ADDIS ABABA UNIVERSITY

DEPARTMENT OF MATHEMATICS

SUBMITTED TO PARTIAL FULFILMENT OF MSC DEGREE

GRADUATE SEMINAR ON MINIMUM COST NETWORK FLOW PROBLEM AND NETWORK SIMPLEX METHOD

Submitted by : Tamiru Kuraga

Advisor : Berhanu Guta (Ph.D)

July 2015

Table of Contents

ABSTRACT

The Minimum Cost Network Flow (MCNF) Problem is to send flow from a set of supply or source nodes, through the arcs of a network, to a set of demand or destination nodes, at minimum total cost, and without violating the lower and upper bounds on flows through the arcs. The MCNF framework is particularly broad, and may be used to model a number of more specialized network problems including Assignment, Transportation, Trans-shipment problems, the Shortest Path Problem and the Maximum Flow problem. The Network Simplex Method (NSM) is a part of the bounded variable primal simplex algorithm, specifically for the MCF problem. The basis is represented as a rooted spanning tree of the underlying network, in which arcs represent variables. The method iterates towards an optimal solution by exchanging basic and non-basic arcs. At each interaction, an entering arc is selected by some pricing strategy, and an arc to leave the basis is ascertained. The construction of a new basis tree is called the pivot. There are many strategies for selecting the entering arc, and these determine the speed of solution.

This seminar report contains three chapters. In the first chapter I tried to discuss some basic idea of network systems such as their mathematical character, modeling the minimum cost network flow problem in standard linear programming form, graphical representation and matrices representation of the minimum cost network flow. Moreover the general minimum cost network flow problems and Application of the modeling in the second chapter and one of the solutions methodologies (network simplex method) is discussed in the third chapter, I apply the concepts of the first and the second chapter for the generating optimum solution for minimum cost Network flow problems using simplex algorism.

Introduction to Networking

Networks are familiar diagrams in electrical theory; they are easily visualized in transportation or communication systems like roads, railways, pipelines, nerves or blood vessels, A large variety of mathematical problems are present by networks. Particularly those which involves sequential operations or different but related states or stages, are conveniently described diagrammatically as networks. Sometimes a problem with no such apparent structure assume a mathematical form which is best understood and solve by interpreting it as a network.

A network, in its more generalized and abstract sense is called a **graph**. In recent years graph theory has been a subject of much study and research by mathematicians and has found more and more applications in diverse areas. In the field of operation research graph theory play a particularly important role as quite often the problem of finding an optimal solution can be looked upon as a problem of choosing the best sequence of operations out of a finite number of alternatives which can be represented by a graph.

Acknowledgement

I would like to express my gratitude to my advisor Dr.Berhanu Guta for his unlimited help suggestions and comments for this seminar report to be completed.

I also express my thanks to Dr. Semu Metku, Dr. Taddess and Dr. Yirgalem to their motivations, efforts and unlimited interest to help me a lot to make this seminar report completed .

I owe special recognition to my friends Samuale, Biniam, Seleshe, Abiy and Paulose for their assistance. Without these people achievement would have been impossible.

My sincere thanks is also to my brothers Endashaw who helped me a lot to complete this paper.

Tamiru Kuraga

Addis Ababa University

July 2015

NOTATIONS

- An activity in a given network is represented by an arrow (edge).
- An event in a given network is represented by a vertex.
- Matrix by A
- Cost of edge i to j by C_{ij}
- Supply/demand values by ”b”, $b \in R$
- We denote the vertex by V_i, i=1,2.3. . . . n
- G is a graph
- Let “e” be the edge from vertex i to vertex j, then we write e = (i j) or e_{ij}.
- Y_i is simplex multiplier or node potential
- r_{ij} is reduced cost
- X_B is basic edge
- X_N is non-basic arc
- C_N is cost of basic arc
- a_{ij} is entries of a matrices

Chapter 1

Network systems

1.1 Network Flows

A wide variety of engineering and management problems involve optimization of network flows that is, how objects move through a network. Examples include coordination of trucks in a transportation system, routing of packets in a communication network, and sequencing of legs for air travel. Such problems often involve few indivisible objects, and this leads to a finite set of feasible solutions. For example, consider the problem of finding a minimal cost sequence of legs for air travel from A.A to Cairo. Though there are many routes that will get a traveler from one place to the other, the number is finite. This may appear as a striking difference that distinguishes network flow problems from linear programs. The latter always involves a polyhedral set of feasible solutions. Surprisingly, as we will see in this chapter, network flows problems can often be formulated and solved as linear programs.

In relation of studying this problem, the network system is very helpful for the fact that the mathematical characteristics of the network models are so special that by exploiting these structural properties, we use it for the study of our problem.

An important economic application of network system is the description of projects, where projects are set of finite sequence of activities. Since projects are represented by networks, the first section of this seminar is devoted to the network systems, since the network system is the basic element.

1.1.1 Network

A directed graph, or network, $G = (V, E)$ consists of a set V of vertices and a set $E \in V \times V$ of edges. When the relation E is symmetric, G is called an **undirected** graph, and we can write edges as unordered pairs $\{u, v\} \in E$ for $u, v \in V$. The degree of vertex $u \in V$ in graph G is the number $|\{v \in V : (u, v) \in E \text{ or } (v, u) \in E\}|$ of other vertices connected to it by an edge.

A walk from u ∈ V to w ∈ V is a sequence of vertices $v_1, \dots v_k \in V$ such that $v_1 = u$, $v_k = w$, and $(v_i, v_{i+1}) \in E$ for $i = 1, \dots, k-1$.

In a directed graph, we can also consider an undirected walk where $(v_i, v_{i+1}) \in E$ or $(v_{i+1}, v_i) \in E$ for $i = 1, \dots, k - 1$. A walk is a path if $v_1, \dots, v_k$ are pair wise distinct, and a cycle if $v_1, \dots, v_{k-1}$ are pair wise distinct and $v_k = v_1$. A graph that does not contain any cycles is called **acyclic**. A graph is called **connected** if for every pair of vertices u, v ∈ V there is an undirected path from u to v.

Networks are especially convenient for modeling because of their simple non mathematical structure that can be easily depicted with a graph. This simplicity also reaps benefits with regard to algorithmic efficiency. An endless variety of models involve optimization over such networks. Many cannot be expressed in any straightforward algebraic way or are very difficult to solve. Our discussion starts with a particular class of network optimization models in which the decision variables represent the amounts of flow on the arcs, and the constraints are limited to two kinds: simple bounds on the flows and conservation of flow at the nodes. They are especially easy to describe and solve, yet are widely applicable. Some of their benefits extend to certain generalizations of the network flow form, which we also touch upon.

In a network, each arc or edge is associated with some numerical value (real number), variably called its cost, weight, length (distance) or some other variable depending on the application. It is denoted as c_{ij} for each arc (i, j) ∈ A. Weights of the arcs are very important because some algorithms impose further restrictions on weights. A path is a sequence of distinct arcs/edges connecting two specified nodes in a network. Each arc/edge must have exactly one node in common with its predecessor in the sequence and no node may be visited more than once. A path may be either directed or undirected. If a path begins and ends at the same node it is called a cycle. In an undirected network there are many cycles. However, note that the definition of path must satisfy.

