Addis Ababa University
School of Graduate studies

Faculty of computer and mathematical sciences
Department of mathematics
Project on

Transportation and multicommodity minimal cost network flow problem

By:- Haftom G/anenya
Advisor:- Berhanu Guta (phD)

A project submitted to the office of graduate programs of Addis Ababa University in partial fulfillment of the requirement for the degree of masters of Science in mathematics

February, 2012
A.A, Ethiopia

ACKNOWLEDGEMENT

First and foremost, it is my pleasure to express my heart-felt appreciation and special gratitude to my advisor Dr. Berhanu Guta for his patience in repeatedly reading the draft manuscript of this project and for making constructive comments and suggestions from which I have immeasurably benefited in sharpening my understanding, predominantly, on the area I study.

Secondly, I would like to extend grateful acknowledgement to my family who support me all the time.

Last but not least, I am also indebted for my friends and to all those who helped me one way or another for the accomplishment of my study.

Haftom G/anenya
mujelektahe@gmail.com
A.A.U

2012

Introduction

Network models played an important role in the development of operational research since its origin. The classical transportation problem (one of the first problems solved in a computer in 1951) is a land mark in the history of operational research. A fundamental problem in a network theory is the minimum cost flow problem which means the movement of a commodity from sources that reside at one or more nodes in a network to meet demands at other nodes at a minimum cost of shipment. When the number of commodities is at least two it is called multicommodity minimal cost flow problem.

The multicommodity flow problem yields formulation of optimization problems that arises in industrial applications such as transportation, telecommunication and logistics. multicommodity flow problem is characterized by a set commodities to be routed through a network at a minimum cost. In practice, commodities may represent messages in telecommunications, vehicles in transportation or product goods in logistics. Each commodity has to be transported from one or several origin nodes to one or several destination nodes.

This paper contains two chapters. The first chapter devoted with some basic ideas of minimum cost flow problem and transportation problem. And the second chapter gives brief description of minimal cost multicommodity flow problem and minimal cost multicommodity transportation problems.

Abstract

The complete set of this paper focuses on minimal cost multicommodity network flow problem especially on transportation problem. decompose it in to minimal cost single commodity transportation problem based on their special structure formed by its constraints.

Table of content

Title **page**

Abstract

The complete set of this paper focuses on finding the optimal solution of minimum cost multicommodity network flow problems by decompose it in to minimum cost single commodity network flow problems based on the special structure formed by the constraints especially deals on transportation problem.

Chapter one
Minimum cost network flow problems

1.0. Preliminary concepts

In this section we introduce some of the basic definitions relating to network, graph and related notions.

Network:

A triple (V, E, C) is said to be network if $G = (V, E)$ is directed graph without directed cycle and passes exactly one source and at least one sink. The elements of V are called nodes or vertices. The elements of E are called arc or edge.

Directed graph:

A graph $G = (V, E)$ consists of a set V of nodes and a set E of pairs of distinct nodes from V called arcs. The numbers of nodes and arcs are denoted by N and A, and it is assume throughout that $1 \leq N < \infty$ and $0 \leq A < \infty$. An arc (i, j) is viewed as an ordered pair, and is to be distinguished from the pair(j, i). If (i, j) is an arc, we say that (i, j) is outgoing from node i and incoming to node j; we also say that j is an outward neighbor of i and that i is an inward neighbor of j . We say that arc (i, j) is incident to i and to$j$, and that i is the start node and j is the end node of the arc. We also say that i and j are the end nodes of arc(i, j).

Undirected graph: where an arc is associated with a pair of nodes regardless of order.

Source (initial) node: is a node which has only outgoing arrows or edges.

Terminal (sink) node: is a node which has only incoming arrows or edges.

Intermediate node: is a node which has both incoming and outgoing arrows (edges).

Path: a path p in a directed graph is a sequence of nodes $(v_1, v_2, \dots, v_n)$ with $k \geq 2$ a corresponding sequence of $k - 1$ arcs such that the i^{th} arc in the sequence is either(v_i, v_{i+1}) (in such case it is called a forward arc of path) or (v_{i+1}, v_i) (in which case it is called a backward arc of the path) nodes $v_1 and v_k$ are called the starting node (origin) and the end node (destination) of p respectively.

Cycle: is a path for which the starting and end nodes are the same.

Connected network: a network is said to be connected if for each pair of node $i$and j there is a path starting at i and ending at.

Tree: - is a connected network that contain no cycle.

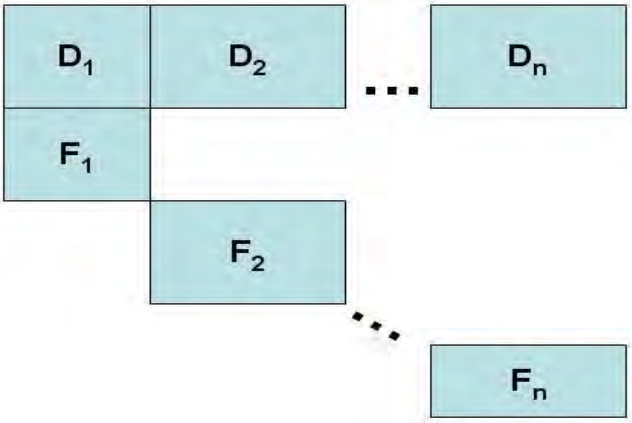
Sub graph: we say that a graph $G' = (V', E')$ is a subgraph of a graph $G = (V, E)$ if $V' \subset V$ and $E' \subset E$.

Convex set: - A set $\chi \subseteq \mathbb{R}^n$ is said to be convex if and only if $\forall\, x, y \in \chi$ and $\lambda \in [0,1]$, $\lambda x + (1-\lambda)y \in \chi$.

Convex combination: - let $\chi \subseteq \mathbb{R}^n$ be a convex set and $x_1, x_2, \dots, x_n \in \chi$ then $Z = \sum_{i=1}^{n} \lambda_i x_i \in \chi$ for $\sum_{i=1}^{n} \lambda_i = 1$, $\lambda_i \geq 0\ \forall i = 1,2, \dots, n$ is called the convex combination of elements of χ.

Polyhedral set: - the intersection of finite set of half spaces in $\mathbb{R}^n$ .polyhedron, if it is bounded.

Block angular structure:- is a structure like below



The *D* matrix represents the coupling constraints and each F_i represents the independent submatrices.

- Master problem: A problem that ties together the subproblems.
- Subproblem: problems corresponding to its independent parts.

1.1 The network-flow problem

A common scenario of a network-flow problem arising in industrial logistics concerns the distribution of a single homogeneous product from plants (origins) to consumer markets (destinations). The total number of units produced at each plant and the total number of units required at each market are assumed to be known. The product need not be sent directly from source to destination, but may be routed through intermediary points reflecting warehouses or distribution centers. Further, there may be capacity restrictions that limit some of the shipping links. The objective is to minimize the variable cost of producing and shipping the products to meet the consumer demand.

The sources, destinations, and intermediate points are collectively called *nodes* of the network, and the transportation links connecting nodes are termed *arcs*. A numerical example of a network-flow problem is given in Fig 1.1. The nodes are represented by numbered circles and the arcs by arrows. The arcs are assumed to be *directed* so that, for instance, material can be sent from node 1 to node 2, but not from node 2 to node 1. Generic arcs will be denoted by $i - j$, so that 4–5 means the arc *from* node 4 *to* node 5. Note that some pairs of nodes, for example 1 and 5, are not connected directly by an arc.

Figure 1.1 exhibits several additional characteristics of network flow problems. First, a flow capacity is assigned to each arc, and second, a per-unit cost is specified for shipping along each arc. These characteristics are shown next to each arc. Thus, the flow on arc 2–4 must be between 0 and 4 units, and each unit of flow on this arc costs \$2.00. The ∞'s in the figure have been used to denote unlimited flow capacity on arcs 2–3 and 4–5. Finally, the numbers in parentheses next to the nodes give the material supplied or demanded at that node. In this case, node 1 is an origin or *source node* supplying 20 units, and nodes 4 and 5 are destinations or *sink nodes* requiring 5 and 15 units, respectively, as indicated by the negative signs. The remaining nodes have no net supply or demand; they are intermediate points, often referred to as *transshipment nodes*.

The objective is to find the minimum-cost flow pattern to fulfill demands from the source nodes. Such problems usually are referred to as *minimum-cost flow* problems. To transcribe the problem into a formal linear program, let

$$x_{ij} = \text{Number of units shipped from node } i \text{ to } j \text{ using arc } i \to j.$$

Then the tabular form of the linear-programming formulation associated with the network of Fig. 1.1 is as shown in Table 1.1.

The first five equations are flow-balance equations at the nodes. They state the conservation-of-flow law,

$$(flow\ out\ of\ a\ node) - (flow\ into\ a\ node) = (netsupply\ at\ a\ node)$$

As examples, at nodes 1 and 2 the balance equations are:

$$x_{12} + x_{13} = 20$$
$$x_{23} + x_{24} + x_{25} - x_{12} = 0$$

It is important to recognize the special structure of these balance equations. Note that there is one balance equation for each node in the network. The flow variables x_{ij} have only $0, +1$, and -1 coefficients in these equations.

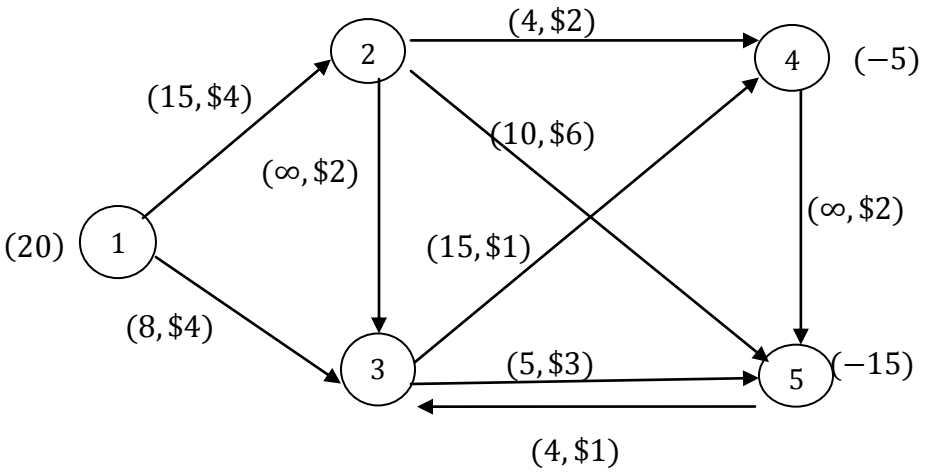


Figure 1.1 minimum-cost flow problem.

	x_{12}	x_{13}	x_{23}	x_{24}	x_{25}	x_{34}	x_{35}	x_{45}	x_{53}	Right hand side
$node\ 1$	1	1								20
$node\ 2$	-1		1	1	1					0
$node\ 3$		-1	-1			1	1		-1	0
$node\ 4$				-1		-1		1		-5
$node\ 5$					-1		-1	-1	1	-15
$capacities$	15	8	∞	4	10	15	5	∞	4	
$objective\ function$	4	4	2	2	6	1	3	2	1	(min)

Tableau1.1 Tableau for minimum-cost flow problem

Further, each variable appears in exactly two balance equations, once with a $+1$ coefficient, corresponding to the node from which the arc emanates; and once with a -1 coefficient, corresponding to the node upon which the arc is incident. This type of tableau is referred to as a *node–arc incidence matrix*; it completely describes the physical layout of the network. It is this particular structure that we shall exploit in developing specialized, efficient algorithms.

The remaining two rows in the table give the upper bounds on the variables and the cost of sending one unit of flow across an arc. For example, x_{12} is constrained by $0 \leq x_{12} \leq 15$ and appears in the objective function as $2x_{12}$. In this example the lower bounds on the variables are taken implicitly to be zero, although in general there may also be nonzero lower bounds.

This example is an illustration of the following general *minimum-cost flow* problem with *n* nodes:

$$minimize\ z = \sum_i \sum_j c_{ij}\, x_{ij}$$

$$subject\ to\ \sum_j x_{ij} - \sum_k x_{ki} = b_i \quad (i = 1,2,\dots,n) \text{ [Flow balance]}$$

$$l_{ij} \leq x_{ij} \leq u_{ij} \qquad \text{[Flow capacities]}$$

The summations are taken only over the arcs in the network. That is, the first summation in the ith flow-balance equation is over all nodes j such that $i - j$ is an arc of the network, and the second summation is over all nodes k such that $k - i$ is an arc of the network. The objective function summation is over arcs $i - j$ that are contained in the network and represents the total cost of sending flow over the network. The ith balance equation is interpreted as above: it states that the flow out

of node i minus the flow into i must equal the net supply (demand if b_i is negative) at the node. u_{ij} is the upper bound on arc flow and may be $+\infty$ if the capacity on arc $i-j$ is unlimited. l_{ij} is the lower bound on arc flow and is often taken to be zero, as in the previous example.

1.2. The special network models

There are a number of interesting special cases of the minimum-cost flow model that have received a great deal of attention. Maximum flow problem, Assignment problem, Transportation problem, shortest path problem etc are some examples of minimum cost flow problems. This section introduces one of these models, since they have had a significant impact on the development of a general network theory. In particular, algorithms designed for these specific models have motivated solution procedures for the more general minimum-cost flow problem.

1.2.1. The Transportation Problem

The transportation problem is a network-flow model without intermediate locations. To formulate the problem, let us define the following terms:

a_i =Number of units available at source $i(i = 1,2,3, \dots, m)$;

b_j =Number of units required at destination $j(j = 1,2,3, \dots, n)$;

c_{ij} =Unit transportation cost from source i to destination j

$(i = 1,2,3, \dots, m; j = 1,2,3, \dots, n)$.

In this paper, we assume that the total product availability is equal to the total product requirements; that is,

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j, \qquad \text{(Rim condition)}$$

If we define the decision variables as:

x_{ij} =Number of units to be distributed from source i to destination j

$(i = 1,2,3, \dots, m; j = 1,2,3, \dots, n)$.

We may then formulate the transportation problem as follows:

$$minimize\ z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\, x_{ij} \qquad (1)$$

$$subject\ to: \ \sum_{j=1}^{n} x_{ij} = a_i \quad (i = 1,2, \dots, m) \qquad (2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \quad (j = 1,2, \dots, n) \qquad (3)$$

$$x_{ij} \geq 0 \quad (i = 1,2, \dots, m; j = 1,2, \dots, n) \qquad (4)$$

Expression (1) represents the minimization of the total distribution cost, assuming a linear cost structure for shipping. Equation (2) states that the amount being shipped from source i to all possible destinations should be equal to the total availability, a_i , at that source. Equation (3) indicates that the amounts being Shipped to destination j from all possible sources should be equal to the requirements, b_j, at that destination. Equation (4) tells us non negativity of the flows. But, this not mean that it has unlimited capacity. This is because the number of units in the source nodes and the destination nodes are finite (known). I.e. the flow from source node i to destination node j is bounded by $0 \leq x_{ij} \leq minimum\{a_i, b_j\}$.

Let us consider a simple example. A compressor company has plants in three locations: Cleveland, Chicago, and Boston. During the past week the total production of a special compressor unit out of each plant has been 35, 50, and 40 units respectively. The company wants to ship 45 units to a distribution center in Dallas, 20 to Atlanta, 30 to San Francisco, and 30 to Philadelphia. The unit production and distribution costs from each plant to each distribution center are given in Table 1.2. What is the best shipping strategy to follow?

The linear-programming formulation of the corresponding transportation problem is:

$$minimize\ z = 8x_{11} + 6x_{12} + 10x_{13} + 9x_{14} + 9x_{21} + 12x_{22} + 13x_{23} + 7x_{24} + 14x_{31} + 9x_{32} + 16x_{33} + 5x_{34}$$

Plants	Distribution centers				Availability(units)
	Dallas	Atlanta	san Francisco	phila.	
Cleveland	8	6	10	9	35
Chicago	9	12	13	7	50
Boston	14	9	16	5	40
Requirement(units)	45	20	30	30	125

Tableau1.2 unit production and shipping costs

$$
\begin{aligned}
s.to: x_{11} + x_{12} + x_{13} + x_{14} &= 35\\
x_{21} + x_{22} + x_{23} + x_{24} &= 50\\
x_{31} + x_{32} + x_{33} + x_{34} &= 40\\
-x_{11} - x_{21} - x_{31} &= -45\\
-x_{12} - x_{22} - x_{32} &= -20\\
-x_{13} - x_{23} - x_{33} &= -30\\
-x_{14} - x_{24} - x_{34} &= -30\\
x_{ij} \geq 0 \quad (i =& \geq 1,2,3; j = 1,2,3,4).
\end{aligned}
$$

Because there is no ambiguity in this case, the same numbers normally are used to designate the origins and destinations. For example, x_{11} denotes the flow from source 1 to destination 1, although these are two distinct nodes. The network corresponding to this problem is given in Fig.1.2.

