c}_e(T^0) - |\bar{c}_{e^0}(T^0)| + |\bar{c}_{e^1}(T^1)|$$

$$\leq \bar{c}_e(T^0) \qquad \text{[from (b)]}$$

$$\text{but } \bar{c}_e(T^0) \begin{cases} = 0 & \text{if } f^0 \in \{u(\delta(V_s{}^1))\} \text{ and } e = f^0 \\ \leq 0 & \text{otherwise} \end{cases} \qquad \text{[from (I)]}$$

hence the result follows.

<u>Case 3. $V^1 \subseteq V_s{}^0$</u>. Which means that $f^0 \in E(T_s{}^0)$ and thus from lemma 2.2.1 we have;

$$y_v{}^1 = \begin{cases} y_v{}^0 & \text{if } v \notin V^1, \\ y_v{}^0 + |\bar{c}_{e^0}(T^0)| & \text{if } v \in V^1. \end{cases} \qquad 5.3.10$$

(a1) Let $A^1 = \{f \in E: x_f{}^1 = 0,\ t(f) \in V^1 \text{ and } h(f) \in V_t^1 \setminus V^1\} \subseteq \{0(\delta(V_s{}^0))\}$,

$A^2 = \{f \in E: x_f{}^1 = 0,\ t(f) \in V_t^1 \setminus V^1 \text{ and } h(f) \in V^1\} \subseteq \{0(\delta(V_t{}^0))\}$

and

$A^3 = \{0(\gamma(V_s{}^1))\} \cup \{0(\gamma(V^1))\} \cup \{0(\gamma(V_t^1 \setminus V^1))\}$ then

$A = A^1 \cup A^2 \cup A^3$. Let $f \in A$ then :

<u>if $f \in A^1$</u>, then from lemma 3.1.2 we have $\bar{c}_f(T^1) \geq 0$,

<u>if $f \in A^2$</u>, then $\bar{c}_f(T^1) = \bar{c}_f(T^0) - |\bar{c}_{e_0}(T^0)|$ [using 5.3.10]

≥ 0 [from (II)]

and

<u>if $f \in A^3$</u>, then from 5.3.10 we have $\bar{c}_f(T^1) = \bar{c}_f(T^0)$

≥ 0 [from (I)].

(a2) Let $B^1 = \{e \in E: x_e^1 = u_e,\ t(e) \in V^1 \text{ and } h(e) \in V_t^1 \setminus V^1\} \subseteq \{u(\delta(V_s^0))\}$,

$B^2 = \{e \in E: x_e^1 = u_e,\ t(e) \in V_t^1 \setminus V^1 \text{ and } h(e) \in V^1\} \subseteq \{u(\delta(V_t^0))\}$,

and

$B^3 = \{u(\gamma(V_s^1))\} \cup \{u(\gamma(V^1))\} \cup \{u(\gamma(V_t^1 \setminus V^1))\}$ then

$B = B^1 \cup B^2 \cup B^3$. Let $e \in B$ then:

<u>if $e \in B^1$</u>, then $\bar{c}_e(T^1) = \bar{c}_e(T^0) + |\bar{c}_{e_0}(T^0)|$ [using 5.3.10]

≤ 0 [from (III)],

<u>if $e \in B^2$</u>, then from lemma 3.1.2 we have $\bar{c}_e(T^1) \leq 0$ and

<u>if $e \in B^3$</u>, then $\bar{c}_e(T^1) = \bar{c}_e(T^0)$ [from 5.3.10]

≤ 0 [from (I)].

(b) From (a) we have $e^1 \in \{O(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\}$ and clearly

$\{O(\delta(V_s^1))\} \cup \{u(\delta(V_t^1))\} \subseteq E \setminus E(T_t^0)$ and therefore:

<u>if $e^1 \in \{O(\delta(V_s^1))\}$</u> then $x^1_{e^1} = x^0_{e^0} = 0$.

If $h(e) \in V_t^1 \setminus V^1 = V_t^0$, then $e^1 \in \{O(\delta(V_s^0))\}$ and

$|\bar{c}_{e^1}(T^1)| = |\bar{c}_{e^1}(T^0)|$ [from 5.3.10]

$\leq |\bar{c}_{e^0}(T^0)|$ [from the choice of e^0].