Network optimization is a special type of linear programming model. Network models have three main advantages over linear programming:

1. They can be solved very quickly. Problems whose linear program would have 1000 rows and 30,000 columns can be solved in a matter of seconds. This allows network models to be used in many applications (such as real-time decision making) for which linear programming would be inappropriate.
2. They have naturally integer solutions. By recognizing that a problem can be formulated as a network program, it is possible to solve special types of integer programs without resorting to the ineffective and time consuming integer programming algorithms.
3. They are intuitive. Network models provide a language for talking about problems that are much more intuitive than the \variables, objective, and constraints" language of linear and integer programming. Of course these advantages come with a drawback: network models cannot formulate the wide range of models that linear and integer programs can. However, they occur often enough that they form an important tool for real decision making

1.2 Defining some basic terminologies that I used in this seminar report

Under this sub section, some words and terms that are used frequently in this seminar report are defined as:-

Definition 1:- A network is a flow diagram consisting of activities and events connected logically and sequentially, or a network is a graph which consists of a set of nodes (vertices) and a set of arcs (edges or links) connecting various pairs of nodes where each arc has specified orientation (in the case where it is termed as directed graph).

Definition 2:- A node is said to be initial node or source if it has only outgoing arrows or edges.

Definition 3:- A node or vertex is called terminal or sink, if it has only incoming arrows or edges.

Definition 4:- An event in a given network is the commencement or completion of an activity.

Definition 5:- An activity in a given network is the actual performance of a given task, that is, it is a work required to complete the specific event.

Definition 6:- Let G = (V .E) be a directed graph, and Let F = $\{e_1,e_2,e_3,e_4,\ldots\ldots\ldots,e_n\}$ in E be a finite sequence of edges , then the sequence F is called a chain if and only if there is a sequence of vertices $\{v_1,v_2,v_3,\ldots\ldots v_n\}$ in V.

Definition 7:- A path in a network is a directed chain.

Definition 8:- A directed chain beginning and ending at the same node is a directed cycle.

Definition 9:- A network is connected provided that there is a directed path between every pair of node.

Definition 10:- A network that is without any directed cycle is called acyclic network,

Definition 11:- A network consisting of directed cycle is called cyclic network.

- **Direct path:** Any two or more nodes connected directly or there exists a directed path that connects every node.

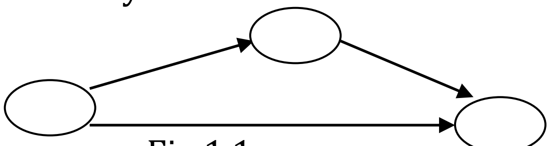

Fig 1.1

- **Indirect path:** for any two or more nodes, there is no single edge direct simple path that connects the nodes.

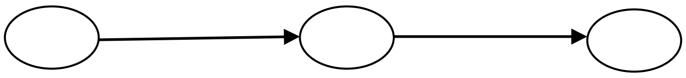

Fig 1.2

Definition 12:- If e = (i, j) then e is said to be incident from vertex i to vertex j.

Definition 13:- Two edges e_1=(i, j) and e_2 =(r, s) are said to be adjacent if j= s or s =i.

Definition 14:- A cycle connecting vertex i and j is called a loop if i = j.

Graphical guideline for network /graphical modeling of network

The following guideline should be followed for the better drawing of network diagrams.

- Arrows are not vectors; they are never used to show duration through its length.
- Orientation between arrows should be chosen to suit drafting convenience.
- As far as possible, we should use straight arrow, curved arrows are not preferable.

1.3 Representation of networks by matrices:

There are several representation of networks in a matrix form, under this sub section we focus on some important representation of networks in a matrix form,

1.3.1 Representation by Incidence matrix

Define the incidence matrix $A = (a_{ij})$ as

$$a_{ij} = \begin{cases} 1 & \text{if arc } p_j \text{ starting in node i} \\ -1 & \text{if arc } p_j \text{ ending in node i} \\ 0 & \text{otherwise} \end{cases}$$

Example 1.1 consider the following network whether the small circles represent nodes with their representative numbers in them and each e_j represent s_j^{th} arrow or arc.

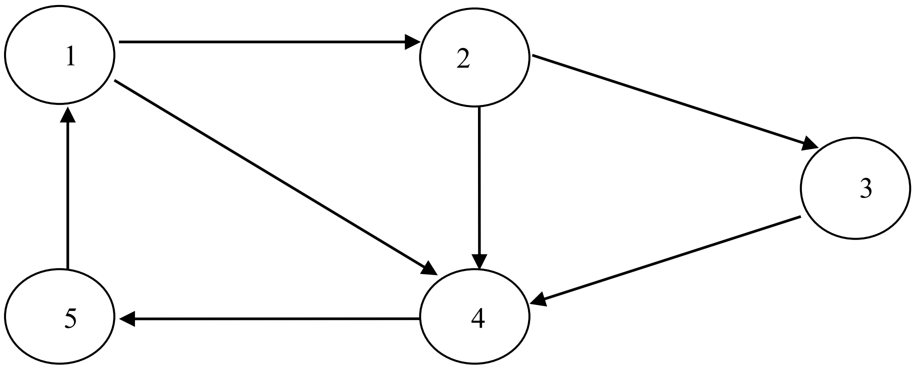

Fig 1.3

The corresponding matrix representation of the above fig 1.3 is given as follows.

$$\begin{array}{ccccccc} X_{12} & X_{14} & X_{23} & X_{24} & X_{34} & X_{45} & X_{51} \\ \end{array}$$
$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ -1 & -1 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Note

1. The first row consists of the numbers 0 and 1 only which shows that V_1 is a source.

2. The 5^{th} row consists of the number 0 and -1 only which shows that V_5 is a sink

3. We see that it is not connected and not cyclic that is start from v_j for any j and one cannot return to v_j.

1.3.2 Representation by adjacency matrix

Define the adjacency matrix A= (a_{ij}), i= 1,2 ………… ,n

j= 1,2,…………,n

$$a_{ij} = \begin{cases} 1 & \text{if there is an arrow from vertex } i \text{ to vertex } j \\ 0 & \text{otherwise} \end{cases}$$

Example 1.2

From the network in the above figure 1.3, we can represent by adjacency **matrix.**

$$\begin{array}{c} \begin{array}{ccccccc} X_{12} & X_{14} & X_{23} & X_{24} & X_{34} & X_{45} & X_{51} \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Note:

1. V_1 is a source since the first column has elements only 0, no arrow comes from any other vertices to v_1.

2. v_5 is a sink since all elements of the 5^{th} row are 0 only, that is no arrow emanates fromv_6

3. The number of 1's in each row shows the number of arrows coming to v_j

4. Given the matrix one can draw a network representing it.

1.3.3 Representation of a network by value matrix.

Let f_i: (i,j) $\longrightarrow$ R^+ be a function that assigns appositive real number to each arrow (i,j) in a network. Let c_{ij} be defined by:

$$c_{ij} = \begin{cases} f(i,j) & \text{if there is an arrow } (i\,,j) \text{ connecting the vertices } v_j \text{ to vj} \\ \infty & \text{if there is no arrow } (i\,,j) \text{ connecting the vertices } v_i \text{ to } v_j \end{cases}$$