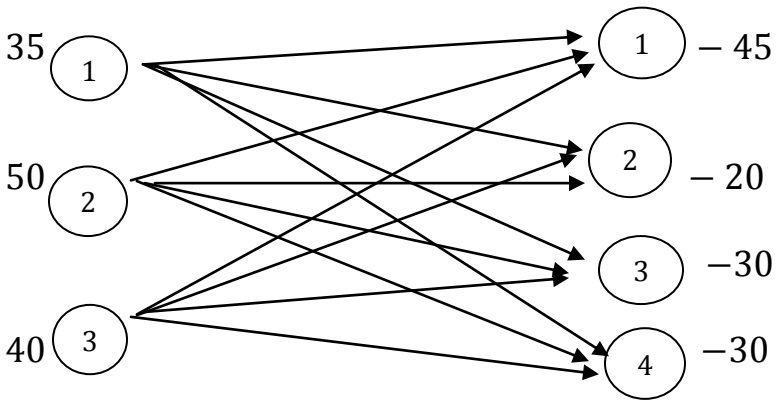


Figure 1.2 transportation network

1.3. SOLVING THE TRANSPORTATION PROBLEM

Ultimately in this chapter we want to develop an efficient algorithm for the general minimum-cost flow problem by specializing the rules of the simplex method to take advantage of the problem structure. However, before taking a somewhat formal approach to the general problem, we will indicate the basic ideas by developing a similar algorithm for the transportation problem. The properties of this algorithm for the transportation problem will then carry over to the more general minimum-cost flow problem in a straightforward manner.

Historically, the transportation problem was one of the first special structures of linear programming for which an efficient special-purpose algorithm was developed.

Many computational algorithms are characterized by three stages:
1. Obtaining an initial solution;
2. Checking an optimality criterion that indicates whether or not a termination condition has been met (i.e., in the simplex algorithm, whether the problem is infeasible, the objective is unbounded over the feasible region, or an optimal solution has been found);
3. Developing a procedure to improve the current solution if a termination condition has not been met.

After an initial solution is found, the algorithm repetitively applies steps 2 and 3 so that, in most cases, after a finite number of steps, a termination condition arises. The effectiveness of an algorithm depends upon its efficiency in attaining the termination condition.

Source	Destination 1	2	⋯	n	Supply
1	c_{11} x_{11}	c_{12} x_{12}	⋯	c_{1n} x_{1n}	a_1
2	c_{21} x_{21}	c_{22} x_{22}	…	c_{21} x_{2n}	a_2
⋮	⋮	⋮	⋯	⋮	⋮
m	c_{m1} x_{m1}	c_{m2} x_{m2}	⋯	c_{mn} x_{mn}	a_m
Demand	b_1	b_2	⋯	b_n	Total

Tableau1.3

Since the transportation problem is a linear program, each of the above steps can be performed by the simplex method. Initial solutions can be found very easily in this

case, however, so phase I of the simplex method need not be performed. Also, when applying the simplex method in this setting, the last two steps become particularly simple.

The transportation problem is a special network problem, and the steps of any algorithm for its solution can be interpreted in terms of network concepts. However, it also is convenient to consider the transportation problem in purely algebraic terms. In this case, the equations are summarized very nicely by a tabular representation like that in Tableau 1.3.

Each row in the tableau corresponds to a source node and each column to a destination node. The numbers in the final column are the supplies available at the source nodes and those in the bottom row are the demands required at the destination nodes. The entries in cell $i - j$ in the tableau denote the flow allocation x_{ij} from source i to destination j and the corresponding cost per unit of flow is c_{ij}. The sum of x_{ij} across row i must equal a_i in any feasible solution, and the sum of x_{ij} down column j must equal b_i .

Initial Solutions

In order to apply the simplex method to the transportation problem, we must first determine a basic feasible solution. Since there are $(m + n)$ equations in the constraint set of the transportation problem, one might conclude that, in a non degenerate situation, a basic solution will have $(m + n)$ strictly positive variables. We should note, however, that, since

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}$$

One of the equations in the constraint set is redundant. In fact, any one of these equations can be eliminated without changing the conditions represented by the original constraints. For instance, in the transportation example, the last equation can be formed by summing the first three equations and subtracting the next three equations. Thus, the constraint set is composed of $(m + n - 1)$ independent equations, and a corresponding nondegenerate basic solution will have exactly $(m + n - 1)$ basic variables.

There are several procedures used to generate an initial basic feasible solution, but we will consider only a few of these. The simplest procedure imaginable would be one that ignores the costs altogether and rapidly produces a basic feasible solution.

In Fig. 1.3, we have illustrated such a procedure for the transportation problem introduced in Section 1.2. We simply send as much as possible from origin 1 to destination 1, i.e., the minimum of the supply and demand, which is 35. Since the supply at origin 1 is then exhausted but the demand at destination 1 is not, we next fulfill the remaining demand at destination 1 from that available at origin 2. At this point destination 1 is completely supplied, so we send as much as possible (20 units) of the

Plants	Distribution centers				Supply
	1. Dallas	2. Atlata	3. San fra.	4. phila	
1.cleveland	35				35
2.chicago	10	20	20		50 40 20
3.boston			10	30	40 30
Demand	45 10	20	30 10	30	

Tableau 1.4 finding an initial basis by the north west corner method

remaining supply of 40 at origin 2 to destination 2, exhausting the demand there. Origin 2 still has a supply of 20 and we send as much of this as possible to destination 3, exhausting the supply at origin 2 but leaving a demand of 10 at destination 3. This demand is supplied from origin 3, leaving a supply there of 30, exactly corresponding to the demand of 30 needed at destination 4. The available supply equals the required demand for this final allocation because we have assumed that the total supply equals the total demand, that is,

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

This procedure is called the *northwest-corner* rule since, when interpreted in terms of the transportation array, it starts with the upper leftmost corner (the northwest corner) and assigns the maximum possible flow allocation to that cell. Then it moves to the right, if there is any remaining supply in the first row, or to the next lower cell, if there is any remaining demand in the first column, and assigns the maximum possible flow allocation to that cell. The procedure repeats itself until one reaches the lowest right corner, at which point we have exhausted all the supply and satisfied all the demand.

Table 1.4 summarizes the steps of the northwest-corner rule, in terms of the transportation tableau, when applied to the transportation example introduced in Section 1.2.

The availabilities and requirements at the margin of the table are updated after each allocation assignment. Although the northwest-corner rule is easy to implement, since it does not take into consideration the cost of using a particular arc, it will not, in general, lead to a cost-effective initial solution.

An alternative procedure, which is cognizant of the costs and straightforward to implement, is the *minimum matrix (least cost)* method. Using this method, we allocate as much as possible to the available arc with the lowest cost. Figure 1.4 illustrates the procedure for the example that we have been considering. The least-cost arc joins origin 3 and destination 4, at a cost of $5/unit, so we allocate the maximum possible, 30 units, to this arc, completely supplying destination 4. Ignoring the arcs entering destination 4, the least-cost remaining arc joins origin 1 and destination 2, at a cost of $6/unit, so we allocate the maximum possible, 20 units, to this arc, completely supplying destination 2. Then, ignoring the arcs entering either destination 2 or 4, the least cost remaining arc joins origin 1 and destination 1, at a cost of $8/unit, so we allocate the maximum possible, 15 units, to this arc, exhausting the supply at origin 1.

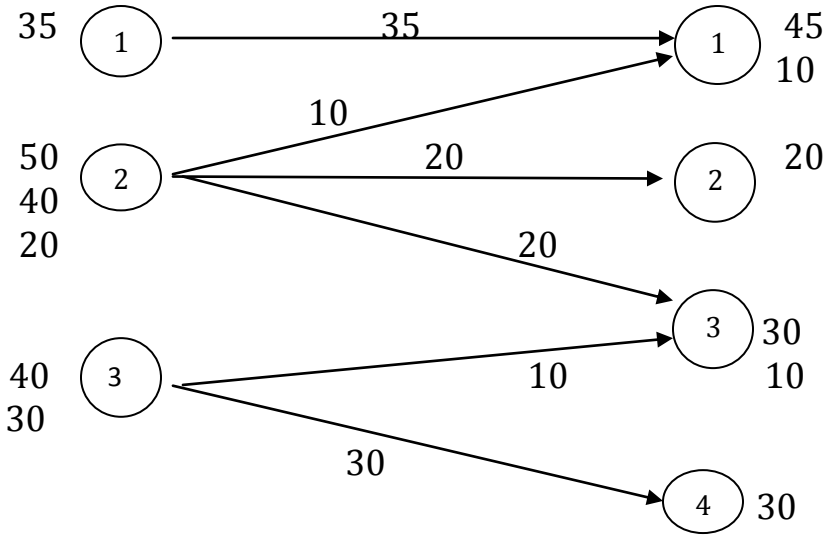


Figure1.3 finding an initial basis by the northwest-corner method

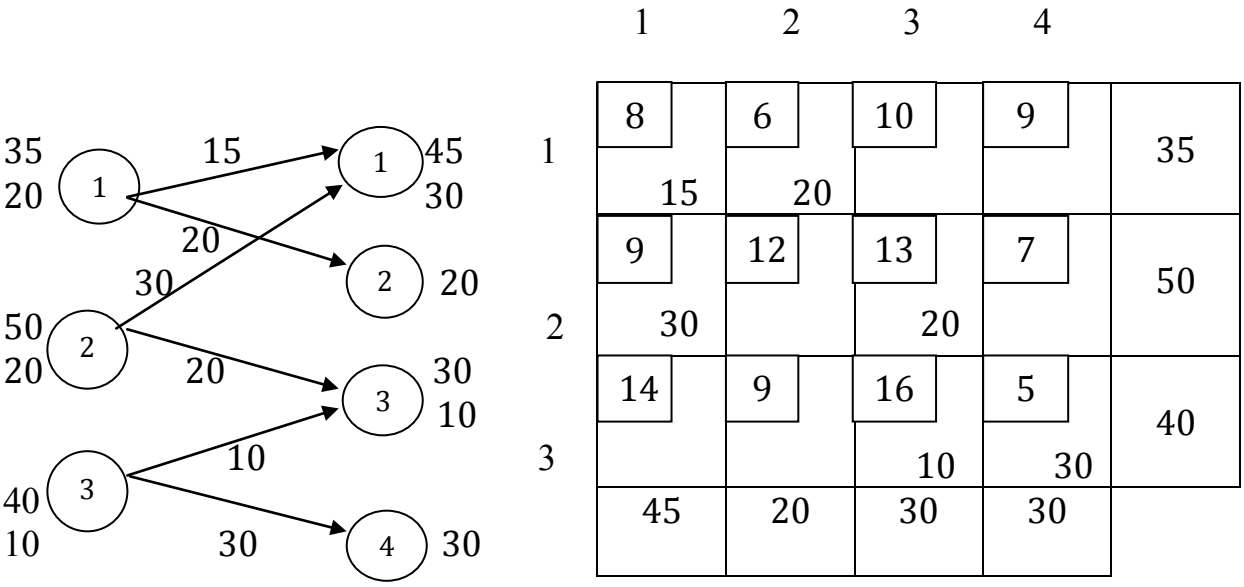


Figure 1.4 finding an initial basis by the minimum matrix method

Note that this arc is the least-cost *remaining* arc but not the next-lowest-cost unused arc in the entire cost array. Then, ignoring the arcs leaving origin 1 or entering destinations 2 or 4, the procedure continues. It should be pointed out that the last two arcs used by this procedure are relatively expensive, costing \$13/unit and \$16/unit. It is often the case that the minimum matrix method produces a basis that simultaneously employs some of the least expensive arcs and some of the most expensive arcs.

It should be pointed out that any comparison among procedures for finding an initial basis should be made only by comparing *solution times*, including both finding the initial basis *and* determining an optimal solution. For example, the northwest-corner method clearly requires fewer operations to determine an initial basis than does the minimum matrix rule, but the latter generally requires fewer iterations of the simplex method to reach optimality.

The two procedures for finding an initial basic feasible solution resulted in different bases; however, both have a number of similarities. Each basis consists of exactly $(m + n - 1)$ arcs, one less than the number of nodes in the network. Further, each basis is a sub network that satisfies the following two properties:

1. Every node in the network is connected to every other node by a sequence of arcs from the sub network, where the direction of the arcs is ignored.
2. The sub network contains no loops, where a loop is a sequence of arcs connecting a node to itself, where again the direction of the arcs is ignored.

A subnetwork that satisfies these two properties is called a spanning tree.

It is the fact that a basis corresponds to a spanning tree that makes the solution of these problems by the simplex method very efficient. Suppose you were told that a feasible basis consists of arcs $1-1,2-1,2-2,2-4,3-2 and 3-3$ then the allocations to each arc can be determined in a straightforward way without algebraic manipulations of tableaus. Start by selecting any *end* (a node with only 1 arc connecting it to the rest of the network) in the sub network . The allocation to the arc joining that end must be the supply or demand available at that end. For example, source 1 is an end node. The allocation on arc $1-1$ must then be 35, decreasing the unfulfilled demand at destination 1 from 45 to 10 units. The end node and its connecting arc are then dropped from the sub network and the procedure is repeated.

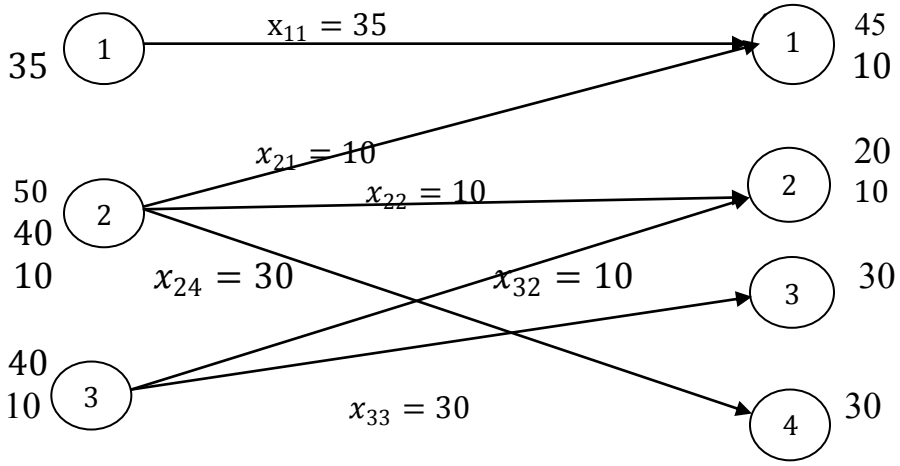


Figure 1.5 determining the values of basic variables.

This example also illustrates that the solution of a transportation problem by the simplex method results in *integer* values for the variables. Since any basis corresponds to a spanning tree, as long as the supplies and demands are integers the amount sent across any arc must be an integer. This is true because, at each stage of evaluating the variables of a basis, as long as the remaining supplies and demands are integer the amount sent to any end must also be an integer. Therefore, if the initial supplies and demands are integers, any basic feasible solution will be an integer solution.

Optimization Criterion

We will give the optimality conditions of this form in terms of the reduced costs of the simplex method.
If x_{ij} for $i = 1,2,3,\dots,m$ *and* $j = 1,2,3,\dots,n$ is a feasible solution to the transportation problem:

$$minimize\ z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\,x_{ij}$$

		Shadow Prices
$subject\ to$: $\sum_{j=1}^{n} x_{ij} = a_i$	$(i = 1,2,\dots,m)$,	u_i
$\sum_{i=1}^{m} x_{ij} = b_j$	$(j = 1,2,\dots,n)$,	v_j
$x_{ij} \geq 0.$		

Then it is optimal if there exist shadow prices (or simplex multipliers) u_i associated with the origins and v_j associated with the destinations, satisfying:

$$\bar{c}_{ij} = c_{ij} - u_i - v_j \geq 0 \quad if\ x_{ij} = 0 \qquad (1)$$

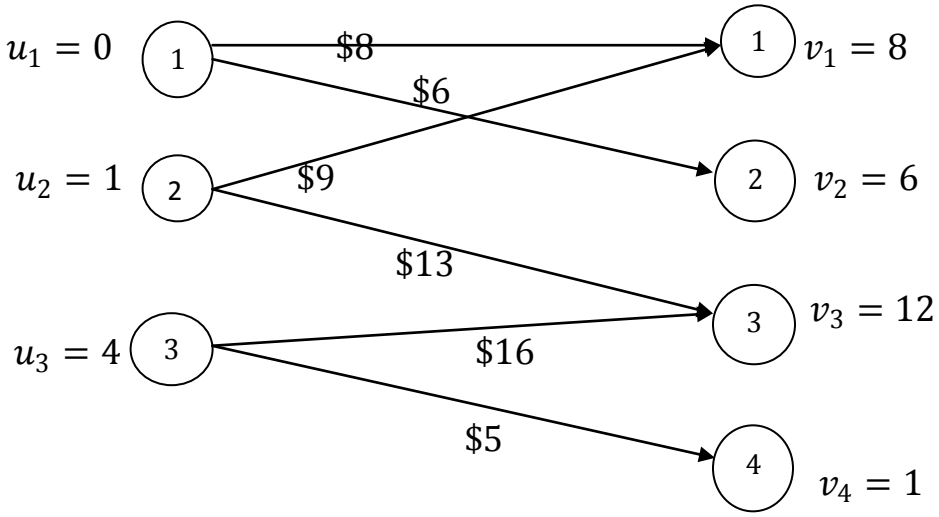


Figure 1.6 determining the simplex multiplier

and $\bar{c}_{ij} = c_{ij} - u_i - v_j = 0 \quad if\ x_{ij} > 0.$ (2)

The simplex method selects multipliers so that condition (2) holds for all basic variables, even if some basic variable $x_{ij} = 0$ due to degeneracy.