If $h(e^1) \in V^1$ then $e^1 \in \{O(\gamma(V_s^0))\}$ and

$|\bar{c}_{e^1}(T^1)| = -\bar{c}_{e^1}(T^1)$ [from 5.3.0]

$= -[\bar{c}_{e^1}(T^0) - |\bar{c}_{e^0}(T^0)|]$ [from 5.3.10]

$\leq |\bar{c}_{e^0}(T^0)|$ [from (I)]

<u>If $e^1 \in \{u(\delta(V_t^1))\}$</u>, then $x^1_{e^1} = x^o_{e^1} = u_{e^1}$.

If $t(e^1) \in V_t^1 \setminus V^1 = V_t^o$, then $e^1 \in \{u(\delta(V_t^o))\}$ and

$$|\bar{c}_{e^1}(T^1)| = |\bar{c}_{e^1}(T^o)| \qquad \text{[from 5.3.10]}$$

$$\leq |\bar{c}_{e^o}(T^o)| \qquad \text{[from the choice of } e^o\text{]}.$$

If $t(e^1) \in V^1$, then $e^1 \in \{u(\gamma(V_s^o))\}$ and

$$|\bar{c}_{e^1}(T^1)| = \bar{c}_{e^1}(T^1) \qquad \text{[from 5.3.0]}$$

$$= \bar{c}_{e^1}(T^o) + |\bar{c}_{e^o}(T^o)| \qquad \text{[from 5.3.10]}$$

$$\leq |\bar{c}_{e^o}(T^o)| \qquad \text{[from (I)]}.$$

(c) If $\{O(\delta(V_t^1))\} = \phi$, then there is nothing to prove; otherwise let $f \in \{O(\delta(V_t^1))\}$ then :

if $t(f) \in V_t^1 \setminus V^1 = V_t^o$, then $f \in \{O(\delta(V_t^o))\}$, $f \neq f^o$ and also from 5.3.10 we have $\bar{c}_f(T^1) = \bar{c}_f(T^o)$. Therefore :

$$\bar{c}_f(T^1) - \bar{c}_{e^1}(T^1) = \bar{c}_f(T^0) - |\bar{c}_{e^1}(T^1)|$$

$$\geq \bar{c}_f(T^0) - |\bar{c}_{e^0}(T^0)| \qquad \text{[from (b)]}$$

$$\geq 0 \qquad \text{[from (II)]}$$

and if $t(f) \in V^1$, then $f \in \{O(\gamma(V_s{}^0))\} \cup \{f^0\}$ and furthermore from 5.3.10 we have $\bar{c}_f(T^1) = \bar{c}_f(T^0) + |\bar{c}_{e^0}(T^0)|$; thus:

$$\bar{c}_f(T^1) - |\bar{c}_{e^1}(T^1)| = \bar{c}_f(T^0) + |\bar{c}_{e^0}(T^0)| - |\bar{c}_{e^1}(T^1)|$$

$$\geq \bar{c}_f(T^0) \qquad \text{[from (b)]}$$

$$\text{but } \bar{c}_f(T^0) \begin{cases} = 0 & \text{if } f^0 \in \{O(\delta(V_t{}^1))\} \text{ and } f = f^0 \\ \geq 0 & \text{otherwise} \end{cases} \qquad \text{[from (I)]}$$

and thus the result follows.

(d) If $\{u(\delta(V_s{}^1))\} = \phi$, then we are done, otherwise let $e \in \{u(\delta(V_s{}^1))\}$ then:

if $h(e) \in V_t^1 \setminus V^1$, then $f \in \{u(\delta(V_s{}^0))\}$ and furthermore from 5.3.10

we have $\bar{c}_e(T^1) = \bar{c}_e(T^0)$ and therefore;

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| = \bar{c}_e(T^0) + |\bar{c}_{e^1}(T^1)|$$

$$\leq \bar{c}_e(T^0) + |\bar{c}_{e^0}(T^0)| \qquad \text{[from (b)]}$$

$$\leq 0 \qquad \text{[from (III)]}$$

and if $h(e) \in V^1$, then $e \in \{u(\gamma(V_s{}^0))\} \cup \{f^0\}$. Moreover, from 5.3.10 we have $\bar{c}_e(T^1) = \bar{c}_e(T^0) - |\bar{c}_{e^0}(T^0)|$; thus:

$$\bar{c}_e(T^1) + |\bar{c}_{e^1}(T^1)| = \bar{c}_e(T^0) - |\bar{c}_{e^0}(T^0)| + |\bar{c}_{e^1}(T^1)|$$

$$\leq \bar{c}_e(T^0) \qquad \text{[from (b)]}$$

$$\text{but } \bar{c}_e(T^0) \begin{cases} = 0 & \text{if } f^0 \in \{u(\delta(V_s{}^1))\} \text{ and } e = f^0 \\ \leq 0 & \text{otherwise} \end{cases} \qquad \text{[from (I)]}$$

and thus the result follows, which completes the proof of the theorem.