We consider the following

1. We have only one sink node 1, since the last row has only ∞ and source (node 8), since the first column has only ∞.
2. In row 1 the number of numbers that are not ∞ shows number of arrows from v_j
3. In column j the number of numbers that are not ∞ shows the number of arrows to $v_{j.}$
4. We see that since the diagonal of the matrix is only ∞ we conclude that the given Network is not cyclic

Chapter 2

Minimum cost network flow

2.1 Minimum cost network

As I have already mentioned it a wide variety of engineering and management problems involve optimization of network flows .that is, how objects move through a network, including coordination of trucks in a transportation system, routing of packets in a communication network, and sequencing of legs for air travel. Such problems often involve few indivisible objects, and this leads to a finite set of feasible solutions. For example, consider the problem of finding a minimal cost sequence of legs for air travel from Addis Ababa to Jimma though many routes that will get a traveler from one place to the other, the number is finite. This may appear as a striking difference that distinguishes network flows problems from linear programs. As we will see in this chapter, network flows problems can often be formulated and solved as linear programs.

Graphs can be used to model many real networked systems. For example, in modeling air travel, each node might represent an airport, and each edge a route taken by some flight. Note that, to solve a specific problem, one often requires more information than the topology captured by a graph. For example, to minimize cost of air travel, one would need to know costs of tickets for various routes. When there are many paths from root or Origen to the destination or sink, naturally want to take the cheapest or shortest one. If c_{ij} represents the cost or distance of travel along arc (i j). Then you want to solve :

$$\textbf{Minimize} \sum c_{ij}$$

Example 2.1:- the numbers 2, 3 & 4 are different place that found between the origin 1and distination5,The arcs represent routes bettween any two placeses .

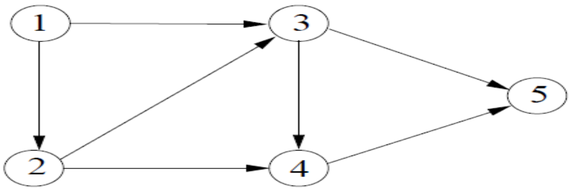

Figure 2.1

Where, the number of arcs limited to those that represent paths within the network. Clearly, if the network is undirected, then the shortest path problem is much the same as the minimum spanning tree problem, except that it minimizes over paths rather than spanning trees. You can also imagine the problem on a directed network, however. In the directed shortest path problem, you must be able to travel a path from origin to destination without going "backwards" along any arc.

The min-cost flow problem consists in determining the most economical way to transport a certain amount of good (e.g. oil, oranges, cars ...) from one or more production facilities to one or more consumption facilities, through a given transportation network (like :- a hydraulic network, a distribution network, a road network etc.). It should be emphasized from the outset that the mathematical model lends itself to represent a variety of problems that have nothing to do with the shipment of goods, and therefore we use the more abstract notion of flow.

As usual, the nodes of the network may be associated with physical places (cities, warehouses, industrial facilities, stations ...), and the arcs to one-way communication links (road sections, railways, pipe line ...) among these places. Note that one does not lose of generality considering the arcs oriented (i.e., one-way) rather than un oriented (i.e., two-way). In fact, each two-way arc can be represented by means of a pair of arcs pointing in opposite directions between the same two nodes.

All of the above models are special types of network flow problems. The minimum cost network flow model represent the broadest class of problem that can be solved much faster than liner programming while still retaining such nice properties as integrality of solution and appeal of concept.

- Like the maximum flow problem, it considers flows in network with capacities.
- Like the shortest path problem, it considers a cost for flow through an arc,
- like the transportation problem ,it allows multiple sources and destinations,

In fact all of these problems can be seen as special cases of the minimum cost network flow problem.

Fundamental information concerns about minimum cost network flow:

- The amount of flow produced or consumed at each node;
- Transportation costs between two nodes;
- Possibly, an upper limit on maximum flow on each arc (i.e., capacity).

For each node i, i =1,... , n, an (integer) number $\mathbf{b_i}$ is given, representing the amount of flow produced; (if $b_i > 0$) or consumed (if $b_i < 0$) at node i. The nodes that produce flow are sometimes referred to as sources and $\mathbf{b_i}$ as supply. Nodes that consume flow are called **sink** and $\mathbf{b_i}$ as **demand**. If $\mathbf{b_i=0}$, node **i** dose not consume nor produce flow that is it is a **transit node**. Note that, this classification into three types of nodes is completely independent of the structure of the network, but it is defined only by the values of supply and demand.

2.2 Spanning Tree:

In a network, spanning tree is a collection of arcs that form a tree and connect to every node. A spanning tree is a useful pattern for cheaply interconnecting all the nodes in a network. The number of arcs in the spanning tree equals the number of nodes minus one, and between any two nodes there is a unique path along the tree. Since there are only finitely many spanning trees within a network, this minimization is well defined. However, a large network may contain a very large number of spanning trees, so it is not immediately clear how much work may be involved in finding a minimum one. Applications include the design of various types of distribution networks in which the nodes represent cities, centers etc.; and edges represent communication links (fiber glass phone lines, data transmission lines, cable TV lines, etc.), high voltage power transmission lines, natural gas or crude oil pipelines, water pipelines, highways, etc. The objective is to design a network that connects all the nodes using the minimum length of cable or pipe or other resource. The minimum cost spanning tree problem also appears as a sub problem in algorithms for many routing problems such as the traveling salesman problem.

Some basic properties of spanning tree

- In a graph with "n" nodes, a tree with n-1 arcs from a spanning tree.
- If an arc added to as panning tree then unique cycle is created.
- If an arc is removed from a spanning tree, then the tree is decomposed in to two new trees.
- A **circuit** in a network is a collection of arcs that are connected together in a circle, while a **tree** is a collection of arcs that are connected together without containing any circuits or Tree is formed by a connected subset of the graph that contains no cycles

Economic Interpretation

- Imagine you are the only company that produces the commodity what price should you sell the commodity for at each node?

2.3 Characteristic of Minimum Cost Network Flow

- Any network flow problem can be cast as a *minimum-cost network flow program*. A min-cost network flow program has the following characteristic :
- **Variables**

  The unknown flows in the arcs, the x_i, are the variables.
- **Flow conservation at the nodes**

  The total flow into a node equals the total flow out of a node. It makes things easier later if we follow the convention of writing the flow conservation equation at a node as:

$$\sum x_k - \sum x_k = 0$$
$$flow\ in - flow\ out$$

- **Source and sink nodes**

Some nodes are connections to the environment surrounding the network. At these entryway nodes, there may be a net gain of flow into the network (source node), or a net loss of flow out of the network (sink node). To emphasize that flow conservation still holds at source and sink nodes shown on the network diagram. The phantom arc will be an inflow for a source node, and an out flow for a sink node.