These conditions not only allow us to test whether the optimal solution has been found or not, but provide us with the necessary foundation to reinterpret the characteristics of the simplex algorithm in order to solve the transportation problem efficiently. The algorithm will proceed as follows: after a basic feasible solution has been found (possibly by applying the northwest-corner method or the minimum matrix method), we choose simplex multipliers u_i and v_j $(i = 1,2,3, \dots, m; j = 1,2,3, \dots, n)$ that satisfy:

$$u_i + v_j = c_{ij} \qquad (3)$$

for basic variables. With these values for the simplex multipliers, we compute the corresponding values of the reduced costs:

$$\bar{c}_{ij} = c_{ij} - u_i - v_j \qquad (4)$$

for all nonbasic variables. If every $\bar{c}_{ij}$ is nonnegative, then the optimal solution has been found; otherwise, we attempt to improve the current solution by increasing as much as possible the variable that corresponds to the most negative (since this is a minimization problem) reduced cost.

First, let us indicate the mechanics of determining from (3) the simplex multipliers associated with a particular basis. Conditions (3) consist of $(m + n - 1)$ equations in $(m + n)$ unknowns. However, any one of the simplex multipliers can be given an arbitrary value since, as we have seen, any one of the $(m + n)$ equation of the transportation problem can be considered redundant. Since there are $(m + n - 1)$ equations in (3), once one of the simplex multipliers has been specified, the remaining values of u_i and v_j is determined uniquely. For example, Fig. 1.6 gives the initial basic feasible solution produced by the minimum matrix method. The simplex multipliers associated with this basis are easily determined by first arbitrarily setting $u_1 = 0$. Given $u_1 = 0, v_1 = 8 \; and \; v_2 = 6$ are immediate from (3); then $u_2 = 1$ is immediate from v_1, and so on. It should be emphasized that the set of multipliers is not unique, since any multiplier could have been chosen and set to any finite value, positive or negative, to initiate the determination.

Once we have determined the simplex multipliers, we can then easily find the reduced costs for all non basic variables by applying (4). These reduced costs are given in Tableau1.5. The —'s indicate the basic variable, which therefore has a reduced cost of zero. This symbol is used to distinguish between basic variables and non basic variables at zero level.

Since, in Tableau1.5, $\bar{c}_{13} = -2 and \bar{c}_{32} = -1$, the basis constructed by the minimum matrix method does not yield an optimal solution. We, therefore, need to find an improved solution.

8	6	10	9	0
−	−	−2	8	
9	12	13	7	1
−	5	−	5	
14	9	16	5	4
2	−1	−	−	
8	6	12	1	u_i / v_j

Table 1.5 reduced costs

Improving the Basic Solution

As we indicated, every basic variable has associated with it a value of $\bar{c}_{ij} = 0$. If the current basic solution is not optimal, then there exists at least one nonbasic variable x_{ij} at value zero with $\bar{c}_{ij}$ negative. Let us select, among all those variables, the one with the most negative $\bar{c}_{ij}$ (ties are broken by choosing arbitrarily from those variables that tie); that is,

$$\bar{c}_{st} = \min_{ij}[\, \bar{c}_{ij} = c_{ij} - u_i - v_j \;|\bar{c}_{ij} < 0]$$

Thus, we would like to increase the corresponding value of x_{st} as much as possible, and adjust the values of the other basic variables to compensate for that increase. In our illustration,$\bar{c}_{st} = \bar{c}_{13} = -2$, so we want to introduce x_{13} into the basis. If we consider Fig. 1.6 we see that adding the arc 1–3 to the spanning tree corresponding to the current basis creates a unique loop $O_1 - D_3 - O_2 - D_1 - O_1$ where $O \; and \; D$ refer to origin and destination, respectively.

It is easy to see that if we make $x_{st} = \theta$, maintaining all other nonbasic variables equal to zero, the flows on this loop must be adjusted by plus or minus θ, to maintain the feasibility of the solution. The limit to which θ can be increased corresponds to the smallest value of a flow on this loop from which θ must be subtracted. In this example, θ may be increased to 15, corresponding to x_{11} being reduced to zero and therefore dropping out of the basis. The basis has

x_{13} replacing x_{11}, and the corresponding spanning tree has arc 1–3 replacing arc 1–1 in Fig. 1.7. Given the new basis, the procedure of determining the shadow prices and then the reduced costs, to check the optimality conditions of the simplex method, can be repeated.

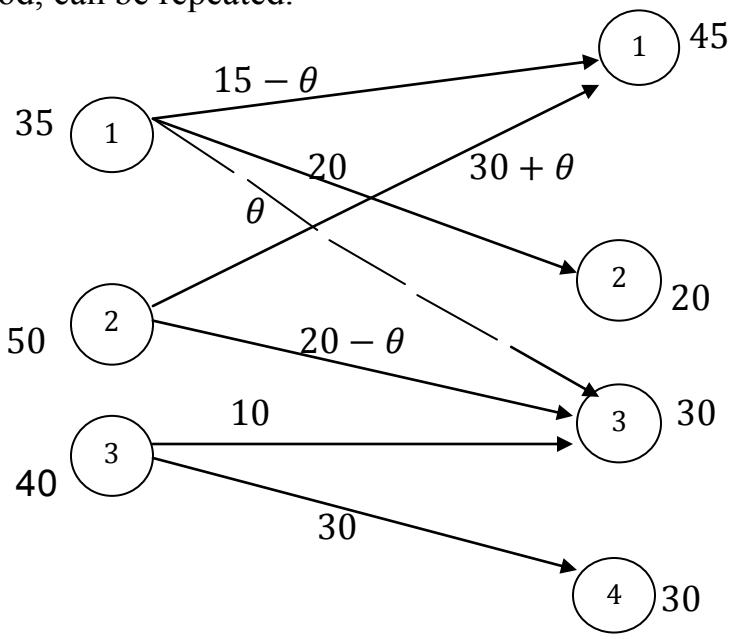


Figure1.7. Introducing a new variable.

It should be pointed out that the current solution usually is written out not in network form but in tableau form, for case of computation. The current solution given in Fig. 1.7 would then be written as in Tableau 1.6.

Note that the row and column totals are correct for any value of θ satisfying $0 \leq \theta \leq 15$. The tableau form for the current basic solution is convenient to use for computations, but the justification of its use is most easily seen by considering the corresponding spanning tree. In what follows we will use the tableau form to illustrate the computations. After increasing θ to 15, the resulting basic solution is given in Tableau 1.7 (ignoringθ) and the new multipliers and reduced costs in Tableau1. 8.

Tableau1.6

$15-\theta$	20	θ		35
$30+\theta$		$20-\theta$		50
		10	30	40
45	20	30	30	

	$20-\theta$	$15+\theta$		35
45		5		50
	θ	$10-\theta$	30	40
45	20	30	30	

8 2	6 –	10 –	9 10	0
9 –	12 3	13 –	7 5	3
14 2	9 −3	16 –	5 –	6
6	6	10	−1	u_i / v_j

Tableau 1.7 current basic solution

Tableau1.8 reduced cost

Since the only negative reduced cost corresponds to $\bar{c}_{32} = -3$, x_{32} is next introduced into the basis. Adding the arc $3-2$ to the spanning tree corresponds to increasing the allocation to cell 3–2 in the tableau and creates a unique loop $O_3 - D_2 - O_1 - D_3 - O_3$. The flow θ on this arc may be increased until θ= 10, corresponding to x_{33} dropping from the basis. The resulting basic solution is Tableau1.9 and the new multipliers and reduced costs are given in Tableau 1.10.

	10	25		35
45		5		50
	10		30	40
45	20	30	30	

Table 1.9 current basic solution

8 2	6 –	10 –	9 7	0
9 –	12 3	13 –	7 2	3
14 5	9 –	16 3	5 –	3
6	6	10	2	u_i / v_j

table 1.10 reduced cost

Since all of the reduced costs are now nonnegative, we have optimal solution.

In the previous section, we described how the simplex method has been specialized in order to solve transportation problems efficiently. There are three steps to the approach:
1. Finding an initial basic feasible solution;
2. Checking the optimality conditions; and
3. Constructing an improved basic feasible solution, if necessary.

1.4 THE SIMPLEX METHOD FOR NETWORKS

The application of the simplex method to the transportation problem presented in the previous section takes advantage of the network structure of the problem and illustrates a number of properties that extend to the general minimum-cost flow problem. All of the models formulated in Section 1.2 are examples of the general Minimum-cost flow problem, although a number, including the transportation problem, exhibit further special structure of their own. Historically, many different algorithms have been developed for each of these models; but, rather than consider each model separately, we will develop the essential step underlying the efficiency of all of the simplex-based algorithms for networks.

We already have seen that, for the transportation problem, a basis corresponds to a spanning tree, and that introducing a new variable into the basis adds an arc to the spanning tree that forms a unique loop. The variable to be dropped from the basis is

then determined by finding which variable in the loop drops to zero first when flow is increased on the new arc. It is this property, that bases correspond to spanning trees, that extends to the general minimum-cost flow problem and makes solution by the simplex method very efficient.

In what follows, network interpretations of the steps of the simplex method will be emphasized; therefore, it is convenient to define some of the network concepts that will be used. Though these concepts are quite intuitive, the reader should be cautioned that the terminology of network flow models has not been standardized, and that definitions vary from one author to another.

Formally, a *network* is defined to be any finite collection of points, called nodes, together with a collection of directed arcs that connect particular pairs of these nodes. By convention, we do not allow an arc to connect a node to itself, but we do allow more than one arc to connect the same two nodes. We will be concerned only with *connected* networks in the sense that *every* node can be reached from every other node by following a sequence of arcs, where the direction of the arcs is ignored. In linear programming, if a network is *disconnected*, then the problem it describes can be treated as separate problems, one for each connected sub network.

A *loop* is a sequence of arcs, where the direction of the arcs is ignored, connecting a particular node to itself. In Fig. 1.1, the node sequences 3–4–5–3 and 1–2–3–1 are both examples of loops.
A *spanning tree* is a connected subset of a network including all nodes and containing no loops.
It is the concept of a spanning tree, which proved most useful in solving the transportation problem in the previous section that will be the foundation of the algorithm for the general minimum-cost flow problem.

Finally, an *end* is a node of a network with exactly one arc incident to it. It is easy to see that every tree must have at least two ends. If you start with any node i in a tree and follow any arc away from it, you eventually come to an end, since the tree contains no loops. If node i is an end, then you have two ends. If node i is not an end, there is another arc from node i that will lead to a second end, since again there are no loops in the tree.

In the transportation problem we saw that there are $(m + n - 1)$ basic variables, since any one of the equations is redundant under the assumption that the sum of the supplies equals the sum of the demands. This implies that the number of arcs in any spanning tree corresponding to a basis in the transportation problem is

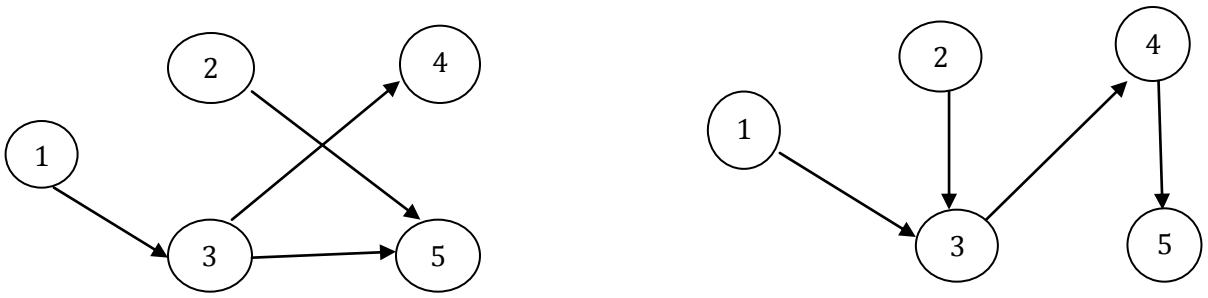


Figure 1.8 examples of spanning trees.

x_{13}	x_{25}	x_{34}	x_{35}	*right hand side*
1				20
	1			0
−1		1	1	0
		−1		−5
	−1		− 1	−15

Tableau1.11 tree variables

always *one less than* the number of nodes in the network. Note that the number of arcs is one less than the number of nodes in each of the trees shown in Fig. 1.8, since they each contain 5 nodes and 4 arcs. In fact, this characterization of spanning trees holds for the general minimum-cost flow problem.

Spanning-Tree Characterization. A subnetwork of a network with n nodes is a spanning tree if and only if it is connected and contains $(n - 1)$ arcs.

We can briefly sketch an inductive proof to show the spanning-tree characterization. The result is clearly true for the two node networks containing one arc. First, we show that is a subnetwork of an n-node network is a spanning tree, it contains $(n - 1)$ arcs. Remove any end and incident arc from the n-node network. The reduced network with $(n - 1)$ nodes is still a tree, and by our inductive assumption it must have $(n - 2)$ arcs. Therefore, the original network with n nodes must have had $(n - 1)$ arcs. Next, we show that if an n-node connected sub network has $(n - 1)$ arcs and no loops, it is a spanning tree. Again, remove any end and its incident arc from the n-node network. The reduced network is connected, has $(n - 1)$ nodes, $(n - 2)$ arcs, and no loops; and by our inductive assumption, it must be a spanning tree.

Therefore, the original network with n nodes and $(n - 1)$ arcs must be a spanning tree.

The importance of the spanning-tree characterization stems from the relationship between a spanning tree and a basis in the simplex method. We have already seen that a basis for the transportation problem corresponds to a spanning tree, and it is this property that carries over to the general network-flow model.

Spanning-Tree Property of Network Bases In a general minimum-cost flow model, a spanning tree for the network corresponds to a basis for the simplex method.

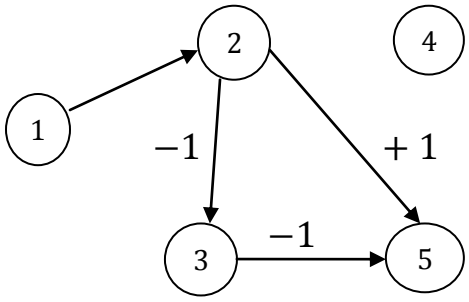
This is an important property since, together with the spanning-tree characterization; it implies that the number of basic variables is always one less than the number of nodes in a general network-flow problem. Now let us intuitively argue that the spanning-tree property holds, first by showing that the variables corresponding to a spanning tree constitute a basis, and second by showing that a set of basic variables constitutes a spanning tree.

First, assume that we have a network with n nodes, which is a spanning tree. In order to show that the variables corresponding to the arcs in the tree constitute a basis, it is sufficient to show that the $(n - 1)$ tree variables are uniquely determined. In the simplex method, this corresponds to setting the nonbasic variables to specific values and uniquely determining the basic variables. First, set the flows on all arcs not in the tree to either their upper or lower bounds, and update the righthand-side values by their flows. Then choose any node corresponding to an end in the subnetwork, say node k. (There must be at least two ends in the spanning tree since it contains no loops.) Node k corresponds to a row in the linear-programming tableau for the tree with exactly one nonzero coefficient in it. To illustrate this the tree variables of the first example in Fig. 1.8 are given in Tableau 1.11.

Since there is only one nonzero coefficient in row k, the corresponding arc incident to node k must have flow across it equal to the righthand-side value for that row. In the example above, $x_{13} = 20$. Now, drop node k from further consideration and bring the determined variable over to the righthand side, so that the righthand side of the third constraint becomes +20. Now we have an $(n - 1)$ node subnetwork with $(n - 2)$ arcs and no loops. Hence, we have a tree for the reduced network, and the process may be repeated. At each iteration exactly one flow variable is determined. On the last iteration, there are two equations corresponding to two nodes, and one arc joining them. Since we have assumed that the total net flow into or out of the network is zero, the last tree variable will satisfy the last two equations. Hence we

have shown that a spanning tree in a network corresponds to a basis in the simplex method. Further, since we have already shown that a tree for a connected network with n nodes contains $(n-1)$ arcs, we have shown that the number of basic variables for a connected network-flow problem is $(n-1)$.

Now assume that we have a network with n nodes and that we know the $(n-1)$ basic variables. To show that these variables correspond to a tree, we need only show that the sub network corresponding to the basic variables does not contain any loops. We establish this property by assuming that there *is* a loop and showing that this leads to a contradiction. In Fig. 1.9, we have a four-arc network containing a loop and its associated tableau.



x_{12}	x_{23}	x_{25}	x_{35}	Right hand side
1				20
−1	1	1		0
	−1		1	0
				−5
		−1	−1	−15

Figure 1.9 network containing loop.

If there exist a loop, then choose weights for the column corresponding to arcs in the loop such that the weight is $+1$ for a forward arc in the loop and -1 for a backward arc in the loop. If we then add the columns corresponding to the loop weighted in this manner, we produce a column containing all zeros. In Fig. 1.9, the loop is 2–5–3–2, and adding the columns for the variables $x_{25}, x_{53}, and x_{32}$ with weights 1, -1, and -1, respectively, produces a zero column. This implies that the columns corresponding to the loop are not independent. Since a basis consists of a set of $(n-1)$ independent columns, any set of variables containing a loop cannot be a basis. Therefore, $(n-1)$ variables corresponding to a basis in the simplex method must be a spanning tree for the network.

If a basis in the simplex method corresponds to a tree, what, then, is the interpretation of introducing a new variable into the basis? Introducing a new variable into the basis adds an arc to the tree, and since every node of the tree is connected to every other node by a sequence of arcs, the addition will form a loop in the subnetwork corresponding to the tree. In Fig. 1.10, arc $4-5$ is being

introduced into the tree, forming the loop $4-5-3-4$. It is easy to argue that adding an arc to a tree creates a *unique* loop in the augmented network. *At least one loop* must be created, since, adding the arc connects two nodes that were already connected. Further, *no more than one loop* is created, since, if adding the one arc created more than one loop, the entering arc would have to be common to two distinct loops, which, upon removal of their common arcs, would yield a single loop that was part of the original network.