Lemma 5.3.2. If T is a tree satisfying the conditions of the theorem 5.3.1 then its associated tree solution X with $x_g = \alpha$ is an optimal solution of amount α:

Proof. Let $G_1 = (V, E \cup \{g\})$ be the digraph of CP, then clearly to prove the lemma it is enough to show that X is an optimal solution to the following problem $P(\alpha)$.

$$P(\alpha) \begin{cases} \min & \sum(c_e x_e : e \in E \cup \{g\}) \\ \text{s.t.} & \\ & v(X) = 0 \quad \text{for all } v \in V, \\ & x_g = \alpha \\ & 0 \le x_e \le u_e \quad \text{for all } e \in E. \end{cases}$$

If $\{O(\delta(V_s))\} \cup \{u(\delta(V_t))\} = \phi$, then X is an optimal flow to $P(\alpha)$; otherwise let $u_g = \alpha$ and perform a degenerate simplex pivot on T by entering the most negative reduced cost edge and deleting the blocking edge g. Let $\hat{X} = X$ and $\hat{Y}$ be the associated tree solution and vertex numbering of the resulting tree $\hat{T}$. Then as in the proof of theorem 5.3.1, we can show that there is no pivot eligible edge relative to $\hat{T}$, which means that the current feasible solution $X = \hat{X}$ is also an optimal solution to $P(\alpha)$.

Theorem 5.3.1 simply means that, in solving the circulation problem by the network simplex method, if an initial feasible tree T^0 satisfying the conditions of the theorem is available, then by pivoting on the most negative reduced cost edge, without a specific choice of leaving edge and regardless of the type of pivot (i.e. degenerate or otherwise),

the resulting tree will satisfy the conditions of theorem 5.3.1. Thus during the algorithm the negative reduced cost edges relative to the current tree T will be only the edges in the set $\{u(\delta(V_t))\} \cup \{O(\delta(V_s))\}$, which guarantees that the unique circuit formed by adding the entering edge to the tree T will contain the return edge g and will be directed in the direction of g and hence if the pivot is non-degenerate then the amount of the flow on the edge g will increase by the amount of flow change "θ" around the circuit. But since $\sum(u_e : h(e) = t) = b^+$; thus for any feasible flow of CP we have $x_g \leq b^+$ and if the capacity vector of MCMFP is an integral vector or equivalently the capacity and supply-demand vectors of the MCFP are integral and the flow vector associated with the initial tree is also integral, then during a non-degenerate pivot, the amount of flow on the return edge g will increase by an integral amount $\theta \geq 1$; thus the flow of maximum amount will be obtained in no more than b^+ non-degenerate pivots. But from lemma 5.3.2 such a flow will be a minimum cost flow of maximum amount. Moreover, if in addition the initial tree is also strongly feasible and strong feasibility[†] is maintained throughout the algorithm, then during a degenerate pivot the leaving edge will be an element of $E(T_t)$ and hence the cardinality of the new V_t will be at least one less than that of the old V_t; hence there could be no more that $|V| - 2$ consecutive degenerate pivots and therefore the following is proven:

[†]It is possible to maintain strong feasibility because it depends only on the choice of leaving edge and the results of theorem 5.3.1 are independent of the choice of leaving edge.

Theorem 5.3.2. Let $T^0, T^1, T^2, \ldots, T^m$ be a sequence of strongly feasible trees for a circulation problem with integral capacities, such that T^0 satisfies the conditions of theorem 5.3.1 and its associated tree solution is also integral, then if the most negative reduced cost edge rule is used to choose the entering edges, it follows that $m \leq (|V| - 2) \cdot b^+$.