Now we use a similar convention for writing the flow conservation equation at the source or sink node:

$$\underset{flow\ in\ -\ flow\ out}{\sum x_k - \sum x_k} = b\mathrm{i}$$

When written this way, **b**$_i$ is a positive constant for a source node, and a negative constant for a sink node. The magnitude of **b**$_i$ is the amount of flow in the phantom source or sink arc. Note also that the relationship may not be an equality relationship: inequalities are common for sources and sinks. For example, the flow of water exiting a supply network must be at least 20 liters per minute, or the flow of oil entering a refinery network must not exceed 10,000 barrels per day.

- When all of the flow conservation equations (or inequalities) are written following the outflows – inflows convention, it is easy to see what type of node is associated with each relationship by looking at the value of the right hand side constant. The constant will be zero for a simple flow conserving node, positive for a source node, and negative for a sink node.

Bounds on the arc flows

There may be upper and lower bounds on the flows in the arcs (i.e. the variables in the model. $\mathbf{x}_j \geq \mathbf{l}_j$ is a lower bound on an arc flow, and $\mathbf{x}_j \leq \mathbf{u}_j$ is an upper bound on an arc flow. For example, the maximum flow of water through a particular pipe is limited because of the pipe diameter and interior roughness, so an upper bound is applied.

Upper bounds are easy to understand, but why might there be a nonzero lower bound? This might represent the minimum required production rate at a factory (e.g. 250 vehicles per day required at the output arc of an automobile plant) or a minimum flow rate through steam piping to prevent condensation.

- The default arc flow bounds are a lower bound of zero and no upper bound. It is not unusual for almost all of the arcs in a network model to have the default bounds, with only a few arcs having specified upper or lower bounds, typically those at the "edges" of the network, representing, say, upper limits on the rates of raw materials flowing into the network and lower limits on the production the rate at the main outflow from the network.

Some modeling systems will allow a negative lower bound on an arc flow. My strong advice is do not do this! The meaning of a negative flow is a backwards flow in the arc. This immediately destroys the best feature of a network model, the intuitive understanding of the system that you gain by looking at the network diagram. You expect the flows to follow the directions indicated by the arrowheads, but the arrowheads may lie if you permit negative flows. There are other ways to accommodate two-way flow, if necessary, such as pairing oppositely oriented arcs between two nodes. Cost per unit of flow. There is a cost per unit of flow, $\mathbf{c}_j$ associated with each arc. In many network models, the cost per unit of flow is zero for most of the arcs, with costs being typically associated with arcs at the "edges" of the network. The default value of cj is zero.

Economic Interpretation

- If the price was lower, then you would lose money
- If the price was higher, then a competitor could undercut your price

Objective function

- In a minimum cost network flow problem, the objective is to find the values of the variables (the $\mathbf{x}_j$) that minimize the total cost of the flows over the network:

  Minimize $\sum_{k=1}^{n} c_k x_k$

Of course, the solution must respect all of the constraints: flow conservation at the nodes, and the upper and lower flow bounds on the arcs.

Economic Interpretation

- You don’t want to lose business, so you also plan to ship over *each arc*
- You want to ship as much as possible
- You must also adjust the rest of your schedule to conform with demand

Network problems can be cast as minimum cost network flow programs.

There are three parameters associated with each arc**:**

- The lower flow bound, l
- The upper flow bound, and u
- The cost per unit of flow. c

The arc labeling convention that we will use shows a triple of numbers in square brackets, [l, u, c]. l is the lower bound on the flow in the arc, with a default value of zero if not explicitly specified; u is the upper bound on the flow in the arc, with a default value of infinity if not explicitly specified; c is the cost per unit of flow in the arc, with a default value of zero if not explicitly specified.

- For example, an arc having a lower flow bound of zero, and upper flow bound of 25, and a cost per unit of flow of $6 would be labeled [0, 25, 6]. If the upper and flow bounds on the arc differ, then the node relationship is an inequality.

✓ As we have seen, the network diagram contains all the information needed to derive an associated linear programming model via a straightforward mechanical writing of the constraints and objective function. For this reason, we will assume from here onward that a properly labeled network diagram is the formulation.

✓ The linear programming version of the network model has some very interesting properties. Look at the left hand sides of all of the constraints. What do you notice? All of the coefficients there are 0, +1, or –1. This is because all of the node relationships are simple summations of flows, and the remainders of the constraints are simple bounds. But this fact has some very important consequences.

- ✓ The second important consequence is this: if all of the constraint right hand side values are integers and if all of the pivot operations are simple additions and subtractions, then we can guarantee that the solution values of the variables at the optimum will also all be integers. This is known as the unimodularity property.
- ✓ Make the origin node a source of exactly n units of flow, with no cost per unit of flow. The label on the phantom arc should be [n, n, 0].
- Make each destination node a sink of exactly one unit of flow, with no cost per unit of flow. The label on the phantom arc for each destination node should be [1,1,0].
- The unimodularity property will permit only integer amounts of flow in an arc. Some arcs will have no flow, and some will have a flow equal to some positive integer less than or equal to n, the maximum amount of flow introduced at the origin node. If you mark the arcs that have a positive flow, then the marked arcs will form a tree on the diagram. The shortest route from the origin to each destination node is actually discovered backwards: trace the route from the destination node back to the origin via marked nodes. Because the marked arcs form a tree, there is only one shortest route for each destination node. The routes to several destination nodes may share an arc, which is why the flow in some arcs may be greater than one.

Network problems can be cast as minimum cost network flow programs.

b_i = net supply (are flow out – are flow in) at node i

u_k = capacity of are k(i.j)

The value of bi is determined by the nature of node i. in particular

$b_i > 0$ if i is a supply node;

$b_i < 0$ if i is a demand node;

$b_i = 0$ if i is a transshipment node. Also, let

K_{oi} = set of arcs leaving node i

K_{Ti} = set arcs terminating at node i.

$$\text{Minimize} \sum_{k=1}^{n} c_k x_k$$

$$Subject\ to \sum_{k \in ko_i} x_k - \sum_{k \in T_i} x_k = b_i\, for\ all\ i = 1 \ldots m$$

$$0 \leq x_k \leq u_k for\ all\ k = 1 \ldots n$$

2.4 Some Application of the model:

- Let **C_{ij}** denote the cost of going from node i to node j, Then the minimum path problem is (p) Find the least cost path.
- Let **C_{ij}** be represented by the time taken to travel from node i to node j, then the minimal path problem is (p) Find the minimal duration path
- Let **C_{ij}** denote the time needed to perform activity (i,j) then, the minimal path problem is (P) Find the minimal time needed to complete the whole project
- Let **C_{ij}** be represented by the distance between nodes i and j than the minimal path problem is given by (P) Find the shortest route between node i and node j.
- Connect a number of base stations minimizing the total cost of construction
- Find a time schedule (start and completion times) for activities in a project
- Find how much goods should be transported from each supplier to each point of demand, using which links in a transport system

Note: Generally in this applications, C_{ij} Further more nodes may not be connected directly, which can be indicated by letting **$C_{ij} = \infty$,** and some times, the so called triangle inequality **$C_{ij} + C_{kj}$** doesn't hold for all possible **i,j** and **k.** Here we assume that if a network contains cycles, the total lengths of every cycles path is not negative, otherwise by repeatedly traversing this cycle, the objective function can be made arbitrarily small, therefore without imposing any further restriction, on the problem as stated, an unbounded solution occur.