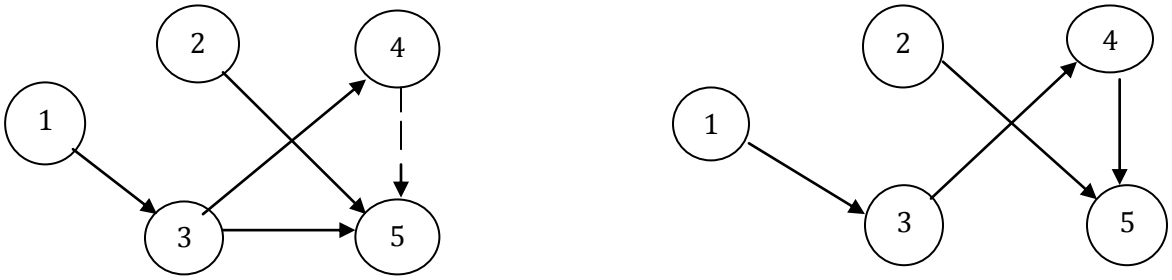


Figure 1.10 introducing a new arc in a tree.

To complete a basis change in the simplex method, one of the variables currently in the basis must be dropped. It is clear that the variable to be dropped must correspond to one of the arcs in the loop, since dropping any arc not in the loop would leave the loop in the network and hence would not restore the spanning-tree Property. We must then be able to determine the arcs that form the loop.

This is accomplished easily by starting with the subnetwork including the loop and eliminating all ends in this network. If the reduced network has no ends, *it* must be the loop. Otherwise, repeat the process of eliminating all the ends in the reduced network, and continue. Since there is a unique loop created by adding an arc to a spanning tree, dropping any arc in that loop will create a new spanning tree, since each node will be connected to every other node by a sequence of arcs, and the resulting network will contain no loops. Figure 1.10 illustrates this process by dropping arc $3-5$. Clearly, any arc in the loop could have been dropped, to produce alternative trees.

In the transportation problem, the unique loop was determined easily, although we did not explicitly show how this could be guaranteed. Once the loop is determined, we increased the flow on the incoming arc and adjusted the flows on the other arcs in the loop, until the flow in one of the arcs in the loop was reduced to zero.
The variable corresponding to the arc whose flow was reduced to zero was then dropped out of the basis. This is essentially the same procedure that will be employed by the general minimum-cost flow problem, except that the special rules

of the simplex method with upper and lower bounds on the variables will be employed. In the next section an example is carried out that applies the simplex method to the general minimum-cost flow problem.

Finally, we should comment on the integrality property of the general minimum-cost flow problem. We saw that, for the transportation problem, since the basis corresponds to a spanning tree, as long as the supplies and demands are integers, the flows on the arcs for a basic solution are integers. This is also true for the general minimum-cost flow problem, so long as the net flows at any node are integers and the upper and lower bounds on the variables are integers.

Integrality Property In the general minimum-cost flow problem, assuming that the upper and lower bounds on the variables are integers and the righthand-side values for the flow-balance equations are integers, the values of the basic variables are also integers when the nonbasic variables are set to their upper or lower bounds.

In the simplex method with upper and lower bounds on the variables, the nonbasic variables are at either their upper or lower bound. If these bounds are integers, then the net flows at all nodes, when the flows on the nonbasic arcs are included, are also integers. Hence, the flows on the arcs corresponding to the basic variables also will be integers, since these flows are determined by first considering all ends in the corresponding spanning tree and assigning a flow to the incident arc equal to the net flow at the node. These assigned flows must clearly be integers. The ends and the arcs incident to them are then eliminated, and the process is repeated, yielding an integer assignment of flows to the arcs in the reduced tree at each stage.

x_{13}	x_{25}	x_{34}	x_{35}	Right hand side	Row no.
1				20	1
	1			0	2
−1		1	1	0	3
		−1		−5	4
	−1		−1	−15	5

Table 1.12 basis variables

The integrality property of the general minimum-cost flow problem was established easily by using the fact that a basis corresponds to a spanning tree. Essentially, all ends could be immediately evaluated, then eliminated, and the procedure repeated. We were able to solve a system of equations by recognizing that at least one

variable in the system could be evaluated by inspection at each stage, since at each stage at least

x_{25}	x_{34}	x_{35}	x_{13}	Right hand side	Row no.
1				0	2
	−1			−5	4
−1		−1		−15	5
	1	1	−1	0	3

Tableau 1.13 a basis is triangular.

One equation would have only one basic variable in it. A system of equations with this property is called *triangular*.

In Tableau 1.12 we have rewritten the system of equations corresponding to the tree variables given in Tableau 1.11. Then we have arbitrarily dropped the first equation, since a connected network with *n* nodes has $(n - 1)$ basic variables. We have rearranged the remaining variables and constraints to exhibit the triangular form in Tableau 1.13.

The variables on the diagonal of the triangular system then may be evaluated sequentially, starting with the first equation. Clearly $x_{25} = 0$.Then, moving the evaluated variable to the righthand side, we have a new triangular system with one less equation. Then the next diagonal variable may be evaluated in the same way and the procedure repeated. It is easy to see that, for our example, the values of the variables are $x_{25} = 0, x_{34} = 5, x_{35} = 15, x_{13} = 20$. Note that the value of x_{13} satisfies the first equation that was dropped. It should be pointed out that many other systems of equations besides network-flow problems can be put in the form of a triangular system and therefore can be easily solved.

1.5 SOLVING THE MINIMUM-COST FLOW PROBLEM

In this section we apply the simplex method to the general minimum-cost flow problem, using the network concepts developed in the previous section. Consider the minimum-cost flow problem given in Section 1.1 and repeated here in Fig. 1.11 for reference.

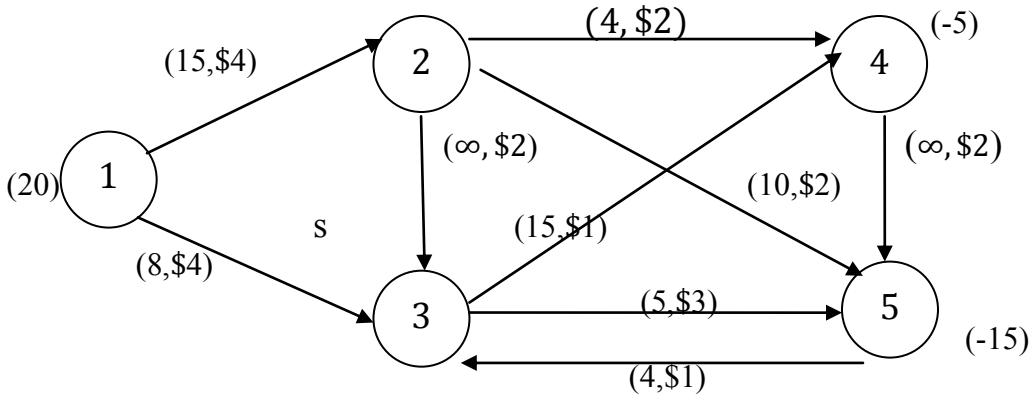


Fig1.11 minimum-cost flow problem

This problem is more complicated than the transportation problem since it contains intermediate nodes (points of transshipment) and capacities limiting the flow on some of the arcs.

In order to apply the simplex method to this example, we must first determine a basic feasible solution. Whereas, in the case of the transportation problem, an initial basic feasible solution is easy to determine (by the northwest-corner method, the minimum matrix method) in the general case an initial basic feasible solution may be difficult to find. The difficulty arises from the fact that the upper and lower bounds on the variables are treated implicitly and, hence, non basic variables may be at either bound. We will come back to this question later. For the moment, assume that we have been given the initial basic feasible solution shown in Fig. 1.12.

The dash–dot arcs $1-3\ and\ 3-5$ indicate non basic variables at their upper bounds of 8 and 5, respectively.

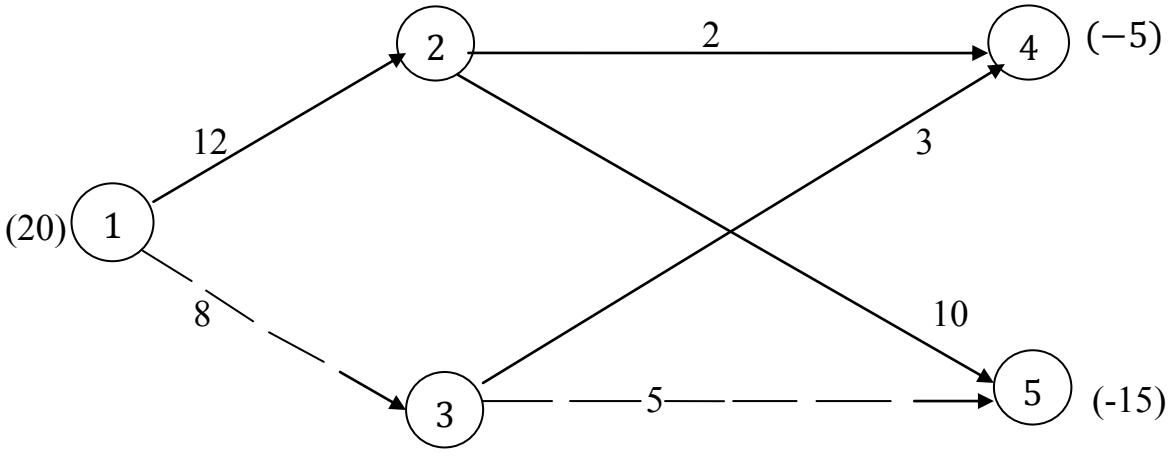


Figure 1.12 Initial basic feasible solution.

The arcs not shown are nonbasic at their lower bounds of zero. The solid arcs form a spanning tree for the network and constitute a basis for the problem.

To determine whether this initial basic feasible solution is optimal, we must compute the reduced costs of all non basic arcs. To do this, we first determine multipliers y_i $(i = 1, 2, \ldots, \mathrm{n})$ and, if these multipliers satisfy:

$$\begin{aligned} \bar{c}_{ij} &= c_{ij} - y_i + y_j \geq 0 \quad \text{if } x_{ij} = \ell_{ij} \\ \bar{c}_{ij} &= c_{ij} - y_i + y_j = 0 \quad \text{if } \ell_{ij} < x_{ij} < u_{ij}, \\ \bar{c}_{ij} &= c_{ij} - y_i + y_j \leq 0 \quad \text{if } x_{ij} = u_{ij}, \end{aligned}$$

Then we have an optimal solution. Since the network-flow problem contains a redundant constraint, any one multiplier may be chosen arbitrarily, as was indicated in previous sections. Suppose $y_2 = 0$ is set arbitrarily; the remaining multipliers are determined from the equations:

$$c_{ij} - y_i + y_j = 0$$

for basic variables. The resulting multipliers for the initial basis are given as node labels in Fig. 1.13. These were determined from the cost data given in Fig. 1.12.

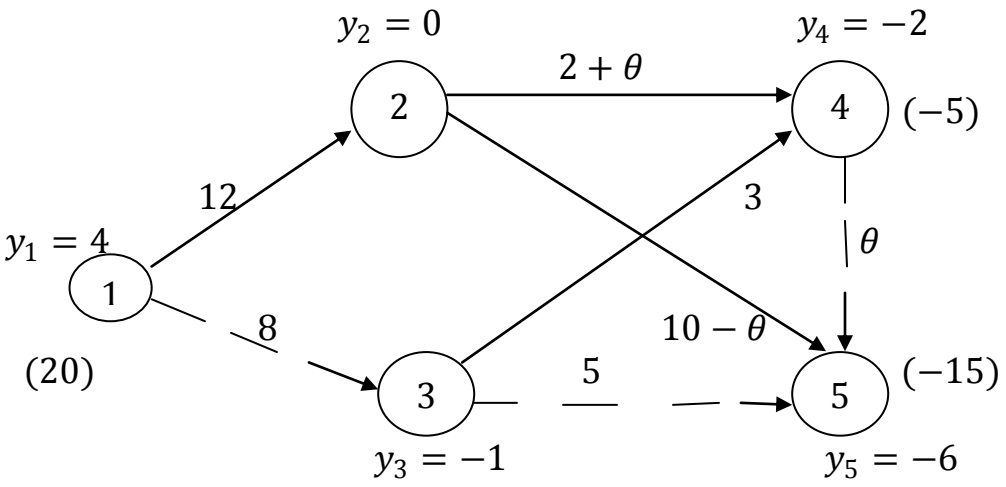


Fig1.13. Iteration 1

Given these multipliers, we can compute the reduced costs of the non basis variables by $\bar{c}_{ij} = c_{ij} - y_i + y_j$.
The reduced costs are determined by using the given cost data in Fig. 1.11 as:

$$\begin{aligned}
\bar{c}_{13} &= 4 - 4 + (-1) = -1, \\
\bar{c}_{23} &= 2 - 0 + (-1) = 1, \\
\bar{c}_{35} &= 3 - (-1) + (-6) = -2, \\
\bar{c}_{45} &= 2 - (-2) + (-6) = -2, \quad \leftarrow \\
\bar{c}_{53} &= 1 - (-6) + (-1) = 6.
\end{aligned}$$

In the simplex method with bounded variables, the non basic variables are at either their upper or lower bounds. An improved solution can be found by either:

1. Increasing a variable that has a negative reduced cost and is currently at its lower bound; or
2. Decreasing a variable that has a positive reduced cost and is currently at its upper bound.

In this case, the only promising candidate is x_{45}, since the other two negative reduced costs correspond to nonbasic variables at their upper bounds. In Fig. 1.13 we have added the arc 4– 5 to the network, forming the unique loop 4– 5– 2– 4 with the basic variables. If the flow on arc 4– 5 is increased by θ , the remaining arcs in the loop must be appropriately adjusted. The limit on how far we can increase θ is given by arc 2– 4, which has an upper bound of 4. Hence, $\theta = 2$and x_{24} becomes nonbasic at its upper bound. The corresponding basic feasible solution is given in Fig. 1.14, ignoring the dashed arc 2– 3.

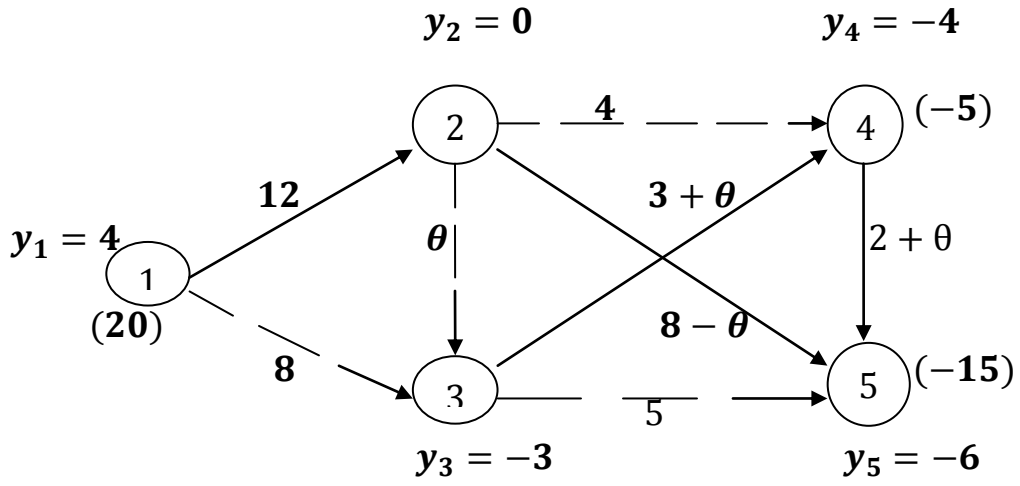


Fig1.14. Iteration 2

The new multipliers are computed as before and are indicated as node labels in Fig. 1.14. Note that not all of the multipliers have to be recalculated. Those multipliers corresponding to nodes that are connected to node 2 by the same sequence of arcs as before will not change labels. The reduced costs for the new basis are then:

$$\begin{aligned}
\bar{c}_{13} &= 4 - 4 + (-3) = -3, \\
\bar{c}_{23} &= 2 - 0 + (-3) = -1, \quad \leftarrow \\
\bar{c}_{24} &= 2 - 0 + (-4) = -2, \\
\bar{c}_{35} &= 3 - (-3) + (-6) = 0, \\
\bar{c}_{53} &= 1 - (-6) + (-3) = 4.
\end{aligned}$$

Again there is only one promising candidate, x_{23}, since the other two negative reduced costs correspond to non basic variables at their upper bounds. In Fig. 1.14 we have added the arc $2-3$ to the network, forming the unique loop $2-3-4-5-2$ with the basic variables. If we increase the flow on arc $2-3$ by θ and adjust the flows on the remaining arcs in the loop to maintain feasibility, the increase in θ is limited by arc 2–5. When $\theta = 8$, the flow on arc 2–5 is reduced to zero and x_{25} becomes non basic at its lower bound. Figure 1.15 Shows the corresponding basic feasible solution.