An important special case of the MCFP is the n by n optimum assignment (OA) problem on a bipartite graph $G' = (V', E')$ with bipartition $V' = (I, J)$ in which $b^+ = \sum(b_v : v \in J) = |J| = n$. If the OA problem is solved as a circulation problem by the strongly feasible network simplex method using the most negative reduced cost edge rule and starting with the strongly feasible tree satisfying the conditions of theorem 5.3.1, then an optimal assignment will be obtained in no more than n non-degenerate pivots and since for all $e \in E'$, $u_e = \infty$ thus relative to the current T, $\{u(\delta(V_t))\} = \phi$. Hence by theorem 5.3.1 the pivot eligible edges are only those in $\{O(\delta(V_s))\}$ and hence if e is the entering edge then $h(e) \in V_t \setminus I = (V_t \cap J) \cup \{t\}$. It follows from the strong feasibility assumption that if the pivot is degenerate then, $h(e) \neq t$ and that the leaving edge is an element of $E(T_t)$; thus each degenerate pivot will decrease the cardinality of $V_t \cap J$ by at least 1. Furthermore, since $(V_t \cap J) \subseteq J$, therefore there can be no more than $|J| = n$ consecutive degenerate pivots; hence the following is proved.

Corollary 5.3.1. If the corresponding circulation problem of an n by n OA problem is solved by the method just described using the most negative reduced cost edge rule and starting with a strongly

feasible tree satisfying the conditions of theorem 5.3.1, then an optimal assignment will be reached in no more than n^2 pivots.

§5.4. INITIAL TREE FOR THE CIRCULATION PROBLEM

Throughout the last section we have assumed that an initial (strongly) feasible tree T^0 satisfying the conditions of theorem 5.3.1 is at hand. In what follows we will show how to obtain such a tree if one is not readily available. If the initial tree T^0 and its associated tree solution X^0 is given and if $x_g{}^0 = \alpha$, then from lemma 5.3.2, X^0 is a minimum cost flow of amount α. Thus to find T^0 we will first find a feasible tree $\hat{T}$ such that its associated tree solution $\hat{X}$ is a minimum cost flow of amount zero (i.e. $\hat{x}_g = 0$) and then we will obtain T^0 from $\hat{T}$. To find $\hat{T}$ and $\hat{X}$ it is clear that we must solve the following MCFP on the digraph $G_1 = (V, E \cup \{g\})$ of CP.

$$P(0) \quad \begin{cases} \min & \sum(c_e x_e : e \in E \cup \{g\}) \\ \text{s.t.} & \\ & v(X) = 0 \qquad \text{for all } v \in V, \\ & x_g = 0, \\ & 0 \le x_e \le u_e \qquad \text{for all } e \in E, \end{cases}$$

or equivalently we must solve the following problem.

$$P'(0) \quad \begin{cases} \min & \sum(c_e x_e : e \in E) \\ \text{s.t.} & \\ & v(X) = 0 \quad \text{for all } v \in V, \\ & 0 \le x_e \le u_e \text{ for all } e \in E. \end{cases}$$

Clearly $X = 0$ is a feasible flow for $P'(0)$ and it is well known that if G (or equivalently[†] the digraph $G' = (V',E')$ of the original MCFP) does not contain negative cost dicircuits then $X = 0$ is also an optimal flow. Start with a spanning arborescence and perform strongly feasible simplex pivots using one ~~of the~~ preventing stalling rules of previous chapters until the optimal tree $\hat{T}$ and optimal flow $\hat{X}$ of amount zero and the optimal vertex numbering $\hat{Y}$ are obtained. If G' does not contain negative cost dicircuits then all of the pivots will be degenerate and thus $\hat{T}$ and $\hat{X} = X = 0$ will be obtained in $O(f(|V|))$ time where $f(|V|)$ is a low degree polynomial in $|V|$. On the other hand if G' contains negative cost dicircuits and a minimum cost flow $\hat{X}$ of amount zero is not readily available then the amount of work to obtain $\hat{T}$ may be exponential in the problem size; thus hereafter we will assume that G' does not contain negative cost dicircuits or that a minimum cost flow $\hat{X}$ of amount zero is given. In most real life MCFP's for one reason or another (e.g. $C \ge 0$ or G' is a (directed) bipartite graph and etc.) the negative cost dicircuits are not present and thus the assumption that G' does not contain negative cost dicircuits is not as big a restriction as it may seem to be. If $G' = (V',E')$ is a bipartite graph with bipartition $V' = (S,T)$ such that $\delta(S) = E'$ then $\hat{T}$ can very easily be obtained as

---

[†]It is easily seen that G does not contain negative cost dicircuits if and only if G' does not.

follows:

1. Let $\hat{T} = \{s\}$ and $y_s = 0$.

2. For every $v \in S$ add v and the unique edge $(s,v) \in E^s$ to $\hat{T}$ and let $\hat{y}_v = 0$.

3. For every $v \in T$ find an edge e such that $c_e = \min\{c_f : h(f) = v\}$, then add e and v to $\hat{T}$ and let $\hat{y}_v = c_e$.