The MCF formulation is particularly broad and can be used as a template upon which a number of network problems may be modeled. The following are some examples.

1. Shortest Paths

2. Maximum Flow
3. The Assignment Problem
4. The Transportation Problem
5. The Transshipment Problem
6. Travel problems

All of the problems described so far in this section involve finding the shortest or cheapest subset of arcs from among a finite, though perhaps very large, collection. You will see later than some of them are reasonably easy to solve, while others are in a certain sense impossibly hard

Chapter 3

Network Simplex Method

3.1 Contemporary Solution Methods

A variety of solution methods for the MCNF problem are described in the literature. These may be categorized into a number of principal approaches, namely primal, dual, primal dual and scaling algorithms. For example, negative cycle and primal simplex (as its name suggests) are both primal algorithms, while dual methods include successive shortest paths and a specialization of the dual simplex method for LP problems. Ford's Out-of-Kilter algorithm is perhaps the best known primal-dual method. But I try to analyzing the Primal Network Simplex Method in greater detail.

These are some of the algorithm that used to finding minimum cost network flow

1. Negative Cycle Algorithm
2. Primal-Dual and Out-of-Kilter Algorithms
3. Specialization of LP Dual Simplex
4. Scaling Algorithms
5. Relaxation Algorithms
6. Network Simplex Method

To use the best method that help us to solve any kind of network is based on the type of constrained and objective function like multiple objective function ,linearity and coefficient of the decision ,supply or demand and cost. Based on these factors the speed of algorithm.

To compare and contrast the speed of the algorithm is difficult to calculate since we perform using computer but we can measure the speed through the number of iteration and steps that we use to find the optimal solution.

So that we try to measure the worst case or the maximum number of that needed to solve linear programming problem having 'n ' variable and 'm' constraints:

1. Using simplex method

   $O(mn)$ number of on arithmetic operation each round and **2^n-1** number of iteration

2. Using network simplex method

   $O(n^2)$ number of operation and **'m'** number of iteration

3.2 Network Simplex Method

We look more closely at a linear programming problem in its standard form, and describe the simplex method, which is one of the most successful approaches for finding the optimal solution for such problems. Let us concentrate on the problem of finding a vector **x** solving such that:

$$\text{Minimize } \mathbf{cx}$$
$$\text{subject to } \mathbf{Ax = b, x \geq 0.}$$

There is no restriction in assuming that the linear system $\mathbf{Ax = b}$ is solvable, for otherwise, there would be no feasible vectors. Moreover, if $\mathbf{A}$ is not a full-rank matrix, we can select a sub matrix A^* by eliminating several rows of $\mathbf{A}$, and the corresponding components of $\mathbf{b^*}$, so that the new matrix $\mathbf{A^*}$ has full rank. In this case we obtain the new, equivalent, LPP

$$\text{Minimize } \mathbf{cx}$$
$$\text{subject to } \mathbf{A^* x = b^* ,x \geq 0,}$$

Where $\mathbf{b}^*$ is the sub vector of $\mathbf{b}$ obtained by eliminating the components corresponding to the rows of $\mathbf{A}$ we have previously discarded. This new LPP is equivalent to the initial one in the sense that they both have the same optimal solutions, but the matrix $\mathbf{A^*}$ for the reduced problem is a full-rank matrix. We shall therefore assume, without loss of generality, that the rank of the $\mathbf{m \times n}$ matrix $\mathbf{A}$ is m (remember $\mathbf{m = n}$) and that the linear system $\mathbf{A^*x = b^*}$ is solvable .There are special feasible vectors that play a central role in the simplex method. These are the solutions of the linear system $Ax = b$ with nonnegative and at least $n - m$ null components. In fact, all of these external points or basic solutions obtained by solving all square $m \times m$ linear systems $Ax = b$ where $n - m$ components of x are set to zero, and discarding those solutions with at least one negative component.

The very special linear structure of an LPP enables us to concentrate on these basic solutions when looking for optimal solutions. we can find optimal solutions for an LPP by looking at all solutions of the system $Ax = b$ with at least $n - m$ zeros, discarding those with some negative component and by computing the cost of the remaining ones, decide on the optimal vector. This process would indeed lead us to one optimal solution, but the simplex method aims to organize these computations in a judicious way so that we can reach the optimal solution as soon as possible without having to go through an exhaustive analysis of all extreme points. In some cases, though, the simplex method actually goes through all basic solutions before finding an external solution. This situation is however rare.

3.3 For network flow problems the simplex algorithm is simplified in the following way

Initialization

- Find a spanning tree for the given network.
- Find the basic solution for the spanning tree using flow balance equation
- $A^*X_B = b^*$.
- If all flows are non-negative $X_B \geq 0$ then the spanning tree flow is basic feasible solution.

Main Iterative step

- The main step start with the basic feasible solution .To check optimality
- First compute the simplex multipliers (y_i)
- Secondly calculate the reduce cost (r_{ij}) by using the simplex multipliers. then
- Check optimality condition

- » $r_{ij} = 0$ if $0 \leq x_{ij} \leq u_{ij}$
- » $r_{ij} \geq 0$ if $x_{ij}=0$
- » $r_{ij} \leq 0$ if $X_{ij}=u_{ij}$

If it satisfies the optimality condition the current basic feasible solution is optimal

If $\mathbf{r_{ij}}$ has some negative components, we can, in principle, lower the cost by letting those components of non-basic arc $\mathbf{X_N}$ become positive.

Choose an arc (i j) with negative reduced cost to send as much flow as possible around the basic cycle and adjust the new spanning tree.

Laving Cycle Rule

The arc entering the spanning tree creates a cycle. In this cycle, choose a leaving are that satisfies.

- The flow is in the opposite direction as the flow of the entering arc
- It has minimum flow of all such arcs.

Update Flows

- Let $\mathbf{X}_{min}$ be the flow of the arc leaving the spanning tree. Note that if both entering and leaving arcs are included, there is a cycle. Only for arcs in this cycle: add $\mathbf{X}_{min}$ to the arcs whose flow is opposite to the leaving are, and subtract $\mathbf{X}_{min}$ from the flows corresponding to arcs whose in the same direction as the leaving are.

Update the simplex multipliers

- If the entering arc goes from the non-root sub tree to the root sub tree, then decrement the duals in the non-root tree by the value of the cost variable of the entering arc.
- Update the reduced cost and check optimality condition

Stopping criterion

- If all components of r turn out to be nonnegative, there is no way to lower the cost, and the present extreme point is indeed optimal. We have found a solution for our problem.

No solution

- Even if we can lower and lower the cost, (some components of $\mathbf{r} \leq \mathbf{0}$) and none of the components in $\mathbf{X}_b$ will ever reach zero. (on choice as leaving arc)The problem does not admit an optimal solution because we can reduce the cost indefinitely.