The new multipliers are computed as before and are indicated as node labels in Fig. 1.3. The reduced costs for the new basis are then:

$$\begin{aligned}
\bar{c}_{13} &= 4 - 4 + (-2) = -2, \\
\bar{C}_{24} &= 2 - 0 + (-3) = -1, \\
\bar{c}_{25} &= 6 - 0 + (-5) = 1, \\
\bar{c}_{35} &= 3 - (-2) + (-5) = 0, \\
\bar{c}_{53} &= 1 - (-5) + (-2) = 4.
\end{aligned}$$

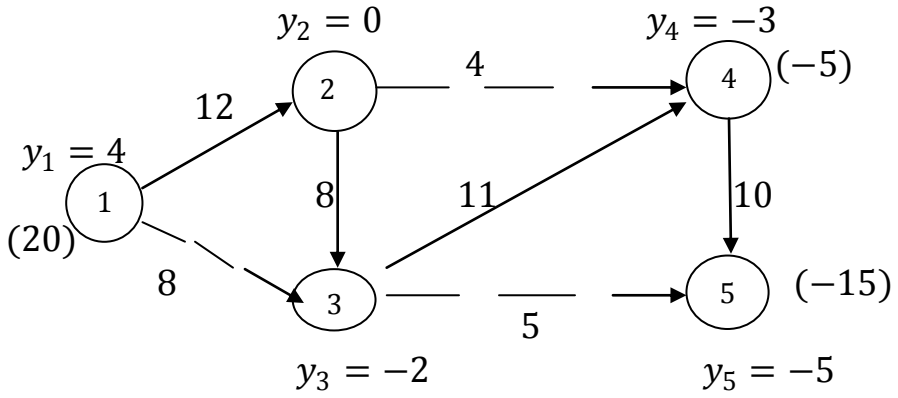


Figure 1.15 optimal solutions.

This is an optimal solution, since all negative reduced costs correspond to non basic variables at their upper bounds and all positive reduced costs correspond to nonbasic variables at their lower bounds. The reduced cost $\bar{c}_{35} = 0$ indicates that alternative optimal solutions may exist. In fact, it is easily verified that the solution given in Fig. 1.16 is an alternative optimal solution.

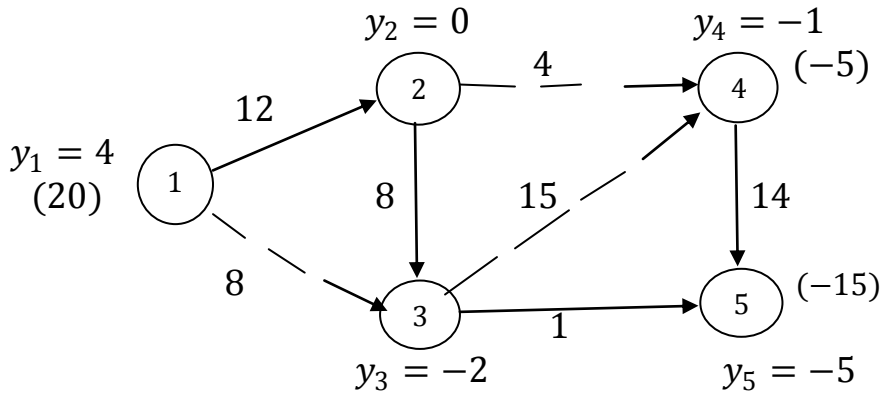


Figure 1.16 alternative optimal solutions

Hence, given an initial basic feasible solution, it is straightforward to use the concepts developed in the previous section to apply the simplex method to the general minimum-cost flow problem. There are two essential points to recognize: (1) a basis for the simplex method corresponds to a spanning tree for the network, and (2) introducing a new variable into the basis forms a unique loop in the spanning tree, and the variable that drops from the basis is the limiting variable in this loop.

CHAPTER TWO

Multicommodity minimal cost netwok flow problem

INTRODUCTION: –

The Multicommodity Minimal Cost Flow Problem is a type of problem that is related to capacitate networks. This capacitated network consists of various nodes and arcs, which can be seen as villages and roads connecting them. Each of these arcs has got costs and capacity. On this network, items (commodities) are carried from one node to another. The problem is to find a path (or various paths) which can be used to transfer these items without violating the maximum capacity of each arc. This path not only has to be capacitated enough to transfer the items, but also has to be as cheap as possible. This problem is called Minimal Cost Flow Problem. If this problem is extended with multiple types of items that are sharing the same arcs and capacities then this is called the Multicommodity Minimal Cost Flow Problem. An example of such a problem is displayed below:

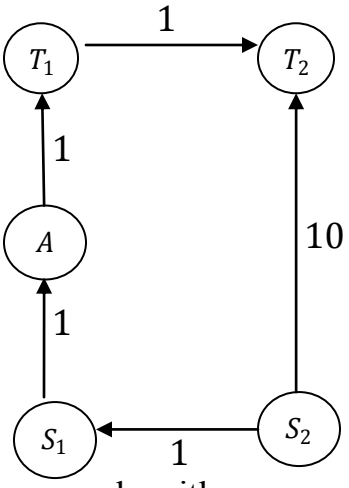


Fig 2.1 an example with corresponding costs

In this example there are two commodities, each with their own source (S_1 and S_2) and their own sinks(T_1 and T_2). They are both of the same size (1) and both are sending 1 item from their source to their sink. All of the arcs have a capacity of 1 and the costs of each arc are 1 or 10. The obvious solution to this example is that commodity 1 goes from S_1 via A to T_1 , and S_2 goes directly toT_2. Although commodity 2 prefers going via S_1, A, T_1 to T_2,this violates the capacity constraints on these arcs.

In many application contexts, several physical commodities, vehicles, or messages, each governed by their own network flow constraints, share the same network. If

the commodities do not share (interact) in any way, then to solve problem with several commodities we would solve each single-commodity problem separately. In other situations, however, because the commodities do share common facilities, the individual single-commodity problems are not independent, so to find an optimal flow, we need to solve the problems in concert with each other. One such model, known as multicommodity flow problem, in which the individual commodities share common arc capacities. That is each arc has a capacity u_{ij} that restrict the total flow of all commodities on that arc.

Definition: A multicommodity minimal cost flow problem are optimization problem where one is interested in identifying the least cost method of moving two or more types of commodities from certain location where they produce to certain location where they are consumed and these different commodities share the same network with capacitated arc.

Here after, when we refer to the $MCNF$ problem, we mean the multicommodity minimal cost network flow problem and $SCNF$ problem to single minimum cost network flow (simply minimum cost network flow problems).

2.1. MCNFP formulation:-

Suppose that we are given a network G with m nodes and n arcs in which there will be a flow of t different commodities. Let u_{kij} is the upper limit on flow for commodity k in arc (i,j). u_{ij} is the upper limit on the sum of all commodity flows in arc(i,j). c_{kij} is the unit cost of commodity k on arc (i,j) and let b_{ki} is the supply $(if\ b_{ki} > 0)$ or demand $(if\ b_{ki} < 0)$ of commodity k at node i. The linear programming formulation for the multicommodity minimal cost flow problem is as follows:

$$minimize \quad \sum_{k=1}^{t} \sum_{(i,j)\in E} c_{kij}\, x_{kij}$$

Subject to $\quad \sum_{k=1}^{t} x_{kij} \le u_{ij} \qquad \forall (i,j) \in E$

$$\sum_{\{j\,/(i,j)\in E\}} x_{kij} - \sum_{\{j\,/(j,i)\in E\}} x_{kij} = b_{ki} \qquad \forall k = 1{:}\,t\,, \forall i \in N$$

$$x_{kij} \ge 0 \qquad \forall (i,j) \in E, \forall k = 1{:}\,t$$

Where x_{kij} is the flows of commodity k in the arc(i,j) and E is the set of n arcs. N is the set of m nodes. The foregoing formulation is called the node-arc formulation for the multicommodity flow problem since it uses the node-arc incidence matrix.

This formulation has a collection of t ordinary mass balance constraints, modeling the flow of each commodity $\mathrm{k} = 1,2,\dots,\mathrm{t}$.

$$Ax_{kij} = b_{kq}, \qquad \text{for k} = 1,2,\dots,\text{t}, \qquad \forall \text{q} \in \text{N}$$

The bundle constraints tie together the commodities by restricting the total flow of all the commodities on each arc (i,j) to at most u_{ij}.

$$\sum_{k=1}^{t} x_{kij} \leq u_{ij}, \qquad \text{for all } (i,j) \in \text{E}$$

Note that we also impose individual flow bounds on the flow of commodity i on arc (p,q).

$$0 \leq x_{kij} \leq u_{kij}, \quad \text{for all } (i,j) \in \text{E} \text{ and all k} = 1,2,\dots,\text{t}.$$

Many applications do not impose these bounds, so for these applications we set each bound to $+\infty$.

In our discussion, it will be more convenient to state the bundle constraints as equalities instead of inequalities. In these instances we introduce nonnegative slack variables s_{ij} and write the bundle constraints as:

$$\sum_{k=1}^{t} x_{kij} + s_{ij} = u_{ij}, \qquad \text{for all } (i,j) \in \text{E}$$

The slack variable s_{ij} for the arc (i,j) measures the unused bundle capacity on that arc.

The constraints of **MCNF** possess special structure called block angular structure. To see what kind of structure is it let's see for simplicity a network which have three nodes and three arcs in which two commodities are flow between the nodes. see the figure below.

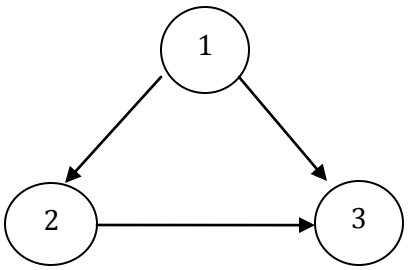


Figure2.2

Corresponding to each arc the following are associated c_{kij} , x_{kij} , u_{kij} , $and\ u_{ij}$ and at each node i there is b_{ki} where $k = 1,2$ and $i \in N$.

Then our objective is to minimize the cost of flow through each arc for both commodities. The linear programming will look like

$$minimize \sum_{k=1}^{2} \sum_{(i,j)\in E} c_{kij}\, x_{kij}$$

$$subject\ to \sum_{k=1}^{2} x_{kij} \leq u_{ij} \quad \forall (i,j)E$$

$$\sum_{\{j\,/(i,j)\in E\}} x_{kij} - \sum_{\{j\,/(j,i)\in E\}} x_{kij} = b_{ki} \qquad \forall i \in N, \forall\, k = 1,2$$

$$x_{kij} \geq 0 \qquad \forall k = 1,2 \quad \forall (i,j) \in E$$

The constraints are

$$x_{112} + x_{212} + s_{12} = u_{11}$$

$$x_{113} + x_{213} + s_{13} = u_{13}$$

$$x_{123} + x_{223} + s_{23} = u_{23}$$

$$x_{112} + x_{113} = b_{11}$$

$$-x_{112} + x_{123} = b_{12}$$

$$-x_{113} - x_{123} = b_{13}$$

$$x_{212} + x_{213} = b_{21}$$

$$-x_{212} + x_{223} = b_{22}$$

$$-x_{213} - x_{223} = b_{23}$$

$$x_{kij} \geq 0 \qquad \forall k = 1,2 \quad \forall (i,j) \in E$$

x_{112}	x_{113}	x_{123}	x_{212}	x_{213}	x_{223}	s_{12}	s_{13}	s_{23}
1	0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0	1
1	1	0						
−1	0	1						
0	−1	−1						
			1	1	0			
			−1	0	1			
			0	−1	−1			

.Node arc incidence matrix of the above figure

From this we can see that it has a block angular structure (see the preliminary part for definition).

Assumptions:-Three assumptions are:

- Homogenous goods assumption
- No congestion assumption
- Indivisible goods assumption

1. Homogenous goods assumption

We are assuming that every unit flow of each commodity uses 1 unit of capacity of each arc.

A more general model would permit the unit flow of each commodity k to consume a given amount p_{kij} of the capacity associated with each arc(i,j), and replace the bundle constraint with a more general resource availability constraint:

$$\sum_{k=1}^{t} p_{kij}\, x_{kij} \leq u_{ij}\,.$$

2. No congestion assumption

We are assuming that we have a hard (i. e. fixed) capacity on each arc and that the cost on each arc is linear in the flow on that arc.
In some application in communication, the commodities interact in a more complex fashion that is at the flow of any commodity increases on an arc, we incur an increasing and a non linear cost on the arc. So, we obtain a non linear objective function in these kinds of problems.

3. Divisible goods assumption

The model assumes that the flow variable can be fractional flow for all the variables. Enforcing the integer constraints on that variable makes the problem complicated and difficult to solve just like that single commodity flow problems. However, the solution with fractional flow provides a linear program bound on the integer program problem that would be obtained by enforcing the integer constraints.

The existence of the bundle constraints makes $MCNF$ problems much more difficult than $SCNF$ problems. For example, many $SCNF$ algorithms exploit the single commodity flow property in which flows in opposition direction on an arc could be canceled out. In $MCNF$ problems, flows do not cancel if they are different commodities. The max-flow min-cut theorem of ford and Fulkerson in $SCNF$ problems guarantees that the maximum flow is equal to the minimum cut. Furthermore, with integral arc capacities and node demands, the maximum flow is guaranteed to be integral. None of these properties can be extended to$MCNF$. The total unimodularity of the constraint matrix in SCNF in the linear programming formulation guarantees integral optimal flows in the cases of integral node supplies/demands and arc capacities. This integrality property also cannot be extended to MCNF linear formulation.

As we shall see, multicommodity flow problems do not enjoy the same special properties as a single commodity flow problem. As an example, consider the network of Fig 2.3 suppose that there are three commodities that flow through the

network. The source of commodity 1 is node1, and the sink for commodity 1 is node 3.that is commodity 1 must orginate only at node 1 and terminate only at node3. Similarly, let the source and sink for commodity 2 be node 2 and 1 respectively. Finally, the source and sink for commodity 3 are

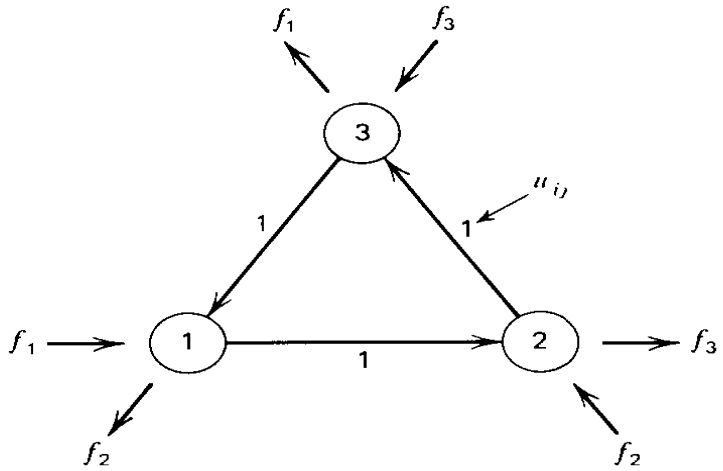


Fig 2.3 a three commodity maximal flows

nodes3 and 2 respectively. With the restriction that the sum of all commodities flowing on an arc should not exceed the arc capacity $u_{ij} = 1$, what is the maximal sum of commodity flows, $f_1 + f_2 + f_3$, possible in the network? Finding the maximum flow for the three commodity problem of Fig 2.3 is relatively simple since there is only one path that each commodity can take on its way from its source to its sink. The paths for commodity 1,2, and 3 respectively are

$$P_1 = \{(1,2),(2,3)\}$$

$$P_2 = \{(2,3),(3,1)\}$$

$$P_3 = \{(3,1),(1,2)\}$$

If we place a single unit of flow on any one of the paths, then the other paths are completely blocked (i.e. must have zero flow) and thus the total flow would be 1. However, there is a better solution availability if we do not require integer flow. Suppose that we place $\frac{1}{2}$ unit of flow of commodity 1 on P_1, $\frac{1}{2}$ unit of flow of commodity 2 on P_2, $\frac{1}{2}$ unit of flow of commodity 3 on P_3. in this case none of the

capacities are violated and the total flow of all commodity is $\frac{3}{2}$. From this we see that multi commodity flow problem do not necessarily provide integer flow.

The block - angular constraint structure of the $MCNF$ problem serve as a best practice for decomposition techniques. The basic solution methods used to solve minimum cost single commodity network flow problems can be modified to solve the multicommodity case. The bases for multicommodity networks are any basis that contains a spanning tree for each commodity. Linear programming also may be used to solve multicommodity network flow problems, although more practical algorithm may exploit both the block-angular structure of the multicommodity formulation and the structure of each block of flow constraints.

Even though multicommodity flow problem do not have as 'nice' a structure as single commodity flow problem, they still are linear programs (if we ignore integration of variable). As we shall soon see, multi commodity flow problem do have special structure that permits the application of decomposition techniques.

Thus we may apply the block diagonal decomposition techniques to the fore going problem. The multicommodity minimal cost flow problem has $(t+1)n$ variables and $n+mt$ constraints. (Including the slack variables for the coupling constraints and ignoring the non negativity and upper bound constraints $0 \leq x_{kij} \leq u_{kij}$).Thus, even for moderate sized problems, the constraint matrix will be large. For example, suppose that we have a problem with 100 nodes, 250 arcs and 10 commodities, the problem will have 2750 variables and 1250 constraints.

2.2.The Decomposition principle

In practice, many linear programming problems are simply too large to fit into today's computer's. It is not unusual in a corporate management model or in a logistics model to produce a linear program with many thousands of rows and a seemingly unlimited number of columns. In such problems some method must be applied to convert the large problems in to one or more smaller problem of manageable size. Fortunately, there is a technique, called the decomposition principle that does exactly this.

Even if a linear program is of manageable size, certain of its constraints may possess special structure that would permit efficient handing. In such cases we would like to separate the linear program in to one with general structure and one with special structure where a more efficient method may be applied. Again the decomposition principle can be applied to such a linear program to achieve the desired effect.