4. Let $\hat{y}_u = \min\{y_v : v \in \hat{T}\}$ then add the unique edge $(u,t) \in E^t$ and t to $\hat{T}$ and let $\hat{y}_t = \hat{y}_u$.

The resulting $\hat{T}$, $\hat{Y}$ and $\hat{X} = 0$ are the desired tree, vertex numbering and tree solution respectively.

Now if the restriction $x_g = 0$ of $P(0)$ is dropped, then, relative to $\hat{T}$, the return edge g has negative reduced cost and it is the only such edge. Perform a (strongly feasible) simplex pivot on q and let T^o be the resulting tree and X^o and Y^o be its associated tree solution and vertex numbering respectively. Then:

<u>Lemma 5.4.1</u>. The tree T^o satisfies the conditions of theorem 5.3.1.

<u>Proof</u>. (I) ($\Rightarrow$): If $e \in \{0(\delta(V_s^o))\} \cup \{u(\delta(V_t^o))\}$ then from the choice of M it is easily seen that e has a negative reduced cost.

(I) ($\Leftarrow$): To show this note that,

$$y_v^o = \begin{cases} \hat{y}_v & \text{if } v \in V_s^o, \\ \hat{y}_v + |\bar{c}_g(\hat{T})| & \text{if } v \in V_t^o. \end{cases} \qquad 5.4.1$$

Now let $A = \{\gamma(V_s^o)\} \cup \{\gamma(V_t^o)\}$ and $B = \{u(\delta(V_s^o))\} \cup \{O(\delta(V_t^o))\}$, then $E = \{O(\delta(V_s^o))\} \cup \{u(\delta(V_t^o))\} \cup A \cup B$.
If $e \in A \cap E(T^o)$ then the reduced cost of e is zero and if $\hat{f}$ was deleted from $\hat{T}$ when g entered, then clearly $\hat{f} \in B$ and thus $[A \setminus E(T^o)] \cap E(\hat{T}) = \phi$; hence for all $e \in A \setminus E(T^o)$ we have $x_e^o = x_e$.
Now from 5.4.1 and properties of T, it is obvious that elements of A $E(T^o)$ are not pivot eligible edges. If $e \in B$, then its pivot eligibility is guaranteed from the choice of M in $c_g = -M$.

To show that T^o satisfies the conditions (II) and (III) of theorem 5.3.1 we will first show that, if e^o is the most negative reduced cost edge relative to T^o, then $|\bar{c}_{e^o}(T^o)| \leq |\bar{c}_g(\hat{T})|$ 5.4.2

If $e^o \in \{O(\delta(V_s^o))\}$, then $x_{e^o}^o = \hat{x}_{e^o} = 0$ and $\bar{c}_{e^o}(T^o) < 0$, thus;

$$|\bar{c}_{e^o}(T^o)| = -\bar{c}_{e^o}(T^o)$$

$$= -[\bar{c}_{e^o}(\hat{T}) - |\bar{c}_g(\hat{T})|] \qquad \text{[from 5.4.1]}$$

$$\leq |\bar{c}_g(\hat{T})| \qquad [\bar{c}_{e^o}(T) \geq 0]$$

If $e^o \in \{u(\delta(V_s{}^o))\}$, then $x^o_{e^o} = \hat{x}_{e^o} = u_{e^o}$ and $\bar{c}_{e^o}(T^o) > 0$, thus;

$$|\bar{c}_{e^o}(T^o)| = \bar{c}_{e^o}(T^o)$$

$$= \bar{c}_{e^o}(\hat{T}) + |\bar{c}_g(\hat{T})| \qquad \text{[from 5.4.1]}$$

$$\leq |\bar{c}_g(\hat{T})| \qquad [\bar{c}_{e^o}(\hat{T}) \leq 0]$$

and thus the validity of 5.4.1 is established.