Network Simplex Method

- We look more closely abut LPP in its standard form, and describe the simplex method, which is one of the most successful approaches for finding the optimal solution for such problems.
- The network simplex algorithm maintains a feasible spanning tree structure at each iteration and successively transforms it into an improved spanning tree structure until it becomes optimal.

Example 3.1

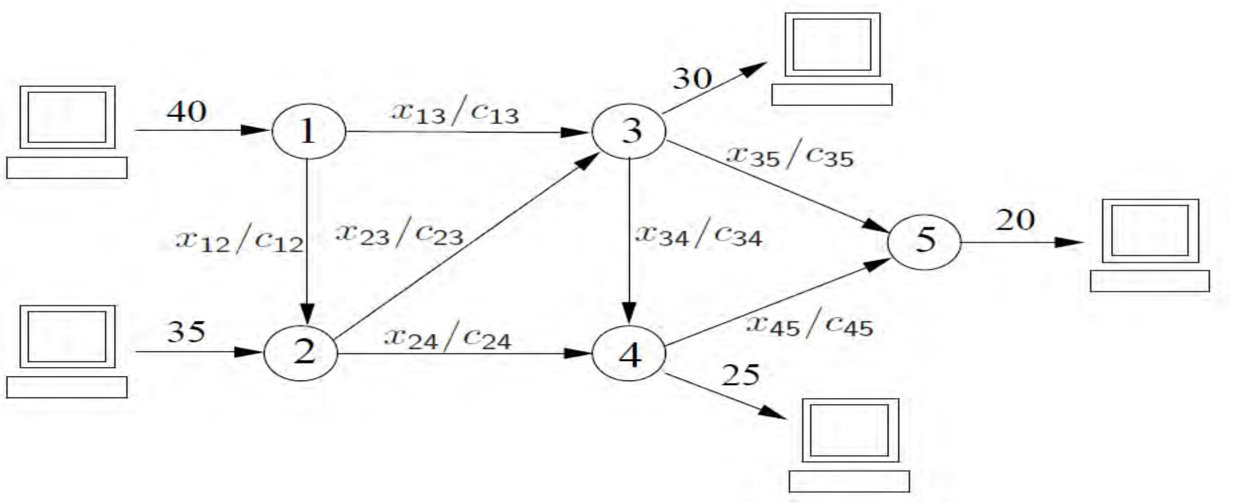

Fig 3.1

Data is sent from serves in nodes 1 and 2 to terminals in nodes 3, 4, 5.

Cost for traffic in the link between i and j is C_{ij}

Aim: minimize the total cost for the data traffic.

THE NETWORK OPTIMIZATION PROBLEMS CAST AS LLP

$$\text{Minimize} \sum_{\text{all arcs}} c_{ij}x_{ij}$$

$$\text{s.t} \quad x_{12}+x_{13}=40$$

$$-x_{12}+x_{23}+x_{24}=35$$

$$-x_{13}-x_{23}+x_{34}+x_{35}=-30$$

$$-x_{24}-x_{34}+x_{45}=-25$$

$$-x_{35}-x_{45}=-20$$

$$x_{ij}\geq 0 \quad \text{all flows in the links.}$$

$b_i=$ denotes external flow in/out to node i.

That is: Flow in if $b_i \geq 0$, and flow out if $b_i \leq 0$.

$C_{ij}=$ cost for traffic /flow

Modeling in graph

G = (N, B)

Where N = set of nodes.

B = set of arcs

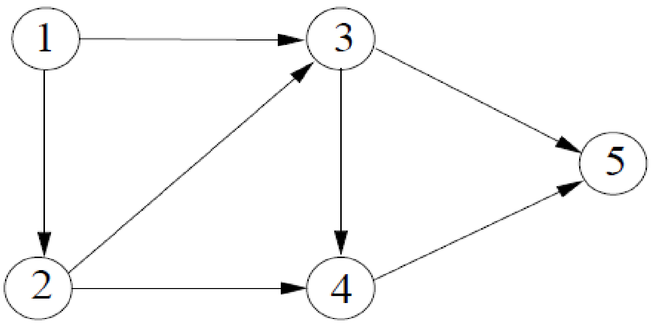

Fig 3.2

B = {(1, 2), (1, 3), (2, 3), (2, 4), (3, 4), (3, 5), (4, 5)}

ρ_1 ρ_2 ρ_3 ρ_4 ρ_5 ρ_6 ρ_7

N = {1, 2, 3, 4, 5}

b = { b_1, b_2, b_3, b_4, b_5}

= {40, -35, 30, -25, 20}

C_{ij} = { 2, 5, 2, 2, 1, 1, 2 }

{ C_{12}, C_{13}, C_{23} , C_{24}, C_{34}, C_{35}, C_{45}}

Then represent by incidence matrix A

x_{ij} = flow in arc (i, j)

If the flow x_{ij} goes in the direction i to j, then $x_{ij} \geq 0$.

Otherwise $x_{ij} \leq 0$.

Where x_{ij}- unity

$$X_{ik} = \begin{cases} 1 & \text{if } k \text{ € } (i, j) \\ -1 & \text{if } k \text{ € } (j, i) \\ 0 & \text{Otherwise} \end{cases}$$

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 \end{bmatrix}$$

It is easy to see that

$\alpha^T A = 0$ where $\alpha^T = (1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)^T$

The rows are thus linearly dependent and it is possible to eliminate the last row to obtain the reduced incidence matrix A*.

That means;

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 \end{bmatrix}$$

Some of the spanning tree of the network are :