The decomposition principle is a systematic procedure for solving large scale linear programs or linear programs that contain constraints of special structure. The constraints are divided in to two sets: general constraints (or complicating constraints) and constraints with special structure.

The strategy of the decomposition procedure is to operate on two separate linear programs: one over the set of general constraints and over the set of special constraints. Information is passed back and forth between the two linear programs until a point is reached where the solution to the original problem is achieved. The linear program over the general constraint is called the master problem and the linear program over the special constraint is called subproblem. The master problem passes down a new set of cost coefficients to the subproblem and receives a new column based on these cost coefficients.

We shall begin by assuming that the special constraint set is bounded. Once the decomposition principle is developed for this case and we have discussed how to get started, we shall relax the boundedness assumption and also extend the procedure to multiple subproblems. But, this project deals only with bounded.

The decomposition algorithm

Consider the following linear program, where χ a polyhedral set representing constraints of special structure, A is an $m \times n$ matrix and b is an m vector.

$$minimize \quad cx$$

$$subject\ to \quad Ax = b$$

$$x \in \chi$$

To simplify the presentation, assume that χ is bounded (this assumption can be relaxed) since χ bounded polyhedral set, then any point $x \in \chi$ can be represented as a convex combination of the finite number of extreme points of χ. Denoting these points by $x_1, x_2, \dots, x_t$ any $x \in \chi$ can be represented as:

$$x = \sum_{j=1}^{t} \lambda_j x_j$$

$$\sum_{j=1}^{t} \lambda_j = 1$$

$$\lambda_j \geq 0, \quad j = 1,2, \dots, t.$$

Substituting for x, the foregoing optimization problem can be transformed in to the following problems in the variables $\lambda_1, \lambda_2, \dots, \lambda_t$.

$$\text{minimize} \quad \sum_{j=1}^{t} (cx_j)\ \lambda_j$$

$$\text{subject to} \quad \sum_{j=1}^{t} (Ax_j)\ \lambda_j = b \qquad (2.1)$$

$$\sum_{j=1}^{t} \lambda_j = 1 \qquad (2.2)$$

$$\lambda_j \geq 0, \quad j = 1,2, \dots, t. \qquad (2.3)$$

Since t, the number of extreme points of the set χ, is usually very large, attempting to explicitly enumerate all the extreme points $x_1, x_2, \dots, x_t$ and explicitly solving this problem is a very difficult task. Rather, we shall attempt to find an optimal solution of the problem (and hence the original problem) without explicitly enumerating all the extreme points.

2.3. Application of the revised simplex method

Consider solving the foregoing problem by the revised simplex method. Suppose that we have a basic feasible solution $\lambda = (\lambda_B, \lambda_N)$. Further suppose that the $(m+1) \times (m+1)$ basis inverse B^{-1} is known (the process of initialization is discussed later). Denoting the dual variables corresponding to equation (2.1) and (2.2) by $w \; and \; \alpha$ we get $(w, \alpha) = \hat{c}_B B^{-1}$, where $\hat{c}_B$ is the cost of the basic variables with $\hat{c}_j = cx_j$ for each basic variable λ_j. the basis inverse, the dual variables, the values of the basic variables, and the objective function are displayed below, where $\bar{b} = B^{-1}\begin{pmatrix} b \\ 1 \end{pmatrix}$.

Basis	RHS
(w, α)	$\hat{c}_B \bar{b}$
B^{-1}	$\bar{b}$

The revised simplex method proceeds by concluding that the current solution is optimal or else by deciding to increase a nonbasic variable. This is done by first calculating

$$z_k - \hat{c}_k = \underset{1 \leq j \leq t}{\text{Maximum}} \; z_j - \hat{c}_j$$

$$= \underset{1 \leq j \leq t}{\text{Maximum}} \; (w, \alpha) \begin{bmatrix} Ax_j \\ 1 \end{bmatrix} - cx_j$$

$$= \underset{1 \leq j \leq t}{\text{Maximum}} \; wAx_j + \alpha - cx_j \qquad (2.4)$$

Since $z_j - \hat{c}_j = 0$ for basic variables, then the foregoing maximum is ≥ 0. thus if $z_k - \hat{c}_k = 0$, then $z_j - \hat{c}_j \leq 0$ for all nonbasic variables and the optimal solution is at hand. On the other hand, $if \; z_k - \hat{c}_k > 0$, then the nonbasic variable x_k is increased.

Determine the index k using equation (2.4) is computationally infeasible because t is very large and the extreme points $x_j's$ corresponding to the nonbasic $\lambda_j's$ are not explicitly known. Therefore an alternative scheme must be devised. Since χ is

bonded polyhedral set, the maximum of any linear objective can be achieved at one of the extreme points. Therefore

$$\underset{1\leq j\leq t}{\text{maximum}}(wA-c)x_j+\alpha=\max_{x\in\chi}(WA-c)x+\alpha$$

To summarize, given a basic feasible solution $(\lambda_B,\lambda_N)\,with$ dual variables (w,α) solve the following linear subproblem which is easy because of the special structure of χ.

$$maximize\quad (wA-c)x+\alpha$$

$$subject\ to\quad x\in\chi$$

Note that the objective function contains a constraint. This is easy handled by initializing the RHS value for z to α instead of the normal value of 0. Let x_k be an optimal solution to the forgoing sub problem with objective value $z_k-\hat{c}_k$. if $z_k-\hat{c}_k=0$, then the basic feasible solution (λ_B,λ_N) is optimal. Otherwise if $z_k-\hat{c}_k>0$, then the variable λ_k enters the basis. As in the revised simplx method the corresponding column $\begin{pmatrix}Ax_k\\1\end{pmatrix}$ is updated by premultiplying it by B^{-1} giving $y_k=B^{-1}\begin{pmatrix}Ax_k\\1\end{pmatrix}$. Note that $y_k\leq 0$ cannot occur since χ was assumed bounded; producing a bounded master problem. The updated column $\begin{pmatrix}z_k-\hat{c}_k\\y_k\end{pmatrix}$ is adjoined to the forgoing above array. The variable λ_{Br} leaving the basis is determined by the usual minimum ratio test. The basis inverse, dual variables and right RHS are updated by pivoting at y_{rk}. After updating, the process is repeated.

Now we have all the ingredients of the decomposition algorithm, assuming of which is given below. Note that the master step given an improved feasible solution of the overall problem, and the sub problem checks whether $z_j-\hat{c}_j\leq 0$ for all λ_j, or else determines the most positive $z_k-\hat{c}_k$.

Summary of the decomposition algorithm

Initialization step: –

Find an initial basic feasible solution of the system defend by equation (2.1), (2.2) and (2.3) (getting an initial basic feasible solution is discussed later). Let the basis be B and from the following master array where

$(w, \alpha) = \hat{c}_B B^{-1}$ (recall that $\hat{c}_j = cx_j$), and $\bar{b} = B^{-1}\begin{bmatrix} b \\ 1 \end{bmatrix}$

Basis inverse	RHS
(w, α)	$\hat{c}_B \bar{b}$
B^{-1}	$\bar{b}$

Main step:

1. Solve the following sub problem

$$maximize \quad (wA - c)x + \alpha$$

$$subject\ to \quad x \in \chi$$

Let x_k be an optimal basic feasible solution with objective value of $z_k - \hat{c}_k$. if $z_k - \hat{c}_k = 0$ stop; the basic feasible solution of the last master step is an optimal solution of the overall problem. Otherwise go to step 2 below.

2. Let $y_k = B^{-1}\begin{pmatrix} Ax_k \\ 1 \end{pmatrix}$ and adjoin the update column $\begin{pmatrix} z_k - \hat{c}_k \\ y_k \end{pmatrix}$ to the master array. Pivot at y_{rk} where the index r is determined as follows:

$$\frac{\overline{b_r}}{y_{rk}} = \underset{1 \le i \le m+1}{\text{Minimum}} \left\{ \frac{\overline{b_i}}{y_{ik}} : y_{ik} > 0 \right\}$$

This updates the dual variables, the basis inverse and the right hand side. After pivoting, the column of λ_k is deleted and step 1 is repeated.

Getting started

In this section we describe a method to obtain a starting basic feasible solution for the master problem using artificial variables if necessary. These artificial variables are eliminated by use of phase I or by the big-M method. If at termination there is a positive artificial variable, then the overall problem has no feasible solution.

Inequality constraints

Consider the following problem

$$\text{minimize} \quad \sum_{j=1}^{t} (cx_j)\ \lambda_j$$

$$\text{subject to} \quad \sum_{j=1}^{t} (Ax_j)\ \lambda_j \leq b$$

$$\sum_{j=1}^{t} \lambda_j = 1$$

$$\lambda_j \geq 0,\ \ j = 1,2, \dots, t.$$

If there is a convenient $x_1 \in \chi$ with $Ax_1 \leq b$, then the following basis is at hand, where the identity corresponds to the slack vector $S \geq 0$.

$$B = \left[\begin{array}{c|c} \mathrm{I} & Ax_1 \\ \hline 0 & 1 \end{array}\right],\ \ B^{-1} = \left[\begin{array}{c|c} \mathrm{I} & -\mathrm{Ax}_1 \\ \hline 0 & 1 \end{array}\right]$$

The initial array is given by the following tableau.

		Basis inverse	RHS
z	0	cx_1	cx_1
S	I	$-Ax_1$	$b - Ax_1$
λ_1	0	1	1

Now suppose that there is no obvious $x \in \chi$ with $Ax \leq b$. In this case after converting the master problem to equality form by adding appropriate slack variables, the constraints are manipulated so that the RHS value are non negative. Then artificial variables are added, as needed, to create an identity matrix. This identity matrix constitutes the starting basis. The two phase or big-M method can be used to drive the artificial variable out of the basis.

Equality constraints

In the case $m + 1$ artificial variable can be introduced to form the initial basis. The artificial variable are eliminated by the two phase or big-M method.

From now on let x_i is the vector of flows of commodity i in the network, u_i represent the vector of upper limits on flow for commodity i in the arcs of the network. Let u represent the vector of upper limits on the sum of all commodities flowing in the arcs of the network, b_i represent the vector of supplies (or demands) of commodity i in the network and c_i represent the vector of arc costs in the network for commodity i.

Consider the application of the decomposition algorithm to the minimal cost multi commodity flow problem, let $\chi_i = \{x_i : Ax_i = b_i,\ 0 \leq x_i \leq u_i\}$ assume that each component of u_i is finite so that χ_i is bounded. (We can relax it). Then any $x_i \in \chi_i$ can be expressed as a convex combination of the extreme point of χ_i as follows:

$$x_i = \sum_{j=1}^{k_i} \lambda_{ij} \chi_{ij}$$

where

$$\sum_{j=1}^{k_i} \lambda_{ij} = 1$$

$$\lambda_{ij} \geq 0,\ \ j = 1,2, \dots, k_i.$$

And $x_{i1}, x_{i2}, \dots, x_{ik_i}$ are extreme points of χ_i. Substituting x_i in the multi commodity minimal cost flow problem and denoting the vector of slacks by s, we get the following.

$$\text{minimize} \quad \sum_{i=1}^{t}\sum_{j=1}^{k_i}\left(c_i x_{ij}\right)\lambda_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{t}\sum_{j=1}^{k_i} x_{ij}\,\lambda_{ij} + s = u$$

$$\sum_{j=1}^{k_i}\lambda_{ij} = 1 \qquad i = 1,2,\dots,t$$

$$\lambda_{ij} \geq 0 \qquad j = 1,2,\dots,k_i, \quad i = 1,2,\dots,t$$

$$s \geq 0$$

Suppose that we have a basic feasible solution to the multi commodity minimal cost flow problem in terms of the $\lambda_{ij}'s$ and let (w, α) be the vector of dual variables corresponding to the basic feasible solution (w has n component and α has t component). Then dual feasibility is given by the following two conditions:

1. $w_{pq} \leq 0$ corresponding to each s_{pq}, and
2. $wx_{ij} + \alpha_i - c_i x_{ij} \leq 0$ corresponding to each λ_{ij}

If any of these conditions is violated, the corresponding variable (s_{pq} or λ_{ij}) is a candidate to enter the master basis. Here s_{pq} is a candidate to enter the basis if $w_{pq} > 0$. For a given commodity i, a non basic variable among the λ_{ij}'s could enter the basis if the optimal objective of the following sub problem is positive.

$$\text{maximize} \quad (w - c_i)x_i + \alpha_i$$

$$\text{subject to} \quad Ax_i = b_i$$

$$0 \leq x_i \leq u_i$$

But, since A is a node-arc incidence matrix, this is simply a single commodity flow problem. Thus, it may be solved by one of the efficient techniques for solving single - commodity network flow problem (network simplex).

Decomposition algorithm for multicommodity minimal cost flow problem

Initialization step: - Begin with a basic feasible solution to the master problem. Store $B^{-1}, \bar{b} = B^{-1}\begin{pmatrix} u \\ 1 \end{pmatrix}$, and $(w, \alpha) = \hat{c}_B B^{-1}$,
Where $\hat{c}_{ij} = c_i x_{ij}$ (the two phase or big-M method may be required).

MAIN STEP

1. Let (w, α) be the vector of dual variables corresponding to the current basic feasible solution to the master problem. If any $w_{pq} > 0$, then the corresponding s_{pq} is a canditate to enter the master basis. if $w_{pq} \leq 0$, for each arc, consider the following i^{th} sub problem.

$$\text{maximize } (w - c_i)x_i + \alpha_i$$
$$\text{subject to } Ax_i = b_i$$
$$0 \leq x_i \leq u_i$$

This is a single commodity flow problem. If the solution x_{ik} to this problem has $z_{ik} - c_{ik} = (w - c_i)x_{ik} + \alpha_i > 0$, then λ_{ik} is a candidate to enter the master basis.

2. If there is no candidate to enter the master basis, then stop; the optimal solution is at hand. Otherwise, select a candidate variable, update its column accordingly, $B^{-1}\begin{pmatrix} e_{pq} \\ 0 \end{pmatrix}$ For s_{pq} and $B^{-1}\begin{pmatrix} x_{ik} \\ e_i \end{pmatrix}$ for λ_{ik} , and pivot.

[Note that e_{pq} is a unit vector with the 1 in the row associated with arc(p, q).] This updates the basis inverse, the dual variables, and the right hand side. Return to step1.

An example of the multi commodity minimal cost flow algorithm

Consider the two-commodity minimal cost flow problem whose data are given in the figure below (Fig 2.4)

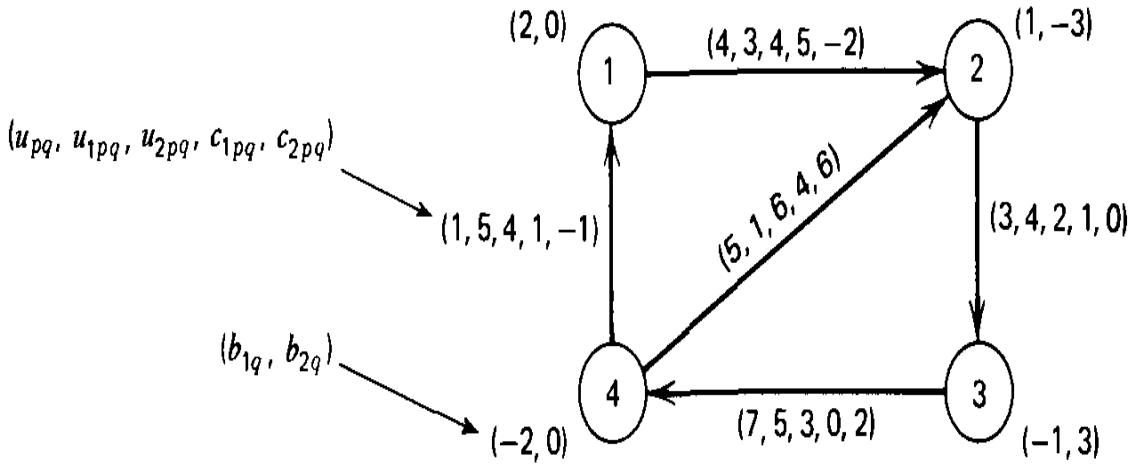


Fig 2.4 A two commodity minimal cost flow problem

The constraint matrix and the right hand side are display in fig (2.5) below (the lower and upper bound constraint $0 \leq x_1 \leq u_1, 0 \leq x_2 \leq u_2$ are not displayed). Notice the structure of the coupling constraints and the special structured block diagonal constraint. Also note that x_1 and x_2 repersent the artificial variables for the two commodities.