(II): Let $f \in \{O(\delta(V_t{}^o))\}$, then

$$\bar{c}_f(T^o) - |\bar{c}_{e^o}(T^o)| \geq \bar{c}_f(T^o) - |\bar{c}_g(\hat{T})| \qquad \text{[from 5.4.2]}$$

$$= \bar{c}_f(\hat{T}) + |\bar{c}_g(\hat{T})| - |\bar{c}_g(\hat{T})| \qquad \text{[from 5.4.1]}$$

$$= \bar{c}_f(\hat{T}) \begin{cases} = 0 & \text{if } f = \hat{f} \\ \geq 0 & \text{otherwise.} \end{cases} \qquad \text{[from the properties of } \hat{T}]$$

(III): Let $e \in \{u(\delta(V_s{}^o))\}$, then

$$\bar{c}_e(T^o) + |\bar{c}_{e^o}(T^o)| \leq \bar{c}_e(T^o) + |\bar{c}_g(\hat{T})| \qquad \text{[from 5.4.2]}$$

$$\leq [\bar{c}_e(T^o) - |\bar{c}_g(\hat{T})|] + |\bar{c}_g(\hat{T})| \quad \text{[from 5.4.1]}$$

$$= \bar{c}_e(T^o) \begin{cases} = 0 & \text{if } e = \hat{f}, \\ \leq 0 & \text{otherwise.} \end{cases}$$

The latest inequality follows from the properties of T and this completes the proof of the lemma.

§5.5 COMMENTS

(5.5.1) As it was shown for the MFP, it is easily demonstrated that in solving the MCFP by the method of this chapter, if an initial tree T^o satisfying the conditions of the theorem 5.3.1 is given then it is not necessary to use the method of strongly feasible trees in order to prevent cycling and stalling. Indeed cycling and stalling can be prevented by using one of the leaving edge rules introduced for the MFP namely:

Rule 1: Delete the nearest blocking edge to root s. Note that if T^o is strongly feasible then this rule will maintain strong feasibility.

Rule 2: Delete the furthest blocking edge to root s i.e. delete the nearest blocking edge to t.

Rule 3: If f_1 and f_2 are the nearest blocking edges to s and t respectively and d_1 and d_2 are their distances from s and t respectively then delete f_1 if and only if $d_1 \leq d_2$. Otherwise choose f_2 to be deleted.

As in lemma 5.2.1 and corollary 5.2.1 it is easily shown that starting with T^0, if any one of these leaving edge rules is used, then the number of consecutive degenerate pivots cannot exceed $|E| - |V| + 1$. Since with the method of strongly feasible trees the number of consecutive degenerate pivots is bounded above by $|V|$, therefore this method is (theoretically) preferable to all of the above rules.

(5.5.2) In the two phase network simplex method, another phase I method is the following: On the digraph $G' = (V', E')$ of the MCFP let $S \subseteq V'$ ($T \subseteq V'$) be the set of supply (demand) vertices. Let T^1 be a tree having an edge set directed from root r to each $v \in [V' \setminus \{r\}] \setminus S$ and an edge directed from each $v \in S \setminus \{r\}$ to r. Let $G_1 = (V_1 = V', E_1)$ be the digraph so obtained. Now T^1 is a strongly feasible basis for the MCFP on G_1 having the same supply-demand vector. The cost vector for the new problem is $d = (d_e : e \in E_1)$, where $d_e = 0$ for $e \in E'$ and $d_e = 1$ for $e \in E_1 \setminus E'$.

It is easily seen that the application of the strongly feasible network simplex method to this problem yields a spanning tree T^0 of G' with the associated optimal basic solution $(x_e^0 : e \in E_1)$. If $x_e^0 = 0$ for all $e \in E_1 \setminus E'$, then $X^0 = (x_e^0 : e \in E')$ is a feasible solution to the MCFP.

This phase I method can also be modified so that the solution technique of the previous sections can apply to the new problem and if the MCFP is feasible then the optimal solution of the new problem will readily provide an optimal solution to the MCFP.

Create two dummy vertices s and t and create an artificial edge directed from each $v \in S$ to s and let E^s denote the set of all such edges and create an edge directed from t to each $v \in T$; let E^t denote the set of all such edges and let g be an edge directed from s to t. Let $G = (V, E \cup \{g\})$ denote the new digraph so obtained. Let P1 be an MCFP on G with the following b, C and u vectors.