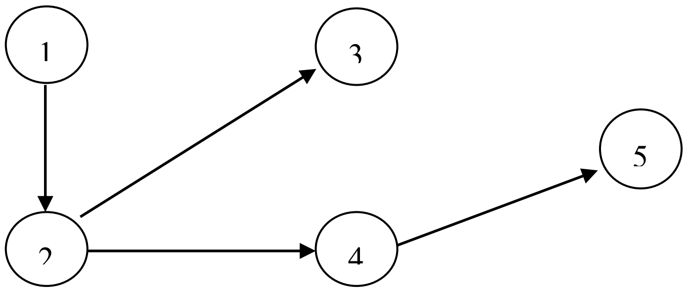

Fig 3.3

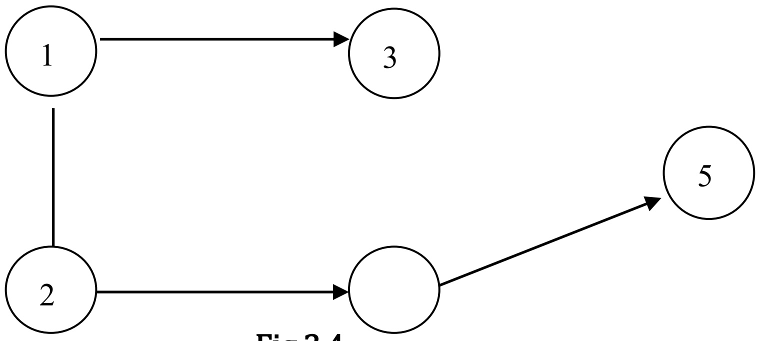

Fig 3.4

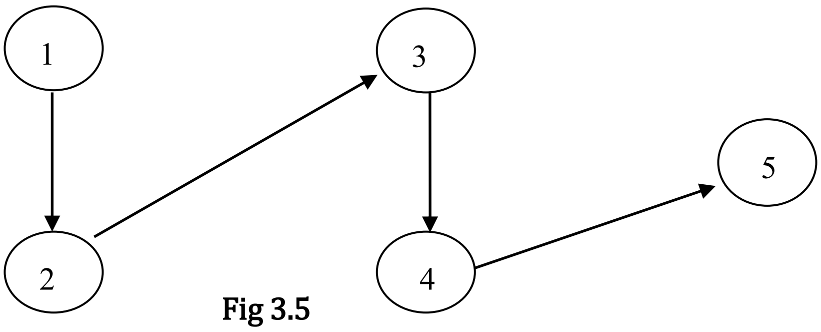

Fig 3.5

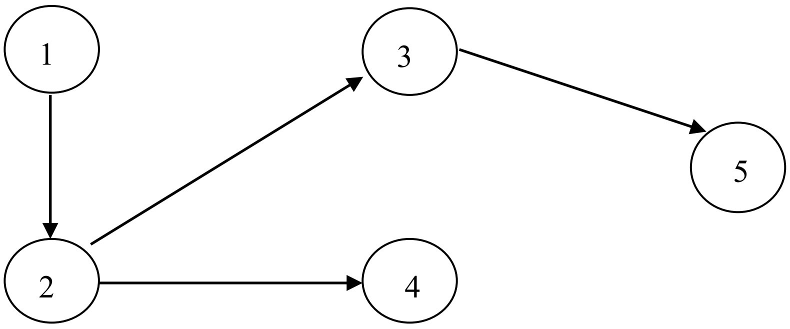

Fig 3.6

Let take one of the spanning trees and then apply network simplex on it.

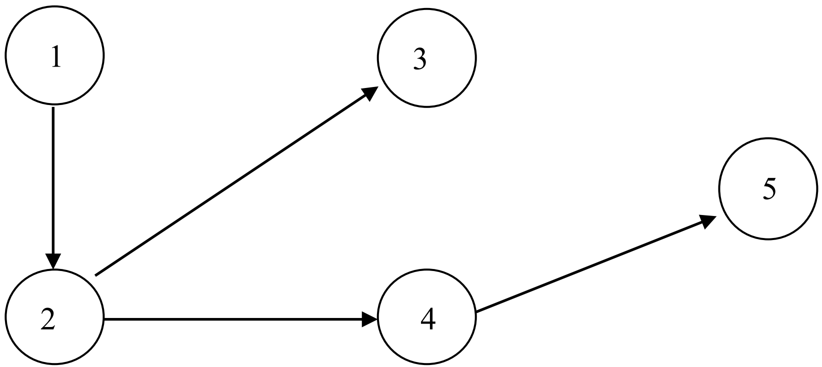

Fig 3.7

In this case

N = {1, 2, 3, 4, 5}

B = {(1, 2), (2, 3), (2, 4), (4, 5)} are the basic arcs and

B_N = {(1, 3), (3, 4),and (3, 5)} are the non-basic arcs.

Now determining feasible basic solution of the given spanning tree by using flow balance

$A_BX_B=b$

N = {1, 2, 3, 4, 5}

B = {(1, 2), (2, 3), (2, 4), (4, 5)} are the basic arcs and

B_N = {(1, 3), (3, 4),and (3, 5)} are the non-basic arcs

The incident matrix is given by

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

b = $(b_1, \quad b_2, \quad b_3, \quad b_4, \quad b_5)$

= 40, 35, 30, -25, 20

Now determining feasible basic solution of the given spanning tree by using flow balance

$$\mathbf{A_B X_B = b_B} \Rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{vmatrix} x_{12} \\ x_{23} \\ x_{24} \\ x_{45} \end{vmatrix} = \begin{bmatrix} 40 \\ 30 \\ -30 \\ -25 \end{bmatrix}$$

Determine basic solution

$X_{12} = 40$

$-X_{12} + X_{23} + X_{24} = 30$

$-X_{24} = -30$

$-X_{24} + X_{45} = -25$

THAT IS; $x_{12}=40$ $x_{23}=45$ $x_{24}=30$ and $x_{45}=20$

Thus

$$X_B = b^* \Rightarrow \begin{vmatrix} b_{12} \\ b_{23} \\ b_{24} \\ b_{45} \end{vmatrix} = \begin{bmatrix} 40 \\ 50 \\ 25 \\ 20 \end{bmatrix} \geq 0$$

X_B = feasible basic solution.

To check is this optimal?

Determining simplex multiplier

Assume $y_m=0$

$\mathbf{y_i - y_j = c_{ij}}$

$(i, j) \in X_B = \{(1, 2), (2, 3), (2, 4), (4, 5)\}$

$Y = \{y_1, y_2, y_3, y_4, y_5\}$

And $C_{ij} = \{ C_{12}, C_{13}, C_{23}, C_{24}, C_{34}, C_{35}, C_{45}\}$

$= (2, 5, 2, 2, 1, 1, 2)$

$$y_4 - y_5 = C_{45}$$

$$y_2 - y_3 = c_{23}$$

$$y_2 - y_4 = c_{24}$$

$$y_1 - y_2 = c_{12}$$

Since $y_5 = 0$

$$y_4 = 2$$

$$y_2 = 4$$

$$y_3 = 2$$

$$y_1 = 6$$

Determine reduced costs.

- $r_{ij} = c_{ij} - y_i + y_j \quad v \in$ (non-basic arcs)

$$A_N = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ -1 & 1 & 1 \\ 0 & -1 & 0 \end{bmatrix} v \in ((1\ 3),(3\ 4),(3\ 5))$$

Determine reduced costs

- $r_{13} = c_{13} - y_1 + y_3 = 5-6+2 = 1$
- $r_{34} = c_{34} - y_3 + y_4 = 1-2+2 = 1$
- $r_{35} = c_{35} - y_3 + y_5 = 1-2+0 = -1$
- Since: $r_{35} < 0$ then the current basic feasible solution (X_B) is not optimal.
- Thus x_{35} is entering arc and now we have to choose (Determine) the leaving arc by using quotient test.
- Since: $r_{35} < 0$
- Let $x_{35} = t =$ arcs (3,5) is enters the basis.