Initialization

To avoid the two phase or the big-M methods, suppose that we begin with the following feasible solutions.

$$X_{11} = \begin{bmatrix} x_{112} \\ x_{123} \\ x_{134} \\ x_{141} \\ x_{142} \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 2 \\ 0 \\ 0 \end{bmatrix} \quad and \quad X_{21} = \begin{bmatrix} x_{212} \\ x_{223} \\ x_{234} \\ x_{241} \\ x_{242} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 3 \\ 0 \\ 3 \end{bmatrix}$$

Note that the master basis (in the space of the slack variable and the $\lambda_{ij}'s$) consists of all the slacks, λ_{11} and λ_{21}. The basis and its inverse are

	FIRST COMMODITY VARIABLES						SECOND COMMODITY VARIABLES						SLACK VARIABLES					
	x_{112}	x_{123}	x_{134}	x_{141}	x_{142}	x_1	x_{212}	x_{223}	x_{234}	x_{241}	x_{242}	x_2	s_{12}	s_{23}	s_{34}	s_{41}	s_{42}	RHS
	−5	−1	0	−1	−4	0	2	0	−2	1	−6	0	0	0	0	0	0	0
Coupling Constraints	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	4
	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	3
	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	7
	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	1
	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	5
Node-arc incidence matrix for subproblem 1	1	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	2
	−1	1	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1
	0	0	−1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	−2
Node-arc incidence matrix for subproblem 2	0	0	0	0	0	0	1	0	0	−1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	−1	1	0	0	−1	0	0	0	0	0	0	−3
	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	0	0	0	3
	0	0	0	0	0	0	0	0	−1	1	1	1	0	0	0	0	0	0

Fig2.5. The constraint matrix for the two-commodity problem of fig 2.4.

$$\mathbf{B} = \begin{array}{c} \begin{matrix} s_{12} & s_{23} & s_{34} & s_{41} & s_{42} & \lambda_{11} & \lambda_{21} \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \quad \mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 1 & 0 & 0 & 0 & -3 & 0 \\ 0 & 0 & 1 & 0 & 0 & -2 & -3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Here $c_1 x_{11} = 13$ and $c_2 x_{21} = 24$ .Denoting $\begin{bmatrix} u \\ 1 \end{bmatrix}$ by $\hat{b}$ we have that

$$(w, \alpha) = \hat{c}_B B^{-1} = (0,0,0,0,0,13,24)B^{-1}$$

$$= (0,0,0,0,0,13,24).$$

$$X_B = B^{-1}\hat{b} = B^{-1}\begin{bmatrix} u \\ 1 \\ 1 \end{bmatrix} = B^{-1}\begin{bmatrix} 4 \\ 3 \\ 7 \\ 1 \\ 5 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 1 \\ 2 \\ 1 \\ 1 \end{bmatrix}$$

$z = \hat{c}_B B^{-1}\hat{b} = 37$

Setting up the revised simplex array

(w, α)	$\hat{c}_B B^{-1}\hat{b}$
B^{-1}	$B^{-1}\hat{b}$

For the master problem, we get the following.

	w_{12}	w_{23}	w_{34}	w_{41}	w_{42}	α_1	α_2	RHS
z	0	0	0	0	0	13	24	37
s_{12}	1	0	0	0	0	− 2	0	2
s_{23}	0	1	0	0	0	− 3	0	0
s_{34}	0	0	1	0	0	− 2	− 3	2
s_{41}	0	0	0	1	0	0	0	1
s_{42}	0	0	0	0	1	0	− 3	2
λ_{11}	0	0	0	0	0	1	0	1
λ_{12}	0	0	0	0	0	0	1	1

Iteration 1

First, all $w_{pq} \leq 0$. Next we check whether a candidate from either subproblem (or commodity) is eligible to enter the master basis.

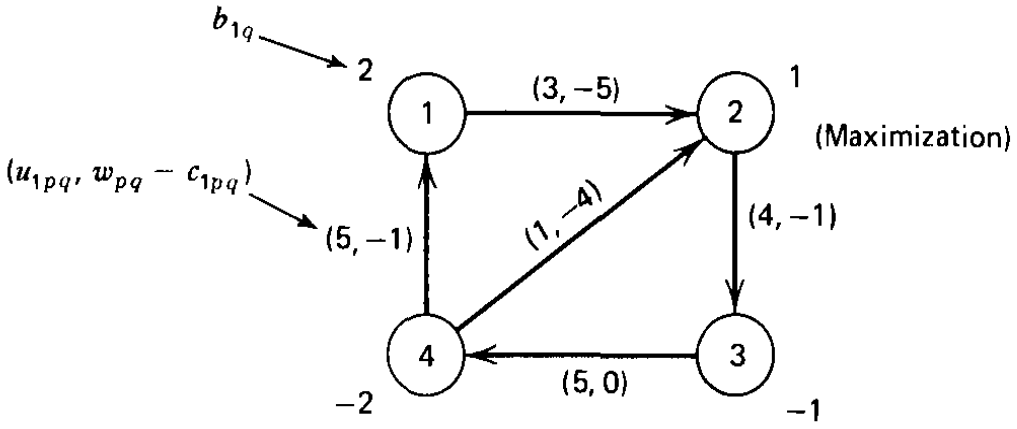


Fig2.6. subproblem 1 at the first iteration

Sub problem 1

$$w - c_1 = 0 - c_1 = (-5, -1, 0, -1, -4)$$

Subproblem 1 is the single commodity flow problem defined in Fig 2.6. The optimal (maximal cost) solution is $x_{12} = (2, 3, 2, 0, 0)^t$ and the value of the subproblem 1 objective is

$$z_{12} - c_{12} = (w - c_1)x_{12} + \alpha_1 = -13 + 13 = 0$$

Thus there is no candidate from subproblem 1.

Sub problem 2

$$w - c_2 = 0 - c_2 = (2, 0, -2, 1, -6)$$

Subproblem 2 is the single - commodity flow problems define in Fig 2.7. The optimal (maximal cost) solution is $x_{22} = (3, 0, 3, 3, 0)^t$ and

$$z_{22} - c_{22} = (w - c_2)x_{22} + \alpha_2 = 3 + 24 = 27.$$

Thus λ_{22} is a canditate to enter the basis. The updated column for λ_{22}

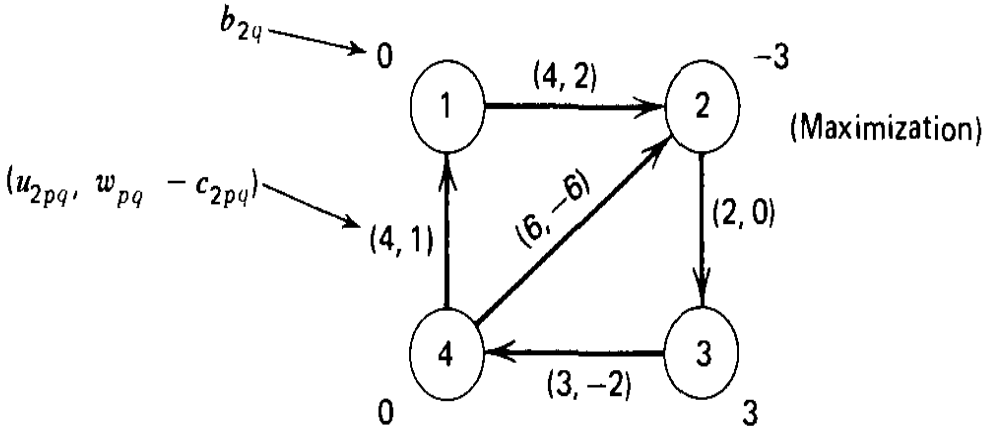


Fig2.7. Subproblem 2 at the first iteration

(exclusive of $z_{22} - c_{22}$) is

$$B^{-1}\begin{bmatrix} x_{22} \\ 0 \\ 1 \end{bmatrix} = (3,0,0,3,-3,0,1)^t$$

The pivoting process is as follows.

	w_{12}	w_{23}	w_{34}	w_{41}	w_{42}	α_1	α_2	RHS	λ_{22}
z	0	0	0	0	0	13	24	37	27
s_{12}	1	0	0	0	0	−2	0	2	3
s_{23}	0	1	0	0	0	−3	0	0	0
s_{34}	0	0	1	0	0	−2	−3	2	0
s_{41}	0	0	0	1	0	0	0	1	(3)
s_{42}	0	0	0	0	1	0	−3	2	−3
λ_{11}	0	0	0	0	0	1	0	1	0
λ_{21}	0	0	0	0	0	0	1	1	1

	w_{12}	w_{23}	w_{34}	w_{41}	w_{42}	α_1	α_2	RHS
z	0	0	0	−9	0	13	24	28
s_{12}	1	0	0	−1	0	−2	0	1
s_{23}	0	1	0	0	0	−3	0	0
s_{34}	0	0	1	0	0	−2	−3	2
λ_{22}	0	0	0	$\frac{1}{3}$	0	0	0	$\frac{1}{3}$
s_{42}	0	0	0	1	1	0	−3	3
λ_{11}	0	0	0	0	0	1	0	1
λ_{21}	0	0	0	$-\frac{1}{3}$	0	0	1	$\frac{2}{3}$

Iteration 2

Again all $w_{pq} \leq 0$ (so no s_{pq} is a canditate to enter the master basis).

Sub problem 1

$w - c_1 = (-5, -1, 0, -10, -4)$

Subproblem 1 is the single commodity flow problem defined in Fig 2.8.

The optimal solution is $x_{13} = (2,\ 3,\ 2,\ 0,\ 0)^t$ with

$$z_{13} - c_{13} = (w - c_1)x_{13} + \alpha_1 = -13 + 13 = 0$$

Thus there is no candidate from subproblem 1.

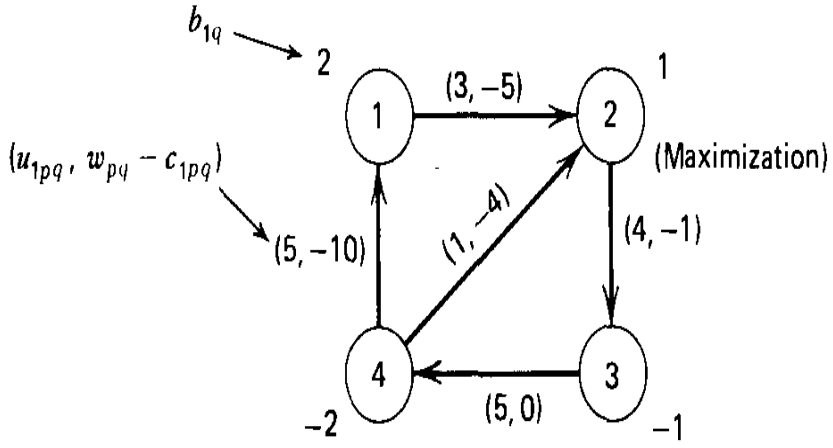


Fig 2.8.subproblem 1 at the second iteration

Sub problem 2

$w - c_2 = (2, 0, -2, -8, -6)$

Subproblem 2 is the single commodity flow problem defined in Fig 2.9. An optimal (maximal cost) solution is $x_{23} = (3,0,3,3,0)^t$ with

$$z_{23} - c_{23} = (w - c_2)x_{23} + \alpha_2 = -24 + 24 = 0.$$

Thus, there is no candidate from sub problem 2.

Therefore we already have the optimal solution as follows:

$$z^* = 28$$

$$x_1^* = \lambda_{11} x_{11} = (2,3,2,0,0)^t$$

$$x_2^* = \lambda_{21} x_{21} + \lambda_{22} x_{22}$$

$$= \frac{2}{3}(0,0,3,0,3)^t + \frac{1}{3}(3,0,3,3,0)^t$$

$$= (1,0,3,1,2)^t$$

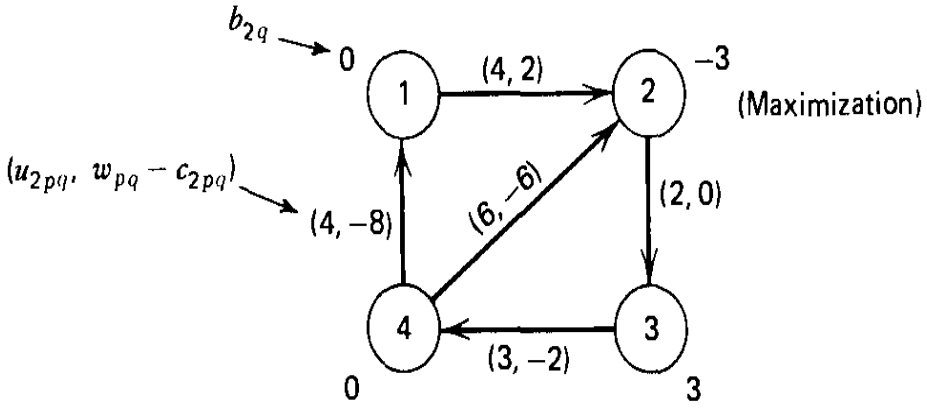


Fig2.9. Subproblem 2 at the second iteration.

2.4. Minimal cost multicommodity transportation problem

Definition:- A minimal cost multicommodity transportation problem is an optimization problem where one is interested in identifying the least cost method of moving two or more types of commodities from certain source (plants) where they are produced to certain destination where they are consumed without violating the (demand/supply) requirement and arc capacities for each commodity.

Here after, when we refer to the **MCTP,** we mean minimal cost multicommodity transportation problem.

2.4.1 MCTP formulation:-

Suppose that we are given a network G with m source and n destnation in which there will be a flow of t different commodities. To formulate the problem, let us define the following terms:

- a_i^k = Number of units of commodity k avilable at source i $(i = 1,2, \dots m)$ and k $(k = 1,2, \dots t)$
- b_j^k = Number of units of commodity k requred at destination j $(j = 1,2, \dots n)$ and k $(k = 1,2, \dots t)$
- c_{ij}^k = unit transportation cost of commodity k from source i to destination j. $(i = 1,2, \dots, m\ ;\ j = 1,2, \dots, n\ ;\ k = 1,2, \dots, t)$
- u_{ij}^k = the upper limit of commodity k from source i to destination j

We assume that the total product availability is equal to the total product requirement for each commodity k $(k = 1,2, \dots t)$.

$$i.e. \quad \sum_{i=1}^{m} a_i^k = \sum_{j=1}^{n} b_j^k = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}^k \qquad \forall k = 1,2, \dots, t \qquad (*)$$

If we define the decision variable as:

- x_{ij}^k = The amount of flow for commodity k on arc (i,j) for $(i = 1,2, \dots, m\ ;\ j = 1,2, \dots, n\ ; k = 1,2 \dots, t)$.

We may then formulate the linear programming for the minimal cost multi commodity transportation problem (**MCTP**) as follow:

$$\text{minimize} \quad \sum_{k=1}^{t}\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}^{k}\, x_{ij}^{k} \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ij}^{k} = a_i^k \qquad \forall i = 1,2,\dots,m \quad \forall k = 1,2,\dots,t \tag{2}$$

$$\sum_{i=1}^{m} x_{ij}^{k} = b_j^k \qquad \forall j = 1,2,\dots,n \quad \forall k = 1,2,\dots,t \tag{3}$$

$$x_{ij}^{k} \geq 0 \qquad \forall k = 1,2,\dots,t \quad \forall j = 1,2,\dots,n \quad \forall i = 1,2,\dots,m \tag{4}$$

This problem has $k(n+m)$ constraints and $k(nm)$ variables.

Expression (1) represents the minimization of the total distribution cost of all commodities. Equation (2) states the amount of commodity k being shipped from source i to all possible destinations nodes j should be equal to the total availability a_i^k at the i^{th}source node. Equation (3) indicates that the amount of commodity k being shipped to destination j from all possible sources should be equal to the requirements b_j^k at the j^{th} destination node.

The last equation (4) $x_{ij}^{k} \geq 0$ tells us the non - negativity of the flow x_{ij}^{k} shipped from source i to destination j for each commodity k. In fact it is not mean it has unlimited capacity. Because, the number of supply available at source node and the number of demand required by the destination node for each commodity is known and use one of the methods (North west corner method or least cost method) x_{ij}^{k} can not exceeds from the weight at the source node or destination node rather it is equal to the minimum number of units available at the source node or destination node at that time. I.e. $0 \leq x_{ij}^{k} \leq min\{a_i^k, b_j^k\}$.

Hence in other word it tells us there is no limitation on the sum of the commodities. If there is a limitation to the sum of the commodities (bundle constraint) then this restricts all commodities to be between some values and this value may not be the

minimum of the available units at the source node and required units at the destination node. And this contradicts the procedure of finding initial basic solutions.

As we can see from the below example the constraint of **MCTP** has a special structure called block diagonal structure, where the diagonal block matrices are the node arc incidence matrix of the network. Such a problem can be decomposes into t single commodity transportation problem one for each commodity called sub problems. Solving the subproblems by the method we discuss in chapter one and add the solution of the sub problems will give you the solution of the original multicommodity transportation problem. To see this let's see the example below.

Example:-

A company produces two types of products A and B at two plants (sources) p_1 and p_2. It then ships these products to three market zones (destinations) M_1, M_2 and M_3. for $k = 1,2;\ i = 1,2$ and $j = 1,2,3$ the following data are given:

- The cost of shipping one unit of product k from plant i to zone j.
- The maximum number of units that can be shipped from each plant to each market zone.
- The demand for product k at market zone j.
- Upper limit for the flow on each commodity k from source i to destination j.

Solve the problem of minimizing transportation cost.

	c_{ij}	M_1	M_2	M_3	supply
Product A	p_1	6	4	5	150
	p_2	5	3	6	130
Product B	p_1	10	3	9	140
	p_2	8	4	6	170
	Demand A	100	70	110	
	Demand B	150	30	130	

Table2.4.1 cost from plants to destinations for both commodities.