$$b_v = \begin{cases} b_v' & \text{if } v \in V', \\ 0 & \text{if } v = s \text{ or } t, \end{cases}$$

$$c_e = \begin{cases} c_e' & \text{if } e \in E', \\ 0 & \text{if } e \in E^s \cup E^t, \\ M & \text{if } e = g, \end{cases}$$

and

$$u_e = \begin{cases} u_e' & \text{if } e \in E', \\ b_v & \text{if } e \in E^t \text{ and } h(e) = v, \\ -b_v & \text{if } e \in E^s \text{ and } t(e) = v, \\ b^+ & \text{if } e = g, \end{cases}$$

where b',C' and u' respectively are the supply-demand vector, the cost vector and the capacity vector of the MCFP respectively and M and b^+ are defined as before. Clearly $(x_e: e \in E \cup \{g\})$ where $x_e = 0$ for all $e \in E'$ and $x_e = u_e$ for all $e \in E^t \cup E^s \cup \{g\}$ is a feasible flow to P1 and it is not difficult to demonstrate that, if $(x_e^*: e \in E \cup \{g\})$ is an optimal flow to P1 with $x_e^* = 0$ for all $e \in E^t \cup E^s \cup \{g\}$, then $X^* = (x_e^*: e \in E)$ is an optimal flow to the MCFP. If $x_e^* > 0$ for some $e \in E^t \cup E^s \cup \{g\}$, then the MCFP is infeasible. Furthermore it can be seen that; P1 can be used in place of CP in previous sections to obtain similar results and that an initial tree can be found as in §5.4.

(5.5.3) It is clear that the method of this chapter can not start with a given feasible solution to the MCFP and indeed it cannot start with any feasible solution to the MCFP at all, but this restriction except in post optimality analysis is not very severe in the sense that, in most MCMP's an initial feasible solution is not readily available and that one must be found with a phase I method.

(5.5.4.) This method resembles the labelling method of Ford and Fulkerson [23], Edmonds and Karp [20] in that the non-degenerate pivots are essentially flow augmentations on s-t flow augmenting paths. Furthermore the worst case computation bound of this method is "theoretically" comparable with that of labelling methods.

REFERENCES

[1] Ali, A.L., R.V. Helgason, J.L. Kennington and H.S. Lall, "Primal Simplex Network Codes: State-of-the-Art Implementation Technology", Networks, Vol. 8 (1978), pp. 315-339.

[2] Avis, D and V. Chvatal, "Notes on Bland's pivoting rule", Mathematical Programming Study 8 (1978), pp. 24-34.

[3] Balinski, M.L. and R.E. Gomory, "A primal method for the assignment and transportation problems", Management Science Vol. 10, No. 3, April 1964, pp. 578-593.

[4] Barr, R.S., F. Glover and D. Klingman, "The alternating basis algorithm for assignment problem", Mathematical Programming 13 (1977), pp. 1-13.

[5] Beale, E.M.L., "Cycling in the dual simplex algorithm", Naval Research Logistics Quarterly, Vol. 2, No. 4, 1955.

[6] Bland, R.G., "New finite pivoting rules for the simplex method", Mathematics of Operations Research, Vol. 2, No. 2, May 1977, pp. 103-107.

[7] Bradley, G.H., G.G. Brown and G.W. Graves, "Design and implementation of large scale primal trans shipment algorithms", Management Science, Vol. 24, No. 1, Sept. 1977, pp. 1-34.

[8] Burkardt, R.E. and U. Zimmerman, "Weakly admissible transformations for solving algebraic assignment and transportation problems", Report 77-3 Mathematisches Institut, University of Cologne, 1977.

[9] Charnes, A., "Optimality and degeneracy in linear programming", Econometrica, 20 (1952), pp. 160-170.

[10] Cunningham, W.H., "A network simplex method", Mathematical Programming 11 (1976), pp. 105-116.

[11] Cunningham, W.H., "Theoretical properties of the network simplex method", Mathematics of Operations Research, Vol. 4, No. 2, May 1979, pp. 196-208.

[12] Cunningham, W.H. and A.B. Marsh, III, "A primal algorithm for optimum matching", Mathematical Programming Study 8, (1978), pp. 50-72.

[13] Dantzig, G.B., "Upper bounds, secondary constraints, and block triangularity in linear programming", Econometrica 23 (1955), pp. 174-183.

[14] Dantzig, G.B., "Linear programming and extensions", Princeton University Press, Princeton, New Jersey (1963).

[15] Dantzig, G.B., A. Orden and P. Wolfe, "A generalized simplex method for minimizing a linear form under linear inequalities", Pacific Journal of Mathematics Vol. 5, No. 2 (1955), and RM-1264, The Rand Corporation. April (1954).

[16] Desler, J.F. and S.L. Hakimi, "A graph-theoretic approach to a class of integer programming problems", Operations Research Vol. 17 (1969), pp. 1017-1033.

[17] Dijkstra, E.W., "A note on two problems in connection with graphs", Numerische Mathematik, Vol. 1(1959), p. 269.