Determine the leaving arc by using quotient test

$$t^{max} = \min\left(\frac{\bar{b}_i}{\bar{a}_{ik}}, \bar{a}_{ik} > 0\right) = \frac{\bar{b}_p}{\bar{a}_{pk}}$$

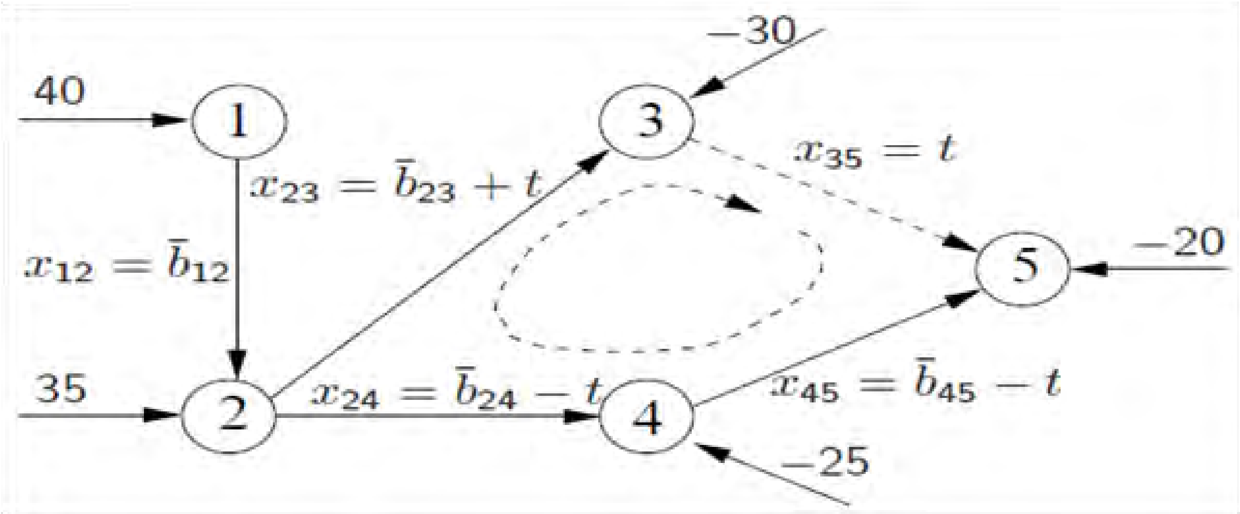

Fig 3.8

- (Determine) the leaving arc by using quotient test.
- $X_{23} = b^*_{23} + t = 30 + t$
- $X_{24} = b^*_{24} - t = 45 - t$
- $X_{45} = b^*_{45} - t = 20 - t$

$$X_B = b^* = \begin{bmatrix} 40 \\ 50 \\ 25 \\ 20 \end{bmatrix} \geq 0$$

$t_{max} = 20$ when $x_{45} = 0$

$X_B = \{(1, 2), (2, 4), (2, 3), (3, 5)$,

- which is a new basic feasible solution is this optimal
- **Simplex multiplicators**

$y_i - y_j = c_{ij}$ Be (all basic arcs)

$y_1 - y_2 = c_{12}$ assume $y_5 = 0$

$y_2 - y_4 = c_{24}$ $y_3 = c_{35} = 1$

$y_2 - y_3 = c_{23}$ $y_2 = c_{23} + y_3 = 2 + 1 = 3$

$y_4 - y_5 = C_{45}$ $y_4 = -c_{24} + y_2 = 3 - 2 = 1$

$Y_1 = C_{12} + Y_2 = 2 + 3 = 5$

$Y` = 5, 3, 1, 1, 0$

- **Reduced cost**

$r_{ij}= c_{ij}- y_i+ y_j$

for all (i, j) € (non-basic arc)

$X_N = \{(1, 3), (3, 4), (4, 5)\}$

$r_{13} = c_{13}- y_1 + y_3=5-5+1=1$

$r_{34} = c_{34}-y_3+y_4=1-1+1=1$

$r_{35} = c_{35}-y_3+y_5=2-0+1=1$

all $r_{ij} \in v > 0$,

Thus the basic feasible solution X_B is optimal

The significance of the network simplex algorithm

- It is a lot faster than the usual simplex algorithm. The number of pivots is the same, but each pivot is much faster
- It gives another view of the simplex algorithm, and its operations.
- It shows how network algorithms can be much faster.

ADVANTAGE OF SOLVING MINIMUM COST NETWORK FLOW PROBLEM

Contrast with the simplex algorithm

- The network simplex algorithm is the simplex algorithm, but without the tableaus
- Bases correspond to spanning trees
- The basic feasible solution is found by sending flow in arcs
- The reduced costs are found by finding the simplex multipliers explicitly.
- The leaving arc is found by sending flow around a cycle.

The Network simplex algorithm provides another opportunity to visualize the simplex algorithm. In this case, one can visualize the algorithm in multiple dimensions (that is lots of variables), as opposed to the lectures on geometry where we were restricted to two or three dimensions (that is, two or three variables).

We will not be using tableaus. Nevertheless, we will still compute basic feasible solutions as well as reduced costs. To carry out these computations, we will work directly on the network.

Conclusion

In this seminar report the first thing that is done is the an alizarin of network systems, which is done in two parts. The first part is concerned with modeling and representing network and the second part is concerning with solving the minimum cost network flow using network simplex method.

In modeling and representing network I try to discuss representing any network flow graphically and in matrix form.

In the second part of the seminar is a deal with the application of the first part of the seminar in such a way that the network system that discussed in the first part is taken to solve any minimum cost network flow problem using simplex algorism.

From the foregoing discussion, we draw the following conclusions

- The Minimum Cost Flow Problem has a rich history, and it forms a framework upon which a wide range of theoretical and practical problems may be formulated and solved.
- There exist, at present, three dominant solution paradigms, namely the Network Simplex Method, Relaxation algorithms and Scaling algorithms. The superiority of any one method is a highly contentious issue.
- The area of PC-based implementations of NSM is under-researched, but we believe that such implementations may offer a real alternative to the large scale codes now prevalent
- Simple pricing strategies, in general, out-perform their more complicated counterparts, mainly due to the elimination of expensive computational overhead.
- The entire field of network flow theory and practice offers a plethora of fertile research opportunities, exemplified by the dedication of the First DIMACS Implementation Challenge to the development of more efficient methods for the solution of the Minimum Cost Flow Problem.

Reference

- [1] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, “Network Flows: Theory, Algorithms and Applications,” Prentice Hall, Englewood Cliffs, New Jersey, U.S.A., 1993
- [2] R. Barr, F. Glover and D. Klingman, Enhancements to spanning tree labeling procedures for network optimization, INFOR, 17 (1979), 16–34.
- [3] E. Cohen and N. Megiddo, Algorithms and complexity analysis for some flow problems, Algorithmic
- [4] [De]R.Deumlich,Optimization theory 1
- [5] De.r.Deumlich,Optimization theory 2
- [6] H.M.Wagner, Principles of operation research.

  Project planning and control with PERT and CPM
- [7] Dr.B.C.Punmia,K.K.KHANDELWAL
- [8] G.MBUXEY,The vehicle scheduling problem and Monte Carlo simulation.
- [9] D.DUBOIS, G.BEL and M.LLIBRE

  a set of methods in transportation network synthesis and Analysis
- [10] PabloPedregal, introduction to optimization
- [11] SinglresuS.Rao Engineering Optimization
- [12] Ann-Brith Stromberg lecture note on shortest paths and network flow models
- [13] Hans JakobLiithi system modeling and optimization theory of linear programming.
- [14] Alessandro Agnetis and Marco Pranzo teaching material