Solution

$$\text{Min } (6x_{11}^1 + 4x_{12}^1 + 5x_{13}^1 + 5x_{21}^1 + 3x_{22}^1 + 6x_{23}^1 + 10x_{11}^2 + 3x_{12}^2 + 9x_{13}^2 + 8x_{21}^2 + 4x_{22}^2 + 6x_{23}^2)$$

Subject to

$$x_{11}^1 + x_{12}^1 + x_{13}^1 = 150$$

$$x_{11}^2 + x_{12}^2 + x_{13}^2 = 140$$

$$x_{21}^1 + x_{22}^1 + x_{23}^1 = 130$$

$$x_{21}^2 + x_{22}^2 + x_{23}^2 = 170$$

$$x_{11}^1 + x_{21}^1 + x_{31}^1 = 100$$

$$x_{11}^2 + x_{21}^2 + x_{31}^2 = 150$$

$$x_{12}^1 + x_{22}^1 + x_{32}^1 = 70$$

$$x_{12}^2 + x_{22}^2 + x_{32}^2 = 30$$

$$x_{13}^1 + x_{23}^1 + x_{33}^1 = 110$$

$$x_{13}^2 + x_{23}^2 + x_{33}^2 = 130$$

$$x_{ij}^1, x_{ij}^2 \geq 0$$

		x_{11}^1	x_{12}^1	x_{13}^1	x_{21}^1	x_{22}^1	x_{23}^1	x_{21}^2	x_{12}^2	x_{13}^2	x_{21}^2	x_{22}^2	x_{23}^2	b_i
Product A	p_1	1	1	1	0	0	0	0	0	0	0	0	0	150
	p_2	0	0	0	1	1	1	0	0	0	0	0	0	130
	m_1	-1	0	0	-1	0	0	0	0	0	0	0	0	-100
	m_2	0	-1	0	0	-1	0	0	0	0	0	0	0	-70
	m_3	0	0	-1	0	0	-1	0	0	0	0	0	0	-110
Product B	p_1	0	0	0	0	0	0	1	1	1	0	0	0	140
	p_2	0	0	0	0	0	0	0	0	0	1	1	1	170
	m_1	0	0	0	0	0	0	-1	0	0	-1	0	0	-150
	m_2	0	0	0	0	0	0	0	-1	0	0	-1	0	-30
	m_3	0	0	0	0	0	0	0	0	-1	0	0	-1	-130

Node - arc incidence matrix

Let $N=\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 \end{pmatrix}$

Then the original problem becomes

$$\text{minimize } Z = \sum_{k=1}^{2}\sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^k x_{ij}^k$$

$$\text{Subject to } \begin{pmatrix} N & 0 \\ 0 & N \end{pmatrix} \begin{pmatrix} x_{ij}^1 \\ x_{ij}^2 \end{pmatrix} = \begin{pmatrix} \hat{b}_1^1 \\ \hat{b}_2^2 \end{pmatrix}$$

$$0 \leq x_{ij}^k$$

Where $x_{ij}^1 = (x_{11}^1\ x_{12}^1\ x_{13}^1\ x_{21}^1\ x_{22}^1\ x_{23}^1)$

$x_{ij}^2 = (x_{11}^2\ x_{12}^2\ x_{13}^2\ x_{21}^2\ x_{22}^2\ x_{23}^2)$

$$\hat{b}_1^1 = \begin{pmatrix} a_1^1 \\ a_2^1 \\ b_1^1 \\ b_2^1 \\ b_3^1 \end{pmatrix} \text{ And } \hat{b}_2^2 = \begin{pmatrix} a_1^2 \\ a_2^2 \\ b_1^2 \\ b_2^2 \\ b_3^2 \end{pmatrix}$$

And since the constraints have special structure called the block diagonal structure we can decompose it into two sub problems as below:

Now let $Z_1 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^1 x_{ij}^1$

$Z_2 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^2 x_{ij}^2$

The original problem can be separated in to two sub problems and solving them separately and adding their solution will give the solution of the original problem.

The original problem is equivalent with the below problem

$$minimize\ Z_1 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^1 x_{ij}^1 \quad + \quad minimize\ Z_2 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^2 x_{ij}^2$$

$$subject\ to \quad Nx_{ij}^1 = \hat{b}_1^1 \qquad subject\ to \quad Nx_{ij}^2 = \hat{b}_2^2$$

$$0 \le x_{ij}^1 \qquad 0 \le x_{ij}^2$$

Now let's solve the two sub problems separately

Solution for sub problem 1(commodity A)

$$minimize\ Z_1 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^1 x_{ij}^1$$

$$subject\ to \quad Nx_{ij}^1 = \hat{b}_1^1$$

$$0 \le x_{ij}^1$$

This is single commodity transportation problem can solve as we discuss in chapter one.

Step 1: find the basic initial solution

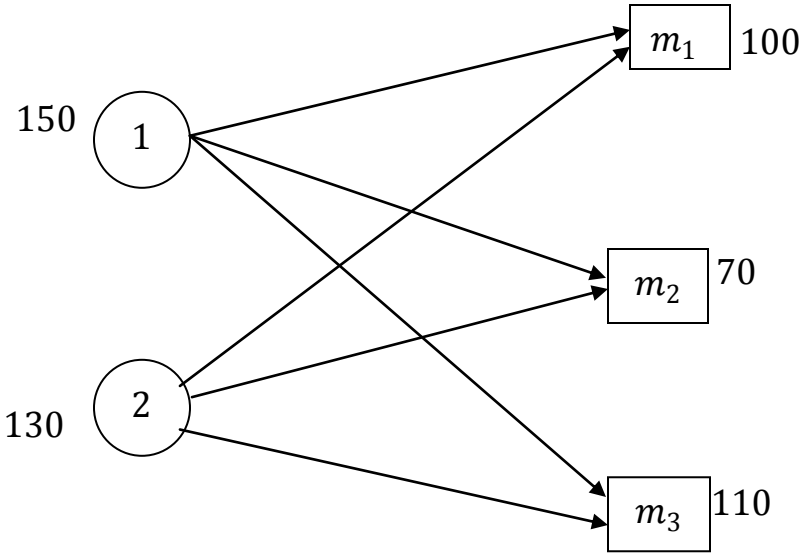


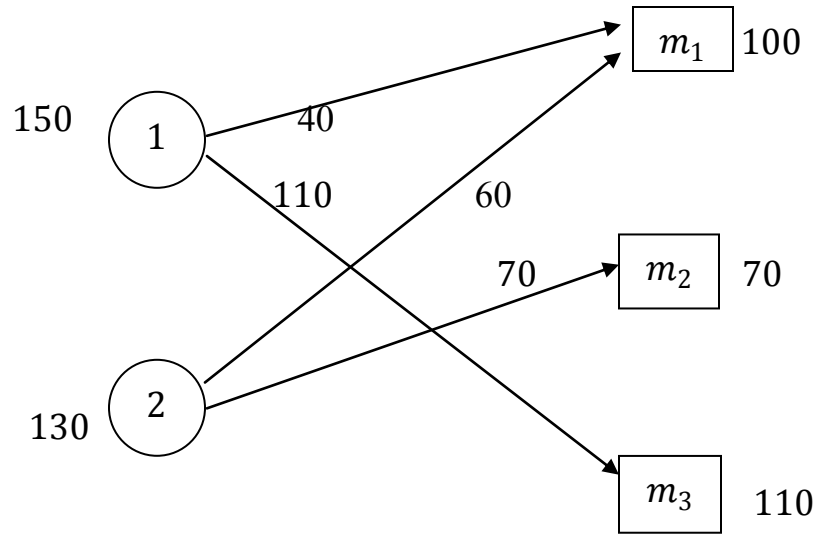
Fig1. Network for the commodity A

To go through let's first find the initial basis using the least cost method

6 40	4	5 110	150
5 60	3 70	6	130
100	70	110	

Table1. Finding an initial basis by the least cost method

The basic variables are $(m + n - 1)$ allocated cells in the table above or the arcs in the spanning tree below.



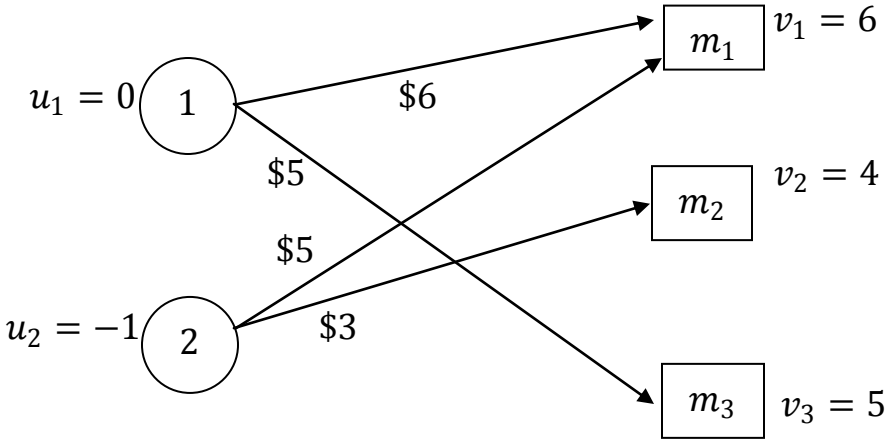
Spanning tree for initial basic solution

Step2: check the optimality of the basis.

Using the basic variables (variables on the spanning tree) we get the simplex multipliers corresponding to all sources and destinations that satisfy

$$c_{ij} = u_i + v_j \qquad (1)$$

And since condition (1) consists $(m + n - 1 = 4)$ equations in $(m + n = 5)$ any one of the simplex multiplier can be given arbitrary value.Let $u_1 = 0$. given $u_1 = 0, v_1 = 6, v_3 = 5$ are immediate from (1); then $u_2 = -1$ is immediate from v_1 and $v_2 = 4$ is immediate from u_2.



Now the reduced costs for the non basic variables using $\bar{c}_{ij} = c_{ij} - u_i - v_j$

6	4	5	0
–	0	–	
5	3	6	−1
–	–	2	
6	4	5	u_i / v_j

Since the reduced cost $\bar{c}_{ij} \geq 0$ for the non basic variables (x_{11}^1, x_{23}^1) the solution is optimal.

$x_{ij}^1 = (x_{11}^1\ x_{12}^1\ x_{13}^1\ x_{21}^1\ x_{22}^1\ x_{23}^1)$ = (40,0,110,60,70,0) is the optimal solution and $Z_1^* = 1300$ is the optimal value.

Solution for sub problem 2(commodity B)

$minimize\ \ Z_2 = \sum_{i=1}^{2}\sum_{j=1}^{3} c_{ij}^2 x_{ij}^2$

$subject\ to \qquad Nx_{ij}^2 = \hat{b}_2^2$

$0 \leq x_{ij}^2$

Step 1: find the basic initial solution

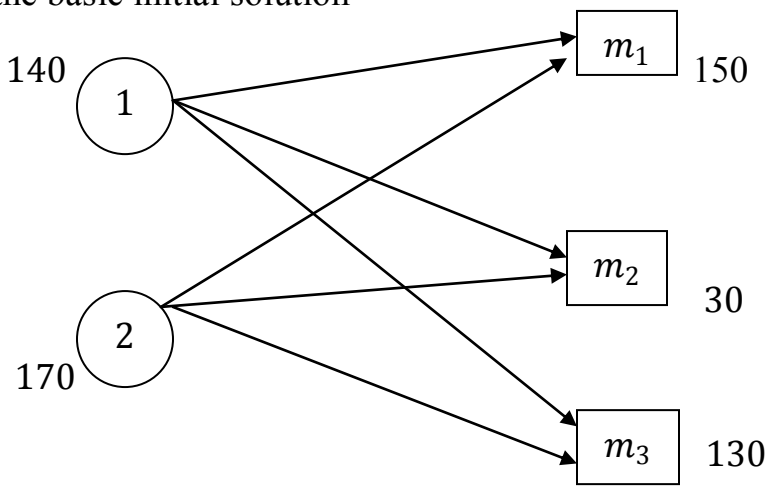
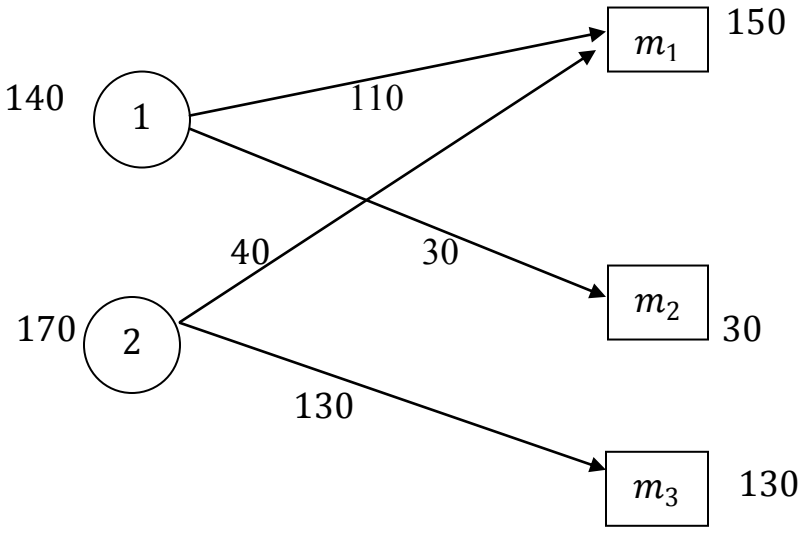


Fig2. Network for the commodity B

To go through let's first find the initial basis using the least cost method

10 110	3 30	9	140
8 40	4	6 130	170
150	30	130	

The basic variables are the $(m + n - 1 = 4)$ allocated cells in the table above or the arcs in the spanning tree below.



Spanning tree for initial basic solution

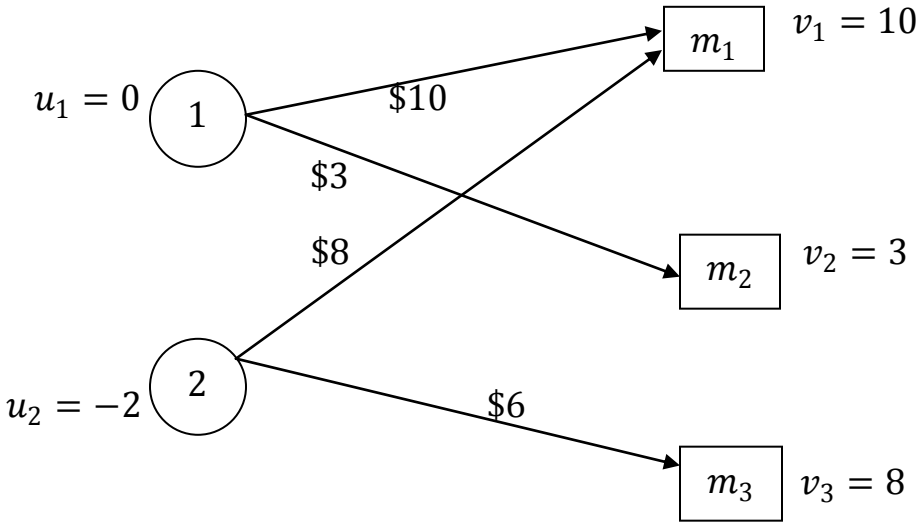
Step2: check the optimality of the basis.

Using the basic variables (variables on the spanning tree) we get the simplex multipliers corresponding to all sources and destinations that satisfy

$$c_{ij} = u_i + v_j \qquad (2)$$

And since condition (2) consists $(m + n - 1 = 4)$ equations in $(m + n = 5)$ any one of the simplex multiplier can be given arbitrary value.

Let $u_1 = 0$. given $u_1 = 0, v_1 = 10, v_2 = 3$ are immediate from (1); then $u_2 = -2$ is immediate from v_1 and $v_3 = 8$ is immediate from u_2.



Determining the simplex multipliers

Now the reduced costs for the non basic variables using $\bar{c}_{ij} = c_{ij} - u_i - v_j$

10	3	9	0
–	–	1	
8	4	6	−2
–	3	–	
10	3	8	u_i / v_j

Since the reduced cost $\bar{c}_{ij} \geq 0$ for the non basic variables (x_{13}^2, x_{22}^2) the solution is optimal.

$x_{ij}^2 = (x_{11}^2, x_{12}^2, x_{13}^2, x_{21}^2, x_{22}^2, x_{23}^2)$ = (110,30,0,40,0,130) is the optimal solution and

$Z_2^* = 2290$ is the optimal value.

Therefore, the optimal solution of the original problem is

$$(x_{ij}^2, x_{ij}^1) = (x_{11}^1, x_{12}^1, x_{13}^1, x_{21}^1, x_{22}^1, x_{23}^1, x_{11}^2, x_{12}^2, x_{13}^2, x_{21}^2, x_{22}^2, x_{23}^2)\text{t}$$

$$= (40,0,110,60,70,0,110,30,0,40,0,130)\text{t}$$

Is the optimal solution and

$Z^* = Z_1^* + Z_2^* =$ 1300+ 2290 = 3590 is the optimal value.

Interpretation: the above solution can interpret it as the minimum cost of shipping 280 units of product A and 310 units of product B from the source nodes to the destination is 3590 and we get this when we send the products through the arcs (paths) $(x_{11}^1, x_{12}^1, x_{13}^1, x_{21}^1, x_{22}^1, x_{23}^1, x_{11}^2, x_{12}^2, x_{13}^2, x_{21}^2, x_{22}^2, x_{23}^2)$t as (40,0,110,60,70,0,110,30,0,40,0,130).

REFERENCES

[1]Mokhtar S. Bazaraa, John J. Jarvis: Linear programming and network flows

[2] Ravindra k.ahuja, Thomas L.magnanti, James B.orlin: network flows

[3] Frederick S. Hillier, Gerald J. Lieberman: introduction to operation research

[4]David G. Luenberger, Yinyu Ye: Linear and nonlinear programming

[5]Mokhtar S. Bazaraa, John J. Jarvis: Linear programming and network flows

[6] Ravindra k.ahuja, Thomas L.magnanti, James B.orlin: network flows