[18] Edmonds, J. "Path, trees, and flowers", Canadian Journal of Mathematics, Vol. 17 (1965), pp. 449-467.

[19] Edmonds, J., "Exponential growth of the simplex method for shortest path problems", University of Waterloo (1970), Unpublished.

[20] Edmonds, J. and R.M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems", Journal of the Association for Computing Machinery, Vol. 19, No. 2, April 1972, pp. 248-264.

[21] Elam, J., F. Glover and D. Klingman, "A strongly convergent primal simplex algorithm for generalized networks", Mathematics of Operations Research, Vol. 4, No. 1, Feb. 1979, pp. 39-59.

[22] Engquist, M. "Successive shortest path algorithms for network problems", to appear.

[23] Ford, Jr., L.R. and D.R. Fulkerson, "Flows in networks", Princeton University Press, Princeton, New Jersey (1962).

[24] Gass, S.L., "Linear programming: Methods and applications", McGraw Hill, 4th edition (1975).

[25] Gassner, B.J., "Cycling in the transportation problem", Naval Research Logistic Quarterly 11 (1964), pp. 43-58.

[26] Glover, F., D. Klingman and J. Stutz, "Augmented threaded index method for network optimization", Infor. Vol. 12, No. 3, Oct. 1974, pp. 293-298.

[27] Hadley, G., "Linear programming", Addison-Wesley (1962).

[28] Hitchcock, F.L., "The distribution of products from several sources to numerous localities", Journal of Mathematics and Physics, Vol. 20, No. 2 (April 1941), p. 224.

[29] Hoffman, A., "Cycling in the simplex algorithm", National Bureau of Standards Report, Washington, D.C. 1953.

[30] Hopcroft, J.E. and R.M. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs", SIAM J. of Computing, Vol. 2, No. 4, Dec. 1973, pp. 225-231.

[31] Hung, M.S. and W.O. Rom, "Solving the assignment problem by relaxation", Journal of Operations Research, Vol. 28, No. 4, July-Aug. 1980, pp. 969-982.

[32] Johnson, E.L., "Networks and basic solutions", Journal of Operations Research, Vol. 14, No. 4, July-Aug. 1966, pp. 619-623.

[33] Johnson, E.L., "Programming in networks and graphs", ORC 65-1, Operations Research Center, University of California, Berkeley (1965).

[34] Klee, V. and G.J. Minty, "How good is the simplex algorithm", O. Shisha, ed., "Inequalities III", Academic Press, New York, 1972, pp. 159-175.

[35] Klingman, D. and R.Russell, "Solving Constrained transportation problems", Operations Research, Vol. 23, No. 1, Jan.-Feb. 1975, pp. 91-106.

[36] Koopmans, T.C., "Optimum utilization of the transportation system", Proceedings of the International Statistical Conferences, Washington, D.C. (1947), Published in Vol. 5 (1949), p. 136. (Also published in Scientific Papers of Tjalling C. Koopmans, Springer-Verlag, New York (1970), p. 184.)

[37] Kuhn, H.W., "The Hungarian method for the assignment problem", Naval Research Logistics Quarterly, Vol. 12 (1955), p. 269.

[38] Lawler, E., "Combinatorial optimization, networks and matroids", Holt, Rinehart and Winston (1976).

[39] Mulvey, J.M., "Pivot strategies for primal-simplex network codes", Journal of the Association for Computing Machinery, Vol. 25, No. 2, April 1978, pp. 266-270.

[40] Munkres, J., "Algorithm for assignment and transportation problem", Journal of SIAM, Vol. 5 (1957), pp. 32-38.

[41] Orden, A., "The trans-shipment problem", Management Science 3 (1956), pp. 276-285.

[42] Queyranne, M., "Theoretical efficiency of the algorithm "Capacity" for the maximum flow problem", Mathematics of Operations Research, Vol. 5, No. 2, May 1980, pp. 258-266.

[43] Silver, R., "An algorithm for the assignment problem", Communications of ACM, Vol. 3 (1960), pp. 603-606.

[44] Wagner, H.M., "On a class of capacitated transportation problem", Management Science 5 (1959), pp. 304-318.

[45] Wolfe, P., "A technique for resolving degeneracy in linear programming", Journal of SIAM 11 (1963), pp. 205-211.

[46] Zadeh, N., "A bad network problem for the simplex method and other minimum cost flow algorithms", Mathematical Programming 5 (1973), pp. 255-266.

END

1 6 1 0 6 8 3

FIN